SCIS 2026 2026 Symposium on
Cryptography and Information Security
Hakodate, Japan, Jan. 26 – 30, 2026
The Institute of Electronics,
Information and Communication Engineers

準同型暗号 TFHE 方式における複数鍵整数型への拡張
Extension of the TFHE Scheme to a Multi-Key Integer-Wise Scheme

北代 雄大 * 服部 大地 † 若杉 飛鳥 † 小寺 雄太 * 野上 保之 *
Yuto Kitadai Daichi Hattori Asuka Wakasugi Yuta Kodeta Yasuyuki Nogami

あらまし 準同型暗号は暗号化状態でのデータ処理を可能にし，プライバシー保護とデータ活用を両立する暗号方式である．完全準同型暗号の一種である TFHE 方式は高速なブートストラップ処理が特徴だが，原論文では単一鍵方式でバイナリ平文を対象とする．複数鍵 TFHE 方式とは，複数組織間での協調計算を可能にする TFHE 方式であるが，平文空間はバイナリに限定される．また，整数型 TFHE 方式とは，平文として整数を対象とする TFHE 方式であるが，単一鍵方式である．本稿では，整数型 TFHE の関数型ブートストラッピングを複数鍵設定に適用した整数型複数鍵 TFHE 方式を提案する．さらに，複数鍵化に伴うノイズ増加を理論的に解析し，ノイズ条件を満たすための最適なパラメータ設計指針を提供する．最後に，本方式での関数型ブートストラッピングの実装を行い，4 パーティで 2 ビット整数の協調計算が実現可能であることを実証した．

キーワード 準同型暗号，TFHE，整数型 TFHE，複数鍵 TFHE

1 背景

クラウドコンピューティングの普及により，クラウド上の豊富な計算資源を用いたデータ処理を行う需要が増している．一方で，医療記録や金融取引データといった機密性の高い情報を外部のサーバー上で処理する場合，従来の技術では平文の状態での処理が必要となるため，セキュリティ上の懸念が存在する．そのような課題に対し，準同型暗号は，データ活用とプライバシー保護を両立する技術として注目されている．特に，完全準同型暗号は，暗号化されたデータを復号することなく任意の計算を実行できる暗号技術であり，秘密計算を実現する有力な手法である．

完全準同型暗号において，ブートストラッピングと呼ばれる処理を実行する必要があるが，この処理は計算コストが高く，準同型暗号を社会実装する上で最大の課題となっている．TFHE 方式はブートストラッピングを最も高速に実行できる準同型暗号方式であり，BFV 方式 [4, 9]，BGV 方式 [5]，CKKS 方式 [7] などの他方式と比べて数十から数百倍速くブートストラッピングを実行できる．TFHE の原論文 [8] では平文としてビットのみを扱う方式として設計されたが，数値計算や機械学習への応用を可能にするために，平文空間を整数へ拡張した整数型 TFHE 方式 [3] が提案された．整数型 TFHE は，整数演算を高速に実行できるように設計されている一方で，単一鍵方式として構成されている．ここで単一鍵方式とは，単一の秘密鍵および公開鍵のもとで暗号化・復号・計算を行う構成である．

一方で，実際の多くのアプリケーション，例えば組織間における医療記録や金融取引データの共同分析では，複数の独立した当事者が自身のデータを直接共有せずに協調的な計算を行うことが求められる．よって，単一鍵方式で協調計算を実現する場合，全ての暗号文が同一の秘密鍵で復号されるため，秘密鍵を所有する任意の参加者が他者の暗号文を復号できてしまうという問題が生じる．複数鍵準同型暗号は，この課題の解決策の 1 つであり，異なる鍵による暗号文に対して準同型演算を実行できる．特に Chen ら [6] は，TFHE を複数鍵方式に拡張した複数鍵 TFHE 方式を提案し，異なる鍵で暗号化されたデータに対する協調的な論理演算を可能にした．

以上の議論から，整数型 TFHE は整数演算を高速に実行するよう設計されている一方で単一鍵方式のみに限定されている．また，複数鍵 TFHE はマルチパーティ計算を可能とするもののビット単位の演算しか扱えない．本稿では，複数の当事者が整数データの暗号文に対

* 岡山大学大学院 環境生命自然科学研究科, 〒 700-8530 岡山県岡山市北区津島中 3-1-1. Graduate School of Environmental, Life, Natural Science and Technology, Okayama University, 3-1-1 Tsushima-Naka, Kita-ku, Okayama-shi, Okayama 700-8530, Japan.
† EAGLYS 株式会社, 〒 151-0051 東京都渋谷区千駄ヶ谷 5 丁目 27-3 やまとビル 7F. EAGLYS Inc., 7F Yamato building, 5-27-3 Sendagaya, Shibuya-ku, Tokyo, 151-0051, Japan.

して高速に協調計算を行うことを可能にするため，整数型 TFHE を複数鍵方式へ拡張した新しい暗号方式を提案する．

1.1 貢献

本稿では，整数型 TFHE 方式と複数鍵 TFHE 方式を統合した新しい暗号方式である整数型複数鍵 TFHE 方式を提案する．提案方式は，複数鍵設定で整数演算を実行できる点で，単一鍵整数型 TFHE およびバイナリ平文のみを扱う複数鍵 TFHE の双方の限界を克服する．具体的には，整数型 TFHE の関数型ブートストラッピングを複数鍵設定に適用し，加算，定数乗算および定数除算などの整数演算を複数鍵環境で 構成可能にする手法を示す．加えて，複数鍵化に伴うノイズ増加について理論的解析を行い，ブートストラッピング後のノイズがノイズ条件を満たすための最適パラメータ設計指針を提供する．そして，110 ビットセキュリティのパラメタ下で，関数型ブートストラッピングの実装による動作検証を通じて，4 パーティに対して 2 ビット整数の協調計算が実現可能であることを示す．最後に，パーティ数 k と許容整数パラメータ τ の間に存在するトレードオフ関係を定量化し，提案方式の実現可能性と実用的な制約を明らかにする．

1.2 関連研究

整数型 TFHE は，Bourse ら [3] によって提案された．Okada ら [13] は関数型ブートストラッピングによる除算および等価性テストを 4 ビット整数に対して 1 秒未満で実現した．Morimura ら [12] は，暗号文での比較演算と除算演算を改良することで，Okada らの高速化に貢献した．

また, 複数鍵準同型暗号は, López-Alt ら [11] によって NTRU に基づく方式として提案された．複数鍵 TFHE は，Chen ら [6] によって提案され，複数鍵準同型暗号の最初の実用的な実装を提供した．しかし，ハイブリッド積の計算コストがパーティ数の二乗に比例するという課題があった．この課題に対して，Kwak ら [10] は，BlindRotation アルゴリズムを 2 段階に分割し，並列化と単一鍵 TFHE との互換性を実現することで，ハイブリッド積の計算コストを削減した．さらに Klemsa ら [1] は，RLWE 鍵の加算により，Kwak らを 4.5 倍から 6.9 倍上回る性能を達成した．

1.3 構成

本稿の構成は以下の通りである．2 章では，本稿で使用する表記および TFHE 方式について概説する．3 章では，整数型 TFHE 方式および複数鍵 TFHE 方式について説明する．4 章では，整数型複数鍵 TFHE の構成とブートストラッピング処理によって生じるノイズに対して，その分散を解析する．5 章では，実装による性能評価を示し，パラメータ空間における実行可能範囲と制約を明らかにする．

2 TFHE 方式

本章では，本稿で使用する表記および TFHE 方式について概説する．

2.1 表記

本稿では，実数全体の集合，整数全体の集合，およびバイナリ集合 $\{0,1\}$ をそれぞれ $\mathbb{R}$，$\mathbb{Z}$，$\mathbb{B}$ と表記する．トーラス $\mathbb{T}=\mathbb{R}/\mathbb{Z}$ は実数の小数部分全体の集合を表し，加法についてアーベル群をなす．a,b をそれぞれベクトルとするとき，ベクトルの内積を $\langle a,b\rangle$ で表す．

セキュリティパラメータ λ に基づき，多項式の次数 N を 2 の冪乗として定める．整数係数多項式環，トーラス係数多項式環，バイナリ係数多項式環をそれぞれ $\mathbb{Z}_N[X]=\mathbb{Z}[X]/(X^N+1)$，$\mathbb{T}_N[X]=\mathbb{T}[X]/(X^N+1)$，$\mathbb{B}_N[X]=\mathbb{B}[X]/(X^N+1)$ と表す．ノイズ分布 χ からのサンプリングを $e \xleftarrow{\$} \chi$，一様分布からのサンプリングを $u \xleftarrow{\$} U(\cdot)$ と記述する．また，四捨五入，切り捨て，切り上げによって整数に丸め込む操作をそれぞれ $\lfloor\cdot\rceil$，$\lfloor\cdot\rfloor$，$\lceil\cdot\rceil$ と表す．

2.2 TLWE (Torus LWE)

TLWE 暗号方式は LWE 暗号方式をトーラス上に拡張したものである．

暗号化: $\mathsf{TLWE.Enc}_s(\mu)$　TLWE 暗号化アルゴリズムは，平文 $\mu\in\mathbb{T}$ に対し，秘密鍵 $s\in\mathbb{B}^n$ を用いて TLWE 暗号文 $c\in\mathbb{T}^{n+1}$ を生成する．セキュリティパラメータ λ により定まる次元数 n とノイズ分布 χ を用いて，乱数 $a \xleftarrow{\$} \mathbb{T}^n$，ノイズ $e \xleftarrow{\$} \chi$ に対し，暗号文 $c=(b,a)$ を $b=\langle a,s\rangle+\mu+e$ で定める．

復号: $\mathsf{TLWE.Dec}_s(c)$　TLWE 復号アルゴリズムは，TLWE 暗号文 $c=(b,a)$ に対し, 秘密鍵 s を用いて平文 μ を計算する．具体的には, 位相 $\varphi_s(c)=b-\langle a,s\rangle=\mu+e$ を計算する．ノイズ e が十分に小さい場合，この位相を最も近い有効な平文へ丸めることで，平文 μ を正しく復元できる．

2.3 TRLWE (Torus Ring LWE)

TRLWE 暗号方式は TLWE 暗号方式を多項式環上に拡張したものである．

暗号化: $\mathsf{TRLWE.Enc}_z(\mu(X))$　TRLWE 暗号化アルゴリズムは，平文 $\mu(X) \in \mathbb{T}_N[X]$ に対し，秘密鍵 $z(X) \in \mathbb{B}_N[X]$ を用いて TRLWE 暗号文 $c \in \mathbb{T}_N[X]^2$ を生成する．セキュリティパラメータ λ により定まる次数 N とノイズ分布 χ を用いて，多項式 $a(X) \xleftarrow{\$} \mathbb{T}_N[X]$，ノイズ $e(X) \xleftarrow{\$} \chi$ に対し，暗号文 $c = (b(X), a(X))$ を $b(X) = a(X) \cdot z(X) + \mu(X) + e(X)$ で定める．

復号: $\mathsf{TRLWE.Dec}_z(c)$　TRLWE 復号アルゴリズムは，暗号文 $c = (b(X), a(X))$ に対し，秘密鍵 $z(X)$ を用いて平文 $\mu(X)$ を計算する．具体的には，位相 $\varphi_z(c) = b(X) - a(X) \cdot z(X) = \mu(X) + e(X)$ を計算する．ノイズ $e(X)$ の各係数が十分に小さい場合，この位相を最も近い有効な平文へ丸めることで，平文 $\mu(X)$ を正しく復元できる．

2.4　ガジェット分解

ガジェットベクトル $g \in \mathbb{T}^\ell$ とは，基数 $B_g \in \mathbb{Z}$ および分解長 $\ell \in \mathbb{Z}$ をパラメータとし，$g = (1/B_g, 1/B_g^2, \ldots, 1/B_g^\ell)$ で定義される列ベクトルである．ガジェット分解とは，値 $v \in \mathbb{T}$ に対して，基数 B_g で近似的に分解する操作であり，出力を $g^{-1}(v) \in \mathbb{Z}^\ell$ と表記する．$g^{-1}(v)$ は $\langle g^{-1}(v), g \rangle \approx v$ を満たし，各成分は $(-B_g/2,\ B_g/2]$ の範囲に収まる．

2.5　ブートストラッピング

外部積，SampleExtraction および KeySwitching の詳細は [8] を参照されたい．

外部積: $\mathsf{TGSW.ExtProd}(C, c)$　外部積 (External Product) は，TGSW 暗号文 $C = \mathsf{TGSW.Enc}_z(m(X))$ と TRLWE 暗号文 $c = \mathsf{TRLWE.Enc}_z(\mu(X))$ を受け取って準同型乗算を行い，TRLWE 暗号文 c' を返す．

CMUX: $\mathsf{TFHE.CMUX}(C, c_0, c_1)$　CMUX は，平文 $m \in \mathbb{B}$ に対する TGSW 暗号文 $C = \mathsf{TGSW.Enc}_z(m)$ と 2 つの TRLWE 暗号文 $c_0 = \mathsf{TRLWE.Enc}_z(\mu_0(X)), c_1 = \mathsf{TRLWE.Enc}_z(\mu_1(X))$ を受け取り，TRLWE 暗号文 $c' = \mathsf{TGSW.ExtProd}(C, c_1 - c_0) + c_0$ を出力する．出力 c' は，平文が $\mu_m(X)$ であるような TRLWE 暗号文となる．

BlindRotation: $\mathsf{TFHE.BR}(c, v(X), \mathrm{BK})$　BlindRotation は，TLWE 暗号文 $c = (b, a) \in \mathbb{T}^{n+1}$，ルックアップテーブル $v(X) \in \mathbb{T}_N[X]$，およびブートストラッピング鍵 $\mathrm{BK} = \{\mathsf{TGSW.Enc}_z(s_i)\}_{i=1}^n$ を受け取り，TRLWE 暗号文 $\mathsf{TRLWE.Enc}_z(v(X) \cdot X^{-\lfloor 2N \cdot \varphi_s(c) \rceil})$ を返す．このアルゴリズムの疑似コードを Algorithm 1 に示す．1-3 行目で TLWE 暗号文の各要素を $2N$ 倍して四捨五入し，整数環 $\mathbb{Z}_{2N}$ 上の元にスケーリングする．続いて，4 行目でアキュムレータ $\mathrm{ACC} \in \mathbb{T}_N[X]^2$ を $(v(X) \cdot X^{-\tilde{b}}, 0)$ で初期化し，5-6 行目で ACC を更新する．

Algorithm 1 TFHE Blind Rotation

Input: $c = \mathsf{TLWE.Enc}_s(\mu) = (b, a)$, $v(X)$, $\mathrm{BK} = \{\mathsf{TGSW.Enc}_z(s_i)\}_{i=1}^n$
Output: $\mathsf{TRLWE.Enc}_z(v(X) \cdot X^{-\lfloor 2N \cdot \varphi_s(c) \rceil})$

1: $\tilde{b} \leftarrow \lfloor 2N \cdot b \rceil \pmod{2N}$
2: **for** $i = 1$ to n **do**
3: 　$\tilde{a}_i \leftarrow \lfloor 2N \cdot a_i \rceil \pmod{2N}$
4: $\mathrm{ACC} \leftarrow (v(X) \cdot X^{-\tilde{b}}, 0)$
5: **for** $i = 1$ to n **do**
6: 　$\mathrm{ACC} \leftarrow \mathsf{TFHE.CMUX}(\mathrm{BK}_i, \mathrm{ACC}, \mathrm{ACC} \cdot X^{\tilde{a}_i})$
7: **return** ACC

Algorithm 2 TFHE Bootstrapping

Input: $c = \mathsf{TLWE.Enc}_s(\mu)$, $v(X) \in \mathbb{T}_N[X]$, BK, KSK
Output: $\mathsf{TLWE.Enc}_s(\mu)$

1: $\mathrm{ACC} \leftarrow \mathsf{TFHE.BR}(c, v(X), \mathrm{BK})$
2: $c_{\mathrm{extracted}} \leftarrow \mathsf{TFHE.SE}(\mathrm{ACC}, 0)$
3: $c_{\mathrm{out}} \leftarrow \mathsf{TFHE.KS}(c_{\mathrm{extracted}}, \mathrm{KSK})$
4: **return** c_{out}

SampleExtraction: $\mathsf{TFHE.SE}(c, k)$　SampleExtraction は，TRLWE 暗号文 $c = \mathsf{TRLWE.Enc}_z(\mu(X)) = (b(X), a(X))$ と整数 k を受け取り，TLWE 暗号文 $c' = (b', a') \in \mathbb{T}^{N+1}$ を返す．

KeySwitching: $\mathsf{TFHE.KS}(c, \mathrm{KSK})$　KeySwitching は TLWE 暗号文 $c = \mathsf{TLWE.Enc}_z(\mu)$ と鍵切替鍵 KSK を受け取り，TLWE 暗号文 $c' = \mathsf{TLWE.Enc}_s(\mu)$ を返す．

Bootstrapping: $\mathsf{TFHE.BS}(c, v(X), \mathrm{BK}, \mathrm{KSK})$
Bootstrapping は，TLWE 暗号文 c，ルックアップテーブル $v(X)$，ブートストラッピング鍵 BK，鍵切替鍵 KSK を受け取り，$v(X)$ を評価した TLWE 暗号文を返す．このアルゴリズムの疑似コードを Algorithm 2 に示す．1 行目で，入力暗号文 c が持つ位相情報 $\varphi_s(c)$ に応じてルックアップテーブル $v(X)$ を回転させ，目的となる値を定数項へ移動させる．続いて，2-3 行目において，その定数項を平文に持ち，入力暗号文と同様の形式を持つ TLWE 暗号文を生成する．

3　TFHE 方式の拡張

本章では TFHE 方式の変種である整数型 TFHE 方式および複数鍵 TFHE 方式について説明する．

3.1 整数型 TFHE 方式 (IWTFHE)

本節では，Okada ら [13] によって一般化された整数型 TFHE 方式について概説する．

3.1.1 パラメータと暗号方式

整数型 TFHE では，パラメータ $\tau \in \mathbb{N}$ を導入し，平文空間 $\mathcal{M} = \{-\tau, \ldots, \tau-1\} \subset \mathbb{Z}$，スケーリング因子 $\Delta = 1/(2\tau)$ を用いる．ここで，平文空間 $\mathcal{M}$ のビット数は $\lceil \log_2(\tau) \rceil$ ビットとして扱う．

暗号化: $\mathsf{IWTFHE.Enc}_s(m)$ 整数型 TFHE における暗号化アルゴリズムは，入力として整数平文 $m \in \mathcal{M}$ を受け取り，秘密鍵 s を用いて TLWE 暗号文 $c = \mathsf{TLWE.Enc}_s(m \cdot \Delta)$ を生成する．

復号: $\mathsf{IWTFHE.Dec}_s(c)$ 整数型 TFHE における復号アルゴリズムは，入力として TLWE 暗号文 c を受け取り，秘密鍵 s を用いて整数平文 m を計算する．具体的には，$m = \lfloor \varphi_s(c)/\Delta \rceil = \lfloor m + e/\Delta \rceil$ を計算し，ノイズ条件 $|e| < 1/(4\tau)$ が満たされる場合，平文 m を正しく復元できる．

3.1.2 関数型ブートストラッピング

関数型ブートストラッピングアルゴリズムは, TFHE 方式のブートストラッピングアルゴリズムを整数型 TFHE 方式へ拡張したものである．このアルゴリズムは，暗号文に対して任意の関数を適用できる．

ルックアップテーブルの構成: $\mathsf{IWTFHE.GenLUT}(f)$ ルックアップテーブル構成アルゴリズムは, 関数 $f : \{0, \ldots, \tau-1\} \to \mathcal{M}$ を受け取り, ルックアップテーブル $v(X) \in \mathbb{T}_N[X]$ を生成する．$v(X) = \mu_0 + \mu_1 X + \cdots + \mu_{N-1} X^{N-1} \in \mathbb{T}_N[X]$ の各係数の設定方法を以下で説明する．円分多項式の性質 $X^{-i} \equiv -X^{N-i}$ を用いて，次のように $v(X)$ を変形する:

$$\begin{aligned} v(X) = &-\mu_{\lceil (-\tau-1/2)N/2\tau \rceil} X^{\lceil (-\tau-1/2)N/2\tau \rceil} \\ &- \cdots - \mu_{-1} X^{-1} + \mu_0 + \mu_1 X + \cdots \\ &+ \mu_{\lceil (\tau-1/2)N/2\tau \rceil} X^{\lceil (\tau-1/2)N/2\tau \rceil} \end{aligned}$$

BlindRotation において, 平文 $m \in \mathcal{M}$ を持つ TLWE 暗号文 c に対する丸め後の位相は $\tilde{\varphi} = \lfloor 2N \cdot \varphi_s(c) \rceil = \lfloor 2N(e + m \cdot \Delta) \rceil$ と表される．ここで，ノイズ条件 $|e| < 1/(4\tau)$ より，$-N/(2\tau) < 2Ne < N/(2\tau)$ が成立する．この条件と $\Delta = 1/(2\tau)$ を用いると，$\lceil (m-1/2)N/\tau \rceil \le \tilde{\varphi} \le \lfloor (m+1/2)N/\tau \rfloor$ が成り立つ．BlindRotation の出力である暗号文に対応する平文は，多項式 $v(X) \cdot X^{-\tilde{\varphi}}$ の定数項, つまり, $\mu_{\tilde{\varphi}}$ となる．以上より，任意の $-(\tau - 1/2)N/2\tau \le i \le (\tau-1/2)N/2\tau$ に対して，ある $m \in \mathcal{M}$ が存在し，$\lceil (m-1/2)N/\tau \rceil \le i \le \lfloor (m+1/2)N/\tau \rfloor$ であるとき，$\mu_i = f(m) \cdot \Delta$ と定める．

Algorithm 3 IWTFHE Functional Bootstrapping

Input: $c = \mathsf{TLWE.Enc}_s(\mu)$, f, BK, KSK
Output: $\mathsf{TLWE.Enc}_s(f(\mu))$
1: $v(X) \leftarrow \mathsf{IWTFHE.GenLUT}(f)$
2: $c_{\mathrm{eval}} \leftarrow \mathsf{TFHE.BS}(c, v(X), \mathrm{BK}, \mathrm{KSK})$
3: **return** c_{eval}

FunctionalBootstrapping: $\mathsf{IWTFHE.FBS}(c, f, \mathrm{BK}, \mathrm{KSK})$ FunctionalBootstrapping は, TLWE 暗号文 $c = \mathsf{IWTLWE.Enc}(m)$, 任意の関数 $f : \{0, \ldots, \tau-1\} \to \mathcal{M}$, ブートストラッピング鍵 BK，鍵切替鍵 KSK を受け取り，TLWE 暗号文 $c_{\mathrm{eval}} = \mathsf{IWTLWE.Enc}(f(m))$ を返す．このアルゴリズムの疑似コードを Algorithm 3 に示す．1 行目で, 関数 f を使用してルックアップテーブル $v(X)$ の係数を適切に設定する．続いて，2 行目において，関数 f を評価した TLWE 暗号文 c_{eval} を作成する．

3.1.3 関数型ブートストラッピングの応用例

本節では，関数型ブートストラッピングによって実現される代表的な演算について説明する．なお，乗算演算と除算演算については，Okada らの論文 [13] を参照されたい．

加算: 加算演算は，t 個の TLWE 暗号文 $c_i = \mathsf{IWTFHE.Enc}_s(m_i)$ $(i \in \{1, \cdots t\})$ を受け取り，TLWE 暗号文 $\mathsf{IWTFHE.Enc}_s(m_1 + \cdots + m_t)$ を返す．暗号文同士の準同型加算は $c_1 + \cdots + c_t$ を計算することで実現される．ただし，加算演算によってノイズが蓄積するため，恒等関数 $f_{\mathrm{id}}(x) = \lfloor x \rfloor$ を IWTFHE.FBS に適用することで，ノイズを低減できる．

定数乗算: 定数乗算演算は，TLWE 暗号文 $c_{\mathrm{in}} = \mathsf{IWTFHE.Enc}_s(m_{\mathrm{in}})$ と定数 $d \in \mathbb{Z}$ を受け取り，TLWE 暗号文 $\mathsf{IWTFHE.Enc}_s(\lfloor m_{\mathrm{in}} \cdot d \rceil)$ を返す．この演算は，関数 $f_{\mathrm{mult},d}(x) = \lfloor x \cdot d \rceil$ を IWTFHE.FBS に適用することで実現される．

定数除算: 定数除算演算は，TLWE 暗号文 $c_{\mathrm{in}} = \mathsf{IWTFHE.Enc}_s(m_{\mathrm{in}})$ と定数 $d \in \mathbb{Z} \setminus \{0\}$ を受け取り，TLWE 暗号文 $\mathsf{IWTFHE.Enc}_s(\lfloor m_{\mathrm{in}}/d \rfloor)$ を返す．この演算は，関数 $f_{\mathrm{div},d}(x) = \lfloor x/d \rfloor$ を IWTFHE.FBS に適用することで実現される．

3.2 複数鍵 TFHE 方式 (MKTFHE)

本節では，Chen ら [6] によって提案された複数鍵 TFHE 方式について概説する．

計算に参加するパーティ数を k とする．各パーティは独立して生成する TLWE 秘密鍵 $s_i \in \mathbb{B}^n$ および TRLWE 秘密鍵 $z_i(X) \in \mathbb{B}_N[X]$ に対し，全パーティの

秘密鍵を連結した連結秘密鍵を $\bar{s} = (s_1, \ldots, s_k) \in \mathbb{B}^{kn}$, $\bar{z} = (z_1(X), \ldots, z_k(X)) \in \mathbb{Z}_N[X]^k$ とする．

3.2.1 MKTLWE (Multi-Key TLWE)

MKTLWE 暗号方式は TLWE 暗号方式を複数鍵に対応させたものである．

暗号化: $\mathsf{MKTLWE.Enc}_{\bar{s}}(\mu)$ MKTLWE 暗号化アルゴリズムは, 平文 $\mu \in \mathbb{T}$ に対し, TLWE 連結秘密鍵 $\bar{s}$ を用いて MKTLWE 暗号文 $\bar{c} \in \mathbb{T}^{kn+1}$ を生成する．乱数ベクトル $a_i \in \mathbb{T}^n$ $(i = 1, \ldots, k)$ とノイズ e に対し, MKTLWE 暗号文 $\bar{c} = (b, a_1, \ldots, a_k)$ を $b = \sum_{i=1}^{k} \langle a_i, s_i \rangle + \mu + e$ で定める．

復号: $\mathsf{MKTLWE.Dec}_{\bar{s}}(\bar{c})$ MKTLWE 復号アルゴリズムは, MKTLWE 暗号文 $\bar{c}$ に対し, TLWE 連結秘密鍵 $\bar{s}$ を用いて平文 μ を計算する．具体的には, 位相 $\varphi'_{\bar{s}}(\bar{c}) = b - \sum_{i=1}^{k} \langle a_i, s_i \rangle$ を計算し，ノイズが十分に小さい場合に平文を復元できる．

拡張: $\mathsf{MKTLWE.Expand}(\bar{c}, I_{\mathbf{in}}, I_{\mathbf{out}})$ MKTLWE 拡張アルゴリズムは, ℓ 個のパーティに関連する MKTLWE 暗号文 $\bar{c} = (b, a_1, \ldots, a_\ell) \in \mathbb{T}^{\ell n+1}$ とそのパーティのインデックス集合 $I_{\text{in}} = \{q_1, \ldots, q_\ell\}$, および拡張後の全パーティのインデックス集合 $I_{\text{out}} = \{p_1, \ldots, p_k\}$ を受け取り, k $(\geq \ell)$ 個のパーティに対応する MKTLWE 暗号文 $(b, a'_1, \ldots, a'_k) \in \mathbb{T}^{kn+1}$ を返す．ここで, $p_i = q_j$ となる j が存在する場合は $a'_i = a_j$ とし，それ以外の場合は $a'_i = 0$ とする．

3.2.2 MKTRLWE (Multi-Key TRLWE)

MKTRLWE 暗号方式は TRLWE 暗号方式を複数鍵に対応させたものである．

暗号化: $\mathsf{MKTRLWE.Enc}_{\bar{z}}(\mu(X))$ MKTRLWE 暗号化アルゴリズムは, 平文 $\mu(X) \in \mathbb{T}_N[X]$ に対し, TRLWE 連結秘密鍵 $\bar{z}(X)$ を用いて, MKTRLWE 暗号文 $\bar{c} \in \mathbb{T}_N[X]^{k+1}$ を生成する．乱数多項式 $a_i(X) \in \mathbb{T}_N[X]$ $(i = 1, \ldots, k)$ とノイズ多項式 $e(X)$ に対し, MKTRLWE 暗号文 $\bar{c} = (b(X), a_1(X), \ldots, a_k(X))$ を $b(X) = \sum_{i=1}^{k} a_i(X) \cdot z_i(X) + \mu(X) + e(X)$ で定める．

復号: $\mathsf{MKTRLWE.Dec}_{\bar{z}}(\bar{c})$ MKTRLWE 復号アルゴリズムは, MKTRLWE 暗号文 $\bar{c}$ に対して, TRLWE 連結秘密鍵 $\bar{z}(X)$ を用いて平文 $\mu(X)$ を計算する．具体的には, 位相 $\varphi'_{\bar{z}}(\bar{c}) = b(X) - \sum_{i=1}^{k} a_i(X) \cdot z_i(X)$ を計算し，ノイズの各係数が十分に小さい場合に平文を復元できる．

共通参照列 (CRS) 全パーティが共有する共通参照列 $a_{\text{crs}} \in \mathbb{T}_N[X]^d$ に対し, 各パーティ i は自身の TRLWE 秘密鍵 $z_i(X)$ およびノイズベクトル $e_i \in \mathbb{T}_N[X]^d$ を用いて, 公開鍵 $\text{PK}_i = -z_i(X) \cdot a_{\text{crs}} + e_i \in \mathbb{T}_N[X]^d$ を生成する．また, $\text{PK}_0 = -a_{\text{crs}}$ と定義する．

Uni-Encryption: $\mathsf{MKTFHE.UniEnc}_{z_i}(m)$
Uni-Encryption アルゴリズムは, パーティ i の TRLWE 秘密鍵 $z_i(X)$ を用いて, 平文 $m \in \mathbb{B}$ を暗号化し, 暗号文 $(d_i, F_i) \in \mathbb{T}_N[X]^{d \times 3}$ を返す．ここで, $g \in \mathbb{T}^d$ はガジェットベクトル, $r(X) \in \mathbb{B}_N[X]$ は乱数多項式, $e_1, e_2 \in \mathbb{T}_N[X]^d$ はノイズベクトルであり, $f_{i,1} \in \mathbb{T}_N[X]^d$ は乱数ベクトルである．暗号文は次式で生成される．ただし, $m \cdot g$ の各成分は定数多項式とみなす:

$$d_i = r(X) \cdot a_{\text{crs}} + m \cdot g + e_1$$
$$F_i = [f_{i,0} \mid f_{i,1}], \quad f_{i,0} = -z_i(X) \cdot f_{i,1} + r(X) \cdot g + e_2$$

各パーティ i は, TLWE 秘密鍵 $s_i = (s_{i,1}, \ldots, s_{i,n}) \in \mathbb{B}^n$ に対し, $(d_{i,j}, F_{i,j}) = \mathsf{MKTFHE.UniEnc}_{z_i}(s_{i,j})$ を計算し, ブートストラッピング鍵 $\text{BK}_i = \{(d_{i,j}, F_{i,j})\}_{j=1}^{n}$ を生成する．

3.2.3 ブートストラッピング

複数鍵 TFHE のブートストラッピングは, TFHE と同様の構成で実現される．複数鍵方式では, 計算量を削減するため, 外部積の代わりに HybridProduct が導入される．

CMUX: $\mathsf{MKTFHE.CMUX}(\bar{c}_0, \bar{c}_1, \text{BK}_{i,j}, \text{PK})$ CMUX は, 2つの MKTRLWE 暗号文 $\bar{c}_0, \bar{c}_1$, ブートストラッピング鍵 $\text{BK}_{i,j}$, および公開鍵 PK を受け取り, MKTRLWE 暗号文 $\bar{c}' = \mathsf{MKTFHE.HybridProd}(\bar{c}_1 - \bar{c}_0, \text{BK}_{i,j}, \text{PK}) + \bar{c}_0$ を返す．

BlindRotation: $\mathsf{MKTFHE.BR}(\bar{c}, v(X), \text{BK}, \text{PK})$
BlindRotation は, MKTLWE 暗号文 $\bar{c} = (b, a_1, \ldots, a_k) \in \mathbb{T}^{kn+1}$, ルックアップテーブル $v(X) \in \mathbb{T}_N[X]$, ブートストラッピング鍵 $\text{BK} = \{\text{BK}_i\}_{i=1}^{k}$, および公開鍵 PK を受け取り, MKTRLWE 暗号文 $\mathsf{MKTRLWE.Enc}_{\bar{z}}(v(X) \cdot X^{-\lfloor 2N \cdot \varphi'_{\bar{s}}(\bar{c}) \rceil})$ を返す．このアルゴリズムの疑似コードを Algorithm 4 に示す．1-5 行目で暗号文の各要素を $2N$ 倍してスケーリングする．続いて, 6 行目で ACC を初期化し, 7-10 行目で ACC を更新する．

SampleExtraction: $\mathsf{MKTFHE.SE}(\bar{c}, k)$ SampleExtraction は, MKTRLWE 暗号文 $\bar{c} = (b(X), a_1(X), \ldots, a_k(X))$ と整数 k を受け取り, MKTLWE 暗号文 $\bar{c}' = (b', a'_1, \ldots, a'_k) \in \mathbb{T}^{kN+1}$ を返す．具体的には, $1 \leq j \leq k$ に対して $(0, a'_j) = \mathsf{TFHE.SE}((0, a_j(X)), k)$ を計算し, $b' = b_k$ を適用することで MKTRLWE 暗号文 $\bar{c}'$ を計算

Algorithm 4 MKTFHE BlindRotation

Input: $\bar{c} = \mathsf{MKTLWE.Enc}_{\bar{s}}(\mu) = (b, a_1, \ldots, a_k)$, $v(X)$, $\mathrm{BK} = \{\mathrm{BK}_i\}_{i=1}^{k}$, PK

Output: $\mathsf{MKTRLWE.Enc}_{\bar{z}}(v(X) \cdot X^{-\lfloor 2N \cdot \varphi'_{\bar{s}}(\bar{c}) \rceil})$

$\tilde{b} \leftarrow \lfloor 2N \cdot b \rceil \pmod{2N}$
for $i = 1$ to k **do**
  for $j = 1$ to n **do**
    $\tilde{a}_{i,j} \leftarrow \lfloor 2N \cdot a_{i,j} \rceil \pmod{2N}$
$\mathrm{ACC} \leftarrow (v(X) \cdot X^{-\tilde{b}}, 0, \ldots, 0)$
for $i = 1$ to k **do**
  for $j = 1$ to n **do**
    $\mathrm{ACC} \leftarrow \mathsf{MKTFHE.CMUX}(\mathrm{ACC}, X^{\tilde{a}_{i,j}} \cdot \mathrm{ACC}, \mathrm{BK}_{i,j}, \mathrm{PK})$
return ACC

Algorithm 5 MKTFHE Bootstrapping

Input: $\bar{c} = \mathsf{MKTLWE.Enc}_{\bar{s}}(\mu)$, $v(X) \in \mathbb{T}_N[X]$, BK, KSK, PK

Output: $\mathsf{MKTLWE.Enc}_{\bar{s}}(\mu)$

$\mathrm{ACC} \leftarrow \mathsf{MKTFHE.BR}(\bar{c}, v(X), \mathrm{BK}, \mathrm{PK})$
$\bar{c}_{\mathrm{extracted}} \leftarrow \mathsf{MKTFHE.SE}(\mathrm{ACC}, 0)$
$\bar{c}_{\mathrm{out}} \leftarrow \mathsf{MKTFHE.KS}(\bar{c}_{\mathrm{extracted}}, \mathrm{KSK})$
return $\bar{c}_{\mathrm{out}}$

する．出力である MKTRLWE 暗号文 $\bar{c}'$ は，$\bar{c}$ の平文多項式の k 次係数を平文に，TRLWE 連結秘密鍵 $\bar{z}$ の係数ベクトル $\bar{z}' = (z_{1,0}, \ldots, z_{1,N-1}, \cdots, z_{k,0}, \ldots, z_{k,N-1}) \in \mathbb{B}^{kN}$ を秘密鍵に持つ．

KeySwitching: $\mathsf{MKTFHE.KS}(\bar{c}, \mathrm{KSK})$ KeySwitching は，MKTLWE 暗号文 $\bar{c} = (b', a'_1, \ldots, a'_k) \in \mathbb{T}^{kN+1}$ と鍵切替鍵 $\mathrm{KSK} = \{\mathrm{KSK}_i\}_{i=1}^{k}$ を受け取り，MKTLWE 暗号文 $\bar{c}$ を返す．具体的には，各パーティの鍵切替鍵 KSK_i を用いて，$(b''_i, a''_i) = \mathsf{TFHE.KS}((0, a'_i), \mathrm{KSK}_i)$ を計算し，MKTLWE 暗号文 $\bar{c} = (b' + \sum_{i=1}^{k} b''_i, a''_1, \ldots, a''_k) \in \mathbb{T}^{kn+1}$ を出力する．ここで，パーティ i の鍵切替鍵 KSK_i は TFHE.KS で使用される KSK と同様の構成である．

Bootstrapping: $\mathsf{MKTFHE.BS}(\bar{c}, v(X), \mathrm{BK}, \mathrm{KSK}, \mathrm{PK})$ Bootstrapping は，MKTLWE 暗号文 $\bar{c}$，ルックアップテーブル $v(X)$，ブートストラッピング鍵 BK，鍵切替鍵 KSK，および公開鍵 PK を受け取り，ノイズを低減した MKTLWE 暗号文を返す．このアルゴリズムの疑似コードを Algorithm 5 に示す．BlindRotation, SampleExtraction，KeySwitching を順に適用することで，ノイズ低減を実現する．

4 整数型複数鍵 TFHE 方式

本章では，整数型 TFHE 方式と 複数鍵 TFHE 方式を組み合わせた整数型複数鍵 TFHE 方式 (IWMKTFHE) について概説する．

4.1 パラメータと暗号化方式

整数型複数鍵 TFHE 方式のパラメータは，整数型 TFHE 方式と 複数鍵 TFHE 方式のパラメータを組み合わせて構成される．

まず，整数型 TFHE のパラメータとして，許容整数パラメータ $\tau \in \mathbb{N}$，平文空間 $\mathcal{M} = \{-\tau, \ldots, \tau - 1\} \subset \mathbb{Z}$，スケーリング因子 $\Delta = 1/(2\tau)$ を用いる．次に，複数鍵 TFHE のパラメータとして，パーティ数を k とし，各パーティ $i \in \{1, \ldots, k\}$ は独立して TLWE 秘密鍵 $s_i \in \mathbb{B}^n$ および TRLWE 秘密鍵 $z_i(X) \in \mathbb{B}_N[X]$ を生成する．全パーティの秘密鍵を連結した連結秘密鍵を $\bar{s} = (s_1, \ldots, s_k) \in \mathbb{B}^{kn}$，$\bar{z} = (z_1(X), \ldots, z_k(X)) \in \mathbb{Z}_N[X]^k$ とする．

暗号化: $\mathsf{IWMKTLWE.Enc}_{\bar{s}}(m)$ 整数型複数鍵 TFHE における暗号化アルゴリズムは，整数平文 $m \in \mathcal{M}$ を受け取り，TLWE 連結秘密鍵 $\bar{s}$ およびスケーリング因子 $\Delta = 1/(2\tau)$ を用いて，MKTLWE 暗号文 $\bar{c} = \mathsf{MKTLWE.Enc}_{\bar{s}}(m \cdot \Delta)$ を生成する．

復号: $\mathsf{IWMKTLWE.Dec}_{\bar{s}}(\bar{c})$ 整数型複数鍵 TFHE における復号アルゴリズムは，入力として MKTLWE 暗号文 $\bar{c}$ を受け取り，TLWE 連結秘密鍵 $\bar{s}$ を用いて整数平文 m を計算する．具体的には，$m = \lfloor \varphi'_{\bar{s}}(\bar{c})/\Delta \rceil = \lfloor m + e/\Delta \rceil$ を計算し，ノイズ条件 $|e| < 1/(4\tau)$ が満たされる場合，平文 m を正しく復元できる．

4.2 関数型ブートストラッピング

整数型複数鍵 TFHE における関数型ブートストラッピングは，整数型 TFHE のルックアップテーブル生成と複数鍵 TFHE のブートストラッピングを組み合わせて実現される．

FunctionalBootstrapping: $\mathsf{IWMKTFHE.FBS}(c, f, \mathrm{BK}, \mathrm{KSK}, \mathrm{PK})$ FunctionalBootstrapping は，MKTLWE 暗号文 $c = \mathsf{IWMKTFHE.Enc}(m)$，任意の関数 $f : \{0, \ldots, \tau - 1\} \to \mathcal{M}$，ブートストラッピング鍵 BK，鍵切替鍵 KSK，および公開鍵 PK を受け取り，MKTLWE 暗号文 $c_{\mathrm{eval}} = \mathsf{IWMKTLWE.Enc}(f(m))$ を返す．このアルゴリズムの疑似コードを Algorithm 6 に示す．IWTFHE.FBS と同様に，1 行目で入力関数 f を使用してルックアップテーブル $v(X)$ の係数を適切に設定する．続いて，2 行目において，関数 f を評価した c_{eval} を出力する．

Algorithm 6 IWMKTFHE Functional Bootstrapping

Input: $c = \mathsf{MKTLWE.Enc}_s(\mu)$, f, BK, KSK, PK

Output: $\mathsf{TLWE.Enc}_s(f(\mu))$

1: $v(X) \leftarrow \mathsf{IWTFHE.GenLUT}(f)$

2: $c_{\mathrm{eval}} \leftarrow \mathsf{MKTFHE.BS}(c, v(X), \mathrm{BK}, \mathrm{KSK}, \mathrm{PK})$

3: **return** c_{eval}

3.1.3 節で紹介した各種演算は，整数型 TFHE と同様に，対応する関数を用いて複数鍵設定で実現される．つまり，加算演算は恒等関数 $f_{\mathrm{id}}(x) = \lfloor x \rceil$ を用いたブートストラッピングによるノイズ除去，定数乗算演算は関数 $f_{\mathrm{mult,d}}(x) = \lfloor x \cdot d \rceil$，定数除算演算は関数 $f_{\mathrm{div,d}}(x) = \lfloor x/d \rceil$ を適用したルックアップテーブル生成により実現される．

4.3 ノイズ解析

本節では，Algorithm 6 により生じるノイズ分散の見積もりを行う．整数型複数鍵 TFHE では，整数型 TFHE のノイズ条件である制約 $|e| < 1/(4\tau)$ に加え，複数鍵への拡張に伴うノイズ増加を考慮する必要がある．以下では，TLWE および TRLWE におけるノイズ分布としてガウス分布を採用し，それらのノイズ標準偏差をそれぞれ α, β とする．また，ガジェット分解パラメータについて，KeySwitching およびブートストラッピング鍵生成で使用する基数と分解長のペアをそれぞれ (B', d'), (B, d) とする．k をパーティ数，n を TLWE 次元数，N を多項式次数とそれぞれ置くとき，Chen ら [6] は，BlindRotation 後のアキュムレータのノイズ分散と KeySwitching によるノイズ分散をそれぞれ $V_{\mathrm{acc}} \approx \frac{knN^2B^2\beta^2}{24}(dk + k + 1)$, $V_{\mathrm{ks}} \approx kN\left(\frac{1}{24}B'^{-2d'} + d'\alpha^2\right)$ と見積もった．

以下では，ブートストラッピング後の総ノイズ分散を $V_{\mathrm{bs}} = V_{\mathrm{acc}} + V_{\mathrm{ks}}$ として，整数型複数鍵 TFHE が正しくブートストラッピングを行える条件を考察する．具体的には，平均 0 で標準偏差が $\sigma_{bs} = \sqrt{V_{\mathrm{bs}}}$ のガウス分布によるサンプリングが区間 $[-1/(4\tau), 1/(4\tau)]$ に収まる確率 P_τ を計算し，その値が復号閾値 T より小さければ，ブートストラッピング後の暗号文に対して，正しく復号が行われるとする．具体的な σ_{bs} や復号閾値 T の値は次章で検討する．

5 実験および結果

本章では，提案手法である整数型複数鍵 TFHE 方式におけるブートストラッピングの動作検証と評価について述べる．

実験の主な目的は，パーティ数 k と，パラメータ τ の実用的な動作範囲を明らかにすることである．特に，両者の間に存在するトレードオフ関係を定量的に評価し，実際の応用において利用可能なパラメータセットを示す．

表 1: MKTFHE のパラメータとノイズ

Set	k	σ_{bs}	P_τ
I	2	4.02×10^{-2}	2.47×10^{-10}
II	4	4.35×10^{-2}	4.70×10^{-9}
III	8	2.46×10^{-2}	1.47×10^{-24}

表 2: 提案手法のパラメータとノイズ

k	τ	(B, d)	σ_{bs}	P_τ
1	2	$(2^9, 3)$	1.92×10^{-2}	4.14×10^{-11}
1	4	$(2^8, 4)$	1.08×10^{-2}	3.48×10^{-9}
1	8	$(2^6, 5)$	3.95×10^{-3}	1.15×10^{-15}
2	2	$(2^8, 4)$	2.03×10^{-2}	4.03×10^{-10}
2	4	$(2^6, 5)$	6.69×10^{-3}	4.56×10^{-21}
2	8	$(2^5, 6)$	4.88×10^{-3}	7.30×10^{-11}
4	2	$(2^6, 5)$	1.20×10^{-2}	9.60×10^{-26}
4	4	$(2^5, 6)$	7.96×10^{-3}	2.11×10^{-15}
8	2	$(2^5, 6)$	1.38×10^{-2}	6.48×10^{-20}

5.1 パラメータ設定

Chen らの複数鍵 TFHE 方式 [6] の Table2, 3 に基づき，(Set, k) = (I, 2), (II, 4), (III, 8) のそれぞれの場合における σ_{bs} および区間内確率 P_τ を表 1 に示す．以下の記述において，表 1 における P_τ の最小値 4.7×10^{-9} を復号閾値 T として設定する．

本稿では，Chen らと同様に 110 ビットセキュリティを満たすパラメータを Lattice Estimator [2] により設定した．4.3 節で定義した変数を $n = 584, N = 1024, \alpha = 3.05 \times 10^{-5}, \beta = 3.29 \times 10^{-9}, (B', d') = (2^2, 8)$ と固定する．一方，ブートストラッピング鍵用のガジェット分解パラメータ (B, d) は，パーティ数 k および許容整数パラメータ τ に応じて調整する必要がある．4.3 節のノイズ分散の式より，総ノイズ分散 V_{bs} は V_{acc} が支配的であり，これは基数 B に強く依存する．そのため，パーティ数 k や許容整数パラメータ τ の増加時には，基数 B を小さく，分解長 d を大きく設定することでノイズを抑制できる．上記の方針に基づき，k，τ，および (B, d) を変更した際の，σ_{bs} および P_τ を表 2 に示す．

表 2 の結果の全ての場合において $P_\tau \leq T$ であり，ブートストラッピング後のノイズ E がノイズ条件 $|E| < 1/(4\tau)$ を満たす範囲に高確率で収まることが期待できる．表 2 の各 k に対して，τ が最大となる場合のパラメータを表 3 に掲載する．

表 3: k に対する最大の τ とパラメータ

k	τ	(B, d)
1	8	$(2^6, 5)$
2	8	$(2^5, 6)$
4	4	$(2^5, 6)$
8	2	$(2^5, 6)$

5.2 実験と結果

本実験では，前節で示した固定パラメータと表 3 のパラメータの設定を使用して，Algorithm 6 を 1000 回 実行した．その結果を復号し，ノイズ条件を満たす場合を復号成功とし，復号成功率を評価した．表 3 掲載のパラメータにおいて，実際に復号成功率が 100% であることを確認した．例えば，パーティ数 $k = 4$ に対して，$\log_2 4 = 2$ ビットまでの協調計算が行えることを実際に明らかにした．ここで，平文整数空間を $\mathcal{M} = \{-\tau, \ldots, \tau - 1\}$ と定めているが，本稿では，$\mathcal{M}$ のビット数は $\lceil \log_2(\tau) \rceil$ ビットとして定義していることに注意されたい．

5.3 連続ブートストラッピングの評価

実用的な準同型演算には，ブートストラッピングの連続実行が必要となる．例えば，複数暗号文の平均を求める際には，加算演算と定数除算演算を連続して実行する必要がある．暗号文に対してブートストラッピングを連続実行できる必要条件は，ブートストラッピング後の出力暗号文に含まれるノイズ E および次段のブートストラッピングの入力暗号文のノイズ E' に対して，ノイズ要件 $|E| < 1/(4\tau)$ かつ $|E'| < 1/(4\tau)$ が成り立つことである．本節では，一度ブートストラッピングされた TLWE 暗号文に対して，再度ブートストラッピングを実行し，その復号成功率を評価した．具体的には，表 3 のパラメータに対して，Algorithm 6 の出力である TLWE 暗号文を再度 Algorithm 6 の入力として実行した．その出力である TLWE 暗号文を復号すると，いずれのパラメータ設定においても復号成功率が 100% であった．この結果は，表 3 のパラメータセットで 1 度でも成功すれば，ブートストラッピングは継続的に実行可能であることを示唆している．

6 結論

本稿では，整数型 TFHE と複数鍵 TFHE を統合した新しい暗号方式である整数型複数鍵 TFHE 方式を提案した．提案方式は，複数の組織が各自の秘密鍵を保持したまま，整数データに対する協調計算を可能にする．具体的には整数型 TFHE の関数型ブートストラッピングを複数鍵設定に適応させることで，整数演算を複数パーティ環境で実現した．さらに，実装による性能評価を通じて，110 ビットセキュリティパラメータの設定下で，パーティ数 $k = 4$ に対して 2 ビット整数の協調計算が可能であることを実証した．

今後の課題として，より高いセキュリティを確保するため 128 ビットセキュリティのパラメータでの実験評価が挙げられる．また，Kwak [10] や Klemsa ら [1] が提案した最適化手法を統合することで，計算効率のさらなる向上が見込まれる．さらに，本稿では未検証である，暗号文同士の乗算や除算といった複雑な演算の実装と評価を行うことも今後の課題である．

参考文献

[1] Yavuz Akın, Jakub Klemsa, and Melek Önen. A practical tfhe-based multi-key homomorphic encryption with linear complexity and low noise growth. In European Symposium on Research in Computer Security, pp. 3–23. Springer, 2023.

[2] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. Cryptology ePrint Archive, 2015.

[3] Florian Bourse, Michele Minelli, Matthias Minihold, and Pascal Paillier. Fast homomorphic evaluation of deep discretized neural networks. In Annual International Cryptology Conference, pp. 483–512. Springer, 2018.

[4] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In Annual cryptology conference, pp. 868–886. Springer, 2012.

[5] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT), Vol. 6, No. 3, pp. 1–36, 2014.

[6] Hao Chen, Ilaria Chillotti, and Yongsoo Song. Multi-key homomorphic encryption from tfhe. In International Conference on the Theory and Application of Cryptology and Information Security, pp. 446–472. Springer, 2019.

[7] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In International conference on the theory and application of cryptology and information security, pp. 409–437. Springer, 2017.

[8] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In international conference on the theory and application of cryptology and information security, pp. 3–33. Springer, 2016.

[9] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, 2012.

[10] Hyesun Kwak, Seonhong Min, and Yongsoo Song. Towards practical multi-key tfhe: parallelizable, key-compatible, quasi-linear complexity. In IACR International Conference on Public-Key Cryptography, pp. 354–385. Springer, 2024.

[11] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Proceedings of the forty-fourth annual ACM symposium on Theory of computing, pp. 1219–1234, 2012.

[12] Koki Morimura, Daisuke Maeda, and Takashi Nishide. Accelerating polynomial evaluation for integer-wise homomorphic comparison and division. Journal of Information Processing, Vol. 31, pp. 288–298, 2023.

[13] Hiroki Okada, Shinsaku Kiyomoto, and Carlos Cid. Integer-wise functional bootstrapping on tfhe: applications in secure integer arithmetics. Information, Vol. 12, No. 8, p. 297, 2021.