NTRU-ν-um: Secure Fully Homomorphic Encryption from NTRU with Small Modulus

Kamil Kluczniak

CISPA Helmholtz Center for Information Security
kamil.kluczniak@cispa.de

Abstract. NTRUEncrypt is one of the first lattice-based encryption schemes. Furthermore, one of the first fully homomorphic encryption (FHE) schemes were built on the NTRU problem. What makes NTRU appealing when designing cryptosystems is the age of the problem and relatively good performance results when compared to ring learning with errors.
Unfortunately, current fully homomorphic schemes based on NTRU became extremely impractical duo to efficient sublattice attacks. Roughly speaking, these types of (leveled) homomorphic encryption schemes, to support a reasonable depth of the circuit we want to evaluate, require publishing RLWE or NTRU encryptions with a very large modulus. Unfortunately, recovering the sublattice and breaking the NTRU problem for such large moduli turns out to be easy, and to compensate, one would need to choose an impractically large dimension. We call NTRU instances with a too large modulus "overstretched". Due to the sublattice attacks, any serious work on practical NTRU-based fully homomorphic encryption essentially stopped.
In this paper, we reactivate research on practical FHE that can be based on NTRU. To do so, we design an efficient bootstrapping scheme in which the noise growth is small enough to keep the modulus to dimension ratio relatively small, thus avoiding the negative consequences of "overstretching" the modulus. Our bootstrapping algorithm is an accumulation-type bootstrapping scheme analogous to FHEW/TFHE. Finally, we show that we can use the bootstrapping procedure to compute any function over $\mathbb{Z}_p$. Consequently, we obtain one of the fastest FHE schemes to compute arithmetic circuits over finite fields.

1 Introduction

A fully homomorphic encryption scheme gives the possibility to compute any function on encrypted data. Early practical homomorphic encryption schemes were built either from the ring learning with errors problem (e.g. BGV [BV11, BGV12] and BFV [Bra12, FV12]) or the NTRU problem[1] (LTV [LTV12] and YASHE [BLLN13]). Both variants demonstrated similar performance characteristics [CS16]. It is worth noting that NTRUEncrypt by Hoffstein, Pipher, and

[1] The problem is called "Decisional Small Polynomial Ratio Assumption" but here we refer to it briefly as NTRU.

Silverman [HPS98] was among the first lattice-based cryptosystems, is currently subject to standardization [IEE09, ANS10] and considered to be a leading candidate for further standards [AASA+20].

The first subfield attacks against NTRU were due to Gentry, and Szydlo [GS02] which was directed against the NTRU signature scheme. However, the attack did not get much attention since the original NTRU encryption algorithm did not require a large modulus. Further, Albrecht, Bai, and Ducas [ABD16] and independently by Cheon, Jeong and Lee [CJL16] apply the subfield attack to, among other, LTV [LTV12] and YASHE [BLLN13].

Roughly speaking, the NTRU lattice contains a sublattice that, when recovered, allows an attacker to recover the secret key almost immediately. Therefore, when the modulus of an NTRU is too large in comparison to the dimension, then NTRU is broken. Kirchner and Faugue [KF17] later studied the attack and showed that finding the basis vector of the sublattice is faster than recovering the secret key already for moduli as small as $n^{2.783+o(1)}$. The same attack does not apply to ring learning with errors. In order to support correct computation, all schemes BGV, BFV, LTV, and YASHE need to increase the modulus with the depth of the circuit. But since for larger moduli, NTRU is broken, to compensate, we would need to increase its dimension making NTRU-based fully homomorphic encryption schemes uncompetitive to RLWE-based schemes. Very recently, Ducas and van Woerden [DvW21] gave a detailed analysis and estimations, backed by experiments, on the hardness of the NTRU problem when the modulus falls into the overstretched regime.

1.1 Our Contribution.

We give a very competitive scheme based on NTRU that we call NTRU-ν-um (read NTRUnium). Our scheme is an accumulator type of scheme. It has also a low ciphertext rate. Concretely, we need only one element in $\mathbb{Z}_Q$ per plaintext $\mathbb{Z}_p$, for $\log_2(Q) = 30, \ldots, 42$ and $\log_2(p) = 4, \ldots, 11$.

We propose to instantiate the scheme over the ring $\mathbb{Z}_Q[X]/(X^N - 1)$ as in [IEE09, ANS10, AASA+20]. We show that for this ring choice our bootstrapping algorithm, alongside reducing the error, can compute all functions $F : \mathbb{Z}_p \mapsto \mathbb{Z}_p$ where $p \in \mathbb{N}$. For the ring $\mathbb{Z}_Q[X]/(X^N+1)$, similarly to FHEW/TFHE, our bootstrapping can compute all negacyclic functions, i.e., $F(x+p/2 \mod p) = -F(x) \mod p$. In other words, for the circulant version of NTRU, we efficiently compute arithmetic circuits over finite fields. With a single bootstrapping invocation, we can compute any power of x, including the multiplicative inverse. Note that for the negacyclic version, the same is not possible unless we assume the input plaintext to be $< q/2$. Nevertheless, to use such bootstrapping to compute arithmetic circuits over $\mathbb{Z}_p$ becomes much more expensive [KS21, YXS+21].

We show several parameters sets to correctly bootstrap plaintexts from $\mathbb{Z}_p$ where $\log_2 p = 4, \ldots, 11$. We note however that some of our parameters sets can bootstrap plaintexts that are even $\log_2 p = 14$ if the application can tolerate errors.

1.2 Techniques.

Let us first start by recalling the structure of NTRU samples and introducing a gadget NTRU version. Denote $\mathcal{R}_Q = \mathbb{Z}_Q[X]/(X^N - 1)$. An NTRU sample is a polynomial $c \in \mathcal{R}_Q$ of the form $c = e_1/f + e_2 + m$, where $f \in \mathcal{R}_Q$ (usually having coefficients in $\{-1, 0, 1\}$) is the secret key, $e_1, e_2 \in \mathcal{R}_Q$ are the error polynomials and have coefficients from some distribution $\mathcal{X}$, and m is such that $\frac{Q}{p} \cdot m'$ with $m' \in \mathcal{R}_p$. Note that if we want to add two NTRU ciphertexts c, and $c' = e_1'/f + e_2' + m'$, we simply compute $c + c' = (e_1 + e_2)/f + e_2 + e_2' + m + m'$ which is a valid ciphertext of $m + m'$ but with larger error. We can also multiply a ciphertext by a scalar $a \in \mathcal{R}_Q$ such that $c'' = c \cdot a = e_1 \cdot a/f + e_2 \cdot a + m \cdot a$.

Note that the above scalar multiplication is quite expensive as the error terms are multiplied by the scalar a. Hence, to preserve correctness and allow for decryption, we can only multiply with "small" scalars, i.e., scalars that have small coefficients. To resolve the issue we introduce a gadget version of NTRU. Gadget NTRU is analogous to the GSW scheme for (R)LWE, but we adapt the GSW technique to NTRU. In this paper, we will use the gadget NTRU to multiply two ciphertexts and use the fact that the resulting error is relatively small. A gadget NTRU sample is a vector $\mathbf{c}_G = [c_i]_{i=1}^{\ell}$ with $\ell = \log_{\mathsf{L}}(Q)$, where each c_i is a NTRU ciphertext of $m_G \cdot \mathsf{L}^{i-1}$. To multiply such ciphertext with a scalar $c \in \mathcal{R}_Q$, we compute the inner product between $\mathbf{c}_G$ and the decomposition of c in L. Concretely, let $\mathbf{c}_D = \mathsf{Decomp}(c, \mathsf{L})$ be such that $\sum_{i=1}^{\ell} \mathbf{c}_D[i] \cdot \mathsf{L}^{i-1} = c$. Then to multiply a gadget NTRU ciphertext $\mathbf{c}_G$ with c we compute $c_{\mathsf{out}} = \langle \mathbf{c}_G, \mathbf{c}_D \rangle = e_{1,G}/f + e_{2,G} + m_G \cdot c$. Note that when computing the inner product, we make N scalar multiplication and additions, where the scalar multiplications are with polynomials from $\mathcal{R}_{\mathsf{L}}$. Furthermore, if c is itself an NTRU ciphertext, then we have

$$\begin{aligned} c_{\mathsf{out}} &= e_{1,G}/f + e_{2,G} + m_G \cdot (e_1/f + e_2 + m) \\ &= (e_{1,G} + m_G \cdot e_1)/f + e_{2,G} + m_G \cdot e_2 + m_G \cdot m. \end{aligned}$$

which is a valid NTRU ciphertext. Note that the error, in this case, depends on the magnitude of m_G. In this paper, m_G is always a monomial with its coefficient in $\{0, 1\}$.

1.3 Bootstrapping.

Following the ideas from [AP14, DM15, CGGI16], we construct a homomorphic accumulator scheme which we can informally summarize as follows. Remind a LWE sample is a vector $\mathbf{c} \in \mathbb{Z}_N^{n+1}$, where $\mathbf{c}[1] = -\mathbf{c}[2 : n+1]^\top \mathbf{s} + e + \frac{N}{p} m$. To partially decrypt $\mathbf{c}$ it is sufficient to compute the linear function $\mathbf{c}[1] + \mathbf{c}[2 : n+1]^\top \mathbf{s} = e + \frac{N}{p} \cdot m$. Given that the error $e < \frac{N}{2 \cdot p}$, we can further decode the message by $\lfloor \frac{p}{N}(e + \frac{N}{p} m) \rceil = m$. Note that each message is encoded in an interval of size $\frac{N}{p}$ to assure correct decryption.

Now let us consider the operation $\mathsf{rotP} \cdot X^{\mathbf{c}[1] + \mathbf{c}[2:n+1]^\top \mathbf{s}} = \mathsf{rotP} \cdot X^{e + \frac{N}{p} \cdot m} \in \mathcal{R}_Q$. Note that when $\mathcal{R}_Q = \mathbb{Z}_Q[X]/(X^N - 1)$, this operation is a cyclic shift of

the coefficients of rotP by $\mathbf{c}[1] + \mathbf{c}[2:n+1]^\top \mathbf{s} = e + \frac{N}{p} \cdot m \mod N$ positions. Hence, the idea is to set the coefficients of the polynomial rotP such that after rotating it, the desired value for $e + \frac{N}{p} \cdot m$ is encoded in the constant coefficient. At this point, let us note that we can set the coefficients of rotP to any values in $\mathbb{Z}_Q$. Therefore, to compute any function $F : \mathbb{Z}_p \mapsto \mathbb{Z}_p$ on the input plaintext we set the coefficients according to the function $F' : \mathbb{Z}_Q \mapsto \mathbb{Z}_Q$ defined as $F'(x) = \frac{Q}{p} \cdot F(\lfloor \frac{p}{Q} x \rceil)$.

Now we are ready to describe the bootstrapping procedure. Let us assume for simplicity that we want to bootstrap an LWE ciphertext with a secret key $\mathbf{s} \in \{0,1\}^n$. We publish n gadget NTRU ciphertexts that encrypt the bits of the LWE secret key. Denote the vector of those gadget NTRU ciphertexts by $\mathbf{c}_{\mathsf{Bk}}$. Furthermore, we have a NTRU ciphertext c_{acc} that encodes rotP. We call c_{acc} the accumulator. To bootstrap a LWE ciphertext $\mathbf{c}$ we compute

$$\begin{aligned} c_{\mathsf{out}} &= c_{\mathsf{acc}} \cdot X^{\mathbf{c}[1]} \cdot \prod_{i=1}^{n} X^{\mathbf{c}[i+1]} \cdot \mathbf{c}_{\mathsf{Bk}}[i] \\ &= c'_{\mathsf{acc}} \cdot X^{\mathbf{c}[2:n+1]^\top \mathbf{s}}. \end{aligned}$$

where c'_{acc} encrypts rotP just as c_{acc} but with a higher error. Finally we have that the message in c_{out} contains the desired result in its constant coefficient.

The problem now is that if we want to continue computing and bootstrapping on the resulting ciphertext, we need to extract the constant coefficient and switch back to LWE. Hence we design a special key switching procedure that homomorphically extracts the dth coefficient, by computing the linear function

$$(c_{\mathsf{acc}} \cdot f)[d] = \sum_{\substack{i=1, j=1, \\ i+j-2 \ mod \ N = d}}^{N} c_{\mathsf{acc}}[i] \cdot f[i]$$

from the coefficient of the NTRU ciphertext and its secret key. Furthermore, we can now use such NTRU to LWE key switching key to pack N messages into a single NTRU ciphertext which we can then extract for bootstrapping. This allows us to obtain the best ciphertext rate for a fully homomorphic encryption to date.

Note, however, that there is a problem with this solution. Namely, when computing $c_{\mathsf{acc}} \cdot f$ we obtain $m \cdot f$ instead of m. In other words, we have the message masked by the secret key f. So how can we possibly continue to bootstrap such ciphertext? We solve this issue, by including $f^{-1} \in \mathcal{R}_Q$ in the accumulator c_{acc}. That is the accumulator will encrypt $f^{-1} \cdot \mathsf{rotP}$, hence when multiplying f we immediately recover rotP (or the cyclic rotation of rotP). Unfortunately, the trick requires us to assume NTRU is key-dependent message (KDM) secure with respect to f^{-1}. While we do not have a formal reduction, we believe that this version preserves security as we can write such NTRU samples as $c = e_1/f + e_2 + m/f = (e_1 + m)/f + e_2$. In this case, the constant coefficient of the e_1 error is shifted by m. If coefficients of e_1 are random variables with expectations equal to zero, then the KDM version of ciphertext simply shifts the

expectation by the respective coefficients of m. To the best of our knowledge, the expectation of e_1 does not play a role in the cryptanalysis of the NTRU problem. Hence we conjecture that such a KDM version of NTRU preserves security.

Another problem appears when using such a scheme in practice. Namely, since we require the message in the accumulator to be key-dependent, an evaluator cannot freely choose rotation polynomials, and we need to publish all potential accumulators together with the bootstrapping key. To resolve this issue, instead of publishing an accumulator with the rotation polynomials, we can publish a gadget NTRU encryption of $\frac{Q}{p} \cdot f^{-1}$. The evaluator can then choose its own rotP and multiply it with the accumulator. Note that if plaintexts are $\mathbb{Z}_p$, then $\mathsf{rotP} \in \mathcal{R}_p$ and $p << Q$. Hence we actually need to publish a smaller gadget that supports the composition of numbers up to p instead of Q.

1.4 Related Work

Gentry's introduction of the bootstrapping technique [Gen09], opened a floodgate of research on fully homomorphic encryption [BV11, Bra12, FV12, BGV12, AP13, GSW13, HS15, CH18, HS21].

The NTRU problem and the corresponding cryptosystem dates back to work by Hoffstein, Pipher, and Silverman [HPS98]. One of the earliest schemes by López-Alt, Tromer and Vaikuntanathan [LTV12], and its scale-invariant version YASHE [BLLN13] are based on the Stehlé and Steinfeld's [SS11] variant of the NTRU problem.

The first accumulator-based bootstrapping scheme is due to Alperin-Sheriff, and Peikert [AP14]. The techniques require representing the decryption circuit as an arithmetic circuit, and we do not rely on Barrington's theorem. Furthermore, the method exploits error characteristics of the cryptosystem by Gentry, Sahai, and Waters [GSW13], in short, called GSW. Hiromasa, Abe, and Okamoto [HAO15] improved upon [AP14] and build a version of GSW that natively encrypts matrices. Genise et al. [GGH$^+$19] showed an encryption scheme that further improves the efficiency of matrix operations, albeit using a novel NTRU-like assumption.

Ducas and Miccancio [DM15], building on [AP14], design a practical bootstrapping algorithm called FHEW. FHEW uses the ring version of the GSW cryptosystem.

Chillotti et al. [CGGI16, CGGI20] showed numerous optimizations to FHEW bootstrapping algorithm. On the other hand, their scheme called TFHE relies on LWE with binary keys, while FHEW was originally designed to support keys with much larger coefficients. We refer to the work by Micciancio and Polyakov [MP21] for an excellent comparison of both methods.

The FHEW and TFHE bootstrapping algorithms by far are the fastest bootstrapping algorithms to date. Further improvements mostly relied on incorporating packing techniques [CGGI17, MS18], and improved lookup tables evaluation [CGGI17, CIM19].

Initially, FHEW/TFHE were designed to bootstrap ciphertexts with binary plaintexts, but a series of works [BDF18, CIM19, GBA21] showed that extending the computation to larger plaintexts may be beneficial in practice.

Concurrently, Chillotti et al. [CLOT21] and Kluczniak and Schild [KS21], who was very quickly followed by Yang et al. [YXS+21], showed how to resolve the limitation of the FHEW/TFHE functional/programmable bootstrap. In particular, while previous schemes could bootstrap larger plaintexts, due to the negacyclicity of the function that the bootstrapping could compute, it wasn't easy to compute arithmetic circuits over $\mathbb{Z}_p$. The works [CLOT21, KS21, YXS+21] resolve the issue by using TFHE as a subroutine. Still, the resulting bootstrapping algorithms are inherently slower then the original TFHE algorithm, and so far only [KS21, YXS+21] implemented their schemes.

Concurrent and Independent Work. We note that Bonte et al. [BIP+22] independently published similar to ours fully homomorphic encryption scheme. In particular, they also define a gadget NTRU cryptosystem and build an accumulator bootstrapping algorithm. We note that there are several differences in our designs. The most crucial difference seems to be that Bonte et al. build their scheme with binary ciphertexts in mind while we compute arbitrary functions on plaintexts from $\mathbb{Z}_p$. Furthermore, we currently instantiate our scheme on $\mathbb{Z}_Q/(X^N-1)$ which was analyzed in [DvW21] and the same choice is made in the standards [IEE09, ANS10] and the round three NIST candidate [AASA+20] for post-quantum encryption and signature schemes. Bonte et al. instantiate their scheme over the ring $\mathbb{Z}_Q[X]/(X^N+1)$. There are also some very technical differences, like the way both works extract LWE ciphertexts. We describe a generalized algorithm that we can later use to extract LWE samples from the packed NTRU ciphertexts. We note that Bonte et al. [BIP+22] show a faster blind rotation algorithm for ternary keys. On the other hand, we perform key switching before modulus reduction, which gives us a much smaller error in the LWE ciphertexts, which is supposed to be bootstrapped. We note that it seems all optimization can be applied in both works.

2 Preliminaries

Notation. We denote as $\mathcal{R}$ the ring of polynomials $\mathbb{Z}[X]/\Psi$ and as $\mathcal{R}_Q = \mathcal{R}/q\mathbb{Z}$ the ring of polynomials with coefficients in $\mathbb{Z}_q$. In this paper $\Psi = (X^N - 1)$ for N-prime, or $\Psi = (X^N + 1)$ for N being a power-of-two. We denote vectors with a bold lowercase letter, e.g., $\mathbf{v}$, and matrices with uppercase letters $\mathbf{V}$. We denote a n dimensional column vector as $[f(.,i)]_{i=1}^{n}$, where $f(.,i)$ defines the i-th coordinate. For brevity, we will also denote as $[n]$ the vector $[i]_{i=1}^{n}$, and more generally $[n,m]_{i=n}^{m}$ the vector $[n,\ldots,m]^\top$. We address the ith entry of a vector $\mathbf{v}$ by $\mathbf{v}[i]$, and denote a slice of the vector by $\mathbf{v}[i,j]$. For a random variable $a \in \mathbb{Z}$ we denote as $\mathsf{Var}(a)$ the variance of a, as $\mathsf{stddev}(x)$ its standard deviation and as $\mathsf{E}(x)$ its expectation. For $a \in \mathcal{R}_Q$, we define $\mathsf{Var}(a)$, $\mathsf{stddev}(a)$ and $\mathsf{E}(a)$ to be the largest variance, standard deviation and expectation respectively among the

coefficients of the polynomial a. By $\mathsf{Ham}(\mathbf{a})$ we denote the hamming weight of vector $\mathbf{a}$, i.e., the number of of non-zero coordinates of $\mathbf{a}$.

Throughout the paper we denote as $q \in \mathbb{N}$ and $Q \in \mathbb{N}$ two moduli. The parameter $n \in \mathbb{N}$ always denotes the dimension of a LWE sample, that we define below. For rings, we always use N to denote the degree of Ψ. We define $\ell = \lceil \log_{\mathsf{L}} q \rceil$ for some decomposition basis $\mathsf{L} \in \mathbb{N}$. Often we mark different decomposition bases $\mathsf{L}_{\mathsf{name}}$ and the corresponding ℓ_{name}, or bounds $\mathcal{B}_{\mathsf{name}}$ with some subscript name.

We recall the learning with errors assumption by Regev [Reg05].

Definition 1 (Learning With Errors). *Let $\mathbf{s} \in \mathbb{Z}_Q$ be a secret key, and $e \in \mathcal{X}_Q$ for a error distribution $\mathcal{X}_Q$. We define a LWE sample of a message $m \in \mathbb{Z}_Q$ as $\mathbf{c} = \mathsf{LWE}_e(\mathbf{s}, m) \in \mathbb{Z}_Q$ where $\mathbf{c}[1] = -\mathbf{c}[2 : n+1]^\top \cdot \mathbf{s} + e + m \in \mathbb{Z}_Q$, and $\mathbf{c}[2 : n+1]$ is a vector that is chosen from the uniform distribution over $\mathbb{Z}_Q$.*

We define the phase of $\mathbf{c}$ as $\mathsf{Phase}(\mathbf{c}) = \mathbf{c}[1] + \mathbf{c}[2 : n+1]^\top \cdot \mathbf{s}$.

Lemma 1 (Linear Homomorphism of GLWE samples). *Let $\mathbf{c} = \mathsf{LWE}_{e_{\mathbf{c}}}(\mathbf{s}, m_{\mathbf{c}})$ and $\mathbf{d} = \mathsf{LWE}_{e_{\mathbf{d}}}(\mathbf{s}, m_{\mathbf{d}})$. If $\mathbf{c}_{\mathsf{out}} \leftarrow \mathbf{c} + \mathbf{d}$, then $\mathbf{c}_{\mathsf{out}} \in \mathsf{LWE}_{e_{\mathsf{out}}}(\mathbf{s}, m)$, where $m = m_{\mathbf{c}} + m_{\mathbf{d}}$, and $Var(e_{\mathsf{out}}) \leq \mathsf{Var}(e_{\mathbf{c}}) + \mathsf{Var}(e_{\mathbf{d}})$. Furthermore, let $d \in \mathbb{Z}_Q$. If $\mathbf{c}_{\mathsf{out}} \leftarrow \mathbf{c} \cdot d$, then $\mathbf{c}_{\mathsf{out}} \in \mathsf{GLWE}_{e_{\mathsf{out}}}(\mathbf{s}, m_{\mathbf{c}} \cdot d)$, where $\mathsf{Var}(e_{\mathsf{out}}) = \frac{d^2-1}{12} \cdot \mathsf{Var}(e_{\mathbf{c}})$.*

3 Homomorphic Encryption Techniques from NTRU

In this section, we describe the algorithms that we use to build the bootstrapping algorithm. Below we recall the basic cryptosystem. First of all, we describe the algorithm as a symmetric key cryptosystem. Specifically, we do not define a public key version, as it is unnecessary in this paper and simplifies the exposition.

Definition 2 (A NTRU Homomorphic Encryption Scheme). *Let $\mathcal{R}_Q = \mathbb{Z}_Q[X]/\psi$ where ψ is a polynomial of degree N. Let $\mathcal{X}_{\mathsf{sk}}$, $\mathcal{X}_i$ for $i \in [2]$ be distributions over $\mathcal{R}_Q$. Set the message modulus as $t < Q$, and define the scaling factor $\Delta_{Q,t} = \lfloor \frac{Q}{t} \rceil$. Let the secret key $f \in \mathcal{X}_{\mathsf{sk}}$ be such that f has an inverse in $\mathcal{R}_Q$. We define an NTRU encryption as $c = \mathsf{NTRU}_{e_1,e_2}(f, \Delta_{Q,t} \cdot m) = e_1 \cdot f^{-1} + e_2 + m$, where $e_1 \in X_1$ and $e_2 \in \mathcal{X}_2$ and $m \in \mathcal{R}_t$.*

To decrypt we compute $f \cdot m = \lfloor \frac{t}{Q} \cdot c \cdot f \rceil \in \mathcal{R}_t$. If additional f has an inverse in $\mathcal{R}_t$, then we can recover $m \in \mathcal{R}_t$.

Lemma 2 (Correctness of the NTRU Decryption Procedure). *Let $c = \mathsf{NTRU}_{e_1,e_2}(f, \Delta_{Q,t} \cdot m)$. We have that $m \cdot f + e = f \cdot c$, where $\mathsf{Var}(e) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2)$. Furthermore if $e < \frac{Q}{2 \cdot t}$, then $\lfloor \frac{t}{Q} \cdot c \cdot f \rceil = m \cdot f \in \mathbb{Z}_t$.*

Proof. From definition we have that $c = e_1 \cdot f^{-1} + e_2 + \Delta_{Q,t} \cdot m \in \mathcal{R}_Q$ where $m \in \mathcal{R}_t$. Then $f \cdot c = e_1 + f \cdot e_2 + f \cdot m = e + f \cdot m$. Note that $e = e_1 + f \cdot e_2$. Hence $\mathsf{Var}(e) = \mathsf{Var}(e_1) + \mathsf{Var}(f \cdot e_2) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2)$. The

above follows from the fact that f and e_2 are independent and centered around zero. Specifically, the variance of the dth coordinate of $f \cdot e_2$ is given by

$$\mathsf{Var}((f \cdot e_2)[d]) = \sum_{\substack{i=1,j=1,\\ i+j-2 \ mod \ q=d}}^{N} \mathsf{Var}(f[i] \cdot e_2[j]).$$

Hence if f has $\mathsf{Ham}(f)$ coordinates of variance $\mathsf{Var}(f)$ and all other set to zero, we have $\mathsf{Var}((f \cdot e_2)[d]) = \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2)$. Finally, if $e < \frac{Q}{2 \cdot t}$, then $\lfloor \frac{t}{Q} \cdot (e + \Delta_{Q,t} \cdot f \cdot m) \rceil = \lfloor \frac{t}{Q} \cdot e + f \cdot m \rceil = f \cdot m \in \mathcal{R}_t$, because $\frac{t}{Q} \cdot e < \frac{1}{2}$.

Below we analyze the variance of the errors for elementary homomorphic operations.

Lemma 3 (Affine Functions on Encrypted Data). *Let $\mathcal{R}$ contain polynomials of degree N. Let $c_1 = \mathsf{NTRU}_{e_1,e_2}(f, m_1)$ and $c_2 = \mathsf{NTRU}_{\tilde{e}_1,\tilde{e}_2}(f, m_2)$, $m, m_1, m_2 \in \mathbb{Z}_Q$ and $a \in \mathcal{R}_Q$. We have the following.*

Scalar Addition: $c_1 + a = \mathsf{NTRU}_{\bar{e}_1,\bar{e}_2}(\mathsf{sk}, m_1 + a)$, *where* $\mathsf{Var}(\bar{e}_1) = \mathsf{Var}(e_1)$ *and* $\mathsf{Var}(\bar{e}_2) = \mathsf{Var}(e_1)$.

Scalar Multiplication by Monomial: $c_1 \cdot m \cdot X^k = \mathsf{NTRU}_{\bar{e}_1,\bar{e}_2}(\mathsf{sk}, m_1 \cdot m \cdot X^k)$, *where* $k < N$, $\mathsf{Var}(\bar{e}_1) = m^2 \cdot \mathsf{Var}(e_1)$ *and* $\mathsf{Var}(\bar{e}_2) = m^2 \cdot \mathsf{Var}(e_2)$. *If m is a uniformly random variable in $\mathbb{Z}_t$, then* $\mathsf{Var}(\bar{e}_1) = \frac{(t^2-1)}{12} \cdot \mathsf{Var}(e_1)$ *and* $\mathsf{Var}(\bar{e}_2) = \frac{(t^2-1)}{12} \cdot \mathsf{Var}(e_2)$.

Scalar Multiplication by Polynomial: $c_1 \cdot a = \mathsf{NTRU}_{\bar{e}_1,\bar{e}_2}(\mathsf{sk}, m_1 \cdot a)$, *where* $\mathsf{Var}(\bar{e}_i) \leq \mathsf{Ham}(a) \cdot ||a||_\infty \cdot \mathsf{Var}(e_i)$ *for $i \in [2]$. If a is a uniformly random variable in $\mathcal{R}_t$, then* $\mathsf{Var}(\bar{e}_i) = \mathsf{Ham}(a) \cdot \frac{(t^2-1)}{12} \cdot \mathsf{Var}(e_i)$.

Addition: $c_1 + c_2 = \mathsf{NTRU}_{\bar{e}_1,\bar{e}_2}(\mathsf{sk}, m_1 + m_2)$, *where* $\mathsf{Var}(\bar{e}_1) = \mathsf{Var}(e_1) + \mathsf{Var}(\tilde{e}_1)$ *and* $\mathsf{Var}(e_2) = \mathsf{Var}(e_1) + \mathsf{Var}(\tilde{e}_1)$.

Proof. From definition we have $c_1 = e_1/f_1 + e_2 + m_1$ and $c_2 = \tilde{e}_1/f_1 + \tilde{e}_1 + m_2$. Clearly scalar addition preserve the error variance. Let $i \in [2]$. For monomial multiplication we have $\bar{e}_i = e_i \cdot m \cdot X^k$. Hence $\mathsf{Var}(\bar{e}_i) = \frac{(t^2-1)}{12} \cdot \mathsf{Var}(e_i)$ in the ring $\mathcal{R}_Q$, because we multiply each coefficient by m and do a cyclic rotation if $\mathcal{R}_Q = \mathbb{Z}_Q[X]/(X^N - 1)$ and a negacyclic rotation if $\mathcal{R}_Q = \mathbb{Z}_Q[X]/(X^q + 1)$. Multiplying with a polynomial a is analogous to multiplying with a monomial, however in that case the variance of the error polynomials $\mathsf{Var}(\bar{e}_i) = \mathsf{Ham}(a) \cdot \frac{(t^2-1)}{12} \cdot \mathsf{Var}(e_i)$. Addition follows from $c_1 + c_2 = e_1/f_1 + e_2 + m_1 + \tilde{e}_1/f_1 + \tilde{e}_1 + m_2 = (e_1 + \tilde{e}_1)/f_1 + e_1 + \tilde{e}_1 + m_1 + m_2$.

3.1 NTRU to LWE Key Switching

Definition 3 (NTRU to LWE Key Switching). *Let $\mathcal{R}_Q = \mathbb{Z}_Q[X]/\Phi$ with $\Phi = X^N - 1$ or $\Phi = X^N + 1$. Let $f \in \mathcal{X}_{\mathsf{sk}}$. Let $\mathbf{s} \in \mathbb{Z}_Q$ be a LWE secret key. Let $\mathcal{X}_{\mathsf{Ksk}}$ be a distribution over $\mathbb{Z}_Q$. Let $\mathsf{L} \in \mathbb{N}$ be a decomposition parameter and denote $\ell = \lceil \log_{\mathsf{L}} Q \rceil$. Let ψ be a function that on input a monomial $a \cdot X^k$ with*

a coefficient $a \in \mathbb{Z}_Q$ and an index $d \in [N]$, outputs the dth coefficient of $a \cdot X^k \bmod \Phi$.

We define the NTRU-to-LWE key switching key as $\mathsf{Ksk}[i,*] \leftarrow [\mathsf{LWE}_{e_{i,k}}(\mathbf{s}, f[i] \cdot \mathsf{L}^{k-1})]_{k=1}^{\ell}$ *for* $i \in [N]$ *and* $e_{i,k} \in \mathcal{X}_{\mathsf{Ksk}}$. *We call* $\mathbf{E} = [e_{i,k}]_{i=1,k=1}^{\ell,N}$ *the error matrix of the key switching key. To indicate that we a key switching key is from an NTRU key f to LWE key $\mathbf{s}$, and that $\mathbf{E}$ is its corresponding error matrix, we write* $\mathsf{Ksk}_{f,\mathbf{s}}^{\mathbf{E}}$.

The key switching procedure KeySwitch *is defined as*

$$\mathsf{KeySwitch}(\mathsf{Ksk}, c, d) = \sum_{\substack{i=1,j=1,\\ i+j-2 \ mod \ N=d}}^{N} \langle \mathsf{Ksk}[i,*], \mathsf{Decomp}(\psi(c[j] \cdot X^{i+j-2}, d), \mathsf{L}) \rangle,$$

where $c = \mathsf{NTRU}_{e_1,e_2}(f, m)$, $m \in \mathbb{Z}_Q$.

Lemma 4 (Correctness of NTRU to LWE Key Switching). *Let* $\mathsf{Var}(e_{\mathsf{Ksk}})$ *be a random variable over $\mathbb{Z}_Q$ such that for all $i \in [N]$ and $k \in [\ell]$ we have that* $\mathsf{Var}(e_{i,k}) \leq \mathsf{Var}(e_{\mathsf{Ksk}})$. *If* $\mathbf{c} = \mathsf{KeySwitch}(\mathsf{Ksk}, c, d)$ *for $d \in [N]$, then* $\mathbf{c} = \mathsf{LWE}_e(\mathbf{s}, (f \cdot m)[d])$, *where* $\mathsf{Var}(e) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2) + N \cdot \ell \cdot \frac{\mathsf{L}^2-1}{12} \cdot \mathsf{Var}(e_{\mathsf{Ksk}})$.

Proof. Let us denote $[a_k^{(i,j)}]_{k=1}^{\ell} = \mathsf{Decomp}(\psi(c[j] \cdot X^{i+j-2}, d), \mathsf{L})$, which are such that $a_k^{(i,j)} < \mathsf{L}$ and $\sum_{k=1}^{\ell} a_k^{(i,j)} \cdot \mathsf{L}^{k-1} = \psi(c[j] \cdot X^{i+j-2}, d)$. In the main summation of the key switching procedure we have

$$\begin{aligned}\langle \mathsf{Ksk}[i,*], \mathsf{Decomp}(\psi(c[j] \cdot X^{i+j-2}, d), \mathsf{L}) \rangle &= \mathsf{LWE}_{\bar{e}_{i,j}}(\mathbf{s}, \sum_{k=1}^{\ell} f[i] \cdot \mathsf{L}^{k-1} \cdot a_k^{(i,j)}) \\ &= \mathsf{LWE}_{\bar{e}_{i,j}}(\mathbf{s}, f[i] \cdot \psi(c[j] \cdot X^{i+j-2}, d))\end{aligned}$$

where $\mathsf{Var}(\bar{e}_{i,j}) = \ell \cdot \frac{\mathsf{L}^2-1}{12} \cdot \mathsf{Var}(e_{\mathsf{Ksk}})$. Then we have

$$\begin{aligned}\mathsf{KeySwitch}(\mathsf{Ksk}, c, d) &= \sum_{\substack{i=1,j=1,\\ i+j-2 \ mod \ N=d}}^{N} \mathsf{LWE}_{\bar{e}_{i,j}}(\mathbf{s}, f[i] \cdot \psi(c[j] \cdot X^{i+j-2}, d)), \\ &= \mathsf{LWE}_{e'}(\mathbf{s}, (f \cdot c)[d]) \\ &= \mathsf{LWE}_{e'}(\mathbf{s}, e_1[d] + (f \cdot e_2)[d] + (f \cdot m)[d]) \\ &= \mathsf{LWE}_{e}(\mathbf{s}, (f \cdot m)[d]),\end{aligned}$$

where $\mathsf{Var}(e') = N \cdot \ell \cdot \frac{\mathsf{L}^2-1}{12} \cdot \mathsf{Var}(e_{\mathsf{Ksk}})$ because there are exactly N pairs $i, j \in [N]$ such that $i + j - 2 \ mod \ N = d$. Then from correctness of the NTRU decryption procedure we have $\mathsf{Var}(e_1 + f \cdot e_2) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2)$. Therefore $\mathsf{Var}(e) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2) + N \cdot \ell \cdot \frac{\mathsf{L}^2-1}{12} \cdot \mathsf{Var}(e_{\mathsf{Ksk}})$.

3.2 Gadget Encryption and Gadget Multiplication

Definition 4 (NTRU Gadget Encryption and Multiplication). *Let $\mathcal{R}_Q$ be a ring, $\mathsf{L} \in \mathbb{N}$ be a decomposition parameter, and $\mathcal{X}_1$, $\mathcal{X}_2$ be error distributions over $\mathcal{R}_Q$. Let $f \in \mathcal{R}_Q$ be such that both have inverses in $\mathcal{R}_Q$. Denote as $\ell = \lceil \log_{\mathsf{L}}(Q) \rceil$. We define a gadget NTRU sample of a message $m_G \in R_Q$ as $\mathbf{c}_G = \mathsf{G\text{-}NTRU}_{\mathbf{e}_1,\mathbf{e_2}}(f, m_G) = [\mathsf{NTRU}_{\mathbf{e}_1[i],\mathbf{e}_2[i]}(f, m_G \cdot \mathsf{L}^{i-1})]_{i=1}^{\ell}$, where $\mathbf{e}_1 \in \mathcal{X}_1^{\ell}$ and $\mathbf{e}_2 \in \mathcal{X}_2^{\ell}$. We define the gadget multiplication procedure GMul as $\mathsf{GMul}(\mathbf{c}_G, c) = \langle \mathbf{c}_G, \mathsf{Decomp}c, \mathsf{L} \rangle$, where $c \in \mathcal{R}_Q$ and in particular $c = \mathsf{NTRU}_{e_1,e_2}(f, m)$.*

Note that the gadget ciphertext is nothing more than a vector of NTRU ciphertexts of $m_G \cdot f \cdot \mathsf{L}^{i-1}$. Addition and scalar multiplication for individual gadget ciphertexts is as for NTRU. In this paper we will never directly decrypt a gadget ciphertext, hence we omit describing the decryption algorithm.

Below we analyze the correctness of the gadget multiplication algorithm. In this paper we use only the case where the message m_G is a monomial with its coefficient in $\mathbb{Z}_2$. Hence we will focus the analysis only to this special case for simplicity.

Lemma 5 (Correctness of NTRU Gadget Multiplication). *If $c_{\mathsf{out}} = \mathsf{GMul}(\mathbf{c}_G, c)$, then $c_{\mathsf{out}} = \mathsf{NTRU}_{e_{\mathsf{out},1},e_{\mathsf{out},2}}(f, c \cdot m_G)$, where*

$$\mathsf{Var}(e_{\mathsf{out},i}) = N \cdot \ell \cdot (\frac{\mathsf{L}^2 - 1}{12}) \cdot \mathsf{Var}(\mathbf{e}_i).$$

for $i \in [2]$.

If additionally $c = \mathsf{NTRU}_{e_1,e_2}(f, m)$, then

$$\mathsf{Var}(e_{\mathsf{out},i}) = \mathsf{Var}(e)_i + N \cdot \ell \cdot (\frac{\mathsf{L}^2 - 1}{12}) \cdot \mathsf{Var}(\mathbf{e}_i).$$

Proof. Let $[c_k]_{k=1}^{\ell} = \mathsf{Decomp}(c, \mathsf{L})$ which is such that $\sum_{k=1}^{\ell} c_k \cdot \mathsf{L}^{k-1} = c \in \mathcal{R}_Q$. From definition we have

$$\begin{aligned} c' &= \langle \mathbf{c}_G, \mathsf{Decomp}c, \mathsf{L} \rangle \\ &= \sum_{k=1}^{\ell} \mathsf{NTRU}_{\mathbf{e}_1[k],\mathbf{e_2}[k]}(f, m_G \cdot \mathsf{L}^{k-1}) \cdot c_k \\ &= \mathsf{NTRU}_{e_1',e_2'}(f, m_G \cdot \sum_{k=1}^{\ell} c_k \cdot \mathsf{L}^{k-1}) \\ &= \mathsf{NTRU}_{e_1',e_2'}(f, m_G \cdot c). \end{aligned}$$

From linear homomorphism of NTRU we have $\mathsf{Var}(e_i') = N \cdot \ell \cdot (\frac{\mathsf{L}^2-1}{12}) \cdot \mathsf{Var}(\mathbf{e}_i)$ for $i \in [2]$. Note that in the above we treat c_k as ring elements uniformly distributed in $\mathcal{R}_{\mathsf{L}}$. Since the coefficient of m_G is smaller than or equal 1 we have

$$\begin{aligned} c' &= e_1'/f + e_2' + m_G \cdot c \\ &= e_1'/f + e_2' + m_G \cdot (e_1/f + e_2 + m) \\ &\leq (e_1' + e_1)/f + e_2' + e_2 + m_g \cdot m. \end{aligned}$$

To summarize we have $\mathsf{Var}(e_{\mathsf{out},i}) = \mathsf{Var}(e)_i + N \cdot \ell \cdot (\frac{\mathsf{L}^2-1}{12}) \cdot \mathsf{Var}(\mathbf{e}_i)$ for $i \in [2]$.

Below we remind the modulus switching algorithm and recall its correctness.

Lemma 6 (Modulus Switching). *Let* $\mathbf{c} = \mathsf{LWE}_e(\mathbf{s}, \Delta_{Q,t} \cdot m)$*, where* $\mathbf{s} \in \mathbb{Z}_Q^n$*. Let us define the modulus switching procedure as* $\mathsf{ModSwitch}(\mathbf{c}, q) = \lfloor [\mathbf{c} \cdot \frac{q}{Q}]_{i=1}^{n+1} \rceil$ *for* $q \leq Q$*. If* $\mathbf{c}_{\mathsf{out}} \leftarrow \mathsf{ModSwitch}(\mathbf{c}, q)$*, then* $\mathbf{c}_{\mathsf{out}} = \mathsf{LWE}_{e_{\mathsf{out}}}(\mathbf{s}, \Delta q, t \cdot m)$*, where*

$$\mathsf{Var}(e_{\mathsf{out}}) = (\frac{q}{Q})^2 \cdot \mathsf{Var}(e) + \frac{1}{12} + \frac{\mathsf{Ham}(\mathbf{s})}{12} \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2) + \frac{q}{2 \cdot Q}$$

Proof. Let $\mathbf{c}[1] + \mathbf{c}[2:n+1]^\top \cdot \mathbf{s} = e + \Delta_{Q,t} \cdot m$. From definition we have

$$\begin{aligned}
\mathbf{c}_{\mathsf{out}}[1] + \mathbf{c}_{\mathsf{out}}[2:n+1]^\top \cdot \mathbf{s} &= \lfloor \frac{q}{Q} \cdot \mathbf{c}[1] \rceil + \lfloor \frac{q}{Q} \mathbf{c}_{\mathsf{out}}[2:n+1]^\top \cdot \mathbf{s} \rceil \\
&= \frac{q}{Q} \cdot \mathbf{c}[1] + r + \frac{q}{Q} \mathbf{c}_{\mathsf{out}}[2:n+1]^\top \cdot \mathbf{s} + \mathbf{r}^\top \cdot \mathbf{s} \\
&= \frac{q}{Q} \cdot (\mathbf{c}[1] + \mathbf{c}_{\mathsf{out}}[2:n+1]^\top \cdot \mathbf{s}) + r + \mathbf{r}^\top \cdot \mathbf{s} \\
&= \frac{q}{Q} \cdot \Delta_{Q,t} \cdot m + \frac{q}{Q} + e + r + \mathbf{r}^\top \cdot \mathbf{s} \\
&= \frac{q}{t} \cdot m + \frac{q}{Q} \cdot (\epsilon \cdot m + e) + r + \mathbf{r}^\top \cdot \mathbf{s}, \\
&\leq \frac{q}{t} \cdot m + \frac{q}{Q} \cdot (\epsilon \cdot t + e) + r + \mathbf{r}^\top \cdot \mathbf{s}
\end{aligned}$$

where $r \in [-1/2, 1/2]$ and $\mathbf{r} \in [-1/2, 1/2]^n$. We assume that r and $\mathbf{r}$ are uniform random. Hence $\mathbf{c}_{\mathsf{out}} = \mathsf{LWE}_{e_{\mathsf{out}}}(\mathbf{c}, \Delta_{q,t} \cdot m)$, where $e_{\mathsf{out}} = \frac{q}{Q} \cdot (\epsilon \cdot t + e) + r + \mathbf{r}^\top \cdot \mathbf{s}$. Therefore, $\mathsf{Var}(e_{\mathsf{out}}) = \mathsf{Var}(\frac{q}{Q} \cdot e) + \mathsf{Var}(r) + \mathsf{Var}(\mathbf{r}^\top \cdot \mathbf{s})$. Clearly we have $\mathsf{Var}(\frac{q}{Q} \cdot e) = (\frac{q}{Q})^2 \cdot \mathsf{Var}(e)$. Then $\mathsf{Var}(r) = \mathsf{Var}(\mathbf{r}[i]) = 1/12$ for $i \in [n]$ and the expectation for all these variables is 0. Since $\mathbf{r}$ and $\mathbf{s}$ are independent, we have $\mathsf{Var}(\mathbf{r}^\top \cdot \mathbf{s}) = \sum_{i=1}^n \mathsf{Var}(\mathbf{r}[i]) \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2)$ in general when all coordinates of $\mathbf{s}$ are random variable. However, if we set $n - \mathsf{Ham}(\mathbf{s})$ coordinates of $\mathbf{s}$ to zero, then we get

$$\begin{aligned}
\mathsf{Var}(\mathbf{r}^\top \cdot \mathbf{s}) &= \sum_{i=1}^{\mathsf{Ham}(\mathbf{s})} \mathsf{Var}(\mathbf{r}[i]) \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2) \\
&= \frac{\mathsf{Ham}(\mathbf{s})}{12} \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2)
\end{aligned}$$

To summarize we have

$$\mathsf{Var}(e_{\mathsf{out}}) = (\frac{q}{Q})^2 \cdot \mathsf{Var}(e) + \frac{1}{12} + \frac{\mathsf{Ham}(\mathbf{s})}{12} \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2) + \frac{q}{2 \cdot Q}$$

4 Computing on Ciphertexts and Bootstrapping

Below we give our bootstrapping algorithm. In the procedure, we use a vector $\mathbf{u} \in \mathbb{Z}^u$ for which the following holds. For all $y \in S$ where $S = \{-1, 0, 1\}$ or

$S = \{0,1\}$ there exist $x \in \{0,1\}^u$ such that $y = \sum_{j=1}^{u} x[i] \cdot \mathbf{u}[i]$. For example for ternary S we have $\mathbf{u} = [-1, 1]$ and for binary we have $\mathbf{u} = [1]$.

Definition 5 (The NTRU-ν-um Bootstrapping Procedure). *Let $\mathbf{s} \in \mathbb{Z}_Q^n$ be a LWE secret key, and $f \in \mathcal{R}_Q$ a NTRU secret key, for a modulus Q and ring $\mathcal{R}_Q$. Let $q \in \mathbb{N}$ be the smallest integer such that $X^q = 1 \in \mathcal{R}_Q$. Let $\mathbf{s}' = \mathbb{Z}_Q^{u \cdot n}$ be such that $\mathbf{s}[i] = \sum_{j=1}^{u} \mathbf{s}'[i,j] \cdot \mathbf{u}[j]$. We define the blind rotation key Bk as*

$$\mathsf{Bk}[i,j] = \mathsf{G\text{-}NTRU}_{e_{\mathsf{Bk},1,i,j}, e_{\mathsf{Bk},2,i,j}}(f, \mathbf{s}'[i,j]),$$

for $i \in [n]$ and $j \in [u]$. We denote $\mathbf{E}_{\mathsf{Bk}} = [[e_{\mathsf{Bk},1,i,j}, e_{\mathsf{Bk},2,i,j}]]_{i=1,j=1}^{n,u}$ the error matrix of the blind rotation key. Similarly to the key switching key, we write $\mathsf{Bk}_{f,\mathbf{s}}^{\mathbf{E}_{\mathsf{Bk}}}$ to indicate that the blind rotation key is with respect to the NTRU secret key f, LWE secret key $\mathbf{s}$ and error matrix $\mathbf{E}_{\mathsf{Bk}}$.

$\mathsf{Bootstrap}(\mathsf{Ksk}_{f,\mathbf{s}}^{\mathbf{E}}, \mathsf{Bk}_{f,\mathbf{s}}^{E_{\mathsf{Bk}}}, c_{\mathsf{acc}}, d)$: *The algorithm takes as input an NTRU ciphertext $c = \mathsf{NTRU}_{e_1,e_2}(f, \Delta_{Q,t_1} \cdot m \cdot f^{-1})$, a NTRU-to-LWE keyswitching key $\mathsf{Ksk}_{f,\mathbf{s}}^{\mathbf{E}}$, the bootstrapping key Bk, an accumulator $c_{\mathsf{acc}} = \mathsf{NTRU}_{e_{\mathsf{acc},1}, e_{\mathsf{acc},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot f^{-1})$, and an index $d \in [N]$. The bootstrapping procedure is as follows.*

1. $\mathbf{c}_{\mathsf{LWE}} \leftarrow \mathsf{KeySwitch}(\mathsf{Ksk}, c, d)$.
2. $\mathbf{c}_{\mathsf{in}} \leftarrow \mathsf{ModSwitch}(\mathbf{c}_{\mathsf{LWE}}, q)$.
3. $c_{\mathsf{acc}} \leftarrow c_{\mathsf{acc}} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]}$.
4. *For $i \in [n]$*
   4.1. *For $j \in |u|$:*

$$c_{\mathsf{acc}} \leftarrow \mathsf{GMul}(\mathsf{Bk}[i,j], c_{\mathsf{acc}} \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{u}[j]} - c_{\mathsf{acc}}) + c_{\mathsf{acc}}$$

5. *Output $c_{\mathsf{out}} = c_{\mathsf{acc}}$.*

4.1 Setting up the Rotation Polynomial and the Accumulator

Before giving the formal analysis of the bootstrapping algorithm, let us briefly explain how to choose the rotation polynomial rotP. Suppose we want to bootstrap a ciphertext that holds the message $m \in \mathbb{Z}_{t_1}$, and along the way we want to compute the function $F : \mathbb{Z}_{t_1} \mapsto \mathbb{Z}_{t_2}$. To do so we need to construct a rotation polynomial $\mathsf{rotP} \in \mathcal{R}_Q$. When working over the ring $\mathbb{Z}_Q[X]/(X^N - 1)$ with N prime, then setting up the polynomial is fairly easy. In this case $N = q$ and we set the coefficients as $\mathsf{rotP}[y+1] = \Delta_{Q,t_1} \cdot F(\lfloor \frac{t_1 \cdot y}{q} \rceil) \mod Q$ for all $y \in \mathbb{Z}_q$.

When working over the ring $\mathbb{Z}_Q[X]/(X^N + 1)$ where N is a power of two we have $q = 2 \cdot N$, and we can compute functions F such that $F(x + \lceil t_1/2 \rceil \mod t_1) = -F(x \mod t_1) \mod t_1$ for $x \in \mathbb{Z}_{t_1}$. We set $\mathsf{rotP}[y+1] = \Delta_{Q,t_1} \cdot F(\lfloor \frac{t_1 \cdot y}{q} \rceil) \mod Q$ for all $y \in \mathbb{Z}_N$. Note that $y \in \mathbb{Z}_N$ where $N = q/2$.

4.2 Correctness of the Bootstrapping Algorithm

Below we give the correctness and noise analysis of our bootstrapping algorithm.

Theorem 1 (Correctness of the Bootstrapping Algorithm).
Let $\mathsf{rotP} \in \mathcal{R}_Q$ *be such that* $(\mathsf{rotP} \cdot X^y)[1] = F(\lfloor \frac{t_1}{q} \cdot y \rceil)$, *where* $y \in \mathbb{Z}_q$ *and* $F : \mathbb{Z}_{t_1} \mapsto \mathbb{Z}_{t_2}$. *Let* $c_{\mathsf{acc}} = \mathsf{NTRU}_{e_{\mathsf{acc},1},e_{\mathsf{acc},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot f^{-1})$ *and* $c = \mathsf{NTRU}_{e_1,e_2}(f, \Delta_{Q,t_1} \cdot m \cdot f^{-1})$.

- *We have* $\mathbf{c}_{\mathsf{in}} = \mathsf{LWE}_{e_{\mathsf{in}}}(\mathbf{s}, (\Delta_{q,t_1} \cdot m)[d])$ *and* $(\mathsf{rotP} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]+\mathbf{c}_{\mathsf{in}}[2:n+1]^\top \cdot \mathbf{s}})[1] = F(m)$ *given that* $e_{\mathsf{in}} < \frac{q}{2 \cdot t_2}$, *where*

$$\mathsf{Var}(e_{\mathsf{in}}) = (\frac{q}{Q})^2 \cdot \mathsf{Var}(e_{\mathsf{LWE}}) + \frac{1}{12} + \frac{\mathsf{Ham}(\mathbf{s})}{12} \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2) + \frac{q}{2 \cdot Q}$$

  and

$$\mathsf{Var}(e_{\mathsf{LWE}}) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2) + N \cdot \ell_{\mathsf{Ksk}} \cdot \frac{\mathsf{L}_{\mathsf{Ksk}}^2 - 1}{12} \cdot \mathsf{Var}(e_{\mathsf{Ksk}})$$

- *If* $c_{\mathsf{out}} \leftarrow \mathsf{Bootstrap}(\mathsf{Ksk}_{f,\mathbf{s}}^{\mathbf{E}}, \mathsf{Bk}_{f,\mathbf{s}}^{E_{\mathsf{Bk}}}, c_{\mathsf{acc}}, d)$, *then* $c_{\mathsf{out}} = \mathsf{NTRU}_{e_{\mathsf{out},1},e_{\mathsf{out},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]+\mathbf{c}_{\mathsf{in}}[2:n+1]^\top \cdot \mathbf{s}} \cdot f^{-1})$, *where for* $l \in [2]$

$$\mathsf{Var}(e_{\mathsf{out},l}) = \mathsf{Var}(e_{\mathsf{acc},l}) + n \cdot u \cdot N \cdot \ell_{\mathsf{Bk}} \cdot (\frac{\mathsf{L}_{\mathsf{Bk}}^2 - 1}{12}) \cdot \mathsf{Var}(\mathbf{E}_{\mathsf{Bk}}).$$

Proof. First note that $c_{\mathsf{acc}} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]} = \mathsf{NTRU}_{e_{\mathsf{acc},1},e_{\mathsf{acc},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot f^{-1} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]})$. Let $c_{\mathsf{acc,curr}} = \mathsf{NTRU}_{e_{\mathsf{curr},1},e_{\mathsf{curr},2}}(f, m_{\mathsf{curr}})$ for some $m_{\mathsf{curr}} \in \mathcal{R}_Q$.

If

$$c_{\mathsf{acc,next}} \leftarrow \mathsf{GMul}(\mathsf{Bk}[i,j], c_{\mathsf{acc,curr}} \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{u}[j]} - c_{\mathsf{acc,curr}}) + c_{\mathsf{acc,curr}}$$

then $c_{\mathsf{acc,next}} = \mathsf{NTRU}_{e_{\mathsf{acc,next},1},e_{\mathsf{acc,next},2}}(f, m_{\mathsf{curr}} \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{u}[j]})$, where $\mathsf{Var}(e_{\mathsf{acc},i,l}) = e_{\mathsf{curr},l} + N \cdot \ell \cdot (\frac{\mathsf{L}^2-1}{12}) \cdot \mathsf{Var}(\mathbf{E}_{\mathsf{Bk}})$ for $l \in [2]$. Note that $\mathsf{Bk}[i,j]$ encrypts $\mathbf{s}'[i,j] \in \{0,1\}$. If $\mathbf{s}'[i,j] = 0$, then from correctness of $GMul$ we have

$$c_{\mathsf{acc,next}} = \mathsf{NTRU}_{e_{\mathsf{GMul},1},e_{\mathsf{GMul},2}}(f, 0) + c_{\mathsf{acc,curr}} = \mathsf{NTRU}_{e_{\mathsf{acc,next},1},e_{\mathsf{acc,next},2}}(f, m_{\mathsf{curr}} \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{u}[j]}),$$

where for $l \in [2]$ we have $\mathsf{Var}(e_{\mathsf{GMul},l}) = N \cdot \ell_{\mathsf{Bk}} \cdot (\frac{\mathsf{L}_{\mathsf{Bk}}^2-1}{12}) \cdot \mathsf{Var}(\mathbf{E}_{\mathsf{Bk}})$. If $\mathbf{s}'[i,j] = 0$, then again from correctness of $GMul$ we have

$$\begin{aligned} c_{\mathsf{acc,next}} &= \mathsf{NTRU}_{e_{\mathsf{GMul},1},e_{\mathsf{GMul},2}}(f, c_{\mathsf{acc,curr}} \cdot \mathbf{s}'[i,j] \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{u}[j]} - c_{\mathsf{acc,curr}}) + c_{\mathsf{acc,curr}} \\ &= \mathsf{NTRU}_{e_{\mathsf{GMul},1},e_{\mathsf{GMul},2}}(f, c_{\mathsf{acc,curr}} \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{s}'[i,j] \cdot \mathbf{u}[j]}) \\ &= \mathsf{NTRU}_{e_{\mathsf{acc,next},1},e_{\mathsf{acc,next},2}}(f, m_{\mathsf{curr}} \cdot X^{\mathbf{c}_{\mathsf{in}}[i+1] \cdot \mathbf{s}'[i,j] \cdot \mathbf{u}[j]}). \end{aligned}$$

In either case we have

$$\mathsf{Var}(e_{\mathsf{acc,next},l}) = \mathsf{Var}(e_{\mathsf{acc,curr},l}) + N \cdot \ell_{\mathsf{Bk}} \cdot (\frac{\mathsf{L}_{\mathsf{Bk}}^2 - 1}{12}) \cdot \mathsf{Var}(\mathbf{E}_{\mathsf{Bk}})$$

Then after $n \cdot u$ iterations we have

$$\begin{aligned} c_{\mathsf{out}} &= \mathsf{NTRU}_{e_{\mathsf{out},1},e_{\mathsf{out},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]+\sum_{i=1}^{n} \mathbf{c}_{\mathsf{in}}[i+1]\cdot\sum_{j=1}^{u} \mathbf{s}'[i,j]\cdot\mathbf{u}[j]}) \\ &= \mathsf{NTRU}_{e_{\mathsf{out},1},e_{\mathsf{out},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]+\sum_{i=1}^{n} \mathbf{c}_{\mathsf{in}}[i+1]\cdot\mathbf{s}[i]}) \\ &= \mathsf{NTRU}_{e_{\mathsf{out},1},e_{\mathsf{out},2}}(f, \Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]+\mathbf{c}_{\mathsf{in}}[2:n+1]^\top\cdot\mathbf{s}} \cdot f^{-1}) \end{aligned}$$

where $\mathsf{Var}(e_{\mathsf{acc},l}) + n \cdot uN \cdot \ell_{\mathsf{Bk}} \cdot (\frac{\mathsf{L}_{\mathsf{Bk}}^2-1}{12}) \cdot \mathsf{Var}(\mathbf{E}_{\mathsf{Bk}})$ as desired.

On the other hand we have $\mathbf{c}_{\mathsf{LWE}} = \mathsf{LWE}_{e_{\mathsf{LWE}}}(\mathbf{s}, (\Delta_{Q,t_2} \cdot m)[d])$, where $\mathsf{Var}(e) = \mathsf{Var}(e_1) + \mathsf{Ham}(f) \cdot \mathsf{Var}(f) \cdot \mathsf{Var}(e_2) + N \cdot \ell \cdot \frac{\mathsf{L}^2-1}{12} \cdot \mathsf{Var}(e_{\mathsf{Ksk}})$ from correctness of key switching. Furthermore, we have $\mathbf{c}_{\mathsf{in}} = \mathsf{LWE}_{e_{\mathsf{in}}}(\mathbf{s}, (\Delta_{q,t_2} \cdot m)[d])$, where $\mathsf{Var}(e_{\mathsf{in}}) = (\frac{q}{Q})^2 \cdot \mathsf{Var}(e_{\mathsf{LWE}}) + \frac{1}{12} + \frac{\mathsf{Ham}(\mathbf{s})}{12} \cdot (\mathsf{Var}(s) + \mathsf{E}(s)^2) + \frac{q}{2\cdot Q}$ from correctness of modulus switching. Finally $(\mathsf{rotP} \cdot X^{\mathbf{c}_{\mathsf{in}}[1]+\mathbf{c}_{\mathsf{in}}[2:n+1]^\top\cdot\mathbf{s}})[1] = F(m)$ follows from the definition of rotP and the size restriction of the error of $\mathsf{Var}(e_{\mathsf{in}})$.

Remark 1. Note that to estimate correctness, we need to take the lowest correctness among the correctness of decryption of c_{out} and correctness of decryption of $\mathbf{c}_{\mathsf{in}}$. Furthermore, it is crucial to estimate the error of the $\mathbf{c}_{\mathsf{in}}$ ciphertext when the bootstrapping algorithm gets as input an NTRU ciphertext that itself was an outcome of bootstrapping. This is important because the error of bootstrapped ciphertexts is usually higher than the error of fresh ciphertexts.

4.3 Computing on Encrypted Data and Packing

To compute on encrypted data, we can use the homomorphism of NTRU ciphertexts to compute affine functions over $\mathbb{Z}_{t_1}$. After bootstrapping a ciphertext holding a message m at position d we obtain a NTRU ciphertexts that has the rotation polynomial rotated such that its first coefficient has $(\Delta_{Q,t_2} \cdot \mathsf{rotP} \cdot f^{-1}) = \Delta_{Q,t_2} \cdot F(m)$. Note that we can still compute affine functions on these ciphertexts with monomials of degree zero. But any further bootstrapping must extract the LWE from position $d = 1$. Note that then working over $\Phi = X^q - 1$, we can compute any function F without the negacyclicity assumption. In this case we can for example correctly compute $x^2 \mod t_1$ for $x \in \mathbb{Z}_{t_1}$. When working over $\Phi = X^q + 1$ we cannot easily compute such functions since x^2 isn't negacyclic, and we would need to resort to much more expensive techniques [CLOT21, KS21, YXS+21]. If furthermore $\mathbb{Z}_{t_1}$ contains inverses of 4 then we can compute $x \cdot y = (\frac{x+y}{2})^2 - (\frac{x-y}{2})^2 \mod t_1$ with only two invocations of the bootstrapping algorithm. This way we can efficiently compute arithmetic circuits. Furthermore, the arithmetic can be easily extended to composite $\mathbb{Z}_p$ with $p = \prod_{i=1}^{m} t_{1,i}$ where all $t_{1,i}$ are pairwise co-prime from the Chinese remainder theorem.

Encrypting Data. To send encrypted data, we have a few options. We can either send LWE ciphertexts, with modulus q and error distribution being a Gaussian with standard deviation matching the standard deviation of the error of the

LWE ciphertext after switching the modulus. Then, instead of key switching the input NTRU ciphertexts, we can immediately start to compute step four on the LWE ciphertexts. The downside of this method is that we require $n+1$ elements in $\mathbb{Z}_q$ to transmit a single message.

Another method is to set a message into a coefficient of the NTRU ciphertext. Then we run the bootstrapping algorithm on this NTRU ciphertext for $d \in [N]$. This way, we may transmit N messages at the cost of only N elements in $\mathbb{Z}_Q$. We note that for the initial NTRU ciphertext, we may actually take a smaller modulus and obtain an even better ciphertext rate. The modulus Q is chosen to support the error induced by the blind rotation part of the bootstrapping as well as the NTRU to LWE key switching part. When sending a fresh NTRU ciphertext, the error doesn't have to be that large; hence we can lower the modulus size for this particular ciphertext. It is also easy to see that we don't need another NTRU to LWE key switch key for such ciphertexts. Since the modulus is smaller than Q, we have enough "powers" in the key switching key to support homomorphic decryption.

Finally, the naive way to return the outcome of the computation is to return the NTRU ciphertext from the last invocation of bootstrapping (or after additionally computing some affine functions on a vector of NTRU ciphertexts). In this case, the ciphertext rate for the result is rather weak since we transmit N elements in $\mathbb{Z}_Q$ per message. What we can do, is either run the NTRU to LWE switching procedure to reduce the rate to $n+1$ elements in $\mathbb{Z}_q$, or we can try to pack the outcome multiple NTRU ciphertexts into a single NTRU ciphertext. For this purpose, we need an additional packing key that works as the NTRU to LWE key switching procedure but has NTRU ciphertexts instead of LWE ciphertexts.

Building Accumulators. Note that the accumulator that we give as input to the bootstrapping procedure is key-dependent. There are a few options on how to provide such an accumulator.

- We send an accumulator as part of the evaluation key. In this case, the evaluation key has fixed rotation polynomials, and we compute only these functions defined by the evaluation key.
- A trivial option is to give and encryption of $\Delta_{Q,t_2} \cdot f^{-1} \in \mathcal{R}_Q$, and let the evaluator multiply the ciphertext by $\mathsf{rotP} \in \mathcal{R}_{t_2}$.
- When t_2 is large, the above method may yield an accumulator with a large error. To make the accumulator's error independent of the magnitude of the coefficients of rotP, we publish a $\mathsf{G\text{-}NTRU}$ ciphertext, and obtain the accumulator by invoking GMul with rotP.

5 Security and Parameters

In this section, we give our parameter sets and estimate the correctness of these parameters for different plaintext moduli. All our parameters are targeted to achieve 128-bits of security. Across different parameter sets we use the same

standard deviation for all NTRU ciphertexts. The NTRU secret key f is assumed to have coefficients from $\{-1, 0, 1\}$, hence $\mathsf{Var}(f) = 2/3$. We choose the errors e_1 and e_2 for the NTRU ciphertexts such that $\mathsf{Var}(e_1 + e_2 \cdot f) = \mathsf{Var}(e \cdot (g + f))$ for some $g \in \mathcal{R}_Q$ with the same parameters as f, where $\mathsf{stddev}e = 3.2$. For the key switching key, we take either ternary or binary keys.

	n	$log_2(Q)$	N	L_{Bk}	$\mathsf{stddev}_{\mathsf{Ksk}}$	Ksk [MB]	Bk [MB]	ct [KB]
Ternary LWE Secret Key								
NTRU-ν-um-C-11-T	$2^9 + 212$	30	$2^{11} - 9$	2^4	2^{12}	178.17	47.44	8.19
NTRU-ν-um-C-12-T	$2^9 + 250$	38	$2^{12} - 3$	2^{10}	2^{19}	593.79	62.42	20.48
NTRU-ν-um-C-13-T	$2^9 + 360$	42	$2^{13} - 1$	2^9	2^{20}	1802.20	214.30	49.15
NTRU-ν-um-C-14-T	$2^9 + 360$	42	$2^{14} - 3$	2^9	2^{20}	3604.41	428.60	98.304
Binary LWE Secret Key								
NTRU-ν-um-C-11-B	$2^9 + 240$	30	$2^{11} - 9$	2^6	2^{12}	185.05	30.80	8.19
NTRU-ν-um-C-12-B	$2^9 + 315$	36	$2^{12} - 3$	2^8	2^{16}	610.46	84.68	20.48
NTRU-ν-um-C-13-B	$2^9 + 355$	41	$2^{13} - 1$	2^9	2^{20}	1749.22	213.07	49.15
NTRU-ν-um-C-14-B	$2^9 + 390$	42	$2^{14} - 3$	2^9	2^{20}	3728.27	443.35	98.30

Table 1. Parameter sets for circulant NTRU (over the ring $\mathbb{Z}[X]_Q/(X^N - 1)$). The hamming weight of these parameters is not enforces, and all coefficients are from the same distribution. In other words we have $\mathsf{Ham}(f) = N$ and $\mathsf{Ham}(\mathbf{s}) = n$.

To estimate security for the LWE samples used the LWE estimator [APS15]. Note that for the NTRU ciphertexts, the modulus Q passes the fatigue point. To estimate security, we use the estimator from [DvW21] to calculate the BKZ block size needed to recover a basis vector of the dense sublattice. Based on the block size, we estimate the running time of BKZ using the cost model from [BDGL16]. Since it is beneficial to work over larger rings, we notice that for most parameter sets, the security bottleneck lies with the LWE samples of the key switching key instead of the NTRU ciphertexts.

Below we show correctness estimates for plaintext space $\mathbb{Z}_{t_1}$, for $t_1 = 2^4, \ldots, 2^{11}$. The sk row gives the share of variance that comes from the LWE secret key when modulus switching. Specifically, we calculate what percentage of the total variance of $\mathbf{c}_{\mathsf{in}}$ consists of the variance of the rounding error. Other rows give the correctness of bootstrapping for $\log_2(t_1) = 4, \ldots, 11$. Each entry contains two lower bounds on the probabilities of failure. The first is the lower-bound on the probability of failing to decrypt the ciphertext c_{out}, the second is the lower-bound on the probability to decrypt $\mathbf{c}_{\mathsf{in}}$. We note that the variance of the error of $\mathbf{c}_{\mathsf{in}}$ is calculated assuming the ciphertext input to the bootstrapping procedure was c_{out} from a previous bootstrapping operation.

NTRU-ν-um	C-11-T	C-12-T	C-13-T	C-14-T
sk	60.13%	72.56%	98.54%	86.51%
4	$0,2^{-48}$	0,0	0.00,0.00	0.00,0.00
5	$2^{-32},2^{-14}$	0,0	0,0	0,0
6	$2^{-10},2^{-5}$	$0,2^{-16}$	0,0	0,0
7	0.11,0.32	$2^{-17},2^{-5}$	$0,2^{-18}$	0,0
8	0.42,0.62	2^{-6},0.29	$0,2^{-6}$	$0,2^{-16}$
9	0.69,0.80	0.27,0.60	0,0.25	$2^{-42},2^{-5}$
10	0.84,0.90	0.58,0.79	2^{-42},0.56	2^{-12},0.28
11	0.92,0.95	0.78,0.89	2^{-12},0.77	0.06,0.59
NTRU-ν-um	C-11-B	C-12-B	C-13-B	C-14-B
sk	10.45%	64.43%	93.36%	82.96%
4	$2^{-14},2^{-13}$	0,0	0,0	0,0
5	0.05,0.06	0,0	0,0	0,0
6	0.32,0.35	$2^{-44},2^{-17}$	0,0	0,0
7	0.62,0.64	2^{-13},0.02	$0,2^{-22}$	0,0
8	0.80,0.81	0.06,0.27	$0,2^{-7}$	$0,2^{-19}$
9	0.90,0.90	0.35,0.58	2^{-42},0.19	2^{-41},0.01
10	0.95,0.95	0.64,0.78	2^{-12},0.52	2^{-12},0.23
11	0.97,0.97	0.81,0.89	0.06,0.74	0.07,0.55

Table 2. Correctness estimates for our circulant parameters.

References

AASA+20. Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, et al. Status report on the second round of the nist post-quantum cryptography standardization process. *US Department of Commerce, NIST*, 2020.

ABD16. Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 153–178. Springer, Heidelberg, August 2016.

ANS10. X9 ANSI. 98: Lattice-based polynomial public key establishment algorithm for the financial services industry. Technical report, Technical report, ANSI, 2010.

AP13. Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasi-linear time. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 1–20. Springer, Heidelberg, August 2013.

AP14. Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 297–314. Springer, Heidelberg, August 2014.

APS15. Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

BDF18. Guillaume Bonnoron, Léo Ducas, and Max Fillinger. Large FHE gates from tensored homomorphic accumulator. In Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *AFRICACRYPT 18: 10th International Conference on Cryptology in Africa*, volume 10831 of *Lecture Notes in Computer Science*, pages 217–251. Springer, Heidelberg, May 2018.

BDGL16. Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In Robert Krauthgamer, editor, *27th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 10–24. ACM-SIAM, January 2016.

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012: 3rd Innovations in Theoretical Computer Science*, pages 309–325. Association for Computing Machinery, January 2012.

BIP+22. Charlotte Bonte, Ilia Iliashenko, Jeongeun Park, Hilder V. L. Pereira, and Nigel P. Smart. Final: Faster fhe instantiated with ntru and lwe. Cryptology ePrint Archive, Report 2022/074, 2022. https://ia.cr/2022/074.

BLLN13. Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In Martijn Stam, editor, *Cryptography and Coding*, pages 45–64, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

Bra12. Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, Heidelberg, August 2012.

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 97–106. IEEE Computer Society Press, October 2011.

CGGI16. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 3–33. Springer, Heidelberg, December 2016.

CGGI17. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 377–408. Springer, Heidelberg, December 2017.

CGGI20. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91, January 2020.

CH18. Hao Chen and Kyoohyung Han. Homomorphic lower digits removal and improved FHE bootstrapping. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 315–337. Springer, Heidelberg, April / May 2018.

CIM19. Sergiu Carpov, Malika Izabachène, and Victor Mollimard. New techniques for multi-value input homomorphic evaluation and applications. In Mitsuru Matsui, editor, *Topics in Cryptology – CT-RSA 2019*, volume 11405 of *Lecture Notes in Computer Science*, pages 106–126. Springer, Heidelberg, March 2019.

CJL16. Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee. An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low-level encoding of zero. *LMS Journal of Computation and Mathematics*, 19(A):255–266, 2016.

CLOT21. Ilaria Chillotti, Damien Ligier, Jean-Baptiste Orfila, and Samuel Tap. Improved programmable bootstrapping with larger precision and efficient arithmetic circuits for tfhe. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 670–699, Cham, 2021. Springer International Publishing.

CS16. Ana Costache and Nigel P. Smart. Which ring based somewhat homomorphic encryption scheme is best? In Kazue Sako, editor, *Topics in Cryptology – CT-RSA 2016*, volume 9610 of *Lecture Notes in Computer Science*, pages 325–340. Springer, Heidelberg, February / March 2016.

DM15. Léo Ducas and Daniele Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 617–640. Springer, Heidelberg, April 2015.

DvW21. Léo Ducas and Wessel van Woerden. Ntru fatigue: How stretched is overstretched? In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 3–32, Cham, 2021. Springer International Publishing.

FV12. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. https://eprint.iacr.org/2012/144.

GBA21. Antonio Guimarães, Edson Borin, and Diego F. Aranha. Revisiting the functional bootstrap in tfhe. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(2):229–253, Feb. 2021.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178. ACM Press, May / June 2009.

GGH+19. Nicholas Genise, Craig Gentry, Shai Halevi, Baiyu Li, and Daniele Micciancio. Homomorphic encryption for finite automata. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part II*, volume 11922 of *Lecture Notes in Computer Science*, pages 473–502. Springer, Heidelberg, December 2019.

GS02. Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 299–320. Springer, Heidelberg, April / May 2002.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, Heidelberg, August 2013.

HAO15. Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. Packing messages and optimizing bootstrapping in GSW-FHE. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 699–715. Springer, Heidelberg, March / April 2015.

HPS98. Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. Ntru: A ring-based public key cryptosystem. In Joe P. Buhler, editor, *Algorithmic Number Theory*, pages 267–288, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

HS15. Shai Halevi and Victor Shoup. Bootstrapping for HElib. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 641–670. Springer, Heidelberg, April 2015.

HS21. Shai Halevi and Victor Shoup. Bootstrapping for HElib. *Journal of Cryptology*, 34(1):7, January 2021.

IEE09. Ieee standard specification for public key cryptographic techniques based on hard problems over lattices. *IEEE Std 1363.1-2008*, pages 1–81, 2009.

KF17. Paul Kirchner and Pierre-Alain Fouque. Revisiting lattice attacks on overstretched NTRU parameters. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 3–26. Springer, Heidelberg, April / May 2017.

KS21. Kamil Kluczniak and Leonard Schild. Fdfb: Full domain functional bootstrapping towards practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2021/1135, 2021. https://ia.cr/2021/1135.

LTV12. Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *44th Annual ACM Symposium on Theory of Computing*, pages 1219–1234. ACM Press, May 2012.

MP21. Daniele Micciancio and Yuriy Polyakov. *Bootstrapping in FHEW-like Cryptosystems*, page 17–28. Association for Computing Machinery, New York, NY, USA, 2021.

MS18. Daniele Micciancio and Jessica Sorrell. Ring packing and amortized FHEW bootstrapping. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP 2018: 45th International Colloquium on Automata, Languages and Programming*, volume 107 of *LIPIcs*, pages 100:1–100:14. Schloss Dagstuhl, July 2018.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93. ACM Press, May 2005.

SS11. Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 27–47. Springer, Heidelberg, May 2011.

YXS+21. Zhaomin Yang, Xiang Xie, Huajie Shen, Shiying Chen, and Jun Zhou. Tota: Fully homomorphic encryption with smaller parameters and stronger security. Cryptology ePrint Archive, Report 2021/1347, 2021. https://ia.cr/2021/1347.