Candidate Multilinear Maps from Ideal Lattices

Sanjam Garg[1⋆], Craig Gentry[2⋆⋆], and Shai Halevi[2⋆⋆]

[1] UCLA
[2] IBM Research

Abstract. We describe plausible lattice-based constructions with properties that approximate the sought-after multilinear maps in hard-discrete-logarithm groups, and show an example application of such multi-linear maps that can be realized using our approximation. The security of our constructions relies on seemingly hard problems in ideal lattices, which can be viewed as extensions of the assumed hardness of the NTRU function.

1 Introduction

Bilinear maps are extremely useful tools in cryptography. After being used to construct non-interactive key agreement [SOK00], tripartite Diffie-Hellman [Jou00], and identity-based encryption [BF01], the applications of bilinear maps have become too numerous to name. Boneh and Silverberg [BS03] argued that multilinear maps would have even more interesting applications, including multipartite Diffie-Hellman and very efficient broadcast encryption. They attempted to construct multilinear maps from abelian varieties (extending known techniques for constructing bilinear maps), but they identified serious obstacles, and concluded that "such maps might have to either come from outside the realm of algebraic geometry, or occur as 'unnatural' computable maps arising from geometry".

Since then, the persistent absence of cryptographically useful multilinear maps as not stopped researchers from proposing applications of them. For example, Rückert and Schröder [RS09] use multilinear maps to construct efficient aggregate and verifiably encrypted signatures without random oracles. Papamanthou et al. [PTT10] show that compact multilinear maps give very efficient

⋆ Research conducted while at the IBM Research, T.J. Watson funded by NSF Grant No.1017660.

⋆⋆ This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center (DoI/NBC) contract number D11PC20202. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

authenticated data structures. Rothblum [Rot13] uses multilinear maps to construct a counterexample to the conjecture that all bit-encryption schemes are KDM-secure (secure when given bit-encryptions of the secret key).

Here, we construct multilinear maps from ideal lattices. Our multilinear maps are "noisy" and bounded to polynomial degree. For very high degree, the "noisiness" overwhelms the signal, somewhat like for ciphertexts in somewhat homomorphic encryption schemes. In light of their noisiness, one could say that our multilinear maps are indeed "unnatural" computable maps arising from geometry. Our candidate multilinear maps differ quite substantially from the "ideal" multilinear maps envisioned by Boneh and Silverberg, in particular some problems that are hard relative to contemporary bilinear maps are easy with our construction (see Section 4.2). Nonetheless, the multilinear analog of the decision Diffie-Hellman problem appears hard in our construction, which gives cause for optimism about its applications in cryptography. In this paper we only demonstrate the applicability of our candidate to the "obvious" application of multipartite Diffie-Hellman key exchange, but other applications are surly possible.

The boundedness of our encodings has interesting consequences, both positive and negative. On the positive side, it hinders an attack based on Boneh and Lipton's subexponential algorithm for solving the discrete logarithm in black box fields [BL96]. This attack cannot be used to solve the "discrete log" problem in our setting, since their algorithm requires exponentiations with exponential degree. On the negative size, the dependence between the degree and parameter-size prevents us from realizing applications such that Papamanthou et al. [PTT10] that needs "compact" maps. Similarly, so far we were not able to use our maps to realize Rothblum's counterexample to the KDM-security of bit encryption conjecture [Rot13]: That counterexample requires degree that is polynomial, but a polynomial that is always just out of our reach of our parameters.

The security of the multilinear-DDH problem in our constructions relies on new hardness assumptions, and we provide an extensive cryptanalysis to validate these assumptions. To make sure that our constructions are not "trivially" insecure, we prove that our constructions are secure against adversaries that merely run a straight-line program. We also analyze our constructions with respect to the best known averaging, algebraic and lattice attacks. Many of these attacks have been published before [CS97,HKL$^+$00,Gen01,GS02,Szy03,HGS04,NR06] in cryptanalysis of the NTRU [HPS01,HHGP$^+$03] and GGH [GGH97] signature schemes. We also present new attacks on *principal* ideal lattices, which arise in our constructions, that are more efficient than (known) attacks on general ideal lattices. Our constructions remain secure against all of the attacks that we present, both old and new. However, we feel that more cryptanalysis needs to be done, and this is partly why we have tried to write our cryptanalysis sections as a thorough survey that will serve as a useful starting point for cryptanalysts.

A brief overview. Our constructions work in polynomial rings and use principal ideals in these rings (and their associated lattices). In a nutshell, an instance of

our construction has a secret short ring element $\mathbf{g} \in R$, generating a principal ideal $\mathcal{I} = \langle \mathbf{g} \rangle \subset R$. In addition, it has an integer parameter q and another secret $\mathbf{z} \in R/qR$, which is chosen at random (and hence is not small).

We think of a term like g^x in a discrete-log system as an "encoding" of the "plaintext exponent" x. In our case the role of the "plaintext exponents" is played by the elements in $R/\mathcal{I}$ (i.e. cosets of $\mathcal{I}$), and we "encode" it via division by $\mathbf{z}$ in R_q. In a few more details, our system provides many levels of encoding, where a level-i encoding of the coset $\boldsymbol{e}_{\mathcal{I}} = \boldsymbol{e} + \mathcal{I}$ is an element of the form $\boldsymbol{c}/\mathbf{z}^i \bmod q$ where $\boldsymbol{c} \in \boldsymbol{e}_{\mathcal{I}}$ is short. It is easy to see that such encodings can be both added and multiplied, so long as the numerators remain short. More importantly, we show that it is possible to publish a "zero testing parameter" that enables to test if two elements encode the same coset at a given level, without violating security (e.g., it should still be hard to compute x from an encoding of x at higher levels). Namely, we add to the public parameters an element of the form $\mathbf{p}_{zt} = h \cdot z^\kappa / g \bmod q$ for a not-too-large h, and show that multiplying an encoding of 0 by $\mathbf{p}_{zt} \pmod q$ yields a small element, while multiplying an encoding of a non-zero by $\mathbf{p}_{zt} \pmod q$ yields a large element. Hence we can distinguish zero from non-zero, and by subtraction we can distinguish two encodings of the same element from encodings of two different elements.

Our schemes are somewhat analogous to graded algebras, hence we sometimes call them *graded encoding schemes*. Our schemes are quite flexible, and for example can be modified to support the analog of asymmetric maps by using several different z's. On the other hand, other variants such as composite-order groups turn out to be insecure with our encodings (at least when implemented in a straightforward manner).

Organization. We define the general notion of encoding that we use in Section 2, as well an abstract notion of our main hardness assumption (which is a multilinear analog of DDH). Then in Section 3 we provide some background on ideal lattices, and in Section 4 we describe our construction.

Applications. In the full version [GGH12]we describe the application to multipartite key agreement. Using our multilinear maps [GGH$^+$13] have provided a construction of an attribute based encryption (ABE) scheme for general circuits. Concurrently and independently Gorbunov, Vaikuntanathan, and Wee [GVW13] also achieve ABE for circuits. One nice feature of their result is that they reduce security to the Learning with Errors (LWE) problem. Goldwasser, Kalai, Popa, Vaikuntanathan, and Zeldovich [GKP$^+$13] show how one can use such an ABE scheme along with fully homomorphic encryption to construct a succinct single use functional encryption scheme. This in turn implies results for reusable Yao garbled circuits and other applications. Subsequent to our work, using our multilinear maps, Garg, Gentry, Sahai, and Waters [GGSW13] constructed a witness encryption scheme where a user's decryption key need not be an actual "key" at all, but rather can be a witness for some arbitrary NP relation specified by the encrypter (the encrypter itself may not know a witness).

2 Multilinear Maps and Graded Encoding Systems

Below we define formally our notion of a "approximate" multilinear maps, which we call *graded encoding schemes* (termed after the notion of graded algebra). Before presenting our notion, we briefly recall *cryptographic multilinear maps* as defined by Boneh and Silverberg [BS03].

For $\kappa + 1$ cyclic groups $G_1, \ldots, G_\kappa, G_T$ (written additively) of the same order p, a *κ-multilinear map* $e : G_1 \times \cdots \times G_\kappa \to G_T$ is non-degenerate (in the sense that if $\{g_i \in G_i\}_{i=1,\ldots,\kappa}$ are all generators of their respective groups, then $e(g_1, \ldots, g_\kappa)$ is a generator of G_T), and it satisfies

$$e(g_1, \ldots, \alpha \cdot g_i, \ldots, g_\kappa) = \alpha \cdot e(g_1, \ldots, g_\kappa),$$

for any elements $\{g_i \in G_i\}_{i=1,\ldots,\kappa}$, index $i \in [\kappa]$ and integer $\alpha \in \mathbb{Z}_p$. (Boneh and Silverberg considered in [BS03] only the *symmetric* case $G_1 = \cdots = G_\kappa$, the asymmetric case with different G_i's was considered, e.g., by Rothblum in [Rot13].)

Cryptographic multilinear maps come with efficient procedures for generating parameters of such κ-multilinear map and for computing the group operation in each group and the map itself. They also come with some hardness properties, at least the discrete logarithm must be hard in the respective groups. Other hardness assumptions include the multilinear-DDH (MDDH) assumption (among others), asserting that given $\kappa+1$ random elements in the source group, it is hard to compute (or even recognize) the target-group element whose discrete logarithm is the product of the logarithms of all these $\kappa + 1$ elements.

2.1 Graded Encoding Schemes

The starting point for our new notion is viewing group elements in multilinear-map schemes as just a convenient mechanism of encoding the exponent: Typical applications of bilinear (or multilinear) maps use $\alpha \cdot g_i$ as an "obfuscated encoding" of the "plaintext integer" $\alpha \in \mathbb{Z}_p$. This encoding supports limited homomorphism (i.e., linear operations and a limited number of multiplications) but no more. In our setting we retain this concept of a somewhat homomorphic encoding, and have an algebraic ring (or field) R playing the role of the exponent space $\mathbb{Z}_p$. However we dispose of the multitude of algebraic groups, replacing them with "unstructured" sets of encodings of ring elements.

Perhaps the biggest difference between our setting and the setting of cryptographic multilinear maps, is that our encoding is randomized, which means that the same ring-element can be encoded in many different ways. (We do not even insist that the "plaintext version" of a ring element has a unique representation.) This means that checking if two strings encode the same element may not be trivial, indeed our constructions rely heavily on this check being feasible for some encodings and not feasible for others.

Another important difference is that our system lets us multiply not only batches of κ encodings at the time, but in fact any subset of encodings. This

stands in stark contrast to the sharp threshold in multi-linear maps, where you can multiply exactly κ encodings, no more and no less.

A consequence of the ability to multiply any number of encodings is that we no longer have a single target group, instead we have a different "target group" for any number of multiplicands. This yields a richer structure, roughly analogous to *graded algebra*. In its simplest form (analogous to symmetric maps with a single source group), we have levels of encodings: At level zero we have the "plaintext" ring elements $\alpha \in R$ themselves, level one corresponds to $\alpha \cdot g$ in the source group, and level-i corresponds to a product of i level-1 encodings (so level-κ corresponds to the target group from multilinear maps).

Definition 1 (κ-Graded Encoding System). *A κ-Graded Encoding System consists of a ring R and a system of sets $\mathcal{S} = \{S_i^{(\alpha)} \subset \{0,1\}^* : \alpha \in R,\ 0 \le i \le \kappa, \}$, with the following properties:*

1. *For every fixed index i, the sets $\{S_i^{(\alpha)} : \alpha \in R\}$ are disjoint (hence they form a partition of $S_i \stackrel{\text{def}}{=} \bigcup_\alpha S_i^{(\alpha)}$).*
2. *There is an associative binary operation '$+$' and a self-inverse unary operation '$-$' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every index $i \le \kappa$, and every $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, it holds that*

$$u_1 + u_2 \in S_i^{(\alpha_1+\alpha_2)} \quad \text{and} \quad -u_1 \in S_i^{(-\alpha_1)}$$

   where $\alpha_1 + \alpha_2$ and $-\alpha_1$ are addition and negation in R.
3. *There is an associative binary operation '$\times$' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every i_1, i_2 with $i_1 + i_2 \le \kappa$, and every $u_1 \in S_{i_1}^{(\alpha_1)}$ and $u_2 \in S_{i_2}^{(\alpha_2)}$, it holds that $u_1 \times u_2 \in S_{i_1+i_2}^{(\alpha_1 \cdot \alpha_2)}$. Here $\alpha_1 \cdot \alpha_2$ is multiplication in R, and $i_1 + i_2$ is integer addition.*

Clearly, Definition 1 implies that if we have a collection of n encodings $u_j \in S_{i_j}^{(\alpha_j)}$, $j = 1, 2 \ldots, n$, then as long as $\sum_j i_j \le \kappa$ we get $u_1 \times \cdots \times u_n \in S_{i_1+\cdots+i_n}^{(\prod_j \alpha_j)}$.

Efficient Procedures, the Dream Version To be useful, we need efficient procedures for manipulating encodings well as as hard computational tasks. To ease the exposition, we first describe a "dream version" of the efficient procedures (which we do not know how to realize), and then explain how to modify them to deal with technicalities that arise from our use of lattices in the realization.

Instance Generation. The randomized $\mathsf{InstGen}(1^\lambda, 1^\kappa)$ takes as inputs the parameters λ, κ, and outputs $(\mathsf{params}, \mathbf{p}_{zt})$, where params is a description of a κ-Graded Encoding System as above, and $\mathbf{p}_{zt}$ is a zero-test parameter for level κ (see below).

Ring Sampler. The randomized $\mathsf{samp}(\mathsf{params})$ outputs a "level-zero encoding" $a \in S_0^{(\alpha)}$ for a nearly uniform element $\alpha \in_R R$. (Note that we require that the "plaintext" $\alpha \in R$ is nearly uniform, but not that the encoding a is uniform in $S_0^{(\alpha)}$.)

Encoding. The (possibly randomized) $\mathsf{enc}(\mathsf{params}, i, a)$ takes a "level-zero" encoding $a \in S_0^{(\alpha)}$ for some $\alpha \in R$ and index $i \leq \kappa$, and outputs the "level-i" encoding $u \in S_i^{(\alpha)}$ for the same α.

Addition and negation. Given params and two encodings relative to the same index, $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, we have $\mathsf{add}(\mathsf{params}, i, u_1, u_2) = u_1 + u_2 \in S_i^{(\alpha_1+\alpha_2)}$, and $\mathsf{neg}(\mathsf{params}, i, u_1) = -u_1 \in S_i^{(-\alpha_1)}$.

Multiplication. For $u_1 \in S_{i_1}^{(\alpha_1)}$, $u_2 \in S_{i_2}^{(\alpha_2)}$ such that $i_1 + i_2 \leq \kappa$, we have $\mathsf{mul}(\mathsf{params}, i_1, u_1, i_2, u_2) = u_1 \times u_2 \in S_{i_1+i_2}^{(\alpha_1 \cdot \alpha_2)}$.

Zero-test. The procedure $\mathsf{isZero}(\mathsf{params}, u)$ output 1 if $u \in S_\kappa^{(0)}$ and 0 otherwise. Note that in conjunction with the subtraction procedure, this lets us test if $u_1, u_2 \in S_\kappa$ encode the same element $\alpha \in R$.

Extraction. This procedure extracts a "canonical" and "random" representation of ring elements from their level-κ encoding. Namely $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u)$ outputs (say) $s \in \{0,1\}^\lambda$, such that:

(a) For any $\alpha \in R$ and two $u_1, u_2 \in S_\kappa^{(\alpha)}$, $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_1) = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_2)$,

(b) The distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u) \ : \ \alpha \in_R R, u \in S_\kappa^{(\alpha)}\}$ is nearly uniform over $\{0,1\}^\lambda$.

Efficient Procedures, the Real-Life Version Our realization of the procedures above over ideal lattices uses noisy encodings, where the noise increases with every operation and correctness is only ensured as long as it does not increase too much. We therefore modify the procedures above, letting them take as input (and produce as output) also a bound on the noise magnitude of the encoding in question. The procedures are allowed to abort if the bound is too high (relative to some maximum value which is part of the instance description params). Also, they provide no correctness guarantees if the bound on their input is "invalid." (When B is a noise-bound for some encoding u, we say that it is "valid" if it is at least as large as the bound produced by the procedure that produced u itself, and moreover any encoding that was used by that procedure (if any) also came with a valid noise bound.) Of course we also require that these procedure do not always abort, i.e. they should support whatever set of operations that the application calls for, before the noise becomes too large. Finally, we also relax the requirements on the zero-test and the extraction routines. Some more details are described next:

Zero-test. We sometime allow false positives for this procedure, but not false negatives. Namely, $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1$ for every $u \in S_\kappa^{(0)}$, but we may have $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1$ also for some $u \notin S_\kappa^{(0)}$. The weakest functionality requirement that we make is that for a uniform random choice of $\alpha \in_R R$, we have

$$\Pr_{\alpha \in_R R}\left[\exists\, u \in S_\kappa^{(\alpha)} \text{ s.t } \mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1\right] = \text{negligible}(\lambda). \quad (1)$$

Additional requirements are considered security features (that a scheme may or may not possess), and are discussed later in this section.

Extraction. Our construction from Section 4 does not support full canonicalization. Instead, we settle for $\mathsf{ext}(\Lambda, \mathbf{p}_{zt}, u)$ that has a good chance of producing the same output when applied to different encoding of the same elements. Specifically, we replace properties (a)-(b) from above by the weaker requirements:

(a′) For a randomly chosen $a \leftarrow \mathsf{samp}(\mathsf{params})$, if we run the encoding algorithm twice to encode a at level κ and then extract from both copies then we get:

$$\Pr\left[\begin{array}{l}\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_1) \\ = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_2)\end{array} \;:\; \begin{array}{l} a \leftarrow \mathsf{samp}(\mathsf{params}) \\ u_1 \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, a) \\ u_2 \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, a)\end{array}\right] \;\geq\; 1 - \text{negligible}(\lambda).$$

(b′) The distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u) : a \leftarrow \mathsf{samp}(\mathsf{params}), u \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, a)\}$ is nearly uniform over $\{0,1\}^\lambda$.

We typically need these two conditions to hold even if the noise bound that the encoding routine takes as input is larger than the one output by samp (upto some maximum value).

Hardness Assumptions Our hardness assumptions are modeled after the discrete-logarithm and DDH assumptions in multilinear groups. For example, the most direct analog of the discrete-logarithm problem is trying to obtain a level-zero encoding $a \in S_0^{(\alpha)}$ for $\alpha \in R$ from an encoding relative to some other index $i > 0$.

The analog of DDH in our case roughly says that it is hard to recognize encoding of products, except relative to indexes upto κ. In other words, given $\kappa + 1$ level-one encoding of random elements it should be infeasible to generate a level-κ encoding of their product, or even to distinguish it from random. One way to formalize it is by the following process. (Below we suppress the noise bounds for readability):

1. $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$
2. For $i = 1, \ldots, \kappa + 1$:
3. Choose $a_i \leftarrow \mathsf{samp}(\mathsf{params})$ // level-0 encoding of random $\alpha_i \in_R R$
4. Set $u_i \leftarrow \mathsf{enc}(\mathsf{params}, 1, a_i)$ // level-1 encoding of the α_i's
5. Set $\tilde{a} = \prod_{i=1}^{\kappa+1} a_i$ // level-0 encoding of the product
6. Choose $\hat{a} \leftarrow \mathsf{samp}(\mathsf{params})$ // level-0 encoding of a random element
7. Set $\tilde{u} \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, \tilde{a})$ // level-κ encoding of the product
8. Set $\hat{u} \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, \hat{a})$ // level-κ encoding of random

(We note that with the noise bound, it may be important that the encoding routines for both $\tilde{a}$ and $\hat{a}$ get as input the same bound, i.e., the largest of the bounds for $\tilde{a}$ and $\hat{a}$.) The GDDH distinguisher gets all the level-one u_i's and either $\tilde{u}$ (encoding the right product) or $\hat{u}$ (encoding a random element), and it needs to decide which is the case. In other words, the GDDH assumption says that for any setting of the parameters, the following two distributions, defined

over the experiment above, are computationally indistinguishable:

$$\mathcal{D}_{GDDH} = \{(\mathsf{params}, \mathbf{p}_{zt}, \{u_i\}_i, \tilde{u})\} \text{ and } \mathcal{D}_{RAND} = \{(\mathsf{params}, \mathbf{p}_{zt}, \{u_i\}_i, \hat{u})\}.$$

3 Preliminaries

Lattices. A lattice $L \subset \mathbb{R}^n$ is an additive discrete sub-group of $\mathbb{R}^n$. Every (nontrivial) lattice has bases: a basis for a full-rank lattice is a set of n linearly independent points $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in L$ such that $L = \{\sum_{i=1}^n z_i \boldsymbol{b}_i : z_i \in \mathbb{Z}\ \forall i\}$. If we arrange the vectors $\boldsymbol{b}_i$ as the columns of a matrix $B \in \mathbb{R}^{n \times n}$ then we can write $L = \{B\boldsymbol{z} : \boldsymbol{z} \in \mathbb{Z}^n\}$. If B is a basis for L then we say that B spans L. For a lattice $L \subset \mathbb{R}^n$, its *dual lattice* consists of all the points in $\mathrm{span}(L)$ that are orthogonal to L modulo one, namely $L^* = \{\boldsymbol{y} \in \mathrm{span}(L) : \forall \boldsymbol{x} \in L, \langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}\}$

Gaussians. For a real $\sigma > 0$, define the (spherical) Gaussian function on $\mathbb{R}^n$ with parameter σ as $\rho_\sigma(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x}\|^2/\sigma^2)$ for all $\boldsymbol{x} \in \mathbb{R}^n$. This generalizes to ellipsoid Gaussians, where the different coordinates are jointly Gaussian but not independent, where we replace the parameter $\sigma \in \mathbb{R}$ by the (square root of the) covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$. For a rank-$n$ matrix $S \in \mathbb{R}^{m \times n}$, the ellipsoid Gaussian function on $\mathbb{R}^n$ with parameter S is defined by $\rho_S(\boldsymbol{x}) = \exp\left(-\pi \boldsymbol{x}^T (S^T S)^{-1} \boldsymbol{x}\right)\ \ \forall \boldsymbol{x} \in \mathbb{R}^n$. Obviously this function only depends on $\Sigma = S^T S$ and not on the particular choice of S. It is also clear that the spherical case can be obtained by setting $S = \sigma I_n$, with I_n the n-by-n identity matrix.

The *ellipsoid discrete Gaussian distribution* over lattice L with parameter S is $\forall\ \boldsymbol{x} \in L, D_{L,S}(\boldsymbol{x}) = \rho_S(\boldsymbol{x})/\rho_S(L)$, where $\rho_S(L)$ denotes $\sum_{\boldsymbol{x} \in L} \rho_S(\boldsymbol{x})$. In other words, the probability $D_{L,S}(\boldsymbol{x})$ is simply proportional to $\rho_S(\boldsymbol{x})$, the denominator being a normalization factor. The same definitions apply to the spherical case, $D_{L,\sigma}(\cdot)$.

Smoothing parameter. Micciancio and Regev defined in [MR07] the *smoothing parameter* for a lattice L and real $\epsilon > 0$, denoted $\eta_\epsilon(L)$, as the smallest s such that $\rho_{1/s}(L^* \setminus \{\mathbf{0}\}) \leq \epsilon$. Intuitively, for a small enough ϵ, the number $\eta_\epsilon(L)$ is sufficiently larger than L's fundamental parallelepiped so that sampling from the corresponding Gaussian "wipes out the internal structure" of L. It is easy to show that the size of vectors drawn from $D_{L,S}$ is roughly bounded by the largest singular value of S. (Recall that the largest and least singular values of a full rank matrix $X \in \mathbb{R}^{m \times n}$ are defined as $\sigma_1(X) = \sup(U_X)$ and $\sigma_n(X) = \inf(U_X)$, respectively, where $U_X = \{\|X\boldsymbol{u}\| : \boldsymbol{u} \in \mathbb{R}^n, \|\boldsymbol{u}\| = 1\}$.)

Lemma 1. *For a rank-n lattice L, constant $0 < \epsilon < 1$ and matrix S s.t. $\sigma_n(S) \geq \eta_\epsilon(L)$, we have* $\Pr_{\boldsymbol{v} \leftarrow \mathcal{D}_{L,S}}\left(\|\boldsymbol{v}\| \geq \sigma_1(S)\sqrt{n}\right) \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-n}$.

Sum of Discrete Gaussians. A recent work [AGHS12] considered the process that begins by choosing "once and for all" m points in a lattice L, drawn independently from a "wide discrete Gaussian" $\boldsymbol{x}_i \leftarrow D_{L,S}$. Once the $\boldsymbol{x}_i$'s are

fixed, they are arranged as the rows of an m-by-n matrix $X = (\boldsymbol{x}_1|\boldsymbol{x}_2|\ldots|\boldsymbol{x}_m)^T$, and we consider the distribution $\mathcal{D}_{X,\sigma}$, induced by choosing an integer vector $\boldsymbol{v}$ from a discrete spherical Gaussian over $\mathbb{Z}^m$ with parameter σ and outputting $\boldsymbol{y} = X^T\boldsymbol{v}$, $\mathcal{E}_{X,\sigma} \stackrel{\text{def}}{=} \{X^T\boldsymbol{v} : \boldsymbol{v} \leftarrow D_{\mathbb{Z}^m,\sigma}\}$. [AGHS12] proved that with high probability over the choice of X, the distribution $\mathcal{D}_{X,\sigma}$ is statistically close to the ellipsoid Gaussian $D_{L,\sigma X}$, and moreover the singular values of X are of size roughly $\sigma\sqrt{m}$:

Theorem 1 ([AGHS12]). *Let L be a full-rank lattice $L \subset \mathbb{R}^n$ and B a matrix whose rows form a basis of L, and denote $\chi = \sigma_1(B)/\sigma_n(B)$. Also let ϵ be negligible in n, and let m, s, s' be parameters such that $s \geq \eta_\epsilon(\mathbb{Z}^n)$, $m \geq 10n\log(8(mn)^{1.5}s\chi)$ and $s' \geq 4mn\chi\ln(1/\epsilon)$.*

Then, when choosing the rows of an m-by-n matrix X from the spherical Gaussian over L, $X \leftarrow (\mathcal{D}_{L,s})^m$, we have with all but probability $2^{-O(m)}$ over the choice of X, that the statistical distance between $\mathcal{E}_{X,s'}$ and the ellipsoid Gaussian $\mathcal{D}_{L,s'X}$ is bounded by 2ϵ.

Lemma 2 ([AGHS12]). *There exists a universal constant $K > 1$ such that for all $m \geq 2n$, $\epsilon > 0$ and every n-dimensional real lattice $L \subset \mathbb{R}^n$, the following holds: Choosing the rows of an m-by-n matrix X independently at random from a spherical discrete Gaussian on L with parameter $\rho > 2K\eta_\epsilon(L)$, $X \leftarrow (D_{L,\rho})^m$, we have*

$$\Pr\left[s\sqrt{2\pi m}/K < \sigma_n(X) \leq \sigma_1(X) < \rho K\sqrt{2\pi m}\right] > 1 - (4m\epsilon + O(\exp(-m/K))).$$

Ideal lattices. For n a power of two, we consider the $2n$'th cyclotomic polynomial ring $R = \mathbb{Z}[X]/(X^n+1)$, and identify an element $\boldsymbol{u} \in R$ with the coefficient vector[3] of the degree-$(n-1)$ integer polynomial that represents $\boldsymbol{u}$. In this way, R is identified with the integer lattice $\mathbb{Z}^n$. Additionally we sometimes consider also the ring $R_q = R/qR = \mathbb{Z}_q[X]/(X^n+1)$ for a (large enough) integer q. Obviously, addition in these rings is done component-wise in their coefficients, and multiplication is polynomial multiplication modulo the ring polynomial $X^n + 1$. In some cases we also consider the corresponding number field $\mathbb{K} = \mathbb{Q}[X]/(X^n+1)$, which is likewise associated with the linear space $\mathbb{Q}^n$.

For an element $\mathbf{g} \in R$, let $\langle\mathbf{g}\rangle$ be the principal ideal in R generated by $\mathbf{g}$ (alternatively, the sub-lattice of $\mathbb{Z}^n$ corresponding to this ideal), namely $\langle\mathbf{g}\rangle = \{\mathbf{g}\cdot\boldsymbol{u} : \boldsymbol{u} \in R\}$. We call $\langle\mathbf{g}\rangle$ an *ideal lattice* to stress its dual interpretation as both an ideal and a lattice. Let $B(\mathbf{g})$ denote the basis of the lattice $\langle\mathbf{g}\rangle$ consisting of the vectors $\{\mathbf{g}, X\mathbf{g}, X^2\mathbf{g}, \ldots, X^{n-1}\mathbf{g}\}$.

For an arbitrary element $\boldsymbol{u} \in R$, denote by $[\boldsymbol{u}]_\mathbf{g}$ the reduction of $\boldsymbol{u}$ modulo the fundamental cell of $B(\mathbf{g})$, which is *symmetric around the origin*. To wit, $[\boldsymbol{u}]_\mathbf{g}$ is the unique element $\boldsymbol{u}' \in R$ such that $\boldsymbol{u} - \boldsymbol{u}' \in \langle\mathbf{g}\rangle$ and $\boldsymbol{u}' = \sum_{i=0}^{n-1}\alpha_i X^i\mathbf{g}$ where all the α_i's are in the interval $[-\frac{1}{2}, \frac{1}{2})$. We use the similar notation $[t]_p$ for integers t, p to denote the reduction of t modulo p into the interval $[-p/2, p/2)$.

[3] Other representations of polynomials are also possible, for example representing a polynomial by its canonical embedding is sometimes preferable to the coefficient representation. Here we stick to coefficient representation for simplicity.

4 The New Encoding Schemes

An instance of our basic construction is parametrized by the security parameter λ and the required multi-linearity level $\kappa \leq \text{poly}(\lambda)$. Based on these parameters, we choose a cyclotomic ring $R = \mathbb{Z}[X]/(X^n + 1)$ (where n is large enough to ensure security), a modulus q that defines $R_q = R/qR$ (with q large enough to support functionality), and another parameter m (chosen so that we can apply Theorem 1). The specific constraints that these parameters must satisfy are discussed at the end of this section, an approximate setting to keep in mind is $n = \tilde{O}(\kappa\lambda^2)$, $q = 2^{n/\lambda}$ and $m = O(n^2)$.

4.1 The Basic Graded Encoding Scheme

An instance of our scheme relative to the parameters above encodes elements of a quotient ring $QR = R/\mathcal{I}$, where $\mathcal{I}$ is a principal ideal $\mathcal{I} = \langle \mathbf{g} \rangle \subset R$, generated by a short vector $\mathbf{g}$. Namely, the "ring elements" that are encoded in our scheme are cosets of the form $\boldsymbol{e} + \mathcal{I}$ for some vector $\boldsymbol{e}$. The short generator $\mathbf{g}$ itself is kept secret, and no "good" description of $\mathcal{I}$ is made public in our scheme. In addition, our system depends on another secret element $\mathbf{z}$, which is chosen at random in R_q (and hence is not short).

A level-zero ("plaintext") encoding of a coset $\boldsymbol{e} + \mathcal{I} \in R/\mathcal{I}$ is just a short vector in that coset (which must exist, since the generator $\mathbf{g}$ is short and therefore the basic cell of $\mathcal{I}$ is quite small). For higher-level encodings, a level-i encoding of the same coset is a vector of the form $\boldsymbol{c}/\mathbf{z}^i \in R_q$ with $\boldsymbol{c} \in \boldsymbol{e} + \mathcal{I}$ short. Specifically, for $i \in \{0, 1, \ldots, \kappa\}$ the set of all level-i encodings is $S_i = \{\boldsymbol{c}/\mathbf{z}^i \in R_q : \|\boldsymbol{c}\| < q^{1/8}\}$, and the set of levle-$i$ encodings of the "plaintext element" $\boldsymbol{e}+\mathcal{I}$ is $S_i^{(\boldsymbol{e}+\mathcal{I})} = \{\boldsymbol{c}/\mathbf{z}^i \in R_q : \boldsymbol{c} \in \boldsymbol{e} + \mathcal{I},\ \|\boldsymbol{c}\| < q^{1/8}\ \}$. Throughout the construction we use the size of the numerator as the "noise level" in the encoding. Namely, with each level-i encoding $\boldsymbol{c}/\mathbf{z}^i$ we produce also an upper bound on $\|\boldsymbol{c}\|$.

Instance generation: $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$**.** Our instance-generation procedure chooses at random the ideal-generator $\mathbf{g}$ and denominator $\mathbf{z}$, as well as several other vectors that are used in the other procedures and are described later in the section. The denominator $\mathbf{z}$ is chosen uniformly at random in R_q. For technical reasons, the generator $\mathbf{g} \in R$ should be chosen so that both $\mathbf{g}$ and $\mathbf{g}^{-1} \in \mathbb{K}$ are short. (Recall that we denote $\mathbb{K} = \mathbb{Q}[X]/(X^n + 1)$. The reason that we need $\mathbf{g}^{-1} \in \mathbb{K}$ to be short is explained when we describe the zero-testing procedure.) We simply draw $\mathbf{g}$ from a discrete Gaussian over $\mathbb{Z}^n$, say $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$ with $\sigma = \tilde{O}(\sqrt{n})$. Clearly $\mathbf{g}$ itself is short (of size less than $\sigma\sqrt{n}$), and we claim that with good probability its inverse in the field of fractions is also rather short. To see this, notice that with probability $1 - o(1/n)$, evaluating $\mathbf{g}$ at any complex n'th root of unity $\zeta \in \mathbb{C}$ yields $\mathbf{g}(\zeta)$ which is not too tiny, say larger than $1/n$. Hence with probability $1 - o(1)$ we have $\mathbf{g}^{-1}(\zeta) = 1/\mathbf{g}(\zeta) < n$ for all the primitive $2n$'th roots of unity ζ, which means that $\mathbf{g}^{-1}$ itself is not too large, say $\|1/\mathbf{g}\| < n^2$. We can draw repeatedly until we get this condition to hold.

Once we have $\mathbf{g}, \mathbf{z}$, we choose and publish some other elements in R_q that will be used for the various procedures below. Specifically we have $m+1$ elements $rand_1, \ldots, \mathbf{x}_m, \mathbf{y}$ that are used for encoding, and an element $\mathbf{p}_{zt}$ that is used as a zero-testing parameter. These elements are described later. finally we also choose a random seed s for a strong randomness extractor. The instance-generation procedure outputs $\mathsf{params} = (n, q, \mathbf{y}, \{\mathbf{x}_i\}_i, s)$ and $\mathbf{p}_{zt}$.

Sampling level-zero encodings: $\boldsymbol{d} \leftarrow \mathsf{samp}(\mathsf{params})$. To sample a level-zero encoding of a random coset, we just draw a random short element in R, $\boldsymbol{d} \leftarrow D_{\mathbb{Z}^n,\sigma'}$, where $\sigma' = \sigma n$ (for σ that was used to sample $\mathbf{g}$). Since whp $\sigma' \geq \eta_{2^{-\lambda}}(\mathcal{I})$, then the induced distribution over the cosets of $\mathcal{I}$ is close to uniform, upto a negligible distance. Also the size of this level-zero encoding is bounded by $\sigma'\sqrt{n}$ (and we use this as our noise-bound for this encoding).

Encodings at higher levels: $\boldsymbol{u}_i \leftarrow \mathsf{enc}(\mathsf{params}, i, \boldsymbol{d})$. To allow encoding of cosets at higher levels, we publish as part of our instance-generation a level-one encoding of $1+\mathcal{I}$, namely an element $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$ where $\mathbf{a} \in 1+\mathcal{I}$ is short. A simplistic method of doing that is drawing $\mathbf{a} \leftarrow D_{1+\mathcal{I},\sigma'}$, then computing $\mathbf{y}$ from $\mathbf{a}$. (Later we describe a somewhat more involved procedure, which we believe is more secure.) Given a level-zero encoding $\boldsymbol{d}$ as above, we can multiply it by $\mathbf{y}$ over R_q to get $\boldsymbol{u}_1 := [\mathbf{y}\boldsymbol{d}]_q$. Note that $\boldsymbol{u}_1 = [\boldsymbol{d}\mathbf{a}/\mathbf{z}]_q$, where $\boldsymbol{d}\mathbf{a} \in \boldsymbol{d}+\mathcal{I}$ as needed, and the size of the numerator is bounded by $\|\boldsymbol{d}\| \cdot \|\mathbf{a}\| \cdot \sqrt{n} = \mathrm{poly}(n)$. More generally we can generate a level-i encoding as $\boldsymbol{u}_i := [\boldsymbol{d}\mathbf{y}^i]_q = [\boldsymbol{d}\mathbf{a}^i/\mathbf{z}^i]_q$. The numerator $\boldsymbol{d}\mathbf{a}^i$ is obviously in $\boldsymbol{d}+\mathcal{I}$, and its size is at most $\|\boldsymbol{d}\| \cdot \|\mathbf{a}\|^i \cdot n^{i/2}$.

The above encoding is insufficient, however, since from $\boldsymbol{u}_1$ and $\mathbf{y}$ it is easy to get back $\boldsymbol{d}$ by simple division in R_q. We therefore include in the public parameters also the "randomizers" $\mathbf{x}_i$, these are just random encodings of zero, namely $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$ where the $\mathbf{b}_i$'s are short elements in $\mathcal{I}$. A simplistic procedure for choosing these randomizers would be to draw short these elements as $\mathbf{b}_i \leftarrow D_{\mathcal{I},\sigma'}$ and publish $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$. As we note in the full version [GGH12], we have reasons to suspect that this simplistic method is insecure so instead we use a somewhat more involved sampling procedure, see details in the full version [GGH12]. Below we denote by $\mathbf{X}$ the matrix with the vectors $\mathbf{x}_i$ as rows, namely $\mathbf{X} = (\mathbf{x}_1 | \ldots | \mathbf{x}_m)^T$. We also use $\mathbf{B}$ to denote the matrix with the numerators $\mathbf{b}_i$ as rows, i.e., $\mathbf{B} = (\mathbf{b}_1 | \ldots | \mathbf{b}_m)^T$.

We use the $\mathbf{x}_i$'s to randomize level-one encodings: Given $\boldsymbol{u}' = [\boldsymbol{c}'/\mathbf{z}]_q$ with noise-bound $\|\boldsymbol{c}'\| < \gamma$, we draw an m-vector of integer coefficients $\boldsymbol{r} \leftarrow D_{\mathbb{Z}^m,\sigma^*}$ for large enough σ^* (e.g. $\sigma^* = 2^\lambda\gamma$), and output

$$\boldsymbol{u} \;:=\; [\boldsymbol{u}' + \mathbf{X}\boldsymbol{r}]_q \;=\; [\boldsymbol{u}' + \sum_{i=1}^{m} r_i\mathbf{x}_i]_q \;(=\; [\frac{\boldsymbol{c}' + \sum_i r_i\mathbf{b}_i}{\mathbf{z}}]_q).$$

We write $\mathbf{B}\boldsymbol{r}$ as a shorthand for $\sum_i r_i\mathbf{b}_i$ and similarly $\mathbf{X}\boldsymbol{r}$ as a shorthand for $\sum_i r_i\mathbf{x}_i$.

Since all the $\mathbf{b}_i$'s are in the ideal $\mathcal{I}$, then obviously $\boldsymbol{c}' + \sum_i r_i\mathbf{b}_i$ is in the same coset of $\mathcal{I}$ as $\boldsymbol{c}'$ itself. Moreover since $\|\mathbf{b}_i\| < \mathrm{poly}(n)$ then $\|\mathbf{B}\boldsymbol{r}\| < \sigma^*\mathrm{poly}(m,n)$. If indeed $\|\boldsymbol{c}'\| < \gamma$, then $\|\boldsymbol{c}' + \mathbf{B}\boldsymbol{r}\| < \gamma + \sigma^*\mathrm{poly}(m,n)$. We also claim that

the distribution of $\boldsymbol{u}$ is nearly independent of original $\boldsymbol{u'}$ (except of course its coset). To see why, note that if the $\mathbf{b}_i$'s are chosen from a wide enough spherical distribution then we can use Theorem 1 to conclude that $\mathbf{B}\boldsymbol{r}$ is close to a wide ellipsoid Gaussian. With our choice of σ^* the "width" of that distribution is much larger than the original $\boldsymbol{c'}$, hence the distribution of $\boldsymbol{c'} + \mathbf{B}\boldsymbol{r}$ is nearly independent of $\boldsymbol{c'}$, except in the coset that it belongs to.

A different approach is to re-randomize $\mathbf{y}$, setting $\mathbf{y}' := \mathbf{y} + \mathbf{X}\boldsymbol{r}$ and then encode via $\boldsymbol{u}_1 := [\mathbf{y}'\boldsymbol{d}]_q$. This does not have the information-theoretic same-distribution guarantee as above (since the distributions $[\mathbf{y}'\boldsymbol{d}]_q$ and $[\mathbf{y}'\boldsymbol{d'}]_q$ may differ, even if $\boldsymbol{d}, \boldsymbol{d'}$ are both short and in the same coset). But on the plus side, it is more convenient to use this re-randomization method for encoding at high levels $i > 1$: After computing the randomized $\mathbf{y}'$, we can use it by setting $\boldsymbol{u}_i := [\boldsymbol{d}(\mathbf{y}')^i]_q$.

Remark 1. Note that in the above description we used the matrix $\mathbf{X}$ to randomize level-one encodings. Using similar pubic parameter $\mathbf{X}_i$ we can generalize the re-randomization procedure to work at any level $i \leq \kappa$. In particular we abstract this procedure as $\mathsf{reRand}(\mathbf{y}, i, \boldsymbol{u}')$: Given $\boldsymbol{u}' = [\boldsymbol{c}'/\mathbf{z}^i]_q$ with noise-bound $\|\boldsymbol{c}'\| < \gamma$, we draw an m-vector of integer coefficients $\boldsymbol{r} \leftarrow D_{\mathbb{Z}^m,\sigma^*}$ for large enough σ^* (e.g. $\sigma^* = 2^\lambda\gamma$), and output $\boldsymbol{u} := [\boldsymbol{u}' + \mathbf{X}_i\boldsymbol{r}]_q$ as a re-randomized version of $\boldsymbol{u}$. Using the same argument as above we can conclude that the distribution generated in this way will be independent of $\boldsymbol{c}'$, except in the coset that it belongs to.

Note that for some applications it might be useful to use the re-randomization operation multiple times. We consider the case in which a constant number of re-randomizations is needed. In this case, with the ℓ^{th} re-randomization (for any constant ℓ) we can generate an encoding by choosing $\boldsymbol{r}$ from $D_{\mathbb{Z}^m,\sigma^*}$ where $\sigma^* = 2^{\lambda^\ell}$ and re-randomizing as above. Since the addition and multiplication of constant number of terms increases noise by a small factor we can claim that each re-randomization wipes the structure that was present previously (even with multiple additions and multiplications).

We define a canonicalizing encoding algorithm $\mathsf{cenc}_\ell(\mathsf{params}, i, \boldsymbol{u}')$ which takes as input an encoding of $\boldsymbol{u}'$ and generates another encoding according with a noise factor of 2^{λ^ℓ}.

Adding and multiplying encodings. It is easy to see that the encoding as above is additively homomorphic, in the sense that adding encodings yields an encoding of the sum. This follows since if we have many short $\boldsymbol{c}_j$'s then their sum is still short, $\|\sum_j \boldsymbol{c}_j\| \ll q$, and therefore the sum $\boldsymbol{c} = \sum_j \boldsymbol{c}_j = [\sum_j \boldsymbol{c}_j]_q \in R_q$ belong to the coset $\sum_j(\boldsymbol{c}_j + \mathcal{I})$. Hence, if we denote $\boldsymbol{u}_j = \boldsymbol{c}_j/\mathbf{z} \in R_q$ then each $\boldsymbol{u}_j$ is an encoding of the coset $\boldsymbol{c}_j + \mathcal{I}$, and the sum $[\sum_j \boldsymbol{u}_j]_q$ is of the form $\boldsymbol{c}/\mathbf{z}$ where $\boldsymbol{c}$ is still a short element in the sum of the cosets.

Moreover, since $\mathcal{I}$ is an ideal then multiplying upto κ encodings can be interpreted as an encoding of the product, by raising the denominator to the

appropriate power. Namely, for $\boldsymbol{u}_j = \boldsymbol{c}_j/\mathbf{z} \in R_q$ as above, we have

$$\boldsymbol{u} = \prod_{j=1}^{\kappa} \boldsymbol{u}_j = \frac{\prod_j \boldsymbol{c}_j}{\mathbf{z}^\kappa} \quad \text{(all the operations in } R_q\text{)}.$$

As long as the $\boldsymbol{c}_j$'s are small enough to begin with, we still have $\|\prod_j \boldsymbol{c}_j\| \ll q$, which means that $[\prod_j \boldsymbol{c}_j]_q = \prod_j \boldsymbol{c}_j$ (operations in R), hence $[\prod_j \boldsymbol{c}_j]_q$ belongs to the product coset $\prod_j (\boldsymbol{c}_j + \mathcal{I})$.

Thus, if each $\boldsymbol{u}_j$ is a level-1 encoding of the coset $\boldsymbol{c}_j + \mathcal{I}$ with short-enough numerator, then their product is a level-κ encoding of the product coset. We note that just like level-1 encoding, level-κ encoding still offers additive homomorphism.

Zero testing: $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{u}_\kappa) \stackrel{?}{=} 0/1$. Since the encoding is additively homomorphic, we can test equality between encodings by subtracting them and comparing to zero. To enable zero-testing, we generate the zero-testing parameter as follows: We draw a "somewhat small" ring element $\boldsymbol{h} \leftarrow D_{\mathbb{Z}^n,\sqrt{q}}$, and the zero-testing parameter is set as $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$. To test if a level-$\kappa$ encoding $\boldsymbol{u} = [\boldsymbol{c}/\mathbf{z}^\kappa]_q$ is an encoding of zero, we just multiply it in R_q by $\mathbf{p}_{zt}$ and check whether the resulting element $\boldsymbol{w} = [\mathbf{p}_{zt} \cdot \boldsymbol{u}]_q$ is short (e.g., shorter than $q^{3/4}$). Namely, we use the test

$$\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{u}) \;=\; \begin{cases} 1 \text{ if } \|[\mathbf{p}_{zt}\boldsymbol{u}]_q\|_\infty < q^{3/4} \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

To see why this works, note that

$$\boldsymbol{w} \;=\; \mathbf{p}_{zt} \cdot \boldsymbol{u} \;=\; \frac{\boldsymbol{h}\mathbf{z}^\kappa}{\mathbf{g}} \cdot \frac{\boldsymbol{c}}{\mathbf{z}^\kappa} \;=\; \boldsymbol{h} \cdot \boldsymbol{c}/\mathbf{g} \quad \text{(all the operations in } R_q\text{)}.$$

If $\boldsymbol{u}$ is an encoding of zero then $\boldsymbol{c}$ is a short vector in $\mathcal{I}$, which means that it is divisible by $\mathbf{g}$ in R. Hence the element $\boldsymbol{c}/\mathbf{g} \in R_q$ is the same as the element $\boldsymbol{c} \cdot \mathbf{g}^{-1} \in \mathbb{K}$, which means that it has size at most $\|\boldsymbol{c}\| \cdot \|\mathbf{g}^{-1}\| \cdot \sqrt{n} = \|\boldsymbol{c}\| \cdot \mathrm{poly}(n)$. This, in turn, implies that $\|\boldsymbol{w}\| \le \|\boldsymbol{h}\| \cdot \|\boldsymbol{c}\| \cdot \mathrm{poly}(n)$, which for our choice of parameter is $q^{1/2} \cdot q^{1/8} \cdot \mathrm{poly}(n) < q^{3/4}$.

If $\boldsymbol{u}$ is an encoding of a different coset, then $\boldsymbol{c}$ is a short vector in some coset of $\mathcal{I}$. In this case we have $\boldsymbol{w} = [\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$, where $\boldsymbol{c}, \mathbf{g}$ are small (and $\boldsymbol{h}$ is "somewhat small"). Intuitively, since $\boldsymbol{h}/\mathbf{g}$ is large whp then for a "random enough" $\boldsymbol{c}$ we expect the size of $\boldsymbol{w}$ to be large. More formally, we argue below that when choosing a uniformly random coset of $\mathcal{I} = \langle \mathbf{g} \rangle$, there are *no short elements* $\boldsymbol{c}$ in that coset such that $[\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$ is small.

Lemma 3. *Let $\boldsymbol{w} = [\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$ and suppose $\|\mathbf{g} \cdot \boldsymbol{w}\|$ and $\|\boldsymbol{c} \cdot \boldsymbol{h}\|$ are each at most $q/2$. Suppose $\langle \mathbf{g} \rangle$ is a prime ideal. Then, either $\mathbf{c}$ or $\boldsymbol{h}$ is in the ideal $\langle \mathbf{g} \rangle$.*

Proof. Since $\mathbf{g} \cdot \boldsymbol{w} = \boldsymbol{c} \cdot \boldsymbol{h} \bmod q$, and since $\|\mathbf{g} \cdot \boldsymbol{w}\|$ and $\|\boldsymbol{c} \cdot \boldsymbol{h}\|$ are each at most $q/2$, we have $\mathbf{g} \cdot \boldsymbol{w} = \boldsymbol{c} \cdot \boldsymbol{h}$ exactly. We also have an equality of ideals $\langle \mathbf{g} \rangle \cdot \langle \boldsymbol{w} \rangle = \langle \boldsymbol{c} \rangle \cdot \langle \boldsymbol{h} \rangle$, and, since $\langle \mathbf{g} \rangle$ is a prime ideal and our cyclotomic ring is a unique factorization domain, we have that $\langle \mathbf{g} \rangle$ divides either $\langle \boldsymbol{c} \rangle$ or $\langle \boldsymbol{h} \rangle$ (or both). The result follows.

Lemma 4. *Let n, q, σ be as in our parameter setting, suppose $q = n^{\omega(1)}$, and consider drawing $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma'}$ subject to $\langle \mathbf{g} \rangle$ being prime and $\boldsymbol{h} \leftarrow D_{\mathbb{Z}^n,\sqrt{q}}$ not being in $\langle \mathbf{g} \rangle$. Then, there is no $\epsilon > 0$ and $\boldsymbol{c}$ in a nonzero coset of $\mathcal{I}$ such that $\|\boldsymbol{c}\| < q^{1/8}$ and $\|[\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q\| < q^{1-\epsilon}$.*

Proof. This follows directly from Lemma 3, our parameter setting (with $\|\mathbf{g}\| = \mathsf{poly}(n)$) and the fact that in the coefficient embedding $\|\mathbf{a} \cdot \mathbf{b}\| \leq n \cdot \|\mathbf{a}\| \cdot \|\mathbf{b}\|$.

Extraction: $s \leftarrow \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_\kappa)$. To extract a "canonical" and "random" representation of a coset from an encoding $\boldsymbol{u} = [\boldsymbol{c}/\mathbf{z}^\kappa]_q$, we just multiply by the zero-testing parameter $\mathbf{p}_{zt}$, collect the $(\log q)/4 - \lambda$ most-significant bits of each of the n coefficients of the result, and apply a strong randomness extractor to the collected bits (using the seed from the public parameters). Namely

$$\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{u}) = \textsc{Extract}_s(\mathsf{msbs}([\boldsymbol{u} \cdot \mathbf{p}_{zt}]_q)) \text{ (msbs of coefficient representation).}$$

This works because for any two encodings $\boldsymbol{u}, \boldsymbol{u'}$ of the same coset we have

$$\|\mathbf{p}_{zt}\boldsymbol{u} - \mathbf{p}_{zt}\boldsymbol{u'}\| = \|\mathbf{p}_{zt}(\boldsymbol{u} - \boldsymbol{u'})\| < q^{3/4},$$

so we expect $\mathbf{p}_{zt}\boldsymbol{u}$, $\mathbf{p}_{zt}\boldsymbol{u'}$ to agree on their $(\log q)/4 - \lambda$ most significant bits. (There is a negligible (in λ) chance that $\boldsymbol{u}$ and $\boldsymbol{u'}$ are such that $\mathbf{p}_{zt}\boldsymbol{u}$ and $\mathbf{p}_{zt}\boldsymbol{u'}$ are on opposite sides of a boundary, such that they have different MSBs.) On the other hand, by Lemma 4, we know that we cannot have $\|\mathbf{p}_{zt}(\boldsymbol{u} - \boldsymbol{u'})\| < q^{1-\epsilon}$ when $\boldsymbol{u} - \boldsymbol{u'}$ encodes something nonzero, and therefore (since $\lambda \ll \log q/4$) the values $\mathbf{p}_{zt}\boldsymbol{u}$ and $\mathbf{p}_{zt}\boldsymbol{u'}$ cannot agree on their $(\log q)/4 - \lambda$ MSBs.

This means, however, that no two points in the basic cell of $\mathcal{I}$ agree on their collected bits when multiplied by $\mathbf{p}_{zt}$, so the collected bits from an encoding of a random coset have min-entropy at least $\log |R/\mathcal{I}|$. We can therefore use a strong randomness extractor to extract a nearly uniform bit-string of length (say) $\lfloor \log |R/\mathcal{I}| \rfloor - \lambda$.

4.2 Security of Our Constructions

The security of our graded encoding systems relies on new, perhaps unconventional assumptions, and at present it seems unlikely that they can be reduced to more established assumptions, such as learning-with-errors (LWE) [Reg05], or even the NTRU hardness assumption [HPS98]. Given that the construction of multilinear maps has been a central open problem now for over a decade, we feel that exploring unconventional assumptions for this purpose is well worth the effort, as long as this exploration is informed by extensive cryptanalysis.

We attempted an extensive cryptanalysis of our scheme, including some new extensions of tools from the literature that we devised in the course of this work. These attempts are described at length in the full version [GGH12].

Easiness of other problems. In light of the apparent hardness of our CDH/DDH analog, we could optimistically hope to get also the analog of other hardness assumptions in bilinear maps, such as decision-linear, subgroup membership, etc. Unfortunately, these problems turn out to be easy in our setting, at least with the simple encoding methods from above.

To see why, observe that publishing level-1 encodings of 0 and 1 enables some "weak discrete log" computation at any level strictly smaller than κ. Specifically, consider one particular encoding of zero $\mathbf{x}_j = [\mathbf{b}_j/\mathbf{z}]_q$ (where $\mathbf{b}_j = \boldsymbol{c}_j\mathbf{g}$ for some $\boldsymbol{c}_j$), which is given in the public parameters together with an encoding of one $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$ and the zero-testing parameter $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$. Given a level-$i$ encoding with $1 \le i \lneq \kappa$, $\boldsymbol{u} = [\boldsymbol{d}/\mathbf{z}^i]_q$, we can multiply it by $\mathbf{x}_j$, $\mathbf{p}_{zt}$, and some power of $\mathbf{y}$ to get

$$\begin{aligned}\boldsymbol{f} = [\boldsymbol{u}\cdot\mathbf{x}_j\cdot\mathbf{p}_{zt}\cdot\mathbf{y}^{\kappa-i-1}]_q &= \left[\frac{\boldsymbol{d}}{\mathbf{z}^i}\cdot\frac{\boldsymbol{c}_j\cdot\mathbf{g}}{\mathbf{z}}\cdot\frac{\boldsymbol{h}\mathbf{z}^\kappa}{\mathbf{g}}\cdot\frac{\mathbf{a}^{\kappa-i-1}}{\mathbf{z}^{\kappa-i-1}}\right]_q \\ = \underbrace{\boldsymbol{d}\cdot\boldsymbol{c}_j\cdot\boldsymbol{h}\cdot\mathbf{a}^{\kappa-i-1}}_{\ll q} &= \boldsymbol{d}\cdot\underbrace{\boldsymbol{c}_j\cdot\boldsymbol{h}}_{\Delta_j} \pmod{\mathcal{I}}.\end{aligned}$$

We stress that the right-hand-side of the equality above is *not reduced modulo* q. This means that from a level-i encoding $\boldsymbol{u}$ of an element $\boldsymbol{d}+\mathcal{I}$, we can get a "plaintext version" of $\boldsymbol{d}\cdot\Delta_j$ from some fixed Δ_j (that depends only on the public parameters but not on $\boldsymbol{u}$). This "plaintext version" is not small enough to be a valid level-zero encoding (because Δ_j is roughly the size of $\boldsymbol{h}$, so in particular $\Delta_j > \sqrt{q}$). Nonetheless, we can still use it in attacks.

For starters, we can apply the above procedure to many of the level-one encodings of zero from the public parameters, thereby getting many elements in the ideal $\mathcal{I}$ itself. This by itself still does not yield a basis of $\mathcal{I}$ (since all these elements have the extra factor of h), but in the full version [GGH12] we show how to remove this extra factor and nonetheless compute a basis for $\mathcal{I}$. This is not a small basis of course, but it tells us that we cannot hope to hide the plaintext space $R/\mathcal{I}$ itself.

Next, consider the subgroup membership setting, where we have $\mathbf{g} = \mathbf{g}_1\cdot\mathbf{g}_2$, we are given a level-1 encoding $\boldsymbol{u} = [\boldsymbol{d}/\mathbf{z}]_q$ and need to decide if $\boldsymbol{d} \in \langle\mathbf{g}_1\rangle$. Using the procedure above we can get $\boldsymbol{f} = \boldsymbol{d}\cdot\Delta_j$, which belongs to the ideal $\langle\mathbf{g}_1\rangle$ if $\boldsymbol{d}$ does. Taking the GCD of the ideals $\langle\boldsymbol{f}\rangle$ and $\mathcal{I}$ will then give us the factor $\langle\mathbf{g}_1\rangle$ with high probability. It follows that the subgroup membership problem is easy for the encoding method above.

Finally, consider getting a matrix of elements $A = (\boldsymbol{a}_{i,j})_{i,j}$, all encoded at some level $i \lneq \kappa$. Using the method above we can get a "plaintext version" of $\Delta_j \cdot M$, which has the same rank as A. Since the decision linear problem is essentially a matrix rank problem, this means that this problem too is easy for this encoding method.

At this point it is worth stressing again that these attacks do not seem to apply to the GDDH problem, specifically because in that problem we need to make a decision about a level-κ encoding, and the "weak discrete log" procedure from

above only applies to encoding at levels strictly below κ. The attacks above make it clear that providing encodings of zero in the public parameters (in conjunction with the zero-testing parameter) gives significant power to the adversary. One interesting direction to counter these attacks is to find different randomization tools that can be applied even when we do not have these encodings of zero in the public parameters.

References

[AGHS12] Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Sampling discrete gaussians efficiently and obliviously. Cryptology ePrint Archive, Report 2012/714, 2012. `http://eprint.iacr.org/`.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, 2001.

[BL96] Dan Boneh and Richard J. Lipton. Algorithms for black-box fields and their application to cryptography (extended abstract). In Neal Koblitz, editor, *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 283–297. Springer, 1996.

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324:71–90, 2003.

[CS97] Don Coppersmith and Adi Shamir. Lattice attacks on ntru. In Walter Fumy, editor, *EUROCRYPT*, volume 1233 of *Lecture Notes in Computer Science*, pages 52–61. Springer, 1997.

[Gen01] Craig Gentry. Key recovery and message attacks on ntru-composite. In *Advances in Cryptology - EUROCRYPT'01*, volume 2045 of *Lecture Notes in Computer Science*, pages 182–194. Springer, 2001.

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 112–131. Springer, 1997.

[GGH12] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. Cryptology ePrint Archive, Report 2012/610, 2012. `http://eprint.iacr.org/`.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. Cryptology ePrint Archive, Report 2013/128, 2013. `http://eprint.iacr.org/`.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *STOC*, 2013.

[GKP+13] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Succinct functional encryption and applications: Reusable garbled circuits and beyond. In *STOC*, 2013.

[GS02] Craig Gentry and Michael Szydlo. Cryptanalysis of the revised ntru signature scheme. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 299–320. Springer, 2002.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits. In *STOC*, 2013.

[HGS04] Nick Howgrave-Graham and Michael Szydlo. A method to solve cyclotomic norm equations. In Duncan A. Buell, editor, *ANTS*, volume 3076 of *Lecture Notes in Computer Science*, pages 272–279. Springer, 2004.

[HHGP+03] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. Ntrusign: Digital signatures using the ntru lattice. In Marc Joye, editor, *CT-RSA*, volume 2612 of *Lecture Notes in Computer Science*, pages 122–140. Springer, 2003.

[HKL+00] Jeffrey Hoffstein, Burton S. Kaliski, Daniel Bennett Lieman, Matthew John Barton Robshaw, and Yiqun Lisa Yin. Secure user identification based on constrained polynomials. *US Patent 6,076,163*, 2000.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. Ntru: A ring-based public key cryptosystem. In Joe Buhler, editor, *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.

[HPS01] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. Nss: An ntru lattice-based signature scheme. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2001.

[Jou00] Antoine Joux. A one round protocol for tripartite diffie-hellman. In *Algorithmic Number Theory - ANTS'00*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer, 2000.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Computing*, 37(1):267–302, 2007.

[NR06] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of ggh and ntru signatures. In *Advances in Cryptology - EUROCRYPT'06*, volume 4004 of *Lecture Notes in Computer Science*, pages 271–288. Springer, 2006.

[PTT10] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Optimal authenticated data structures with multilinear forms. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *Pairing*, volume 6487 of *Lecture Notes in Computer Science*, pages 246–264. Springer, 2010.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.

[Rot13] Ron Rothblum. On the circular security of bit-encryption. In *TCC*, pages 579–598, 2013.

[RS09] Markus Rückert and Dominique Schröder. Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. In Jong Hyuk Park, Hsiao-Hwa Chen, Mohammed Atiquzzaman, Changhoon Lee, Tai-Hoon Kim, and Sang-Soo Yeo, editors, *ISA*, volume 5576 of *Lecture Notes in Computer Science*, pages 750–759. Springer, 2009.

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000.

[Szy03] Michael Szydlo. Hypercubic lattice reduction and analysis of ggh and ntru signatures. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 433–448. Springer, 2003.