修士論文

Full Domain Functional Bootstrapping の改良

雨宮 岳

主指導教員 藤﨑 英一郎

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和 7 年 3 月

Abstract

Fully Homomorphic Encryption (FHE), referred to as the "Holy Grail" of cryptography, is an encryption scheme that allows arbitrary computations to be performed on encrypted data. Here, arbitrary computations can be expressed as combinations of basic operations such as AND, OR, and NOT, which in turn can be represented using addition and multiplication. Since FHE allows addition and multiplication on encrypted data, it enables the computation of arbitrary operations.

This property makes it promising for applications in privacy-preserving computing, such as performing machine learning on encrypted data or executing computations in outsourced computing protocols while keeping the client' s data encrypted.

However, repeated homomorphic operations cause noise to accumulate in the ciphertext, leading to decryption failure. Therefore, FHE requires an operation called Bootstrapping to reduce noise. In conventional schemes, Bootstrapping could not be performed within a practical time frame, making it the biggest challenge in achieving practical FHE. Since Gentry first realized FHE in 2009, various schemes have been proposed and improved, but practical implementation has yet to be achieved.

Recently, a type of FHE called Torus Fully Homomorphic Encryption (TFHE), proposed by Chillotti et al., has achieved fast execution of 1-bit Gate Bootstrapping in approximately 10 milliseconds. Furthermore, an extension of Bootstrapping in TFHE, known as Functional Bootstrapping, exists. Functional Bootstrapping not only reduces noise but also enables the evaluation of arbitrary functions using a Look-Up Table. As a result, it is more efficient to evaluate nonlinear functions directly using Functional Bootstrapping to process a Look-Up Table rather than approximating them by combining basic operations through Gate Bootstrapping.

However, in TFHE's Functional Bootstrapping, the domain of the function must be restricted to evaluate an arbitrary function. Full Domain Functional Bootstrapping (FDFB) is an approach that eliminates this limitation, allowing the function' s domain to cover the entire plaintext space of the ciphertext. Existing FDFB methods required at least two Blind Rotations, two Sample Extractions, and two Key Switchings. In this study, we propose a method that achieves FDFB using only two Blind Rotations and two simple Sample Extractions.

目次

図目次

表目次

第 1 章

はじめに

完全準同型暗号 (Fully Homomorphic Encryption) はデータを暗号化したまま任意の演算を計算できる暗号である．ここで任意の演算は基本演算 AND,OR,NOT の組み合わせで表現でき，その基本演算は加算と乗算の組み合わせで表現できる．FHE はデータを暗号化したまま加算と乗算ができるため，任意の演算を計算できる．

しかし準同型演算を繰り返し行うことで，暗号文にノイズが蓄積し復号に失敗する．そのため FHE は Bootstrapping[Gen09] と呼ばれるノイズを小さくする操作を必要とする．従来の方式では Bootstrapping は現実的な時間で行うことができず，実用化のボトルネックになっていた．2009 年に Gentry に初めて FHE が実現してからこれまでにさまざまな方式が考案され改良が重ねられているが，いまだ実用化には至っていない．

近年，Chillotti 等により提案された FHE の一種であるトーラス型完全準同型暗号 (TFHE) [CGGI20] は 1bit の Gate Bootstrapping を 10ms 程度で実行できる．そして，TFHE の Bootstrapping には Functional Bootstrapping という拡張が存在する．Functional Bootstrapping はノイズを小さくすると同時に任意の関数を評価できる．これにより，非線形関数などは Gate Bootstrapping を組み合わせて近似するよりも Functional Bootstrapping で直接評価する方が効率的になる．

しかし TFHE の Functional Bootstrapping で任意の関数を評価するためには，関数の定義域を制限する必要がある．その制限を解消した，暗号文の平文空間全体を定義域とするような Functional Bootstrapping が Full Domain Functional Bootstrapping(FDFB) である．これまで提案されていた FDFB は少なくとも 2 回の Blind Rotation，2 回の Sample Extraction, 2 回の Key Switching を必要としていたが, 本研究では 2 回の Blind Rotation と 2 回の簡単な Sample Extraction のみで FDFB を実現する方式を提案する．

第 2 章

準備

本章では提案方式の説明に必要な記法と背景知識について説明する.

2.1 記法

本論文の記法は [Joy22] を参考にしている.

記号	意味
$\lfloor x \rfloor$	実数 x 以下の最大の整数
$\lceil x \rceil$	実数 x 以上の最小の整数
$\lfloor x \rceil$	実数 x に最も近い整数
	$\lfloor x \rceil = \begin{cases} \lfloor x + 0.5 \rfloor & \text{for } x \geq 0 \\ \lceil x - 0.5 \rceil & \text{for } x < 0 \end{cases}$
$\lfloor p(X) \rceil$	実数係数多項式 $p(X)$ の係数を最も近い整数に変換した整数係数多項式
$x \xleftarrow{\$} D$	集合 D から x を一様ランダムに抽出
$x \leftarrow \chi$	x を離散ガウス分布 $\mathcal{N}(0, \sigma^2)$ に従って抽出
$\mathbb{T}$	実トーラス，$\mathbb{T} = \mathbb{R}/\mathbb{Z}$
$\mathbb{T}_N[X]$	実トーラス多項式，$\mathbb{T}_N[X] = \mathbb{T}[X]/(X^N + 1)$
$\mathbb{T}_q$	離散トーラス，$\mathbb{T}_q = \frac{1}{q}\mathbb{Z}/q\mathbb{Z}$
$\mathbb{T}_{N,q}[X]$	離散トーラス多項式，$\mathbb{T}_{N,q}[X] = \mathbb{T}_q[X]/(X^N + 1)$
$\mathbb{Z}$	整数の集合
$\mathbb{Z}_p$	$\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$
$\mathbb{R}$	実数の集合
$\mathbb{Z}_N[X]$	$\mathbb{Z}_N[X] = \mathbb{Z}[X]/(X^N + 1)$
$\mathbb{B}$	$\mathbb{B} = \{0, 1\}$
$\mathbb{B}_N[X]$	$\mathbb{B}_N[X] = \mathbb{B}[X]/(X^N + 1)$

2.2 TFHE スキーム

本節では TFHE の構成に必要な暗号とその性質について説明する．[Joy22, Kle23] を参考にした．

2.2.1 トーラス

$\mathbb{T} = \mathbb{R}/\mathbb{Z}$ であり，$\forall t_1, t_2 \in \mathbb{T}$ に対して $t_1 + t_2 \mod 1 \in \mathbb{T}$ が成り立つ．加法に関して可換群である．

TFHE スキームで用いる離散化されたトーラスは $q \in \mathbb{Z}$ として

$$\begin{aligned}\mathbb{T}_q &= \frac{1}{q}\mathbb{Z}/q\mathbb{Z} \subset \mathbb{T} \\ &= \left\{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\right\}\end{aligned}$$

である．$\forall t_1, t_2 \in \mathbb{T}_q$ に対して $t_1 + t_2 \mod 1 \in \mathbb{T}_q$ が成り立ち，加法に関して可換群である．また，$p \in \mathbb{Z}$ が $p|q$ を満たすならば

$$\mathbb{T}_p \subset \mathbb{T}_q$$

が成り立つ．

2.2.2 トーラス多項式

係数がトーラスの元であるような多項式の集合である．$q \in \mathbb{Z}$，$\mathbb{Z}_N[X] = \mathbb{Z}[X]/(X^N + 1)$，$\mathbb{R}_N[X] = \mathbb{R}[X]/(X^N + 1)$ とし

$$\mathbb{T}_N[X] = \mathbb{R}_N[X]/\mathbb{Z}_N[X] = \mathbb{T}[X]/(X^N + 1)$$

TFHE スキームで用いる離散化されたトーラス多項式は，$\mathbb{Z}_{N,q}[X] = \mathbb{Z}_q[X]/(X^N + 1)$，$\mathbb{R}_{N,q}[X] = \mathbb{R}_q[X]/(X^N + 1)$ として

$$\mathbb{T}_{N,q}[X] = \mathbb{R}_{N,q}[X]/\mathbb{Z}_{N,q}[X] = \mathbb{T}_q[X]/(X^N + 1)$$

また，$p \in \mathbb{Z}$ が $p|q$ を満たすならば

$$\mathbb{T}_{N,p}[X] \in \mathbb{T}_{N,q}[X]$$

が成り立つ．

2.2.3 TLWE

離散トーラス $\mathbb{T}_q$ とその部分群を $\mathbb{T}_p$ とする．χ は $\mathbb{Z}_q$ 上の離散ガウス分布とする．

■鍵生成 $\forall i \in [1, n]$ に対し，

- $a_i \xleftarrow{\$} \mathbb{T}_q$
- $s_i \xleftarrow{\$} \mathbb{B}$
- $\bar{e} \leftarrow \chi$

とし，秘密鍵 $\mathbf{s} = (s_1, \ldots, s_n)$ とする．

■暗号化　マスクベクトル $\mathbf{a} = (a_1, \ldots, a_n)$ とする．メッセージ $m \in \mathbb{Z}_p$，$p \mid q$ として平文 $\mu = \frac{m}{p} \in \mathbb{T}_p$ とする．$e = \frac{\bar{e}}{q}$ とする．$b = \langle \mathbf{a}, \mathbf{s} \rangle + \mu + e$ とし

$$\mathbf{c} = (\mathbf{a}, b) = \mathsf{TLWE}_{\mathbf{s}}(\mu) \in \mathbb{T}_q^{n+1}$$

を暗号文として出力する．

■復号　$\mathbf{s}$ を用いて $b = \langle \mathbf{a}, \mathbf{s} \rangle + \mu + e$ から

$$\mu^* = b - \sum_{i=0}^{n-1} a_i \cdot s_i$$

を計算する．そして

$$\mu = \frac{\lfloor p\mu^* \rceil \mod p}{p}$$

を平文として出力する．

また，μ を平文とする TLWE 暗号文 $\mathsf{TLWE}_{\mathbf{s}}(\mu)$ と小さい $k \in \mathbb{Z}$ に対して

$$\mathsf{TLWE}_{\mathbf{s}}(\mu_1) + \mathsf{TLWE}_{\mathbf{s}}(\mu_2) = \mathsf{TLWE}_{\mathbf{s}}(\mu_1 + \mu_2)$$
$$k \cdot \mathsf{TLWE}_{\mathbf{s}}(\mu) = \mathsf{TLWE}_{\mathbf{s}}(k \cdot \mu)$$

が成り立つ．

2.2.4　TGLWE

離散トーラス多項式 $\mathbb{T}_{N.q}[X]$ とその部分群を $\mathbb{T}_{N,p}[X]$ とする．変数 X は $X^N \equiv -1 \mod X^N + 1$ を満たす．χ は $\mathbb{Z}_{N,q}[X]$ 上の離散ガウス分布とする．

■鍵生成　$\forall i \in [1, n]$ に対し，

- $\mathfrak{a}_i \xleftarrow{\$} \mathbb{T}_{N,q}[X]$
- $\mathfrak{s}_i \xleftarrow{\$} \mathbb{B}_N[X]$
- $\bar{\mathfrak{e}} \leftarrow \chi$

とし，秘密鍵 $\mathfrak{s} = (\mathfrak{s}_1, \ldots, \mathfrak{s}_k)$ とする．

■暗号化　マスクベクトル $(\mathfrak{a}_1, \ldots, \mathfrak{a}_n)$ とする．メッセージ $\mathfrak{m} \in \mathbb{Z}_{N,p}[X]$，$p \mid q$ として平文 $\mu(X) = \frac{1}{p}\mathfrak{m} \in \mathbb{T}_{N,p}[X]$ とする．$\mathfrak{e} = \frac{\bar{\mathfrak{e}}}{q}$ とする．$\mathfrak{b} = \sum_{i=1}^{k} \mathfrak{a}_i \cdot \mathfrak{s}_i + \mu(X) + \mathfrak{e}$ とし

$$\mathfrak{c} = (\mathfrak{a}_1, \ldots, \mathfrak{a}_k, \mathfrak{b}) = \mathsf{TGLWE}_{\mathfrak{s}}(\mu(X)) \in \mathbb{T}_{N,q}[X]^{k+1}$$

を暗号文として出力する．

■復号　$\mathfrak{s}$ を用いて $\mathfrak{b} = \sum_{i=1}^{k} \mathfrak{a}_i \cdot \mathfrak{s}_i + \mu(X) + \mathfrak{e}$ から

$$\mu^*(X) = \mathfrak{b} - \sum_{i=1}^{k} \mathfrak{a}_i \cdot \mathfrak{s}_i$$

を計算する．そして

$$\mu(X) = \frac{\lfloor p\mu^*(X) \rceil \mod p}{p}$$

を平文として出力する．

また，$\mu(X)$ を平文とする TGLWE 暗号文 $\mathsf{TGLWE}_{\mathfrak{s}}(\mu(X))$ と小さい $\mathfrak{k} \in \mathbb{Z}_N[X]$ に対して

$$\begin{aligned}
&\mathsf{TGLWE}_{\mathfrak{s}}(\mu_1(X)) + \mathsf{TGLWE}_{\mathfrak{s}}(\mu_2(X)) \\
&= \mathsf{TGLWE}_{\mathfrak{s}}(\mu_1(X) + \mu_2(X)) \\
&\mathfrak{k} \cdot \mathsf{TGLWE}_{\mathfrak{s}}(\mu(X)) \\
&= \mathsf{TGLWE}_{\mathfrak{s}}(\mathfrak{k} \cdot \mu(X))
\end{aligned}$$

が成り立つ．

2.2.5　ガジェット分解

$B, l \in \mathbb{Z}^*$ が，$B^l \mid q$ を満たすとする．ガジェットベクトルを

$$\mathbf{g} = (\frac{1}{B}, \ldots, \frac{1}{B^l}) \in \mathbb{T}_q^l$$

と定義する．このとき $\mathbf{g}$ は

$$\begin{aligned}
\mathbf{g} : \mathbb{Z}_{N,q}[X]^l &\to \mathbb{T}_{N.q}[X] \\
(\mathfrak{v}_1 \ldots, \mathfrak{v}_l) &\mapsto \langle (\mathfrak{v}_1 \ldots, \mathfrak{v}_l), \mathbf{g} \rangle
\end{aligned}$$

である．ここで多項式 $\mathfrak{v}$ の l 桁 B 進展開を

$$\begin{aligned}\bar{\mathfrak{v}} &= \lfloor B^l \cdot \mathfrak{v} \rceil \\ &\equiv \sum_{j=1}^{l} u_j B^{l-j} \pmod{B^l}\end{aligned}$$

として，$\mathfrak{v}$ に対するガジェットベクトルの逆変換 $g^{-1}(\mathfrak{v})$ は，

$$g^{-1}(\mathfrak{v}) = (u_1, \ldots, u_l) \in \mathbb{Z}_{N,q}[X]$$

とする．このとき

$$g^{-1}(\mathfrak{v})\mathbf{g}^\top \approx \mathfrak{v}$$

が成り立つ．この操作で得られた $\mathfrak{v}$ は本来の $\mathfrak{v}$ を l 桁 B 進展開を計算し，本来の $\mathfrak{v}$ の l 桁目以降を切り捨てているため，近似値となる．仮に $B^l = q$ であれば $g^{-1}(\mathfrak{v})\mathbf{g}^\top = \mathfrak{v}$ である．

ガジェット行列を

$$\mathbf{G}^\top = \begin{pmatrix} \mathbf{g}^\top & & & \\ & \mathbf{g}^\top & & \\ & & \ddots & \\ & & & \mathbf{g}^\top \end{pmatrix} \in \mathbb{T}_{q,}^{(k+1)l \times (k+1)}$$

と定義する．多項式を要素とするベクトル $\vec{\mathfrak{v}} = (\mathfrak{v}_1, \ldots, \mathfrak{v}_{k+1})$ に対する逆変換は

$$\begin{aligned}G^{-1}(\vec{\mathfrak{v}}) &= (g^{-1}(\mathfrak{v}_1), \ldots, g^{-1}(\mathfrak{v}_{k+1})) \\ &= (u_{1,1}, \ldots, u_{1,l}, \ldots, u_{k+1.l}, \ldots, u_{k+1,l})\end{aligned}$$

とする．このとき

$$G^{-1}(\vec{\mathfrak{v}})\mathbf{G}^\top \approx \vec{\mathfrak{v}}$$

が成り立つ．

2.2.6 TGGSW

$$\mathcal{Z} = \begin{pmatrix} \mathsf{TGLWE}_s(0) \\ \mathsf{TGLWE}_s(0) \\ \vdots \\ \mathsf{TGLWE}_s(0) \end{pmatrix} \in \mathbb{Z}_N[X]^{(k+1)l}$$

とする．平文を $\mathfrak{m} \in \mathbb{Z}_N[X]$ とする TGGSW の暗号文 $\mathcal{C}$ は

$$\mathcal{C} = \mathcal{Z} + \mathfrak{m} \cdot G^{\top}$$

である．

TGLWE 暗号文と TGGSW 暗号文で External Product と呼ばれる以下のような準同型演算を実現できる．

$$\begin{array}{rccl} \boxdot : & \mathsf{TGGSW}_{\mathfrak{s}}(\mathfrak{m}) \times \mathsf{TGLWE}_{\mathfrak{s}}(\mu(X)) & \to & \mathsf{TGLWE}_{\mathfrak{s}}(\mathfrak{m} \cdot \mu(X)) \\ & (\mathcal{C}, \mathfrak{c}) & \mapsto & \mathcal{C} \boxdot \mathfrak{c} = G^{-1}(\mathfrak{c}) \cdot \mathcal{C} \end{array}$$

2.3 Bootstrapping

TFHE の Bootstrapping は，Blind Rotation，Sample Extraction，Key Switching の 3 ステップを行う．

2.3.1 Blind Rotation

TFHE の Bootstrapping は，Blind Rotation アルゴリズムを通してノイズを小さくする．入力の TLWE 暗号文に含まれているノイズは Blind Rotation を正しく実行できるかに影響する．基本的なアイデアとしては，テスト多項式 $v(X) = \sum_{i=0}^{N-1} v_i \cdot X^i$ に対して $X^{-j} \cdot v(X)$ を計算し

$$X^{-j} \cdot v(X) = \begin{cases} v_j + \cdots & j \in [0, N) \\ -v_{N+j} + \cdots & j \in [-N, 0) \end{cases} \tag{2.1}$$

という多項式に変換する．

暗号文に対しては，Bootstrap Key $\mathsf{BK}_i \leftarrow \mathsf{TGGSW}_{\mathfrak{s}}(s_i)$ を生成し，$\tilde{a} = \lfloor 2N \cdot a \rceil \mod 2N$，$\tilde{b} = \lfloor 2N \cdot b \rceil \mod 2N$ を計算する．これは $X^{2N} = 1$ であるため指数の部分も $2N$ まで定義する必要があるためである．$i \in [0, N-1]$ に対して $s_i = 1$ ならばテスト多項式に $X^{\tilde{a}_i}$ をかけ，$s_i = 0$ ならばかけない，という操作を繰り返し，$X^{-\tilde{b}+\sum_{i=0}^{N-1} a_i \cdot s_i} \cdot v(X)$ を平文とした TGLWE 暗号文に変換する．

$\mathsf{BlindRotate}(\mathbf{BK}, \mathsf{TLWE}_{\mathfrak{s}}(\mu), \mathsf{TGLWE}_{\mathfrak{s}}(v(X)))$ を Blind Rotation アルゴリズムを行い，$\mathsf{TGLWE}_{\mathfrak{s}}(X^{-\tilde{\mu}^*} \cdot v(X))$ を出力する関数と定義する．

Algorithm 1 BlindRotate

Input: TLWE sample$(a_1, \ldots, a_n, b) \in \mathbb{T}_q^{n+1}$

TGLWE sample $\mathsf{TGLWE}_{\mathfrak{s}}(v(X)) \in \mathbb{T}_{N,p}[X]$

Bootstrap Keys $\mathbf{BK} = (\mathsf{BK}_1, \ldots, \mathsf{BK}_n) \in (\mathbb{T}_{N,q}[X]^{(k+1)l \times (k+1)})^n$

Output: $\mathsf{TGLWE}_{\mathfrak{s}}(X^{-\tilde{\mu}^*} \cdot v(X))$

1: $\tilde{a}_i \leftarrow \lfloor 2N \cdot a_i \rceil$

2: $\tilde{b} \leftarrow \lfloor 2N \cdot b \rceil$

3: $\mathsf{ACC} \leftarrow X^{-\tilde{b}} \cdot \mathsf{TGLWE}_{\mathfrak{s}}(v(X))$

4: **for** $i \in [1, n]$ **do**

5: $\quad \mathsf{ACC} \leftarrow \mathsf{ACC} + \mathsf{BK}_i \boxdot (X^{\tilde{a_i}} \cdot \mathsf{ACC} - \mathsf{ACC})$

6: **end for**

7: **return** ACC

2.3.2 Sample Extraction

Blind Rotation が出力する TGLWE 暗号文の平文の定数項を暗号化した TLWE 暗号文をつくる．Blind Rotation で得た TGLWE 暗号文は

$$\begin{aligned}&(\mathfrak{a}_1, \ldots, \mathfrak{a}_k, \sum_{i=1}^{k} \mathfrak{a}_i \cdot \mathfrak{s}_i + \mu(X) + \mathfrak{e})\\ =&(\mathfrak{a}_1, \ldots, \mathfrak{a}_k, \mathfrak{b}) \in \mathbb{T}_{N,q}[X]\end{aligned}$$

このとき，$\mathfrak{a}_i = \sum_{j=0}^{N-1} (a_i)_j \cdot X^j$，$\mathfrak{s}_i = \sum_{j=0}^{N-1} (s_i)_j \cdot X^j$ とすると

$$\begin{aligned}\mathfrak{b} = \sum_{i=1}^{k} &((s_j)_0 + \cdots + (s_j)_{N-1} X^{N-1}) \cdot\\ &((a_j)_0 + \cdots + (a_j)_{N-1} X^{N-1}) + \mu(X) + \mathfrak{e}\end{aligned}$$

であり，$\mathfrak{b} = b_0 + \cdots + b_{N-1} X^{N-1}$ としたとき b_0 は

$$\begin{aligned}&\sum_{j=1}^{k} ((s_j)_0 \cdot (a_j)_0 - (s_j)_1 \cdot (a_j)_{N-1} - \cdots - (s_j)_{N-1} \cdot (a_j)_1)\\ &\quad + \mu + e_0\\ =&((s_1)_0, (s_1)_1, \ldots, (s_1)_{N-1}, \ldots, (s_k)_0, \ldots, (s_k)_{N-1}) \cdot\\ &((a_1)_0, -(a_1)_{N-1}, \ldots, -(a_1)_1, \ldots, -(a_k)_1) + \mu + e_0\end{aligned}$$

と表される．

$$\mathbf{s}' = ((s_1)_0, (s_1)_1, \ldots, (s_1)_{N-1}, \ldots, (s_k)_0, \ldots, (s_k)_{N-1})$$
$$\mathbf{a}' = ((a_1)_0, -(a_1)_{N-1}, \ldots, -(a_1)_1, \ldots, -(a_k)_1)$$

として，Sample Extraction は

$$\mathsf{TLWE}_{\mathbf{s}'}(\mu) = (\mathbf{a}', b_0) \in \mathbb{T}_q^{kN+1}$$

を出力する．

$\mathsf{SampleExtract}(\mathsf{TGLWE}_{\mathfrak{s}}(v_0 + \cdots))$ を Sample Extraction アルゴリズムを行い，$\mathsf{TLWE}_{\mathbf{s}'}(v_0)$ を出力する関数と定義する．

2.3.3 Key Switching

Sample Extraction が出力する TLWE 暗号文 $\mathsf{TLWE}_{\mathbf{s}'}(\mu) \in \mathbb{T}_q^{kN+1}$ を $\mathsf{TLWE}_{\mathbf{s}}(\mu) \in \mathbb{T}_q^{n+1}$ に変換する．Sample Extraction で定義した $\mathbf{s}', \mathbf{a}'$ を

$$\mathbf{s}' = (s'_1, \ldots s'_{kN}) \in \mathbb{B}^{kN}$$
$$\mathbf{a}' = (a'_1, \ldots, a'_{kN}) \in \mathbb{T}_q^{kN}$$

とする．そして Key Switching Key を

$$\mathsf{KSK}_{i,j} = \mathsf{TLWE}_{\mathbf{s}}(s'_i \cdot B^{-j}) \quad (i \in [1, kN], j \in [1, l])$$

と定義する．またガジェットベクトルを $\mathbf{g} = (1/B, \ldots, 1/B^l)$ とし，

$$\mathbf{g}^{-1}(a'_i) = ((\bar{a'_i})_1 \ldots, (\bar{a'_i})_l)$$

と定義する．Key Switching は

$$(\mathbf{0}, b_0) - \sum_{i=1}^{kN} \sum_{j=1}^{l} (\bar{a'_i})_j \cdot \mathsf{KSK}_{i,j}$$

を計算し，$k+1$ 次元の TLWE 暗号文を出力する．しかし，正確には $\mathbf{g}^{-1}(a'_i)\mathbf{g} = \sum_{j=1}^{l} (\bar{a_i})_j \cdot B^{-j} = a'_i + \epsilon_i$ となるような丸め誤差が生じる．この誤差を考慮すると，Key Switching は $\mathsf{TLWE}_{\mathbf{s}}(\mu)$ でノイズが $e_0 + \sum_{i=1}^{kN} (s'\epsilon_i)$ である TLWE 暗号文を出力する．

$\mathsf{KeySwitch}(\mathsf{TLWE}_{\mathbf{s}'}(v_0))$ を Key Switching アルゴリズムを行い，$\mathsf{TLWE}_{\mathbf{s}}(v_0)$ を出力する関数と定義する．

NOT	$(\mathbf{0}, \frac{1}{4}) - \mathbf{c}$
AND	$(\mathbf{0}, -\frac{1}{8}) + \mathbf{c}_1 + \mathbf{c}_2$
NAND	$(\mathbf{0}, \frac{5}{8}) - \mathbf{c}_1 - \mathbf{c}_2$
OR	$(\mathbf{0}, \frac{1}{8}) + \mathbf{c}_1 + \mathbf{c}_2$
XOR	$2 \cdot (\mathbf{c}_1 + \mathbf{c}_2)$

表 2.1　準同型論理回路

2.4　Gate Bootstrapping

$f : \{0,1\}^2 \to \{0,1\}$ の任意の論理回路 f を評価する Bootstrapping である．テスト多項式を

$$\mathfrak{v} = -\frac{1}{8}\left(\sum_{i=0}^{\frac{N}{2}-1} X^i - \sum_{i=\frac{N}{2}}^{N-1} X^i\right)$$

とし，$\mathbf{c} = \mathsf{TLWE}_{\mathbf{s}}(\mu), \mathbf{BK} = (\mathsf{BK}_1, \ldots, \mathsf{BK}_n), \mathsf{BK}_i = \mathsf{TGGSW}_{\mathfrak{s}}(s_i)$ を用いて $\mathsf{BlindRotate}(\mathfrak{v}, \mathbf{c}, \mathbf{BK})$ を計算する．このとき，回路の入力 $\mu_1, \mu_2 \in \{0, \frac{1}{4}\}$ を暗号化した TLWE 暗号文 $\mathbf{c}_1, \mathbf{c}_2$ を用いて表 2.1 のように $\mathbf{c}$ を設定することで任意の論理回路を計算できる．

Blind Rotation は平文について，ノイズが十分小さいと仮定すれば

$$X^{-\tilde{\mu^*}} \cdot \mathfrak{v} = X^{-2N \cdot k} \cdot \mathfrak{v}$$

を計算しており，このとき k は $\mathbf{c} = \mathsf{TLWE}_{\mathbf{s}}(k)$ を満たす．例えば NAND 回路を評価したいとき表 2.1 より $k = \frac{5}{8} - \mu_1 - \mu_2$ である．ここで $\mu_1, \mu_2 \in \{0, \frac{1}{4}\}$ であるから

$$2N \cdot k = \begin{cases} \frac{5}{4}N & (\mu_1 = \mu_2 = 0) \\ \frac{3}{4}N & (\mu_1 = \frac{1}{4}, \mu_2 = 0 \text{ or } \mu_1 = 0, \mu_2 = \frac{1}{4}) \\ \frac{1}{4}N & (\mu_1 = \mu_2 = \frac{1}{4}) \end{cases} \tag{2.2}$$

が成り立つ．$X^{-2N \cdot k} \cdot \mathfrak{v}$ を計算するときの X の指数と $\mathfrak{v}$ の係数は図 2.1 のように対応している．図 2.1 と (2.2) から得られる係数と，μ_1, μ_2 を入力とする NAND の出力は一致している．

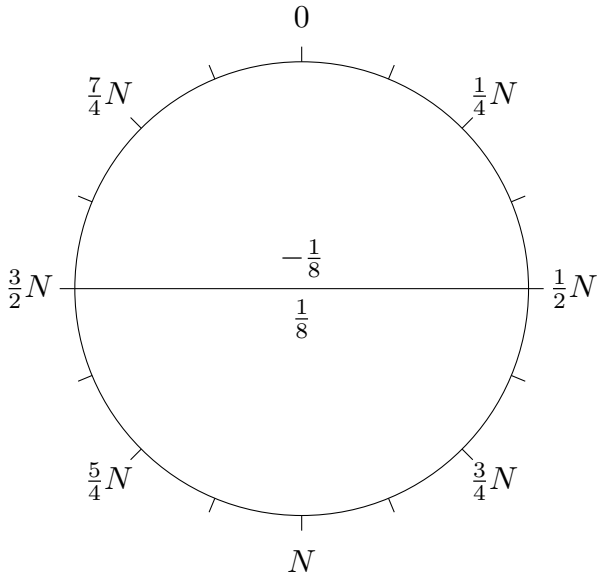

図 2.1　Gate Bootstraping における指数と係数の対応

2.5　Functional Bootstrapping

Blind Rotation は，$v(X) = \sum_{i=0}^{N-1} v_i \cdot X^i$，平文 $\mu \in \mathbb{T}_p$ として $X^{-\tilde{\mu}^*} \cdot v(X)$ を暗号文のまま計算し，(2.1) を平文とする TGLWE 暗号文を得る．ここで

$$\tilde{\mu}^* \in [0, N) \iff \mu \in [0, \frac{1}{2}) \cap \mathbb{T}_p$$
$$\tilde{\mu}^* \in [-N, 0) \iff \mu \in [-\frac{1}{2}, 0) \cap \mathbb{T}_p$$

であるから，テスト多項式の係数をそのまま定数項として得たいときは，平文 μ は $\mu \in [0, \frac{1}{2}) \cap \mathbb{T}_p$ である必要がある．

この性質を踏まえて，テスト多項式を

$$v(X) = \sum_{i=0}^{N-1} f(\frac{\lfloor \frac{p}{2N} i \rceil \mod p}{p}) \cdot X^i$$

とし，平文 μ を $\mu \in [0, \frac{1}{2}) \cap \mathbb{T}_p$ とすることで関数を評価する．

$p = 4$ のとき，Functional Bootstrapping の指数 ($\lfloor 2N \cdot \mu^* \rceil$) とテスト多項式の係数の対応は図 2.2 のようになる．

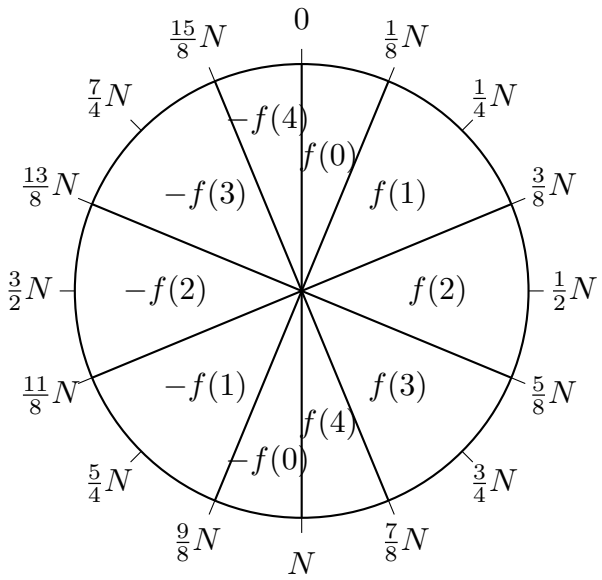

図 2.2　Functional Bootstraping ($p = 4$) における指数と係数の対応

第 3 章

提案方式

3.1 提案方式のメインアイデア

提案方式のメインのアイデアは TLWE 暗号文を TGLWE 暗号文とみなし，TLWE 暗号文に整数係数多項式をかけて TGLWE 暗号文を得ることと, TGSW 暗号文を TGGSW 暗号文とみなし，Blind Rotation を行うことである．どちらも多項式同士の計算から，多項式と単項式の乗算に変換している．

TLWE 暗号文 $\mathsf{TLWE}_{\mathbf{s}}(\mu) = (a_1, \ldots, a_k, \langle \mathbf{s}, \mathbf{a} \rangle + \mu + e)$ に対し整数係数多項式 $v(X) \in \mathbb{Z}_{N,q}[X]$ をかけると

$$
\begin{aligned}
&(a_1, \ldots, a_k, \langle \mathbf{s}, \mathbf{a} \rangle + \mu + e) \cdot v(X) \\
=&(a_1 \cdot v(X), \ldots, a_k \cdot v(X), \langle \mathbf{s}, \mathbf{a} \cdot v(X) \rangle + \mu \cdot v(X) + e \cdot v(X)) \\
=&\mathsf{TGLWE}_{\mathbf{s}}(\mu \cdot v(X))
\end{aligned}
$$

が得られる．

また，秘密鍵が多項式でない TGLWE 暗号文に対して Sample Extraction を行うとき

$$
\begin{aligned}
&\langle \mathbf{s}, (\mathfrak{a}_1, \ldots, \mathfrak{a}_k) \rangle \\
=&(s_1 \cdot (a_1)_1 + \cdots + s_k \cdot (a_k)_1) + \cdots + (s_1 \cdot (a_1)_{N-1} + \cdots + s_k \cdot (a_k)_{N-1}) \cdot X^{N-1} \\
=&(s_1, \ldots, s_k) \cdot ((a_1)_1, \ldots, (a_k)_1) + \cdots
\end{aligned}
$$

となる．このように簡単な Sample Extraction を行うことができ，Key Switching が不要になる．

次に，TGSW 暗号文を TGGSW 暗号文とみなして Blind Rotation を行うことを考える．Blind Rotation はサブルーチンとして TGLWE 暗号文と TGGSW 暗号文の

External Product を行うが，TGGSW 暗号文を TGSW 暗号文に置き換えたとき以下のような計算となる．$i \in [1, k]$ に対して

$$
\begin{aligned}
&\mathsf{TGSW}_{\mathbf{s}}(s_i) \boxdot \mathsf{TGLWE}(v(X)) \\
&= G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot \mathsf{TGSW}_{\mathbf{s}}(s_i) \\
&= G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot \left(\begin{pmatrix} \vdots \\ \mathsf{TLWE}_{\mathbf{s}}(0) \\ \vdots \end{pmatrix} + s_i \cdot \mathbf{G}^{\top} \right) \\
&= G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot \begin{pmatrix} \vdots \\ \mathsf{TLWE}_{\mathbf{s}}(0) \\ \vdots \end{pmatrix} + G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot s_i \cdot \mathbf{G}^{\top}
\end{aligned}
$$

ここで，$G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \in \mathbb{Z}_{N,q}[X]^{l(k+1)}$ であるから

$$
G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot \begin{pmatrix} \vdots \\ \mathsf{TLWE}_{\mathbf{s}}(0) \\ \vdots \end{pmatrix} = \mathsf{TGLWE}_{\mathbf{s}}(0)
$$

また $G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot \mathbf{G}^{\top} \approx \mathsf{TGLWE}_{\mathbf{s}}(v(X))$ であるから以下が成り立つ．

$$
G^{-1}(\mathsf{TGLWE}_{\mathbf{s}}(v(X))) \cdot s_i \cdot \mathbf{G}^{\top} \approx s_i \cdot \mathsf{TGLWE}_{\mathbf{s}}(v(X))
$$

したがって，TGSW 暗号文を TGGSW 暗号文とみなした場合でも External Product を計算できるため，Blind Rotation が問題なくできる．

3.2 提案方式の詳細

提案方式では，FDFB を 2 回の Blind Rotation，2 回の簡単な Sample Extraction，0 回の Key Switching で実現する．

TFHE で任意の Full Domain Function h を評価するために，任意の関数 $f : [0, \frac{1}{2}) \cap \mathbb{T}_p \to \mathbb{T}_p$，$g : [-\frac{1}{2}, 0) \cap \mathbb{T}_p \to \mathbb{T}_p$ を用いて h を以下のように定義する．

$$
h(x) \coloneqq \begin{cases} f(x) & x \in [0, \frac{1}{2}) \cap \mathbb{T}_p \\ g(x) & x \in [-\frac{1}{2}, 0) \cap \mathbb{T}_p \end{cases}
$$

ここで $\mathsf{sgn}(x) \coloneqq \begin{cases} 1 & x \in [0, \frac{1}{2}) \cap \mathbb{T}_p \\ -1 & x \in [-\frac{1}{2}, 0) \cap \mathbb{T}_p \end{cases}$ とすると $h(x)$ は以下のように表せる.

$$h(x) = \frac{1 + \mathsf{sgn}(x)}{2} f(x) + \frac{1 - \mathsf{sgn}(x)}{2} g(x)$$

Blind Rotation で関数 h を評価するために, 平文 $\mu \in \mathbb{T}_p$, 関数 f, g を係数にマップした多項式 $v_f(X) = \sum_{i=0}^{N-1} v_{f,i} X^i, v_g(X) = \sum_{i=0}^{N-1} v_{g,i} X^i \in \mathbb{T}_{N,p}[X]$, 秘密鍵 $\mathbf{s} = (s_1, \ldots, s_k)$, Bootstrap Key $\mathbf{BK} = (\mathsf{BK}_1, \ldots, \mathsf{BK}_k), \mathsf{BK} = \mathsf{TGSW}_{\mathbf{s}}(s_i)$ を用いて

$$\begin{aligned} &\mathsf{BlindRotate}(\mathbf{BK}, \mathsf{TLWE}_{\mathbf{s}}(\mu), \mathsf{TGLWE}_{\mathbf{s}}(v_h(X)) \\ =&\mathsf{TGLWE}_{\mathbf{s}}(h(\mu) + \cdots) \end{aligned}$$

を計算する.

そのために, $\mathsf{TGLWE}_{\mathbf{s}}(v_h(X))$ を計算する必要がある. まず, $\tau = \mathsf{min}\{\tau \in \mathbb{Z}_p \mid \tau \cdot v_f(X), \tau \cdot v_g(X) \in \mathbb{Z}_{N,p}[X]\}$ を用いて

$$\begin{aligned} &\mathsf{BlindRotate}(\mathbf{BK}, \mathsf{TLWE}_{\mathbf{s}}(\mu), \frac{1}{2\tau} \sum_{i=0}^{N-1} X^i) \\ =&\mathsf{TGLWE}_{\mathbf{s}}(X^{-\tilde{\mu}^*} \cdot \frac{1}{2\tau} \sum_{i=0}^{N-1} X^i) \\ =&\begin{cases} \mathsf{TGLWE}_{\mathbf{s}}(\frac{1}{2\tau} + \cdots) & \mu \in [0, \frac{1}{2}) \cap \mathbb{T}_p \\ \mathsf{TGLWE}_{\mathbf{s}}(-\frac{1}{2\tau} + \cdots) & \mu \in [-\frac{1}{2}, 0) \cap \mathbb{T}_p \end{cases} \end{aligned}$$

を計算し, $\mathsf{TGLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2\tau} + \cdots)$ を得る.

秘密鍵が定数の TGLWE 暗号文であるから Sample Extraction より $\mathsf{TLWE}_{\mathbf{s}}(\mu) = ((a_1)_1, \ldots, (a_k)_1, \sum_{i=1}^{k} s_i \cdot (a_i)_1 + \mu + e)$ を得る.

したがって, $\mathsf{TGLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2\tau} + \cdots)$ に対して Sample Extraction を行い, $\mathsf{TLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2\tau})$ を得る.

次に, $\mathsf{TLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2\tau})$ を TGLWE 暗号文とみなし, 整数係数多項式 $\tau \cdot v_f(X), \tau \cdot v_g(X)$ をかけて以下を得る.

$$\mathsf{TGLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2} v_f(X))$$
$$\mathsf{TGLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2} v_g(X))$$

そして

$$(\mathbf{0}, \frac{v_f(X)}{2}) + \mathsf{TGLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2} v_f(X)) = \mathsf{TGLWE}_{\mathbf{s}}(\frac{1+\mathsf{sgn}(\mu)}{2} v_f(X))$$
$$(\mathbf{0}, \frac{v_g(X)}{2}) - \mathsf{TGLWE}_{\mathbf{s}}(\frac{\mathsf{sgn}(\mu)}{2} v_g(X)) = \mathsf{TGLWE}_{\mathbf{s}}(\frac{1-\mathsf{sgn}(\mu)}{2} v_g(X))$$

を計算する．したがって

$$\mathsf{TGLWE}_{\mathbf{s}}(v_h(X)) = \mathsf{TGLWE}_{\mathbf{s}}(\frac{1+\mathsf{sgn}(\mu)}{2} v_f(X) + \frac{1-\mathsf{sgn}(\mu)}{2} v_g(X))$$

を得る．

最後に

$$\begin{aligned} &\mathsf{BlindRotate}(\mathbf{BK}, \mathsf{TLWE}_{\mathbf{s}}(\mu), \mathsf{TGLWE}_{\mathbf{s}}(v_h(X)) \\ =&\mathsf{TGLWE}_{\mathbf{s}}(h(\mu) + \cdots) \end{aligned}$$

を計算し，Sample Extraction を行なって

$$\mathsf{TLWE}_{\mathbf{s}}(h(\mu)) = \begin{cases} f(\mu) & \mu \in [0, \frac{1}{2}) \cap \mathbb{T}_p \\ g(\mu) & \mu \in [-\frac{1}{2}, 0) \cap \mathbb{T}_p \end{cases}$$

を得る．したがって 2 回の Blind Rotation，2 回の簡単な Sample Extraction，0 回の Key Switching で FDFB を実現できる．

3.3 提案方式の安全性

提案方式では秘密鍵 $\mathbf{s}$ は TGSW 暗号文で，平文 μ は TLWE 暗号文で常に秘匿されている．そして，あとの操作はすべて公開のものであるため秘密鍵と平文が漏洩することはない．

Algorithm 2 提案方式

Input: $\mathsf{TLWE}_{\mathbf{s}}(\mu) = (a_1, \ldots, a_k, b = \langle \mathbf{s}, \mathbf{a} \rangle + \mu + e) \in \mathbb{T}_q^{n+1}$

$v_f(X) \in \mathbb{T}_{N,p}[X]$, $v_g(X) \in \mathbb{T}_{N,p}[X]$

$\tau = \mathsf{min}\{\tau \in \mathbb{Z}_p \mid \tau \cdot v_f(X), \tau \cdot v_g(X) \in \mathbb{Z}_{N,p}[X]\}$

$\mathbf{BK} = (\mathsf{BK}_1, \ldots, \mathsf{BK}_k) \in (\mathbb{T}_{,q}^{(k+1)l \times (k+1)})^n$

$\mathsf{BK}_i = \mathsf{TGSW}_{\mathbf{s}}(s_i)$

Output: $\begin{cases} \mathsf{TLWE}_{\mathbf{s}}(f(m)) & \text{for } m \in [0, N) \\ \mathsf{TLWE}_{\mathbf{s}}(g(m)) & \text{for } m \in [-N, 0) \end{cases}$

1: $\mathfrak{c}_0 \leftarrow \mathsf{BlindRotate}(\frac{1}{2\tau} \sum_{i=0}^{N-1} X^i, \mathsf{TLWE}_{\mathbf{s}}(\mu), \mathbf{BK})$

2: $\mathbf{c} \leftarrow \mathsf{SampleExtract}(\mathfrak{c}_0)$

3: $\mathfrak{c}_f \leftarrow \mathbf{c} \cdot \tau \cdot v_f(X), \mathfrak{c}_g \leftarrow \mathbf{c} \cdot \tau \cdot v_g(X)$

4: $\mathfrak{c}_h \leftarrow (\mathbf{0}, \frac{v_f(X)}{2}) + \mathfrak{c}_f + (\mathbf{0}, \frac{v_g(X)}{2}) - \mathfrak{c}_g$

5: $\mathfrak{c} \leftarrow \mathsf{BlindRotate}(\mathfrak{c}_h, \mathsf{TLWE}_{\mathbf{s}}(\mu), \mathbf{BK})$

6: **return** $\mathsf{SampleExtract}(\mathfrak{c})$

第 4 章

比較

本章では提案手法と既存の FDFB の比較を行う．[CZB+22] では FDFB の手法の提案に加え，代表的な方式が紹介されている．

■FDFB[KS21]　平文の正負により評価する関数を切り替える．提案方式との差異は Key Switching の回数である．

■TOTA[YXS+21]　平文の正負によって生じる nega-cyclic 性 (2.1) を排除する．

■Comp，CMV[CZB+22]　Comp は，関数を奇関数と偶関数の和で表せることを利用する．CMV は Multi-Value Bootstrapping[CIM19] を用いた Comp である．

方式	BlindRotate(回)	KeySwitch(回)
[KS21]	2	2
[YXS+21]	2	2
Comp[CZB+22]	4	4
CMV[CZB+22]	3	3
提案方式	2	0

表 4.1　比較

第 5 章

おわりに

TLWE 暗号文, TGSW 暗号文をそれぞれ TGLWE 暗号文, TGGSW 暗号文とみなすことで FDFB の構成法を改良し，2 回の Blind Rotation，2 回の簡単な Sample Extraction で実現した．これにより，Key Switching によるノイズの増加を解消し，多項式を単項式に置き換えていることによる効率化を実現した．ただ，Bootstrapping アルゴリズムにおける計算時間のボトルネックは Blind Rotation であり，その回数は同じであることから劇的な効率化は実現できていないと考えられる．

また本論文では Blind Rotation1 回で FDFB を構成することはできなかったが，仮に Blind Rotation せずに TGLWE 暗号を TGGSW 暗号に変換することができれば，Blind Rotation1 回で FDFB を構成できる．

謝辞

本研究を進めるにあたり，主指導教員である藤﨑先生には多くの助言をいただきました．心より感謝申し上げます．

また，沼畑さん，川野さん，長谷川さんをはじめとする研究室メンバーには研究室内外で大変お世話になりました．心より感謝申し上げます．

参考文献

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

[CIM19] Sergiu Carpov, Malika Izabachène, and Victor Mollimard. New techniques for multi-value input homomorphic evaluation and applications. In *Topics in Cryptology–CT-RSA 2019: The Cryptographers' Track at the RSA Conference 2019, San Francisco, CA, USA, March 4–8, 2019, Proceedings*, pages 106–126. Springer, 2019.

[CZB+22] Pierre-Emmanuel Clet, Martin Zuber, Aymen Boudguiga, Renaud Sirdey, and Cédric Gouy-Pailler. Putting up the swiss army knife of homomorphic calculations by means of tfhe functional bootstrapping. *Cryptology ePrint Archive*, 2022.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.

[Joy22] Marc Joye. Sok: Fully homomorphic encryption over the [discretized] torus. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 661–692, 2022.

[Kle23] Jakub Klemsa. Hitchhiker's guide to the tfhe scheme. *Journal of Cryptographic Engineering*, 2023.

[KS21] Kamil Kluczniak and Leonard Schild. FDFB: full domain functional bootstrapping towards practical fully homomorphic encryption. *IACR Cryptol. ePrint Arch.*, page 1135, 2021.

[YXS+21] Zhaomin Yang, Xiang Xie, Huajie Shen, Shiying Chen, and Jun Zhou.

Tota: fully homomorphic encryption with smaller parameters and stronger security. *Cryptology ePrint Archive*, 2021.