# Ideal Multilinear Maps based on Ideal Lattices

Chunsheng Gu

School of Computer Engineering, Jiangsu University of Technology, Changzhou 213001, China
E-mail: chunsheng_gu@163.com

Aug 7, 2015

**Abstract.** Cryptographic multilinear maps have many applications, such as multipartite key exchange and software obfuscation. However, the encodings of three current constructions are "noisy" and their multilinearity levels are fixed and bounded in advance. In this paper, we describe a candidate construction of ideal multilinear maps by using ideal lattices, which supports arbitrary multilinearity levels. The security of our construction depends on new hardness assumptions.
**Keywords.** Ideal multilinear maps, ideal lattices, multipartite Diffie–Hellman key exchange, witness encryption, zeroizing attack

## 1 Introduction

The construction of multilinear maps has been a long-standing open problem since 2003. Many studies on the applications of bilinear maps, such as [SOK00, Jou00, BF01, Sma03], have influenced research on cryptographic multilinear maps [BS03, RS09, PTT10, Rot13]. Boneh and Silverberg [BS03] first introduced the notion of multilinear maps, which are an extension of bilinear maps. However, they suspected that such maps come from the realm of algebraic geometry.

Garg, Gentry, and Halevi (GGH) recently described the first candidate construction of multilinear maps from ideal lattices [GGH13]. The GGH construction, whose encodings are randomized with noise and bounded with a fixed maximum degree, is different from the ideal multilinear maps envisioned by Boneh and Silverberg [BS03]. Construction security depends on the new hardness assumptions of GCDH/GDDH, which provided extensive cryptanalysis in [GGH13]. Langlois, Stehlé, and Steinfeld [LSS14] presented a variant of GGH by reanalyzing its re-randomization process to improve its efficiency. However, by using the zeroizing attack proposed by [GGH13], the application of multipartite key exchange (MPKE) based on GGH was broken by Hu and Jia [HJ15a].

One line of work focused on new constructions of multilinear maps. Following the GGH framework, the second candidate construction of multilinear maps was presented by Coron, Lepoint, and Tibouchi (CLT) [CLT13]. The CLT construction changes from working over ideal lattices to working over integers and is implemented by using many heuristic optimization techniques. However, by using the zeroizing attack, the CLT construction was broken by Cheon et al. [CHL+14]. Boneh, Wu, and Zimmerman [BWZ14] and Garg, Gentry, Halevi, and Zhandry [GGHZ14] proposed two independent approaches to fix the CLT construction [CLT13]. However, Coron, Lepoint, and Tibouchi [CTL14] showed that two fixes can be broken by using an extension of the attack proposed by Cheon et al. [CHL+14]. Recently, Coron, Lepoint, and Tibouchi [CTL15] presented a new variant of CLT by modifying the zero-testing parameter.

The third candidate construction of graph-induced multilinear maps from lattices was proposed by Gentry, Gorbunov, and Halevi [GGH15]. The security of their construction depends on new hardness assumptions and cannot be reduced to LWE or other classic hardness assumptions.

Another line of work focused on the new cryptographic applications of multilinear maps: witness encryption [GGS+13], general program obfuscation [GGH+13b, Zim15], function encryption [GGH+13b], and other applications [GGH+13a, BZ14].

However, all known constructions are noisy multilinear maps. These noisy encodings restrict the number of operations that can be performed and further restrict their applications. In this study, we propose a candidate construction of ideal multilinear map that supports any multilinearity degree.

### 1.1 Our Results

Our main contribution describes a candidate construction of ideal multilinear map by using ideal lattices. The security of our construction depends on new hardness assumptions. The starting point of our work is that, given $k+1$ ring elements $\mathbf{y}_i = \mathbf{a}_i\mathbf{g} + \mathbf{t}_i\mathbf{f}$, all products of the form $\mathbf{b}_j = \mathbf{a}_j \prod_{i \neq j} \mathbf{y}_i \bmod \mathbf{f}$ are identical to element $\mathbf{g}^k \prod_{i=1}^{k+1} \mathbf{a}_i \bmod \mathbf{f}$.

Our construction includes two layers, namely, the inner layer that works over polynomial rings $R = \mathbb{Z}[x]/(x^n+1)$ and $R_q = R/qR$ and the outer layer that works over matrix ring $\mathbb{Z}^{n \times n}$. First, we select secret short ring elements $\mathbf{f}_j, \mathbf{g}_j \in R, j = 1,...,m$ and denote $\mathbf{f} = \prod_{j=1}^{m} \mathbf{f}_j$ and $\mathbf{g} = \mathbf{g}_j \bmod \mathbf{f}_j$. Given an element $\mathbf{a} \in R$, the level-$1$ encoding is $\mathbf{c} = (\mathbf{ag}) \bmod \mathbf{f}$. However, one can compute $\mathbf{g} = (\mathbf{c}/\mathbf{a}) \bmod \mathbf{f}$ when $\mathbf{f}$, $\mathbf{a}$ and $\mathbf{c}$ are known. To prevent this simple division attack, we replace $\mathbf{f}$ with $\mathbf{q} = \mathbf{q}_0\mathbf{f}$ and add noise $\mathbf{rf}$ to encoding $\mathbf{c}$, where $\mathbf{q}_0$ is a short ring element. Thereafter, we transform $\mathbf{a}$ and $\mathbf{c}$ into new encodings $\mathbf{x} = (\mathbf{a} + \mathbf{ef}) \bmod \mathbf{q}$ and $\mathbf{y} = \mathbf{c} + \mathbf{tf} = (\mathbf{ag} + \mathbf{tf}) \bmod \mathbf{q}$, where $\mathbf{e}$ and $\mathbf{t}$ are short random elements drawn from $R$. In our construction, we regard $\mathbf{g}^k$ as level-$k$ encoding and $\mathbf{rf}$ as random encoding of zero.

The aforementioned encodings support the addition and multiplication operations. Given level-$1$ encodings $\mathbf{u}_i = \mathbf{c}_i + \mathbf{r}_i\mathbf{f} = (\mathbf{a}_i\mathbf{g} + \mathbf{r}_i\mathbf{f}) \bmod \mathbf{q}$, $i = 1, 2$, we derive the following expressions:

$$\begin{aligned} \mathbf{u}_1 + \mathbf{u}_2 &= (\mathbf{c}_1 + \mathbf{r}_1\mathbf{f}) + (\mathbf{c}_2 + \mathbf{r}_2\mathbf{f}) \\ &= ((\mathbf{a}_1 + \mathbf{a}_2)\mathbf{g} + (\mathbf{r}_1 + \mathbf{r}_2)\mathbf{f}) \bmod \mathbf{q} \\ \mathbf{u}_1 \cdot \mathbf{u}_2 &= (\mathbf{c}_1 + \mathbf{r}_1 \cdot \mathbf{f}) \cdot (\mathbf{c}_2 + \mathbf{r}_2 \cdot \mathbf{f}) \\ &= ((\mathbf{c}_1 \cdot \mathbf{c}_2) + (\mathbf{c}_1\mathbf{r}_2 + \mathbf{c}_2\mathbf{r}_1 + \mathbf{r}_1\mathbf{r}_2\mathbf{f})\mathbf{f}) \bmod \mathbf{q} \\ &= ((\mathbf{a}_1\mathbf{a}_2)\mathbf{g}^2 + (\mathbf{c}_1\mathbf{r}_2 + \mathbf{c}_2\mathbf{r}_1 + \mathbf{r}_1\mathbf{r}_2\mathbf{f})\mathbf{f}) \bmod \mathbf{q} \end{aligned}$$

Notably, for simplicity, we also consider our scheme as a graded encoding scheme. That is, only same level encodings are added and any level encodings can be multiplied. In fact, our scheme supports addition and multiplication between arbitrary encodings. This is different from the graded encoding in [GGH13, CLT13]. Thus, our construction is a genuine multilinear map. It is easy to verify that given arbitrary encodings $\mathbf{u}_i, i = 1, 2$, the sum $\mathbf{u} = \mathbf{u}_1 + \mathbf{u}_2$ also is an encoding.

One can be decided whether an encoding is $0$. Given $\mathbf{q} = \mathbf{q}_0\mathbf{f}$, all encodings can have the form $(\mathbf{ag}^k + \mathbf{rf}) \bmod \mathbf{q}$. That is, the norm of any encoding is less than $\|\mathbf{q}\|_\infty$, where $\|\mathbf{q}\|_\infty$ is the maximum norm of $\mathbf{q}$. Thus, a zero-testing parameter $\mathbf{p}_{zt} = (\sum_{j=1}^{m} \mathbf{h}_j \cdot (\mathbf{f}_j^{-1} \bmod q)) \bmod q$ is included in the public parameters, where $q \gg \|\mathbf{q}\|_\infty$, to decide whether $\mathbf{u}$ is an encoding of zero. If the norm of $[\mathbf{p}_{zt} \cdot \mathbf{u}]_q$ is small, $\mathbf{u}$ is an encoding of zero; otherwise, $\mathbf{u}$ is an encoding of a nonzero element.

However, this construction can be broken when $\mathbf{q}$ and $\tau$ pairs of encodings

$\mathbf{x}_i = (\mathbf{a}_i + \mathbf{e}_i\mathbf{f}) \bmod \mathbf{q}$ and $\mathbf{y}_i = (\mathbf{a}_i\mathbf{g} + \mathbf{t}_i\mathbf{f}) \bmod \mathbf{q}$ are included in the public parameters. Because one can compute the basis of $\mathbf{f}$, the inverse of $Rot(\mathbf{a}_1)$, and the inverse of $Rot(\mathbf{g})$ by cross-multiplying $\mathbf{x}_{i_1}\mathbf{y}_{i_2} - \mathbf{x}_{i_2}\mathbf{y}_{i_1} = \mathbf{r}_{i_1,i_2}\mathbf{f}$ of $\mathbf{x}_i, \mathbf{y}_i$, where notation $Rot(\mathbf{v})$ denotes the anti-cyclic matrix of $\mathbf{v} \in R$.

Very recently, Pellet-Mary and Stehlé [PS15] presented an attack for our previous version. This is because one can compute $\mathbf{y}^{-1} \bmod \mathbf{q}$ if $\mathbf{y}$ and $\mathbf{q}$ are coprime. To avoid this attack, we multiply all encodings in the public parameters by a random element $\mathbf{h} \in R$. For simplicity, we reload notations in the public parameters in the following. In this case, one can no longer find an invertible encoding over modulo $\mathbf{q}$.

We use a unimodular matrix to enclose the ring elements in the aforementioned construction to prevent the cross-multiplying attack. We first select two short random unimodular matrices $\mathbf{T}$, $\mathbf{S}$. Then we transform $\mathbf{x}_i$, $\mathbf{y}_i$ into $\mathbf{X}_i = \mathbf{S}Rot(\mathbf{x}_i)\mathbf{S}^{-1}$, $\mathbf{Y}_i = \mathbf{T}^{-1}Rot(\mathbf{y}_i)\mathbf{T}$, and $\mathbf{p}_{zt}$ into $\mathbf{P}_{zt} = \left[\mathbf{S}Rot(\mathbf{p}_{zt})\mathbf{T}\right]_q$. For $\mathbf{q}$, we generate two list encodings of zero $\mathbf{M}_t = \mathbf{S}Rot(\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1}$, $\mathbf{N}_t = \mathbf{T}^{-1}Rot(\mathbf{n}_t\mathbf{q})\mathbf{T}$, $t \in [\![n]\!]$, where $\mathbf{m}_t$ and $\mathbf{n}_t$ are short random elements drawn from $R$. We represent $\mathbf{M}$ and $\mathbf{N}$ as $n^2 \times n$ matrices whose column vectors are $\mathbf{M}_t$ and $\mathbf{N}_t$, respectively. Thus, we can easily generate encodings by scalar product; that is, sampling $\tau$ small random integers $d_i$, a level-$1$ encoding is generated as $\mathbf{U} = \sum_{i=1}^{\tau} d_i\mathbf{Y}_i \bmod \mathbf{N}$, and the level-$0$ encoding corresponding to $\mathbf{U}$ is $\mathbf{D} = \sum_{i=1}^{\tau} d_i\mathbf{X}_i \bmod \mathbf{M}$. Similarly, whether the encoding $\mathbf{U}$ is an encoding of zero can be determined by computing $\mathbf{V} = \left[\mathbf{E} \cdot \mathbf{P}_{zt} \cdot \mathbf{U}\right]_q$ and checking the norm of $\mathbf{V}$, where $\mathbf{E} = \sum_{i=1}^{\tau} r_i\mathbf{X}_i \bmod \mathbf{M}$ and can be set an identity matrix. By cross-multiplication, the adversary can obtain $\mathbf{V} = \left[\mathbf{E} \cdot \mathbf{P}_{zt} \cdot \mathbf{U}\right]_q = \mathbf{S}Rot(\mathbf{r})\mathbf{T}$, which is not reduced modulus $q$. We modify the zero-testing parameter to map $\mathbf{V}$ into an integer to damage the structure of the principal ideal lattice $\mathbf{r}$ and remove the relationship between $\mathbf{T}$ and $\mathbf{S}$. This describes our construction of ideal multilinear maps.

Our second contribution is to describe two applications using our ideal construction: multipartite Diffie–Hellman key exchange protocol (MPKE), which supports any number of participants, and witness encryption scheme (WE). In our construction, the size of modulus $q$ does not depend on the multilinearity levels. Thus, the MPKE and WE using our ideal multilinear maps are practical.

### 1.2 Organization

The remainder of this paper is organized as follows. We recall several preliminaries in Section 2. We describe the construction of ideal multilinear maps that use ideal lattices in Section 3. We extend our construction to asymmetric and commutative variants in Section 4. Finally, we propose two applications by using our ideal construction in Section 5.

## 2 Preliminaries

### 2.1 Notations

We denote $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ as the integer ring, rational number field, and real number field,

respectively. The vectors and matrices are denoted in bold. For a positive integer $k$, we let $[\![k]\!]=\{1,2,\cdots,k\}$. For $n$ the power of 2, we let $R=\mathbb{Z}[x]/<x^n+1>$, $R_q=R/qR$, and $\mathbb{k}=\mathbb{Q}[x]/<x^n+1>$. We denote an element of the polynomial ring as a coefficient vector for simplicity. For the element $\mathbf{a}\in R$, we denote $\|\mathbf{a}\|_\infty$ ($\|\mathbf{a}\|$ for short) as the maximum norm of $\mathbf{a}$.

Throughout this study, we use the absolute minimum residual system, that is, $[a]_q=a \bmod q\in(-q/2,q/2]$. Similarly, notation $[\mathbf{a}]_q$ denotes each entry (or each coefficient) $a_i\in(-q/2,q/2]$ of $\mathbf{a}$.

### 2.2 Lattices and Ideal Lattices

The *n*-dimensional full-rank lattice $L\subset\mathbb{R}^n$ is the set of all integer linear combinations $\sum_{i=1}^n y_i\mathbf{b}_i$ of *n* linearly independent vector $\mathbf{b}_i\in\mathbb{R}^n$. If we arrange the vectors of $\mathbf{b}_i$ as the columns of matrix $\mathbf{B}\in\mathbb{R}^{n\times n}$, then $L=\{\mathbf{By}:\mathbf{y}\in\mathbb{Z}^n\}$. We can state that $\mathbf{B}$ spans $L$ if $\mathbf{B}$ is a basis for $L$. For basis $\mathbf{B}$ of the lattice, we denote its parallelization cell as $P(\mathbf{B})=\{\mathbf{Bz}\mid\mathbf{z}\in\mathbb{R}^n,\forall i:-1/2\le z_i<1/2\}$. We let $\det(\mathbf{B})$ be the determinant of $\mathbf{B}$.

For elements $\mathbf{a},\mathbf{g}\in R$, we let $I=<\mathbf{g}>$ be the principal ideal lattice generated by $\mathbf{g}$, and $Rot(\mathbf{g})=(\mathbf{g},x\cdot\mathbf{g},...,x^{n-1}\cdot\mathbf{g})$ the basis of $R$. We denote $[\mathbf{a}]_\mathbf{g}$ as the reduction of $\mathbf{a}$ modulo $Rot(\mathbf{g})$, that is, $[\mathbf{a}]_\mathbf{g}\in P(Rot(\mathbf{g}))$ and $(\mathbf{a}-[\mathbf{a}]_\mathbf{g})\in L(Rot(\mathbf{g}))$.

Given $\mathbf{c}\in\mathbb{R}^n$, $\sigma>0$, we define $D_{L,\sigma,\mathbf{c}}=\rho_{\sigma,\mathbf{c}}(\mathbf{x})/\rho_{\sigma,\mathbf{c}}(L)$ as the Gaussian distribution of lattice $L$, where $\mathbf{x}\in L$, $\rho_{\sigma,\mathbf{c}}(\mathbf{x})=\exp(-\pi\|\mathbf{x}-\mathbf{c}\|^2/\sigma^2)$, and $\rho_{\sigma,\mathbf{c}}(L)=\sum_{x\in L}\rho_{\sigma,\mathbf{c}}(\mathbf{x})$. Thereafter, we write $D_{\mathbb{Z}^n,\sigma,0}$ as $D_{\mathbb{Z}^n,\sigma}$ for simplicity. We denote a Gaussian sample as $\mathbf{x}\leftarrow D_{L,\sigma}$ (or $\mathbf{d}\leftarrow D_{I,\sigma}$) over the lattice $L$ (or the ideal lattice $I$).

### 2.3 Multilinear Maps

**Definition 2.1 (Multilinear Map [BS03]).** For $k+1$ cyclic groups $G_1,...G_k,G_T$ of the same order $p$, a $k$-multilinear map $e:G_1\times\cdots\times G_k\to G_T$ has the following properties:

(1) Elements $\{g_j\in G_j\}_{j=1,...,k}$, index $j\in[\![k]\!]$, and integer $a\in Z_p$ hold that

$$e(g_1,\cdots,a\cdot g_j,\cdots,g_k)=a\cdot e(g_1,\cdots,g_k).$$

(2) Map $e$ is a nondegenerate in the following sense: if elements $\{g_j\in G_j\}_{j=1,...,k}$ are generators of their respective groups, then $e(g_1,\cdots,g_k)$ is a generator of $G_T$.

**Definition 2.2 ($k$-Graded Encoding System [GGH13]).** A $k$-graded encoding system over $R$ is a set system of $S=\{S_j^{(\alpha)}\subset R:\alpha\in R,j\in[\![k]\!]\}$ with the following properties:

(1) For every index $j\in[\![k]\!]$, the sets $\{S_j^{(\alpha)}:\alpha\in R\}$ are disjoint.

(2) Binary operations "+" and "−" exist, such that every $\alpha_1,\alpha_2$, every index $j\in[\![k]\!]$, and every $u_1\in S_j^{(\alpha_1)}$ and $u_2\in S_j^{(\alpha_2)}$ hold that $u_1+u_2\in S_j^{(\alpha_1+\alpha_2)}$ and

$u_1 - u_2 \in S_j^{(\alpha_1-\alpha_2)}$, where $\alpha_1+\alpha_2$ and $\alpha_1-\alpha_2$ are the addition and subtraction operations in $R$, respectively.

(3) Binary operation "×" exists, such that every $\alpha_1, \alpha_2$, every index $j_1, j_2 \in [\![k]\!]$ with $j_1 + j_2 \le k$, and every $u_1 \in S_{j_1}^{(\alpha_1)}$ and $u_2 \in S_{j_2}^{(\alpha_2)}$ hold that $u_1 \times u_2 \in S_{j_1+j_2}^{(\alpha_1\times\alpha_2)}$, where $\alpha_1 \times \alpha_2$ is the multiplication operation in $R$ and $j_1 + j_2$ is the integer addition.

## 3 Ideal Multilinear Maps

In this section, we describe the construction of ideal multilinear maps applying unimodular matrices. For our construction, its inner layer works over the polynomial ring, and its outer layer works over the matrix ring. We use different unimodular matrices for elements of level-$0$ and level-$1$ encodings to prevent the cross-multiplication between level-$0$ and level-$1$ encodings. For simplicity, we only consider our construction as graded encoding.

### 3.1 Construction

**Setting the parameters.** We let $\lambda$ be the security parameter and $n$ be the dimension of polynomial ring $R$. Concrete parameters are set as $\sigma = \sqrt{\lambda n}$, $\sigma' = \lambda n^{1.5}$, $n = O(\lambda^2)$, $m = O(\lambda)$, $\xi = \lambda^{O(m)}$, $q \ge mn^{21}\xi^2 2^{\eta}$, and $\tau = O(n^2)$.

**Instance generation**: $(\text{par}) \leftarrow \text{InstGen}(1^{\lambda})$.

(1) Select a sufficiently large prime $q$.

(2) Generate parameters in the inner layer:

(2.1) Sample $\mathbf{f}_j \leftarrow D_{\mathbb{Z}^n,\sigma}$, $j \in [\![m]\!]$, and $\mathbf{h}, \mathbf{q}_0 \leftarrow D_{\mathbb{Z}^n,\sigma'}$ such that $\mathbf{f}_j$, $j \in [\![m]\!]$ are pairwise relatively co-prime, $\mathbf{f}_j^{-1} \in \mathbb{k}$, and $\|\mathbf{f}_j^{-1}\| \le O(n^2)$.

(2.2) Set $\mathbf{f} = \prod_{j=1}^{m} \mathbf{f}_j$ and $\mathbf{q} = \mathbf{h}\mathbf{q}_0\mathbf{f}$ such that matrix $Rot(\mathbf{q})$ is invertible.

(2.3) Sample $\mathbf{g}_0, \mathbf{g}_1 \leftarrow D_{\mathbb{Z}^n,\mu}$, $\mathbf{a}_i \leftarrow D_{\mathbb{Z}^n,\mu}$, and $\mathbf{e}_i, \mathbf{t}_i \leftarrow D_{\mathbb{Z}^n,\sigma'}, i \in [\![\tau]\!]$, where $\mu = \|\mathbf{f}\|$.

(2.4) Set $\mathbf{x}_i = \mathbf{h}(\mathbf{a}_i\mathbf{g}_0 + \mathbf{e}_i\mathbf{f}) \bmod \mathbf{q}$, and $\mathbf{y}_i = \mathbf{h}(\mathbf{a}_i\mathbf{g}_1 + \mathbf{t}_i\mathbf{f}) \bmod \mathbf{q}, i \in [\![\tau]\!]$.

(2.5) Set $\mathbf{p}_{zt} = (\sum_{j=1}^{m} \mathbf{h}_j \cdot (\mathbf{f}_j^{-1} \bmod q)) \bmod q$, where $\mathbf{h}_j \leftarrow D_{\mathbb{Z}^n,\sigma}$, $j \in [\![m]\!]$.

(2.6) Sample $\mathbf{m}_t, \mathbf{n}_t \leftarrow D_{\mathbb{Z}^n,\sigma}$, $t \in [\![n]\!]$, such that $\mathbf{M}' = (\mathbf{m}_1\mathbf{q}, \cdots, \mathbf{m}_n\mathbf{q})$ and $\mathbf{N}' = (\mathbf{n}_1\mathbf{q}, \cdots, \mathbf{n}_n\mathbf{q})$ are invertible.

(3) Generate parameters in the outer layer:

(3.1) Sample randomly unimodular matrices $\mathbf{S}$ and $\mathbf{T}$ such that $\|\mathbf{S}\| = \|\mathbf{T}\| \le O(n^2)$.

(3.2) Set $\mathbf{X}_i = \mathbf{S}Rot(\mathbf{x}_i)\mathbf{S}^{-1}$, $\mathbf{Y}_i = \mathbf{T}^{-1}Rot(\mathbf{y}_i)\mathbf{T}$, $i \in [\![\tau]\!]$.

(3.3) Set $\mathbf{M}_t = \mathbf{S}Rot(\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1}$, $\mathbf{N}_t = \mathbf{T}^{-1}Rot(\mathbf{n}_t\mathbf{q})\mathbf{T}$, $t \in [\![n]\!]$. We denote $\mathbf{M}, \mathbf{N} \in \mathbb{Z}^{n^2\times n}$ as the matrix of column vectors $\mathbf{M}_t$ and $\mathbf{N}_t$, where $\mathbf{M}_t$ and $\mathbf{N}_t$ are considered $n^2$-dimensional column vectors. We let $\|\mathbf{M}\| = \|\mathbf{N}\| = \xi$.

(4) Generate the parameters of zero-testing and extraction:

(4.1) Sample $\mathbf{s},\mathbf{t} \leftarrow D_{\mathbb{Z}^n,\sigma}$.

(4.2) Randomly select $\mathbf{z}_s, \mathbf{z}_t \in R_q$, such that $(\mathbf{z}_s)^{-1}, (\mathbf{z}_t)^{-1} \in R_q$.

(4.3) Set $\mathbf{s}^* = \left[\mathbf{s}^T Rot(\mathbf{z}_s)^{-1}\mathbf{S}^{-1}\right]_q$ and $\mathbf{t}^* = \left[\mathbf{T}^{-1} Rot(\mathbf{z}_t)^{-1}\mathbf{t}\right]_q$.

(4.4) Set $\mathbf{P}_{zt} = \left[\mathbf{S} Rot(\mathbf{z}_s\mathbf{z}_t\mathbf{p}_{zt})\mathbf{T}\right]_q$.

(5) Output the public parameters $\text{par} = \left\{q, \mathbf{M}, \mathbf{N}, \{\mathbf{X}_i, \mathbf{Y}_i\}_{i\in[\![\tau]\!]}, \mathbf{P}_{zt}, \mathbf{s}^*, \mathbf{t}^*\right\}$.

**Generating level-$k$ random encodings:** $\mathbf{U} \leftarrow \text{Enc}(\text{par}, k, \{w_i\}_{i\in[\![\tau]\!]})$.

Select $w_i \leftarrow D_{\mathbb{Z},\sigma'}$, $i \in [\![\tau]\!]$ and generate a level-$0$ encoding $\mathbf{D} = \sum_{i=1}^{\tau} w_i \cdot (\mathbf{X}_i)^k \bmod \mathbf{M}$ and a level-$k$ encoding $\mathbf{U} = \sum_{i=1}^{\tau} w_i \cdot (\mathbf{Y}_i)^k \bmod \mathbf{N}$.

**Adding encodings:** $\mathbf{U} \leftarrow \text{Add}(\text{par}, k, \mathbf{U}_1, \cdots, \mathbf{U}_\beta)$.

Given $\beta$ level-$k$ encodings $\mathbf{U}_\alpha, \alpha \in [\![\beta]\!]$, their sum $\mathbf{U} = \sum_{\alpha=1}^{\beta} \mathbf{U}_\alpha \bmod \mathbf{N}$ is a level-$k$ encoding.

**Multiplying encodings:** $\mathbf{U} \leftarrow \text{Mul}(\text{par}, 1, \mathbf{U}_1, \cdots, \mathbf{U}_k)$.

Given $k$ level-$1$ encodings $\mathbf{U}_\alpha, \alpha \in [\![k]\!]$, their product $\mathbf{U} = \prod_{\alpha=1}^{k} \mathbf{U}_\alpha \bmod \mathbf{N}$ is a level-$k$ encoding.

**Zero-testing:** $\text{isZero}(\text{par}, \mathbf{D}, \mathbf{U})$.

Given a level-$k$ encoding $\mathbf{U}$ and a level-$0$ encoding $\mathbf{D}$, we determine whether $\mathbf{U}$ is an encoding of zero. We compute $v = \left[\mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U} \cdot \mathbf{t}^*\right]_q$ in $\mathbb{Z}_q$ and check whether $v$ is small, as follows:

$$\text{isZero}(\text{par}, \mathbf{D}, \mathbf{U}) = \begin{cases} 1 & \text{if } |v| < q/2^\eta \\ 0 & \text{otherwise} \end{cases}.$$

**Extract:** $sk \leftarrow \text{Ext}(\text{par}, \mathbf{D}, \mathbf{U})$.

Given a level-$k$ encoding $\mathbf{U}$ and a level-$0$ encoding $\mathbf{D}$, $\text{Ext}(\text{par}, \mathbf{D}, \mathbf{U}) = \text{Extract}_s(\text{msbs}_\gamma(\left[\mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U} \cdot \mathbf{t}^*\right]_q))$, where $\text{msbs}_\gamma$ extracts the $\gamma = \eta - \lambda$ most significant bits from the result. $\text{Extract}_s$ is a strong randomness extractor using the seed $s$.

**Remark 3.1** (1) Our scheme support addition and multiplication operations for arbitrary encodings. The aim using graded encoding is to conveniently describe our construction. That is, our construction is a genuine multilinear map scheme. (2) One can take $\mathbf{g}_1 = 1$. (3) One can set $m = 1$, that is, $\mathbf{f} = \prod_{j=1}^{m} \mathbf{f}_j$. Given that an integer $v = \left[\mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U} \cdot \mathbf{t}^*\right]_q$, we do not know how to attack our construction by using the integers of this form. In fact, we can change vectors $\mathbf{s},\mathbf{t} \leftarrow D_{\mathbb{Z}^n,\sigma}$ into matrices $\mathbf{S}_1 \leftarrow D_{\mathbb{Z}^{k_1\times n},\sigma}, \mathbf{T}_1 \leftarrow D_{\mathbb{Z}^{n\times k_2},\sigma}$, and replace $\mathbf{s}^*, \mathbf{t}^*$ with $\mathbf{S}^* = \mathbf{S}_1 Rot(\mathbf{z}_s)^{-1}\mathbf{S}^{-1}$, $\mathbf{T}^* = \mathbf{T}^{-1} Rot(\mathbf{z}_t)^{-1}\mathbf{T}_1$. In this case, we compute a $k_1 \times k_2$-matrix $\left[\mathbf{S}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U} \cdot \mathbf{T}^*\right]_q$ as the final result, where $k_1 \times k_2 < n$ to guarantee the security of construction. (3) We set $\tau \geq n^2 + \lambda$ to prevent the algebraic equation attack. However, we can take $\tau = \lambda n$ or $\tau = 2n$ according to an optimization in [HJ15c]. (4) The matrix $\mathbf{D}$ can be taken the identity matrix. Our aim is to demonstrate how to use level-$0$ encodings when constructing the MPKE protocol. (5) When our construction is applied to multipartite Diffie–Hellman key exchange, $\mathbf{P}_{zt}, \mathbf{X}_i$ in the public parameters can be replaced

by $\mathbf{P}_{zt,i}=\left[\mathbf{X}_i\mathbf{P}_{zt}\right]_q$ and $\mathbf{S}$ does not require a unimodular matrix. Moreover, the matrix $\mathbf{P}_{zt,i}=\left[\mathbf{X}_i\mathbf{P}_{zt}\right]_q$ may be further modified into a vector $\mathbf{p}_{zt,i}=\left[\mathbf{s}^*\mathbf{X}_i\mathbf{P}_{zt}\right]_q$.

### 3.2 Correctness

**Lemma 3.2** $\mathrm{InstGen}(1^\lambda)$ is a probabilistic polynomial time algorithm.

**Proof**. Unimodular matrices $\mathbf{S}$ and $\mathbf{T}$ can be generated by the method of [GGH15]. All other elements in $\mathrm{InstGen}(1^\lambda)$ can be computed in polynomial time. ■

**Lemma 3.3** The ranks of $\mathbf{M}$ and $\mathbf{N}$ in the public parameter $\mathrm{par}$ are $n$.

**Proof**. We prove the result by contradiction and assume that the rank of $\mathbf{M}$ is less than $n$. Without loss of generality, assume that there exist $n$ non-all-zero real numbers $k_t$ such that $\sum_{t=1}^{n}k_t\mathbf{M}_t=\mathbf{0}^{n\times n}$. Thus, we derive the following expression:

$$\sum_{t=1}^{n}k_t\mathbf{M}_t=\mathbf{S}(\sum_{t=1}^{n}k_t Rot(\mathbf{m}_t\mathbf{q}))\mathbf{S}^{-1}=\mathbf{0}^{n\times n}.$$

Given that $\mathbf{S}, Rot(\mathbf{q})$ are invertible over $\mathbb{R}$, we derive $\sum_{t=1}^{n}k_t Rot(\mathbf{m}_t\mathbf{q})=\mathbf{0}^{n\times n}$, that is, $\sum_{t=1}^{n}k_t\mathbf{m}_t\mathbf{q}=\mathbf{0}^{n}$.

On the basis of the rank $n$ of $\mathbf{M}'=(\mathbf{m}_1\mathbf{q},\cdots,\mathbf{m}_n\mathbf{q})$, a contradiction is generated. Similarly, we can prove that the rank of $\mathbf{N}$ is $n$. ■

**Lemma 3.4** If we assume that $Space(\mathbf{M})$ and $Space(\mathbf{N})$ are linear spaces spanned by $\mathbf{M}$ and $\mathbf{N}$, respectively, then $\mathbf{X}_i\in Space(\mathbf{M})$ and $\mathbf{Y}_i\in Space(\mathbf{N})$ for $\{\mathbf{X}_i\}_{i\in[\![\tau]\!]},\{\mathbf{Y}_i\}_{i\in[\![\tau]\!]}$ in the public parameter $\mathrm{par}$.

**Proof**. Given that $\mathbf{M}'$ is invertible, vector $\mathbf{k}=(\mathbf{M}')^{-1}\cdot\mathbf{x}_i$ for $\mathbf{x}_i$ of $\mathbf{X}_i=\mathbf{S}Rot(\mathbf{x}_i)\mathbf{S}^{-1}$, that is, $\mathbf{M}'\cdot\mathbf{k}=\mathbf{x}_i$. Thus, we derive the following expression:

$$\sum_{t=1}^{n}k_t\mathbf{m}_t\mathbf{q}=\mathbf{x}_i.$$

The $j$-th column of $Rot(\mathbf{x}_i)$ is $\mathbf{x}_i\cdot x^j$, $(\sum_{t=1}^{n}k_t\mathbf{m}_t\mathbf{q})\cdot x^j=\mathbf{x}_i\cdot x^j$. As such, $\mathbf{X}_i=\sum_{t=1}^{n}k_t\mathbf{M}_t$ and $\mathbf{X}_i\in Space(\mathbf{M})$. Similarly, $\mathbf{Y}_i\in Space(\mathbf{N})$. ■

**Lemma 3.5** Encoding $\mathbf{U}\leftarrow\mathrm{Enc}(\mathrm{par},k,\{w_i\}_{i\in[\![\tau]\!]})$ is a level-$k$ encoding.

**Proof**. Given that $\mathbf{D}=\sum_{i=1}^{\tau}w_i\cdot(\mathbf{X}_i)^k \bmod \mathbf{M}$, we derive the following expressions:

$$\begin{aligned}\mathbf{D}&=\sum_{i=1}^{\tau}w_i\cdot(\mathbf{X}_i)^k \bmod \mathbf{M}\\&=\sum_{i=1}^{\tau}w_i\cdot(\mathbf{X}_i)^k-\sum_{t=1}^{n}\alpha_t\cdot\mathbf{M}_t\\&=\mathbf{S}(\sum_{i=1}^{\tau}w_i\cdot Rot(\mathbf{x}_i)^k-\sum_{t=1}^{n}\alpha_t\cdot Rot(\mathbf{m}_t\mathbf{q}))\mathbf{S}^{-1}\\&=\mathbf{S}Rot(\mathbf{d})\mathbf{S}^{-1}\end{aligned},$$

$$
\begin{aligned}
&\mathbf{d} \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{x}_i)^k - \sum_{t=1}^{n} \alpha_t \cdot (\mathbf{m}_t \mathbf{q})) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{x}_i)^k) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{h}(\mathbf{a}_i + \mathbf{e}_i \mathbf{f}) \bmod \mathbf{q})^k) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{h}\mathbf{a}_i \bmod \mathbf{f}_j)^k) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{a}_{i,j})^k) \bmod \mathbf{f}_j
\end{aligned}
$$

Given that $\mathbf{U} = \sum_{i=1}^{\tau} w_i \cdot (\mathbf{Y}_i)^k \bmod \mathbf{N}$, we derive the following expressions:

$$
\begin{aligned}
\mathbf{U} &= \sum_{i=1}^{\tau} w_i \cdot (\mathbf{Y}_i)^k \bmod \mathbf{N} \\
&= \sum_{i=1}^{\tau} w_i \cdot (\mathbf{Y}_i)^k - \sum_{t=1}^{n} \beta_t \cdot \mathbf{N}_t \\
&= \mathbf{T}^{-1} (\sum_{i=1}^{\tau} w_i \cdot Rot(\mathbf{y}_i)^k - \sum_{t=1}^{n} \beta_t \cdot Rot(\mathbf{n}_t \mathbf{q})) \mathbf{T} \\
&= \mathbf{T}^{-1} Rot(\mathbf{u}) \mathbf{T}
\end{aligned}
$$

$$
\begin{aligned}
&\mathbf{u} \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{y}_i)^k - \sum_{t=1}^{n} \beta_t \cdot (\mathbf{n}_t \mathbf{q})) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{y}_i)^k) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{h}(\mathbf{a}_i \mathbf{g} + \mathbf{t}_i \mathbf{f}) \bmod \mathbf{q})^k) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{h}\mathbf{a}_i \bmod \mathbf{f}_j)^k (\mathbf{g} \bmod \mathbf{f}_j)^k) \bmod \mathbf{f}_j \\
&= (\sum_{i=1}^{\tau} w_i \cdot (\mathbf{a}_{i,j})^k \cdot (\mathbf{g}_j)^k) \bmod \mathbf{f}_j \\
&= ((\mathbf{d} \bmod \mathbf{f}_j) \cdot (\mathbf{g}_j)^k) \bmod \mathbf{f}_j
\end{aligned}
$$

In the previously presented expressions, we use notations $\mathbf{a}_{i,j} = \mathbf{h}\mathbf{a}_i \bmod \mathbf{f}_j$ and $\mathbf{g}_j = \mathbf{g} \bmod \mathbf{f}_j$.

As such, $\mathbf{U}$ is a level-$k$ encoding of $\mathbf{D}$. ■

**Lemma 3.6** Encoding $\mathbf{U} \leftarrow \mathrm{Add}(\mathrm{par}, k, \mathbf{U}_1, \cdots, \mathbf{U}_s)$ is a level-$k$ encoding.

**Proof**. According to the modulo arithmetic rule, the sum $\mathbf{U}$ of level-$k$ encodings $\mathbf{U}_t, t \in [\![s]\!]$ is a level-$k$ encoding. ■

**Lemma 3.7** $\mathbf{U} \leftarrow \mathrm{Mul}(\mathrm{par}, 1, \mathbf{U}_1, \cdots, \mathbf{U}_k)$ is a level-$k$ encoding.

**Proof**. Given that $\mathbf{U}_t, t \in [\![k]\!]$ are level-1 encodings, we obtain $\mathbf{U}_t = \mathbf{T}^{-1} Rot(\mathbf{u}_t) \mathbf{T}$ and $\mathbf{u}_t = \mathbf{h}(\mathbf{d}_t \mathbf{g} + \mathbf{r}_t \mathbf{f}) \bmod \mathbf{q}$. As such, we derive the following expressions:

$$
\begin{aligned}
\mathbf{U} &= \prod_{t=1}^{k} \mathbf{U}_t \bmod \mathbf{N} \\
&= \mathbf{T}^{-1} Rot(\prod_{t=1}^{k} \mathbf{u}_t) \mathbf{T} - \sum_{t=1}^{n} \beta_t \cdot \mathbf{N}_t \\
&= \mathbf{T}^{-1} Rot(\prod_{t=1}^{k} \mathbf{u}_t - \sum_{t=1}^{n} \beta_t \cdot \mathbf{n}_t \mathbf{q}) \mathbf{T} \\
&= \mathbf{T}^{-1} Rot(\mathbf{u}) \mathbf{T}
\end{aligned}
$$

$$
\begin{aligned}
&\mathbf{u} \bmod \mathbf{f}_j \\
&= (\prod_{t=1}^{k} \mathbf{u}_t - \sum_{t=1}^{n} \beta_t \cdot \mathbf{n}_t \mathbf{q}) \bmod \mathbf{f}_j \\
&= (\prod_{t=1}^{k} \mathbf{h}(\mathbf{d}_t \mathbf{g} + \mathbf{r}_t \mathbf{f}) \bmod \mathbf{f}_j)) \bmod \mathbf{f}_j \ , \\
&= (\prod_{t=1}^{k} \mathbf{d}_{t,j} \mathbf{g}_j) \bmod \mathbf{f}_j \\
&= (\prod_{t=1}^{k} \mathbf{d}_{t,j})(\mathbf{g}_j)^k \bmod \mathbf{f}_j
\end{aligned}
$$

where $\mathbf{d}_{t,j} = \mathbf{h}\mathbf{d}_t \bmod \mathbf{f}_j$. Thus, $\mathbf{U}$ is a level-$k$ encoding. ■

**Lemma 3.8** For an arbitrary integer $k > 0$, the zero-testing algorithm $\text{isZero}(\text{par}, \mathbf{D}, \mathbf{U})$ correctly determines whether $\mathbf{U}$ is an encoding of zero.

**Proof**. Given a level-$0$ encoding $\mathbf{D}$ and an arbitrary level-$k$ encoding $\mathbf{U}$, we obtain $\mathbf{D} = \mathbf{S}Rot(\mathbf{d})\mathbf{S}^{-1}$ , $\mathbf{U} = \mathbf{T}^{-1}Rot(\mathbf{u})\mathbf{T}$ , $\mathbf{u} = \mathbf{a} + \mathbf{r} \cdot \mathbf{f}$ and $\|\mathbf{d}\| \le n^2 \|\mathbf{S}\| \|\mathbf{D}\| \|\mathbf{S}^{-1}\|$ , $\|\mathbf{a}\| < \|\mathbf{f}\|$, $\|\mathbf{u}\| \le n^2 \|\mathbf{T}\| \|\mathbf{U}\| \|\mathbf{T}^{-1}\|$.

On the basis of $\|\mathbf{D}\| \le \|\mathbf{M}\|$ and $\|\mathbf{U}\| \le \|\mathbf{N}\|$, we obtain $\|\mathbf{d}\| \le n^6 \xi$ and $\|\mathbf{u}\| \le n^6 \xi$.

(1) If $\mathbf{U}$ is an encoding of zero, then $\mathbf{u} \bmod \mathbf{f}_j = 0, j \in [\![m]\!]$. Given that $\mathbf{f}_j, j \in [\![m]\!]$ are pairwise relatively prime, $\mathbf{u} \bmod \mathbf{f} = 0$, that is, $\mathbf{a} \bmod \mathbf{f} = 0$ and $\mathbf{a} = 0$ based on $\|\mathbf{a}\| < \|\mathbf{f}\|$. As such, we derive the following expression:

$$
\begin{aligned}
|v| &= \left| \left[ \mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U} \cdot \mathbf{t}^* \right]_q \right| \\
&= \left| \mathbf{s}^T Rot(\sum_{j=1}^{m} \mathbf{d}\mathbf{h}_j \mathbf{r} \cdot \mathbf{f} / \mathbf{f}_j) \mathbf{t} \right| \\
&\le n^2 \|\mathbf{s}\| \left\| Rot(\sum_{j=1}^{m} \mathbf{d}\mathbf{h}_j \mathbf{r} \cdot \mathbf{f} / \mathbf{f}_j) \right\| \|\mathbf{t}\| \\
&\le n^2 \|\mathbf{s}\| \left\| \sum_{j=1}^{m} \mathbf{d}\mathbf{h}_j \mathbf{r} \cdot \mathbf{f} / \mathbf{f}_j \right\| \|\mathbf{t}\| \quad . \\
&\le mn^5 \cdot \|\mathbf{d}\| \cdot \|\mathbf{h}_j\| \cdot \|\mathbf{r}\mathbf{f}\| \cdot \|(\mathbf{f}_j)^{-1}\| \\
&\le mn^5 \cdot n^6 \xi \cdot n^2 \cdot n^6 \xi \cdot n^2 \\
&\le q / 2^{\eta}
\end{aligned}
$$

(2) If $\mathbf{U}$ is not an encoding of zero, then $\mathbf{u} \bmod \mathbf{f} \neq 0$ , that is, $\exists j \in [\![m]\!], \mathbf{a} \bmod \mathbf{f}_j \neq 0$. As such, we derive the following expression:

$$
\begin{aligned}
|v| &= \left|\left[\mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U} \cdot \mathbf{t}^*\right]_q\right| \\
&= \left|\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \cdot \left[\mathbf{f}_j^{-1}\right]_q \cdot (\mathbf{a}+\mathbf{rf}))\mathbf{t}\right]_q\right| \\
&= \left\|\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \cdot \left[\mathbf{f}_j^{-1}\right]_q \cdot (\mathbf{a}+\mathbf{rf}))\mathbf{t}\right]_q\right\| \\
&= \left\|\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{a} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t} + \mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{r} \cdot \mathbf{f}/\mathbf{f}_j)\mathbf{t}\right]_q\right\| \\
&\geq \left\|\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{a} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q\right\| - n^2 \left\|\mathbf{s}^T\right\| \left\|Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{r} \cdot \mathbf{f}/\mathbf{f}_j)\right\| \|\mathbf{t}\| \\
&\geq q^{1-\varepsilon} - q/2^{\eta} \\
&\geq q^{1-\varepsilon'}
\end{aligned}
$$

where $\varepsilon, \varepsilon'$ are small positive constants.

As such, $\text{isZero}(\text{par}, \mathbf{D}, \mathbf{U})$ correctly decides whether $\mathbf{U}$ is an encoding of zero. ■

**Lemma 3.9** Suppose that $\mathbf{D} = \mathbf{S}Rot(\mathbf{d})\mathbf{S}^{-1}$ is a level-$0$ encoding, and $\mathbf{U}_t = \mathbf{T}^{-1}Rot(\mathbf{u}_t)\mathbf{T}$, $t \in [\![2]\!]$ two level-$k$ encodings. If $\mathbf{U}_t, t \in [\![2]\!]$ encode the same level-$0$ elements, namely, $\mathbf{u}_1 = \mathbf{u}_2 = \mathbf{a}_j(\mathbf{g}_j)^k \bmod \mathbf{f}_j, \forall j \in [\![m]\!]$, then we derive the following expression:

$$\text{Ext}(\text{par}, \mathbf{D}, \mathbf{U}_1) = \text{Ext}(\text{par}, \mathbf{D}, \mathbf{U}_2).$$

**Proof**. Given that $\mathbf{u}_1 = \mathbf{u}_2 = \mathbf{a}_j(\mathbf{g}_j)^k \bmod \mathbf{f}_j, \forall j \in [\![m]\!]$ and $\mathbf{f}_j$, $j \in [\![m]\!]$ are pairwise relatively prime, we obtain $\mathbf{u}_1 = \mathbf{a} + \mathbf{r}_1 \cdot \mathbf{f}, \mathbf{u}_2 = \mathbf{a} + \mathbf{r}_2 \cdot \mathbf{f}$. As such, we derive the following expressions:

$$
\begin{aligned}
v_1 &= \left[\mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U}_1 \cdot \mathbf{t}^*\right]_q \\
&= \left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \cdot \left[\mathbf{f}_j^{-1}\right]_q \cdot (\mathbf{a}+\mathbf{r}_1\mathbf{f}))\mathbf{t}\right]_q \\
&= \left(\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{a} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q + \left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{r}_1\mathbf{f} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q\right) \bmod q \\
v_2 &= \left[\mathbf{s}^* \cdot \mathbf{D} \cdot \mathbf{P}_{zt} \cdot \mathbf{U}_2 \cdot \mathbf{t}^*\right]_q \\
&= \left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \cdot \left[\mathbf{f}_j^{-1}\right]_q \cdot (\mathbf{a}+\mathbf{r}_2\mathbf{f}))\mathbf{t}\right]_q \\
&= \left(\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{a} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q + \left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{r}_2\mathbf{f} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q\right) \bmod q
\end{aligned}
$$

On the basis of Lemma 3.8, we obtain $\left|\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{r}_t\mathbf{f} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q\right| \leq q/2^{\eta}, t \in [\![2]\!]$. Thus, the $\gamma = \eta - \lambda$ most significant bits of $v_1, v_2$, which are the same with high probability, are decided on the basis of the first term $\left[\mathbf{s}^T Rot(\textstyle\sum_{j=1}^{m} \mathbf{dh}_j \mathbf{a} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q$. ■

### 3.3 Security

The security of our constructions depends on new hardness assumptions and cannot be reduced to classic hard problems, such as lattice hard problem or LWE.

We adaptively extend the definition of ext-GCDH/ext-GDDH in [LSS14] to our construction. Consider the following process:

(1) $(\text{par}) \leftarrow \text{InstGen}(1^{\lambda})$.

(2) Select an arbitrary positive integer $k$.

(3) For $t = 0$ to $k$:

Sample $w_{t,i} \leftarrow D_{\mathbb{Z},\sigma'}, i \in [\![\tau]\!]$

Generate level-$1$ encoding of $\mathbf{D}_t = (\sum_{i=1}^{\tau} w_{t,i} \cdot \mathbf{X}_i) \bmod \mathbf{M}$:

$\mathbf{U}_t = (\sum_{i=1}^{\tau} w_{t,i} \cdot \mathbf{Y}_i) \bmod \mathbf{N}$.

(4) Sample $r_{0,i} \leftarrow D_{\mathbb{Z},\sigma'}, i \in [\![\tau]\!]$ and generate $\mathbf{R}_0 = (\sum_{i=1}^{\tau} r_{0,i} \cdot \mathbf{X}_i) \bmod \mathbf{M}$.

(5) Compute $\mathbf{U} = \prod_{t=1}^{k} \mathbf{U}_t \bmod \mathbf{N}$.

(6) Set $v_C = v_D = \text{Ext}(\text{par}, \mathbf{D}_0, \mathbf{U})$.

(7) Set $v_R = \text{Ext}(\text{par}, \mathbf{R}_0, \mathbf{U})$.

**Definition 3.10** (ideal-ext-GCDH/ideal-ext-GDDH). The extraction $k$-graded computational Diffie-Hellman problem (ideal-ext-GCDH) is on input $\{\text{par}, \mathbf{U}_0, \cdots, \mathbf{U}_k\}$ to output an extraction encoding $v_C \in \mathbb{Z}$. The extraction $k$-graded decisional Diffie-Hellman problem (ideal-ext-GDDH) distinguishes between $v_D$ and $v_R$, that is, between the distributions $D_{GDDH} = \{\text{par}, \mathbf{U}_0, \cdots, \mathbf{U}_k, v_D\}$ and $D_{RAND} = \{\text{par}, \mathbf{U}_0, \cdots, \mathbf{U}_k, v_R\}$.

In this study, we assume that the ideal-ext-GCDH/ideal-ext-GDDH is hard.

### 3.4 Cryptanalysis

#### 3.4.1 Easily Computable Quantities

Given that $\mathbf{M}_t = \mathbf{S}Rot(\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1}, \mathbf{N}_t = \mathbf{T}^{-1}Rot(\mathbf{n}_t\mathbf{q})\mathbf{T}$ and $\mathbf{S}, \mathbf{T}$ are unimodular matrices, we derive the following expressions:

$$\det(\mathbf{M}_t) = \det(\mathbf{S}Rot(\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1}) = \det(Rot(\mathbf{m}_t\mathbf{q})) = \det(Rot(\mathbf{m}_t))\det(Rot(\mathbf{q})),$$

$$\det(\mathbf{N}_t) = \det(\mathbf{S}Rot(\mathbf{n}_t\mathbf{q})\mathbf{S}^{-1}) = \det(Rot(\mathbf{n}_t\mathbf{q})) = \det(Rot(\mathbf{n}_t))\det(Rot(\mathbf{q})).$$

As such, the determinants $\det(Rot(\mathbf{q}))$, $\det(Rot(\mathbf{m}_t))$, and $\det(Rot(\mathbf{n}_t))$ can be computed by using the GCD algorithm.

Given that $\mathbf{N}_t, t \in [\![n]\!]$ are also encodings of zero, we derive the following expression:

$$\begin{aligned}
\mu_t &= \left[\mathbf{s}^* \mathbf{P}_{zt} \mathbf{N}_t \mathbf{t}^*\right]_q \\
&= \left[\mathbf{s}^T Rot(\mathbf{p}_{zt}\mathbf{n}_t\mathbf{q})\mathbf{t}\right]_q \\
&= \left[\mathbf{s}^T Rot(\sum_{j=1}^{m} \mathbf{h}_j \mathbf{n}_t \mathbf{p}_0 \cdot \mathbf{f} \cdot \left[\mathbf{f}_j^{-1}\right]_q)\mathbf{t}\right]_q \\
&= \left[\mathbf{s}^T Rot(\sum_{j=1}^{m} \mathbf{h}_j \mathbf{n}_t \mathbf{p}_0 \cdot \mathbf{f} / \mathbf{f}_j)\mathbf{t}\right]_q
\end{aligned}.$$

By using our parameter settings, $\mu_t$ is not reduced modulo $q$, that is, $\mu_t = \mathbf{s}^T Rot(\sum_{j=1}^{m} \mathbf{h}_j \mathbf{n}_t \mathbf{p}_0 \cdot \mathbf{f} / \mathbf{f}_j)\mathbf{t}$.

For $\mathbf{M}_t, t \in [\![n]\!]$, we derive $\nu_t = \left[\mathbf{s}^* \mathbf{M}_t \mathbf{P}_{zt} \mathbf{t}^*\right]_q = \mathbf{s}^T Rot(\sum_{j=1}^{m} \mathbf{h}_j \mathbf{m}_t \mathbf{p}_0 \cdot \mathbf{f} / \mathbf{f}_j)\mathbf{t}$.

Similarly, $\theta_{s,t} = \left[\mathbf{s}^* \mathbf{M}_s \mathbf{P}_{zt} \mathbf{N}_t \mathbf{t}^*\right]_q = \mathbf{s}^T Rot(\sum_{j=1}^{m} \mathbf{h}_j \mathbf{m}_s \mathbf{n}_t \cdot \mathbf{q}^2 / \mathbf{f}_j)\mathbf{t}$, where $s, t \in [\![n]\!]$.

By using the cross-multiplication of $\mathbf{X}_i, \mathbf{Y}_i$, we can also obtain the following quantities:

$$\begin{aligned}\pi_{i_1,i_2} &= \left[\mathbf{s}^* (\mathbf{X}_{i_1} \mathbf{P}_{zt} \mathbf{Y}_{i_2} - \mathbf{X}_{i_2} \mathbf{P}_{zt} \mathbf{Y}_{i_1}) \mathbf{t}^*\right]_q \\ &= \left[\mathbf{s}^T Rot(\mathbf{p}_{zt}((\mathbf{a}_{i_1} + \mathbf{e}_{i_1}\mathbf{f})(\mathbf{a}_{i_2}\mathbf{g} + \mathbf{t}_{i_2}\mathbf{f}) - (\mathbf{a}_{i_2} + \mathbf{e}_{i_2}\mathbf{f})(\mathbf{a}_{i_1}\mathbf{g} + \mathbf{t}_{i_1}\mathbf{f})))\mathbf{t}\right]_q . \\ &= \left[\mathbf{s}^T Rot(\mathbf{p}_{zt}\mathbf{f}(\mathbf{a}_{i_1}\mathbf{t}_{i_2} + \mathbf{e}_{i_1}\mathbf{a}_{i_2}\mathbf{g} + \mathbf{e}_{i_1}\mathbf{t}_{i_2}\mathbf{f} - \mathbf{a}_{i_2}\mathbf{t}_{i_1} - \mathbf{e}_{i_2}\mathbf{a}_{i_1}\mathbf{g} - \mathbf{e}_{i_2}\mathbf{t}_{i_1}\mathbf{f}))\mathbf{t}\right]_q\end{aligned}$$

For these quantities generated from encodings of zero, building a system of equations is possible. In fact, if we define a function $f_{\mathbf{s},\mathbf{t}}(\mathbf{w}_i) = \mathbf{s}^T \cdot Rot(\mathbf{w}_i) \cdot \mathbf{t}$, then integers, such as $u_t$, $\nu_t$, $\theta_{s,t}$, and $\pi_{i_1,i_2}$, are values of $f_{\mathbf{s},\mathbf{t}}$. Given only several values of $f_{\mathbf{s},\mathbf{t}}$, we cannot solve $\mathbf{s}, \mathbf{t}$ when unknown $\mathbf{w}_i$. Currently, we do not find other attack method for the function $f_{\mathbf{s},\mathbf{t}}$.

In addition, we notice the decisional linear (DLIN) and the subgroup membership (SubM) problems are easy in our construction since $\mathbf{M}_t = \mathbf{S}Rot(\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1}$ and $\mathbf{N}_t = \mathbf{T}^{-1}Rot(\mathbf{n}_t\mathbf{q})\mathbf{T}$ are encodings of zero.

### 3.4.2 Attack of Pellet-Mary and Stehlé

The key point of Pellet-Mary and Stehlé attack [PS15] is to find inverse element of an encoding over modulo encoding of zero. That is, one must find $(\mathbf{Y}_i)^{-1} \bmod \mathbf{N}_t$ given the public parameters $\mathrm{par}$. If $\mathbf{h} = 1$ in our scheme, then one can compute $(\mathbf{Y}_i)^{-1} \bmod \mathbf{N}_t$. Without loss of generality, assume that $\det(\mathbf{Y}_i), \det(\mathbf{N}_t)$ are coprime. One first computes the adjacent matrices $\mathbf{Y}_i^*, \mathbf{N}_t^*$ of $\mathbf{Y}_i, \mathbf{N}_t$ such that $\mathbf{Y}_i \times \mathbf{Y}_i^* = \det(\mathbf{Y}_i)\mathbf{I}$, $\mathbf{N}_t \times \mathbf{N}_t^* = \det(\mathbf{N}_t)\mathbf{I}$, where $\mathbf{I}$ is identity matrix. then, one finds two integer $a, b$ using Euclid algorithm such that $a\det(\mathbf{Y}_i) + b\det(\mathbf{N}_t) = 1$. Finally, one inputs inverse element $(\mathbf{Y}_i)^{-1} = (a\mathbf{Y}_i^*) \bmod \mathbf{N}$. However, when $\mathbf{h} \neq 1$, we currently do not know how to compute $(\mathbf{Y}_i^*)^{-1} \bmod \mathbf{N}$. If one uses the above method, then he only obtains $\mathbf{Y}_i \times (a\mathbf{Y}_i^*) + \mathbf{N}_t \times (b\mathbf{N}_t^*) = \det(Rot(\mathbf{h}))\mathbf{I}$. Thus, the Pellet-Mary and Stehlé attack [PS15] does not work in our construction.

### 3.4.3 Lattice Reduction Attack

Given that $\mathbf{M}_t = \mathbf{S}Rot(\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1} \in \mathbb{Z}^{n \times n}$ and $\mathbf{N}_t = \mathbf{T}^{-1}Rot(\mathbf{n}_t\mathbf{q})\mathbf{T} \in \mathbb{Z}^{n \times n}$, we can attempt to use the lattice reduction algorithm to determine the secret elements in our construction. By using $\mathbf{M}_t$ and $\mathbf{N}_t$, we can generate the following lattices:

$$L_1(t_1,t_2)=\begin{pmatrix}\mathbf{M}_{t_1}\\ \mathbf{M}_{t_2}\end{pmatrix},\ L_2(t_1,t_2)=\begin{pmatrix}\mathbf{N}_{t_1}\\ \mathbf{N}_{t_2}\end{pmatrix}.$$

By applying the lattice reduction algorithm [LLL82], we obtain $\mathbf{E}_1 Rot(\mathbf{q})\mathbf{S}^{-1}$ and $\mathbf{E}_2 Rot(\mathbf{q})\mathbf{T}$. However, $\mathbf{E}_1$ and $\mathbf{E}_2$ are not identity matrices according to current lattice reduction algorithm. If $\mathbf{E}_1=\mathbf{E}_2=\mathbf{I}$, we can further derive $Rot(\mathbf{q})$ by using $Rot(\mathbf{q})\mathbf{T}$ and $Rot(\mathbf{q})\mathbf{S}^{-1}$ and finally solve $\mathbf{T}$ and $\mathbf{S}$. When $\mathbf{T}$ and $\mathbf{S}$ are known, our construction can be broken, as will be described in the following subsection. Thus, we must set a sufficiently large dimension for our construction to prevent lattice reduction attack.

#### 3.4.4 Attack of the Principal Ideal Generator

Given that $\mathbf{X}_i=\mathbf{S}Rot(\mathbf{x}_i)\mathbf{S}^{-1}\in\mathbb{Z}^{n\times n}$ and $\mathbf{Y}_i=\mathbf{T}^{-1}Rot(\mathbf{y}_i)\mathbf{T}\in\mathbb{Z}^{n\times n}$, we can compute $p_i=\det(Rot(\mathbf{x}_i))$ and $q_i=\det(Rot(\mathbf{y}_i))$.

Without loss of generality, we assume that $p_1,q_1$ are primes. We first factor $(x^n+1)\bmod p_1$ and derive the $n$ roots of $\alpha_j$ to obtain $\mathbf{x}_1$. We then solve the principal ideal generator of $(p_1,\alpha_j), j\in[\![n]\!]$ by using an arbitrary efficient algorithm (if existing). We assume that the principal ideal generator generated by $(p_1,\alpha_1)$ is $\mathbf{x}_1$. Finally, we derive $\mathbf{S}$ by using $\mathbf{x}_1$. Similarly, we can also obtain $\mathbf{T}$.

If there exists an efficient algorithm that solves the short principal ideal generator, can we prevent this attack? In fact, previous algorithms computing the principal ideal generator do not guarantee that we can derive the original $\mathbf{x}_1$ in $\mathbf{X}_i=\mathbf{S}Rot(\mathbf{x}_i)\mathbf{S}^{-1}$ [CDPR15]. If we can derive the short principal ideal generator $\mathbf{x}$ of $\mathbf{x}_1$ but $\mathbf{x}\neq\mathbf{x}_1$, we can obtain a nontrivial unit element $\boldsymbol{\delta}=\mathbf{x}_1/\mathbf{x}$, that is, $\det(Rot(\boldsymbol{\delta}))=1$. Once we obtain the short nontrivial unit element $\boldsymbol{\delta}$, we can modify our construction, as follows:

$$\mathbf{Y}_i=\mathbf{T}^{-1}Rot(\boldsymbol{\delta}\mathbf{y}_i)\mathbf{T}\ \text{ and }\ \mathbf{X}_i=\mathbf{S}Rot(\boldsymbol{\delta}\mathbf{x}_i)\mathbf{S}^{-1},$$
$$\mathbf{M}_t=\mathbf{S}Rot(\boldsymbol{\delta}\mathbf{m}_t\mathbf{q})\mathbf{S}^{-1}\ \text{ and }\ \mathbf{N}_t=\mathbf{T}^{-1}Rot(\boldsymbol{\delta}\mathbf{n}_t\mathbf{q})\mathbf{T}.$$

In this case, we must exactly solve $\boldsymbol{\delta}\mathbf{y}_i$ and $\boldsymbol{\delta}\mathbf{x}_i$ to obtain $\mathbf{T}$ and $\mathbf{S}$. Otherwise, we cannot derive $\mathbf{T}$ and $\mathbf{S}$. So, this countermeasure improves the security of our construction.

## 4 Variants

### 4.1 Asymmetric Variant

Asymmetric ideal multilinear maps with different groups are required in some applications. Similar to [GGH13], we briefly describe the asymmetric variant as follows:

In this variant, we use different ideal generators $\mathbf{g}_t\leftarrow D_{\mathbb{Z}^n,\mu}, t\in[\![\beta]\!]$ to represent asymmetric maps. An element of the form $(\mathbf{a}_t\cdot\mathbf{g}_t)\bmod\mathbf{f}$ is a level-$1$ encoding corresponding to the $t$-th generator $\mathbf{g}_t$. We denote the index vectors of different levels of encoding. For a level-$0$ encoding $\mathbf{a}$, an encoding $\mathbf{c}$ with an index vector $\mathbf{w}=(w_1,...,w_\beta)\in\mathbb{N}^\beta$ has the form $\mathbf{c}\bmod\mathbf{f}=(\mathbf{a}\prod_{t=1}^{\beta}(\mathbf{g}_t)^{w_t})\bmod\mathbf{f}$.

We select the parameters of the inner layer $\mathbf{x}_{i,t} = \mathbf{h}(\mathbf{a}_{i,t} + \mathbf{s}_{i,t} \cdot \mathbf{f}) \bmod \mathbf{q}$, $\mathbf{y}_{i,t} = \mathbf{h}(\mathbf{a}_{i,t}\mathbf{g}_t + \mathbf{t}_{i,t} \cdot \mathbf{f}) \bmod \mathbf{q}$, and $i \in [\![\tau]\!], t \in [\![\beta]\!]$ and generate the parameters of the outer layer $\mathbf{X}_{i,t} = \mathbf{S}Rot(\mathbf{x}_{i,t})\mathbf{S}^{-1}$ and $\mathbf{Y}_{i,t} = \mathbf{T}^{-1}Rot(\mathbf{y}_{i,t})\mathbf{T}$ to enable encoding in this variant. Thus, the public parameters become $\mathrm{par}_1 = \left\{q, \mathbf{M}, \mathbf{N}, \left\{\mathbf{X}_{i,t}, \mathbf{Y}_{i,t}\right\}_{i\in[\![\tau]\!], t\in[\![\beta]\!]}, \mathbf{P}_{zt}, \mathbf{s}^*, \mathbf{t}^*\right\}$.

### 4.2 Commutative Variant

In the commutative variant, we switch from the integer ring to the polynomial ring to decrease the size of the public parameters. On one hand, the dimension $n$ in our construction must be sufficiently large to guarantee security. On the other hand, the number of level-$1$ encodings in the public parameter $\tau \geq n^2 + O(\lambda)$ should be able to prevent algebraic equation attack. Thus, the size of the public parameters is significantly large to be practical. As such, we use $R^{(y)} = \mathbb{Z}[y] / < y^{n_1} + 1 >$ and $R_q^{(y)} = R^{(y)} / qR^{(y)}$ instead of $\mathbb{Z}$ and $\mathbb{Z}_q$, where $n_1 = O(\lambda)$. In this case, the size of the public parameters is relative practical.

This commutative variant is the same as our construction in Section 3, except all operations are conducted over the ring $R^{(y)}$ and $R_q^{(y)}$. We can easily verify that this variant construction is correct.

### 4.3 Variant without Noise

According to analysis in 3.4.2, we can further construct new variant. Assume that the public parameters $\mathrm{par}_2 = \left\{\mathbf{Y}, \left\{\mathbf{N}_t\right\}_{t\in[\![n]\!]}\right\}$, where $\mathbf{Y} = \mathbf{T}^{-1}Rot(\mathbf{y})\mathbf{T}$ with $\mathbf{y} = (\mathbf{haf}) \bmod \mathbf{q}$, and $\mathbf{N}_t = \mathbf{T}^{-1}Rot(\mathbf{n}_t\mathbf{q})\mathbf{T}$, $t \in [\![n]\!]$. Given a random integer $a$, one generate an encoding $\mathbf{U} = (a\mathbf{Y}) \bmod \mathbf{N}$. This variant can be extended to the commutative variant to improve its efficiency. The security of this variant depends on new hardness assumption. That is, we suppose that computing $\mathbf{Y}^{-1} \bmod \mathbf{N}$ is hard.

Given $k+1$ encodings $\mathbf{U}_j = (a_j\mathbf{Y}) \bmod \mathbf{N}$, it is easy to verify that $\mathbf{C}_t = (a_t \prod_{j\neq t} \mathbf{U}_j) \bmod \mathbf{N}$, $t \in [\![k+1]\!]$ are same. However, only given $\left\{\mathrm{par}_2, \left\{\mathbf{U}_j\right\}_{j\in[\![k+1]\!]}\right\}$, one currently cannot obtain $\mathbf{C}_t$.

## 5 Applications

Using our construction of ideal multilinear maps, we describe two applications: the one-round multipartite Diffie–Hellman key exchange protocol and witness encryption. Their security relies on the hardness assumption of the ideal-ext-GDDH.

### 5.1 Multipartite Key Exchange Protocol

***Setup***$(1^\lambda)$. The output $(\mathrm{par}) \leftarrow \mathrm{InstGen}(1^\lambda)$ is used as the public parameter.

***Publish***$(\mathrm{par}, j)$. We let $N$ be the number of participants. For $j \in [\![N]\!]$, each party $j$

samples random elements $w_{j,i} \leftarrow D_{\mathbb{Z},\sigma'}, i \in [\![\tau]\!]$, which are used as secret keys. Thereafter, level-$1$ encoding $\mathbf{U}_j \leftarrow \mathrm{Enc}\left(\mathrm{par}, 1, \{w_{j,i}\}_{i\in[\![\tau]\!]}\right)$ is computed and published as a public key.

***KeyGen***$(\mathrm{par}, j, \{w_{j,i}\}_{i\in[\![\tau]\!]}, \{\mathbf{U}_j\}_{j\neq i})$. Each party $j$ computes a level-$N-1$ encoding $\mathbf{C}_j = \prod_{k\neq j} \mathbf{U}_k$ and a level-$0$ encoding $\mathbf{D}_j = (\sum_{i=1}^{\tau} w_{j,i} \cdot \mathbf{X}_i) \bmod \mathbf{M}$, and extracts the common secret key $sk = \mathrm{Ext}(\mathrm{par}, \mathbf{D}_j, \mathbf{C}_j)$.

**Remark 5.1** Given that each party merely requires the level-$1$ encoding be generated in the MPKE protocol, the parameters $\mathbf{s}^*$, $\mathbf{X}_i$, $\mathbf{P}_{zt}$ in $\mathrm{par}$ can be combined into a vector $\mathbf{p}_{zt,i} = \left[\mathbf{s}^* \cdot \mathbf{X}_i \cdot \mathbf{P}_{zt}\right]_q$. As such, the public parameters can be $\mathrm{par}_2 = \left\{q, \mathbf{N}, \{\mathbf{p}_{zt,i}, \mathbf{Y}_i\}_{i\in[\![\tau]\!]}, \mathbf{t}^*\right\}$.

**Theorem 5.2** Suppose that the ideal-ext-GDDH is hard. Then our protocol is a one-round multipartite Diffie–Hellman key exchange protocol.

**Proof**. The proof is similar to the proof presented in Theorem 2 in [GGH13]. ■

### 5.2 Witness Encryption

#### 5.2.1 Construction

Let integer $K$ be a multiple of 3. An instance of 3-exact cover problem consists of a number $K$ and a collection $Set$ of subsets $S_1, S_2, ..., S_\beta \subset [\![K]\!]$. The problem is to find a 3-exact cover of $[\![K]\!]$. For an instance of witness encryption, the public key is a collection $Set$ and the public parameters $\mathrm{par}$ in our ideal construction, the secret key is a hidden 3-exact cover of $[\![K]\!]$.

***Encrypt***$(1^\lambda, \mathrm{par}, M)$**:**

(1) For each $k \in [\![K]\!]$, generate a level-$1$ encoding $\mathbf{U}_k = \sum_{i=1}^{\tau} d_{k,i} \mathbf{Y}_i \bmod \mathbf{N}$, where $\mathbf{d}_k \leftarrow D_{Z^\tau,\sigma}$.

(2) Generate an encryption key $sk = \mathrm{Ext}(\mathrm{par}, \mathbf{I}, \mathbf{U})$, where $\mathbf{U} = (\prod_{k=1}^{K} \mathbf{U}_k) \bmod \mathbf{N}$, and encrypt a message $M$ into ciphertext $C$, where $\mathbf{I}$ is the $n \times n$ identity matrix.

(3) For each subset $S_i = \{i_1, i_2, i_3\}$ of $Set$, generate a level-$3$ encoding $\mathbf{U}_{S_i} = (\mathbf{U}_{i_1} \mathbf{U}_{i_2} \mathbf{U}_{i_3}) \bmod \mathbf{N}$.

(4) Output the ciphertext $C$ and all level-$3$ encodings $E = \left(\mathbf{U}_{S_i}, S_i \in Set\right)$.

***Decrypt***$(C, E, W)$:

(1) Given $C$, $E$ and a witness set $W$, compute $\mathbf{U} = (\prod_{S_i \in W} \mathbf{U}_{S_i}) \bmod \mathbf{N}$.

(2) Generate $sk = \mathrm{Ext}(\mathrm{par}, \mathbf{I}, \mathbf{U})$, and decrypt $C$ to get the plaintext $M$.

**Correctness**. The correctness of witness encryption directly follows that of our ideal construction.

**Security**. Similar to [GGSW13], the security of our construction depends on the assumption of the Decision Graded Encoding No-Exact-Cover.

**Theorem 5.3** Suppose that the Decision Graded Encoding No-Exact-Cover is hard. Then our construction is a witness encryption scheme.
**Proof**. The proof is identical to one presented in Theorem 5.2 in [GGSW13]. ■

### 5.2.2 The Hu-Jia Attacks

(1) The Hu-Jia attack in [HJ15b] does not work in our construction. This is because one cannot compute the inverse $(\mathbf{U}_{S_i})^{-1} \bmod \mathbf{N}$ of $\mathbf{U}_{S_i} = (\mathbf{U}_{i_1}\mathbf{U}_{i_2}\mathbf{U}_{i_3}) \bmod \mathbf{N}$. Without loss of generality, assume that $\mathbf{U}_{S_i} = \mathbf{T}^{-1}Rot(\mathbf{b}+\mathbf{r}_1\mathbf{f})\mathbf{T}$. Given $\mathbf{U}_{S_i}$, find $(\mathbf{U}_{S_i})^{-1} \bmod \mathbf{N}$. The problem is to solve a matrix $\mathbf{U} = \mathbf{T}^{-1}Rot(\mathbf{s}+\mathbf{r}_2\mathbf{f})\mathbf{T} \bmod \mathbf{N}$ such that $\mathbf{T}^{-1}Rot(\mathbf{b}+\mathbf{r}_1\mathbf{f})\mathbf{T} \cdot \mathbf{T}^{-1}Rot(\mathbf{s}+\mathbf{r}_2\mathbf{f})\mathbf{T} = \mathbf{T}^{-1}Rot(1+\mathbf{r}_3\mathbf{f})\mathbf{T} \bmod \mathbf{N}$. On the one hand, if considering $\mathbf{U}_{S_i}$ as a vector, and $\mathbf{N}$ as a matrix, one does not know how to compute $\mathbf{U}$. On the other hand, if considering the $t$-th column vector of $\mathbf{N}$ as a matrix $\mathbf{N}_t$, one can generate two matrices $\mathbf{E}_1, \mathbf{E}_2$ such that $\mathbf{E}_1\mathbf{U}_{S_i} + \mathbf{E}_2\mathbf{N}_t = \mathbf{I}$. This is impossible because $\mathbf{U}_{S_i}$ and $\mathbf{N}_t$ are not coprime. Moreover, one cannot also ensure that $\mathbf{E}_1, \mathbf{E}_2$ are of the form $\mathbf{T}^{-1}Rot(\mathbf{r})\mathbf{T}$, where $\mathbf{r} \in R$.

(2) The Hu-Jia attack in [HJ15a] does not apply to our construction. To attack the GGH-based WE [GGH13], Hu and Jia [HJ15a] first generate a combined 3-exact cover $EC$ and compute the collection of positive factors $CPF$ and the collection of negative factors $CNF$ of $EC$ (see [HJ15a]). Then, they compute $\mathbf{v}_{PPF} = \prod_{pf \in CPF} \mathbf{v}'^{(pf)}$ and $\mathbf{v}_{PNF} = \prod_{nf \in CNF} \mathbf{v}'^{(nf)}$, where $\mathbf{v}'^{(pf)}, \mathbf{v}'^{(nf)}$ are equivalent secrets. Finally, they solve $\mathbf{v}_{PTS} = \prod_{k=1}^{K} \mathbf{v}^{(k)}$ by using equation $(\mathbf{v}_{PPF} - \mathbf{v}_{PTS} \times \mathbf{v}_{PNF}) \in < \mathbf{g} >$ and a basis of $\mathbf{g}$.

Now, we adapt their notation to our construction to obtain the following equation

$$(\prod_{pf \in CPF} \mathbf{U}_{(pf)}) \bmod \mathbf{N} = (\prod_{k=1}^{K} \mathbf{U}_k) \bmod \mathbf{N} \times (\prod_{nf \in CNF} \mathbf{U}_{(nf)}) \bmod \mathbf{N},$$

Given $\mathbf{U}_{PPF} = (\prod_{pf \in CPF} \mathbf{U}_{(pf)}) \bmod \mathbf{N}$ and $\mathbf{U}_{PNF} = (\prod_{nf \in CNF} \mathbf{U}_{(nf)}) \bmod \mathbf{N}$, find $\mathbf{U}_{PTS} = (\prod_{k=1}^{K} \mathbf{U}_k) \bmod \mathbf{N}$. Similar as the above (1), one cannot solve the inverse $(\mathbf{U}_{PNF})^{-1} \bmod \mathbf{N}$ of $\mathbf{U}_{PNF}$.

**Remark 5.4** We select an element $\varphi \in [\![K]\!]$ such that $\left|S^{(\varphi)}\right| = \max\left\{\left|S^{(\pi)}\right|, \pi \in [\![K]\!]\right\}$, where $S^{(\pi)} = \{S_i \mid (\pi \in S_i) \bigcap (S_i \in Set)\}$. For $S_i \in S^{(\varphi)}$, we modify $\mathbf{U}_{S_i} = (\mathbf{U}_{i_1}\mathbf{U}_{i_2}\mathbf{U}_{i_3}) \bmod \mathbf{N}$ into $\mathbf{u}_{S_i} = \left[\mathbf{s}^* \cdot \mathbf{I} \cdot \mathbf{P}_{zt} \cdot \mathbf{U}_{S_i}\right]_q$. In this case, we do not increase the size of $q$ since the level-$0$ encoding $\mathbf{I}$ (identity matrix) compensates an increase multiplied by $\mathbf{U}_{S_i}$. When decrypting, we compute the secret key as follows:

$$sk = \mathrm{Extract}_s(\mathrm{msbs}_\gamma(\left[\mathbf{u}_{S_i \in W \cap S_i \in S^{(\varphi)}} \cdot (\prod_{S_i \in W \cap S_i \notin S^{(\varphi)}} \mathbf{U}_{S_i} \bmod \mathbf{N}) \cdot \mathbf{t}^*\right]_q)).$$

Using this countermeasure, we merely increase the difficulty that adversary attacks our witness encryption using a combined 3-exact cover. Since any subset in $S^{(\varphi)}$ cannot be used in any combined subset.

## References


[BF03] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing, SIAM Journal on Computing, 32(3):586–615, 2003.

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

[BWZ14] D. Boneh, D. J. Wu, and J. Zimmerman. Immunizing multilinear maps against zeroizing attacks. http://eprint.iacr.org/2014/930.

[BZ14] D. Boneh and M. Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. CRYPTO 2014, Part I. LNCS 8616, pp. 480–499.

[CDPR15] R. Cramer, L. Ducas, C. Peikertz, and O. Regev. Recovering Short Generators of Principal Ideals in Cyclotomic Rings. http://eprint.iacr.org/2015/313.

[CHL+15] J. H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehle. Cryptanalysis of the Multilinear Map over the Integers. EUROCRYPT 2015, Part I, LNCS 9056, pp. 3–12.

[CN11] Y. Chen and P. Q. Nguyen. BKZ 2.0 Better Lattice Security Estimates, ASIACRYPT 2011, LNCS 7073, pp. 1–20.

[CLT13] J. S. Coron, T. Lepoint, and M. Tibouchi. Practical multilinear maps over the integers. CRYPTO 2013, LNCS 8042, pp. 476–493.

[CLT14] J. S. Coron, T. Lepoint, and M. Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. http://eprint.iacr.org/2014/975.

[CLT15] J. S. Coron, T. Lepoint, and M. Tibouchi. New Multilinear Maps over the Integers. http://eprint.iacr.org/2015/162.

[GGH13] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. EUROCRYPT 2013, LNCS 7881, pp. 1–17.

[GGS+13] S. Garg, C. Gentry, A. Sahai, and B. Waters. Witness encryption and its applications. STOC 2013, pp. 467–476.

[GGH+13a] S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters. Attribute-based encryption for circuits from multilinear maps. Cryptology ePrint Archive, Report 2013/128 (2013)

[GGH+13b] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. FOCS 2013, pp. 40-49.

[GGH15] C. Gentry, S. Gorbunov, and S. Halevi. Graph-induced multilinear maps from lattices. TCC 2015, Part II, LNCS 9015, pp. 498–527.

[GGHZ14] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Fully secure functional encryption without obfuscation. http://eprint.iacr.org/2014/666.

[GGSW13] S. Garg, C. Gentry, A. Sahai, and B. Waters. Witness encryption and its applications. STOC 2013, pp. 467–476.

[GLSW14] C. Gentry, A. Lewko, A. Sahai, and B. Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. http://eprint.iacr.org/2014/309.

[GLW14] C. Gentry, A. Lewko, and B. Waters. Witness Encryption from Instance Independent Assumptions. CRYPTO 2014. LNCS 8616, pp. 426 - 443.

[HJ15a] Yupu Hu and Huiwen Jia. Cryptanalysis of GGH Map. http://eprint.iacr.org/2015/301.

[HJ15b] Yupu Hu and Huiwen Jia. A Comment on Gu Map-1. http://eprint.iacr.org/2015/448.

[HJ15c] Yupu Hu and Huiwen Jia. An Optimization of Gu Map-1. http://eprint.iacr.org/2015/453.

[HPS98] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a ring based public key cryptosystem. ANTS 1998, LNCS 1423, pp. 267-288.

[Jou00] A. Joux. A one round protocol for tripartite Diffie-Hellman. ANTS 2000, LNCS 1838, pp. 385–394.

[LLL82] H.W. Lenstra, A.K. Lenstra and L. Lovasz. Factoring polynomials with rational coefficients. Mathematische Annalen, 1982, 261(4): 515–534.

[LPSS14] S. Ling, D. Phan, D. Stehlé, and R. Steinfeld. Hardness of k-lwe and applications in traitor tracing. CRYPTO 2014, LNCS 8616, pp. 315-334.

[LSS14] A. Langlois, D. Stehlé, and R. Steinfeld. GGHLite: More Efficient Multilinear Maps from Ideal Lattices, EUROCRYPT 2014, LNCS 8441, 2014, pp. 239–256.

[PS15] A. Pellet-Mary and D. Stehlé. Cryptanalysis of Gu's ideal multilinear map. http://eprint.iacr.org/2015/759.

[PTT10] C. Papamanthou, R. Tamassia, and N. Triandopoulos. Optimal authenticated data structures with multilinear forms. Pairing 2010, LNCS 6487, pp. 246–264.

[Rot13] R. Rothblum. On the circular security of bit-encryption. TCC 2013, LNCS 7785, 2013, pp. 579–598.

[RS09] M. Rückert and D. Schröder. Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. ISA 2009, LNCS 5576, pp. 750–759.

[Sma03] N.P Smart. An identity based authenticated key agreement protocol based on the Weil pairing, Electronics Letters, 38(13), pp. 630-632, 2002.

[SOK00] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing, the 2000 Symposium on Cryptography and Information Security, Okinawa, Japan, 2000.

[SS11] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices, EUROCRYPT 2011, LNCS 6632, pp. 27–47.

[Zim15] J. Zimmerman. How to obfuscate programs directly. EUROCRYPT 2015, Part II, LNCS 9057, pp. 439–467.