On the Security and Privacy of CKKS-based Homomorphic Evaluation Protocols

Intak Hwang, Seonhong Min, Jinyeong Seo, and Yongsoo Song

Seoul National University
{intak.hwang, minsh, jinyeong.seo, y.song}@snu.ac.kr

Abstract. CKKS is a homomorphic encryption (HE) scheme that supports arithmetic over complex numbers in an approximate manner. Despite its utility in PPML protocols, formally defining the security of CKKS-based protocols is challenging due to its approximate nature. To be precise, in a sender-receiver model, where the receiver holds input ciphertexts and the sender evaluates its private circuit, it is difficult to define sender's privacy in terms of indistinguishability, whereas receiver's privacy is easily achieved through the semantic security of CKKS.

In this paper, we present a new definition for CKKS-based protocols, called Differentially Private Homomorphic Evaluation (DPHE) protocols, along with a general method to achieve this. In our definition, we relax the sender's privacy condition from indistinguishability to differential privacy notion. We focus on the fact that most security concern for PPML protocols is differential privacy on evaluation results, rather than the simulatability of the evaluation. We prove that if the ideal functionality satisfies differential privacy and a protocol satisfies DPHE, then the output of the protocol also satisfies differential privacy.

Next, we provide a general compiler that transforms a plain CKKS-based protocol into a DPHE one. We achieve this by mixing the Laplace mechanism and zero-knowledge argument of knowledge (ZKAoK) for CKKS. This approach allows us to achieve sender's privacy with a moderate noise, whereas the previous indistinguishability-based approach requires exponentially large overhead.

Finally, we provide a concrete instantiation of ZKAoK for CKKS in the form of PIOP. To prove the well-formedness of CKKS ciphertexts and public keys, we devise new proof techniques that use homomorphic evaluation during verification. We also provide an implementation to demonstrate the practicality of our ZKAoK for CKKS by compiling PIOPs using the HSS polynomial commitment scheme (Crypto'24).

Keywords: Homomorphic Encryption, Zero-knowledge Proof, Differential Privacy, CKKS

1 Introduction

Homomorphic encryption (HE) is an encryption scheme that enables computation on encrypted data without requiring decryption. Since Gentry's seminal work [47], numerous HE schemes have been proposed, with the current best-performing schemes [14,15,30,32,41] being based on the hardness of the Learning with Errors (LWE) problem [72] or its ring variant (RLWE) [67]. In (R)LWE-based HE schemes, plaintexts are encrypted with small noises, and decryption yields both the plaintext and the noise. These HE schemes can be categorized based on the supported homomorphic operations: BGV [15] and BFV [14, 41] support modular arithmetic over integers, TFHE [32] supports Boolean operations, and CKKS [30] supports approximate arithmetic over complex numbers.

One direct application of homomorphic encryption (HE) is a secure evaluation protocol in the sender-receiver model. In this protocol, the sender possesses a circuit, while the receiver holds an input for that circuit. The objective of the protocol is to deliver the evaluation result to the receiver without revealing the input to the sender. Assuming semi-honest security, HE offers a straightforward two-round solution, which we refer to as a homomorphic evaluation protocol. In the first round, the receiver encrypts its input using a homomorphic encryption scheme and sends it along with a public key. The sender then evaluates the circuit on the encrypted data using homomorphic operations and sends the resulting ciphertext back to the receiver. Finally, the receiver decrypts the output ciphertext to obtain the evaluation result. This construction is not only round-optimal but also communication-efficient, particularly when the size of the sender's circuit significantly exceeds the size of the receiver's input.

When the sender's circuit is private, the sender additionally runs a randomization process to ensure that an output ciphertext can be simulated using only the protocol output and the receiver's input. For exact HE schemes, including BGV, BFV, and TFHE, this randomization process is typically achieved through the noise flooding method [3, 35], where the sender erases any remaining circuit information in the output ciphertext's noise by adding exponentially large noise. This approach works for exact HE schemes because the plaintext and noise in a ciphertext are strictly distinguished.

However, for the CKKS scheme, the noise flooding method does not work well because plaintext and noise are fused[1], making it impossible for even the secret key owner to discriminate between them without knowing the full trace of homomorphic operations performed on the ciphertext. As a result, adding exponentially large noise typically spoils the plaintext, which compromises the correctness and usefulness of the protocol. To the best of our knowledge, there is no general randomization method for CKKS ciphertexts that is comparable to the noise flooding method.

When extending a homomorphic evaluation protocol to cover a malicious receiver, the sender must validate the well-formedness of the receiver's ciphertext and public key. This is crucial because the randomization process, such as noise flooding, guarantees the sender's privacy only when the receiver's ciphertext and public key are correctly generated. This validation can be performed using a zero-knowledge argument of knowledge (ZKAoK) on the receiver's ciphertext and public key. There exists a general compiler [3] that transforms a semi-honestly

[1] Here, we do not consider discrete variants of CKKS, such as [6]

secure BGV or BFV based homomorphic evaluation protocol into a maliciously secure one, assuming the ideal functionality of ZKAoK for BGV or BFV. In this context, the construction of efficient ZKAoK for the BGV and BFV schemes has been continuously studied in the literature [3, 13, 22, 52].

However, for CKKS, no tailored ZKAoK exists, and existing ZKAoKs for HE do not extend well to cover CKKS. This is due to challenges in proving the validity of the plaintext and the sparsity of the secret key when designing a ZKAoK for CKKS. More specifically, the validity of CKKS plaintext is typically represented in relations over complex numbers, whereas existing ZKAoKs usually prove relations over a prime field. For the CKKS secret key, proving its sparsity is crucial not only for the precision of homomorphic operations but also for efficient CKKS bootstrapping [28], but previous constructions of ZKAoKs for HE either do not consider this or fail to prove it efficiently.

1.1 Our Contribution

In this paper, we address the forementioned issues arises from CKKS based homomrphic evaluation protocols.

DPHE Protocol. To begin with, we introduce a new tailored definition that captures the context of CKKS-based protocols. Previously, the security of HE-based protocols was analyzed within the framework of secure multiparty computation (MPC), where an efficient simulator for protocol execution is constructed to achieve computational indistinguishability. This stems from a modular design approach for privacy-preserving protocols, in which the evaluation process is protected via MPC, while privacy leakage from the evaluation result is typically mitigated using differential privacy (DP) [38]. While this approach extends well to exact HE schemes, it is challenging to achieve in CKKS.

However, we focus on the fact that final privacy leakage is analyzed using differential privacy in most applications of CKKS, such as secure inference, secure training, and secure aggregation. Based on this observation, we define a new security notion, which we call differentially private homomorphic evaluation (DPHE) protocols, inspired by the definition of differential privacy.

We specifically focus on defining the sender's privacy within the framework of differential privacy and formalize it as the existence of an efficient simulator whose max divergence from the real protocol execution is bounded, analogous to the definition of DP. The key implication of our definition is that if the ideal functionality is a DP mechanism and a protocol satisfies our sender's privacy definition, then the output of the real execution remains a DP mechanism. This precisely captures the expected privacy guarantees of privacy-preserving protocols. Furthermore, we show that our definition is compatible with the key properties of the universal composability framework [18], such as the hybrid execution model.

DPHE Compiler. To design DPHE protocols, we also develop an efficient compiler that transforms a plain CKKS-based homomorphic evaluation protocol into a DPHE protocol. Our compiler is inspired by [3], which constructs an efficient compiler that transforms a BGV-based semi-honest protocol into a maliciously secure one using the noise flooding method in the hybrid execution model for the ideal functionality of ZKAoK for BGV. We adapt this approach to our DPHE definition and prove that it is possible to compile a plain CKKS protocol into a DPHE protocol, using the Laplace mechanism in the hybrid execution model for the ideal functionality of ZKAoK for CKKS. We note that the Laplace mechanism also introduces additional noise for randomization, but it allows for moderate noise, whereas the noise flooding requires an exponentially large amount. Thus, it enables achieving the sender's privacy without significantly compromising the usefulness of the protocol.

ZKAoK for CKKS. Although we have designed an efficient DPHE compiler, there is still a lack of efficient ZKAoK for CKKS, as mentioned earlier. To address this, we design a ZKAoK for CKKS using a polynomial interactive oracle proof (PIOP), an interactive proof system that can later be compiled into a succinct non-interactive argument of knowledge (SNARK) with a polynomial commitment scheme [17, 31] and the Fiat-Shamir heuristic [43]. Our ZKAoK is adapted from [52], which constructs a ZKAoK for the BFV scheme using PIOPs. However, to prove CKKS-specific relations, such as sparsity of the secret key and the validity of plaintexts, we develop new methods tailored to these challenges.

To prove the sparsity of the secret key, we design a new PIOP that checks the L^2-norm bound of a witness vector. We then show the sparsity by showing that the L^2-norm of the secret key is much smaller than the polynomial degree. For the validity of plaintexts, solving the problem using only PIOPs is challenging. To resolve this, we incorporate a homomorphic operation called the coeff-to-slot operation. First, the prover generates a ciphertext in an intermediate form, which allows them to prove the validity of the plaintext via PIOP. The verifier then verifies the proof and applies the coeff-to-slot operation to obtain ciphertexts with valid plaintexts. Based on these solutions, we construct the first efficient ZKAoK for CKKS.

Implementation. To demonstrate the practicality of our ZKAoK for CKKS, we present a proof-of-concept implementation. We compile our PIOP using the HSS polynomial commitment scheme (PCS) [53], which is based on lattice cryptography, supports HE-friendly prime fields, and provides fast proving performance. Benchmark results show that proving the validity of a ciphertext and public key with a total size of 41.1MB results in a proof size of 17.9MB, with the prover's time taking 324 seconds and the verifier's time taking 51 seconds. Specifically, we achieve a proof size that is about two times smaller than the combined size of the ciphertext and public key, thanks to the sublinear complexity of the HSS scheme. We also note that the proof size can be further reduced by compiling our PIOP with other PCS. However, since DPHE protocols require the proofs to be sent along with the ciphertext and public keys, the efficiency gain in communication costs would be moderate. Therefore, we prioritize fast proof generation in our implementation.

1.2 Related Work

DP-MPC. Similar to our definition of differentially private homomorphic evaluation, the concept of a differentially private multiparty computation (DP-MPC) protocol is discussed in [7]. It also defines the security of DP-MPC protocols through the existence of an efficient simulator whose max divergence is bounded with respect to the adversary's view in the semi-honest setting. However, its main focus is on how an existing secure MPC protocol can be transformed into a DP-MPC one when the ideal functionality of the protocol is replaced with a DP mechanism in the semi-honest setting, which aligns with the conventional approach of integrating DP techniques into MPC protocols.

In contrast, our definition considers malicious receiver cases within the UC framework, and our DPHE compiler builds upon a CKKS homomorphic evaluation protocol, which is not a secure MPC protocol by definition. Thus, we view our contribution as a generalization of the results presented in [7].

Approximate MPC. Besides conventional secure multiparty computation, whose objective is to exactly evaluate a target function f, there is a generalization called approximate MPC protocols [42], which considers scenarios where the protocol evaluates an approximation $\hat{f}$ instead of f. Its main focus is the scenario where evaluating $\hat{f}$ is more efficient within a secure MPC protocol and how to define security in such cases so that evaluating $\hat{f}$ does not reveal additional information about the inputs compared to evaluating f. In other words, it considers the informational difference between evaluating f and $\hat{f}$ using an MPC protocol that always produces an exact computation result, either $f(x)$ or $\hat{f}(x)$, respectively.

However, in the case of CKKS evaluation protocols, the context is different since even an honest receiver cannot obtain the exact evaluation result for the target function f. Thus, we note that its definition is not directly applicable to our case.

ZKAoK for HE. As mentioned earlier, several works in the literature have constructed ZKAoKs for HE, especially for BGV and BFV. Boschini et al. [13] construct a proof system for BGV ciphertexts based on Aurora [9]. Bell et al. [8] also construct a proof system for BGV ciphertexts based on the inner product argument of Bulletproofs [16]. Chatel et al. [22] construct a ZKAoK for BFV ciphertexts and public keys, based on the LANES [4,40,66] framework, a lattice-based zero-knowledge proof system. The most relevant to our work is [52], which constructs a PIOP-based ZKAoK for BFV ciphertexts and public keys. However, to our knowledge, there has been no prior work addressing the issues that arise with ZKAoKs for CKKS.

Vector Range Proof. In proving the sparsity of the secret key and the validity of ciphertexts, we frequently utilize range proofs for vectors that check the L^2 or L^∞-norm of witness vectors. Efficient range proofs for vectors have been studied in various works [33,34,48], differing primarily in the proof systems they employ. The most relevant work to ours is [46], which constructs an efficient L^∞-norm range proof by extending univariate PIOPs to bivariate PIOPs. However, its optimization is particularly effective for the KZG [55] polynomial commitment scheme, so we did not include it in our implementation. Additionally, to our knowledge, no univariate PIOP construction has addressed the L^2-norm range proof, which we have developed in this paper.

IND-CPA-D. Recently, a new security notion for CKKS, called IND-CPA-D, has been proposed [62]. Concrete attacks [23, 27, 49] are continuously being raised, and preventive measures [63] are being extensively studied. In short, the definition of IND-CPA-D primarily focuses on the case where decryption results of CKKS are shared with an evaluator, which may leak partial information about the secret key.

In our case, we restrict our protocol to a sender-receiver model, ensuring that the sender does not receive any evaluation results, similar to other oblivious evaluation protocols [71]. Therefore, we do not consider IND-CPA-D security in our protocol, leaving it as an orthogonal research direction.

Verifiable HE. Another research direction in homomorphic evaluation protocols focuses on addressing the malicious behavior of the sender, often referred to as verifiable HE. One frequently used approach is the incorporation of succinct non-interactive arguments (SNARG) to validate the sender's computation [11, 44, 45, 64]. We believe that this approach can be naturally extended to our case, although in this paper, we focus on the semi-honest sender scenario.

2 Background

2.1 Notation

For a positive integer q, we use $\mathbb{Z} \cap (-q/2, q/2]$ as a representative set of $\mathbb{Z}_q$, and denote by $[a]_q$ the reduction of a modulo q. Vectors over $\mathbb{Z}$ or $\mathbb{Z}_q$ are denoted with regular lowercase letters and arrows, such as $\vec{v}$, and matrices over $\mathbb{Z}$ or $\mathbb{Z}_q$ are represented by regular uppercase letters. We regard all vectors as column vectors, and we use the symbol $\|$ for the concatenation of two vectors.

Let N be a power of two. We denote by $R = \mathbb{Z}[X]/(X^N + 1)$ the ring of integers of the $2N$-th cyclotomic field, and by $R_q = R/qR$ the residue ring of R modulo q. For polynomials, we use bold lowercase letters to denote them e.g., $\boldsymbol{f}$. For a vector $\vec{v} = (v_0, \ldots, v_{n-1}) \in \mathbb{Z}^n$, the L^p and L^∞ norms are defined as follows for $p \geq 1$.

$$\|\vec{v}\|_p := \sqrt[p]{\sum_{i=0}^{n-1} |v_i|^p} \quad \text{and} \quad \|\vec{v}\|_\infty := \max_{0 \leq i < n} |v_i|$$

The Hadamard product is denoted by $\odot$. For a polynomial $\boldsymbol{f}$ or a vector of polynomials $\vec{\boldsymbol{f}}$, $\|\boldsymbol{f}\|_p$ and $\left\|\vec{\boldsymbol{f}}\right\|_p$ are calculated by regarding them as coefficient vectors. For a matrix $A \in \mathbb{R}^{n \times n}$, we denote the matrix norm of A by $\|A\|_2 := \max_{0 \neq \vec{x} \in \mathbb{R}^n} \frac{\|A\vec{x}\|_2}{\|\vec{x}\|_2}$.

2.2 Probability Distributions

We denote sampling x from the distribution $\mathcal{D}$ by $x \leftarrow \mathcal{D}$. For distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over a countable set S (e.g. $\mathbb{Z}^n$), the statistical distance of $\mathcal{D}_1$ and $\mathcal{D}_2$ is defined as $\frac{1}{2} \cdot \sum_{x \in S} |\mathcal{D}_1(x) - \mathcal{D}_2(x)| \in [0, 1]$. We denote the uniform distribution over S by $\mathcal{U}(S)$ when S is finite.

For $\sigma > 0$, we define the Gaussian function $\rho_\sigma : \mathbb{R} \to (0, 1]$ as $\rho_\sigma(x) := \exp(-\pi \cdot (x/\sigma)^2)$. The discrete Gaussian distribution $\mathtt{DG}(\sigma)$ with parameter σ is defined as a distribution over $\mathbb{Z}$, whose probability mass function is $\rho_\sigma(x)/\rho_\sigma(\mathbb{Z})$, where $\rho_\sigma(\mathbb{Z}) := \sum_{x \in \mathbb{Z}} \rho_\sigma(x) < \infty$. For a polynomial $\boldsymbol{f}$ of degree d, we denote by $\boldsymbol{f} \leftarrow \mathtt{DG}(\sigma)^d$ if each of its coefficients is sampled from $\mathtt{DG}(\sigma)$. For $b > 0$, the Laplace distribution $\mathtt{Lap}(b)$ with parameter b is defined as a distribution over $\mathbb{R}$, whose probability density function is $\frac{1}{2b} \exp(-|x|/b)$.

2.3 The RLWE Problem

First, we review the definition of the RLWE problem [67]. The computational hardness of the RLWE problem is a key component for achieving IND-CPA security in many lattice-based HE schemes [14,15,41], including CKKS [30].

Definition 1 (RLWE [67]). *Let χ_s, χ_e be distributions over R. Then, the goal of the Ring-LWE (MLWE) problem is to distinguish $(\boldsymbol{a}, \boldsymbol{u})$ from $(\boldsymbol{a}, \boldsymbol{as} + \boldsymbol{e})$ for $\boldsymbol{a} \leftarrow \mathcal{U}(R_q)$, $\boldsymbol{e} \leftarrow \chi_e$, and $\boldsymbol{s} \leftarrow \chi_s$. We say that a PPT algorithm $\mathtt{D}$ has an advantage ε in solving $\mathsf{RLWE}_{q,\chi_s,\chi_e}$ if the following holds.*

$$\Big| \Pr[\mathtt{D}(\boldsymbol{a}, \boldsymbol{u}) = 1] - \Pr[\mathtt{D}(\boldsymbol{a}, \boldsymbol{as} + \boldsymbol{e}) = 1] \Big| \geq \varepsilon$$

When χ_s, χ_e are discrete Gaussian distributions, we denote them simply by their width parameters.

Next, we review the Hint-RLWE problem [56], a variant of the RLWE problem. The Hint-RLWE problem is frequently used in lattice-based zero-knowledge proof systems [53,56] or signature schemes [36,39] to achieve simulatability without the rejection sampling method [65]. Under a suitable parameter setting, it can be reduced from the RLWE problem, as shown in [56].

Definition 2 (Hint-RLWE [56]). *Let $\boldsymbol{c}_s, \boldsymbol{c}_e$ be elements in R, and let χ_s, χ_e, χ_f, χ_g be distributions over R. Then, the goal of the Hint-RLWE problem is to distinguish $(\boldsymbol{a},\ \boldsymbol{u},\ \boldsymbol{c}_s\boldsymbol{s} + \boldsymbol{f},\ \boldsymbol{c}_e\boldsymbol{e} + \boldsymbol{g})$ from $(\boldsymbol{a},\ \boldsymbol{as} + \boldsymbol{e},\ \boldsymbol{c}_s\boldsymbol{s} + \boldsymbol{f},\ \boldsymbol{c}_e\boldsymbol{e} + \boldsymbol{g})$ for $\boldsymbol{a} \leftarrow \mathcal{U}(R_q)$, $\boldsymbol{e} \leftarrow \chi_e$, $\boldsymbol{f} \leftarrow \chi_f$, $\boldsymbol{g} \leftarrow \chi_g$ and $\boldsymbol{s} \leftarrow \chi_s$. We say that a PPT algorithm $\mathtt{D}$ has an advantage ε in solving $\mathsf{HintRLWE}^{\boldsymbol{c}_s,\boldsymbol{c}_e,\chi_f,\chi_g}_{q,\chi_s,\chi_e}$ if the following holds.*

$$\Big| \Pr[\mathtt{D}(\boldsymbol{a},\ \boldsymbol{u},\ \boldsymbol{c}_s\boldsymbol{s} + \boldsymbol{f},\ \boldsymbol{c}_e\boldsymbol{e} + \boldsymbol{g}) = 1] - \Pr[\mathtt{D}(\boldsymbol{a},\ \boldsymbol{as} + \boldsymbol{e},\ \boldsymbol{c}_s\boldsymbol{s} + \boldsymbol{f},\ \boldsymbol{c}_e\boldsymbol{e} + \boldsymbol{g}) = 1] \Big| \geq \varepsilon$$

Definition 3 (Smoothing parameter [68]). *For an n-dimensional lattice Λ and $\varepsilon > 0$, the smoothing parameter $\eta_\varepsilon(\Lambda)$ is the smallest s such that $\rho_{1/s}(\Lambda^* \backslash \{\vec{0}\}) \leq \varepsilon$.*

Theorem 1 (Theorem 1 in [56]). *Let $\sigma_e, \sigma_s > 0$, and $\sigma_e, \sigma_f, \sigma > 0$ be real numbers, which satisfy the following.*

$$\frac{1}{\sigma^2} = 2\left(\frac{1}{\sigma_s^2} + \frac{\|\boldsymbol{c}_s\|_1^2}{\sigma_f^2}\right) = 2\left(\frac{1}{\sigma_e^2} + \frac{\|\boldsymbol{c}_e\|_1^2}{\sigma_g^2}\right)$$

If $\sigma \geq \sqrt{2} \cdot \eta_{(1/2^\lambda)}(\mathbb{Z}^N)$, then there exists an efficient reduction from $\mathsf{RLWE}_{q,\sigma,\sigma}$ to $\mathsf{HintRLWE}_{q,\sigma_s,\sigma_e}^{\boldsymbol{c}_s,\boldsymbol{c}_e,\sigma_f,\sigma_g}$ that reduces the advantage by at most $\mathsf{negl}(\lambda)$.

2.4 Differential Privacy

Below, we review the definition of differential privacy (DP) [38] and its extension to computationally bounded adversaries, known as computational differential privacy (CDP) [70]. Differential privacy is a well-established notion for privacy-preserving statistical analyses. However, when combining DP techniques with cryptographic protocols, a computational indistinguishability extension of the DP notion is necessary for analyzing security, as the original definition is information-theoretic. In this context, CDP is frequently utilized in secure aggregation protocols [12, 75].

Definition 4 (Differential Privacy [38]). *A randomized algorithm $\mathtt{f} : \mathcal{X} \to \mathcal{Y}$ provides ϵ-differential privacy (DP) if for all adjacent inputs $x, x' \in \mathcal{X}$, and all subsets $S \subseteq \mathcal{Y}$, the following holds.*

$$\Pr[\mathtt{f}(x) \in S] \leq e^{\epsilon} \cdot \Pr[\mathtt{f}(x') \in S].$$

Definition 5 (Computational Differential Privacy [70]). *A randomized algorithm $\mathtt{f} : \mathcal{X} \to \mathcal{Y}$ provides ε-computational differential privacy (CDP) if for all adjacent inputs $x, x' \in \mathcal{X}$, the following holds for all PPT algorithms $\mathtt{D}$.*

$$\Pr[\mathtt{D}(\mathtt{f}(x)) = 1] \leq e^{\varepsilon} \cdot \Pr[\mathtt{D}(\mathtt{f}(x')) = 1] + \mathsf{negl}(\lambda)$$

2.5 The CKKS Scheme

We review the CKKS scheme [30], especially for its residue number system (RNS)-variant [29].

Basic Operations. The CKKS scheme is an encryption scheme whose plaintext space is R, and ciphertext space is $R_{Q_L}^2$. Basic operations, such as setup, key generation, encryption, and decryption algorithms, are defined as follows.

- $\underline{\mathtt{Setup}}(1^\lambda, N) \to \mathsf{pp}$: Given a security parameter λ and a ring dimension N, choose an RNS modulus chain $q_0, q_1, \ldots, q_L$, a secret key distribution χ_s, an error distribution χ_e, and a scaling factor $\Delta \in \mathbb{Z}$. Output a public parameter $\mathsf{pp} = (N, \Delta, Q_L = \prod_{i=0}^{L} q_i, \chi_s, \chi_e)$.

- $\underline{\mathtt{KeyGen}}(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{pk})$: Given a public parameter pp and a set of automorphisms, generate a secret key sk and a public key pk as follows, where $\mathsf{pk} = (\mathsf{ek}, \mathsf{rlk}, \mathsf{rtk}, \mathsf{cjk})$, and $\vec{g} = (Q_L/q_0, \ldots, Q_L/q_{L-1}) \in \mathbb{Z}^L$.

  Secret key. Sample $\boldsymbol{s} \leftarrow \chi_s$, and set $\mathsf{sk} = \boldsymbol{s} \in R$.

  Encryption key. Sample $\boldsymbol{u}_{\mathsf{ek}} \leftarrow \mathcal{U}(R_{Q_L})$, $\boldsymbol{e}_{\mathsf{ek}} \leftarrow \chi_e$, and set $\mathsf{ek} = (-\boldsymbol{u}_{\mathsf{ek}}\boldsymbol{s} + \boldsymbol{e}_{\mathsf{ek}},\ \boldsymbol{u}_{\mathsf{ek}}) \in R_{Q_L}^2$.

  Relinearization key. Sample $\vec{\boldsymbol{u}}_{\mathsf{rlk}} \leftarrow \mathcal{U}(R_{Q_L}^L)$, $\vec{\boldsymbol{e}}_{\mathsf{rlk}} \leftarrow \chi_e^L$, and set $\mathsf{rlk} = (-\boldsymbol{s} \cdot \boldsymbol{u}_{\mathsf{rlk}} + \boldsymbol{s}^2 \cdot \vec{g} + \vec{\boldsymbol{e}}_{\mathsf{rlk}},\ \vec{\boldsymbol{u}}_{\mathsf{rlk}}) \in R_{Q_L}^{2L}$.

  Rotation key. Sample $\vec{\boldsymbol{u}}_{\mathsf{rtk}} \leftarrow \mathcal{U}(R_{Q_L}^L)$, $\vec{\boldsymbol{e}}_{\mathsf{rtk}} \leftarrow \chi_e^L$, and set $\mathsf{rtk} = (-\boldsymbol{s} \cdot \boldsymbol{u}_{\mathsf{rtk}} + \varphi(\boldsymbol{s}) \cdot \vec{g} + \vec{\boldsymbol{e}}_{\mathsf{rtk}},\ \vec{\boldsymbol{u}}_{\mathsf{rtk}}) \in R_{Q_L}^{2L}$, where $\varphi : X \mapsto X^5$ is an automorphism over R.

  Conjugation key. Sample $\vec{\boldsymbol{u}}_{\mathsf{cjk}} \leftarrow \mathcal{U}(R_{Q_L}^L)$, $\vec{\boldsymbol{e}}_{\mathsf{cjk}} \leftarrow \chi_e^L$, and set $\mathsf{cjk} = (-\boldsymbol{s} \cdot \boldsymbol{u}_{\mathsf{cjk}} + \psi(\boldsymbol{s}) \cdot \vec{g} + \vec{\boldsymbol{e}}_{\mathsf{cjk}},\ \vec{\boldsymbol{u}}_{\mathsf{cjk}}) \in R_{Q_L}^{2L}$, where $\psi : X \mapsto X^{-1}$ is an automorphism over R.

- $\underline{\mathtt{Enc}_{\mathsf{sk}}}(\boldsymbol{m}) \to \mathsf{ct}$: Given a secret key $\mathsf{sk} = \boldsymbol{s}$ and a plaintext $\boldsymbol{m} \in R$, output a ciphertext $\mathsf{ct} = (-\boldsymbol{a} \cdot \boldsymbol{s} + \boldsymbol{m} + \boldsymbol{e}, \boldsymbol{a}) \in R_{Q_L}^2$, where $\boldsymbol{a} \leftarrow \mathcal{U}(R_{Q_L})$, and $\boldsymbol{e} \leftarrow \chi_e$.

- $\underline{\mathtt{Dec}_{\mathsf{sk}}}(\mathsf{ct}) \to \boldsymbol{m}$: Given a secret key $\mathsf{sk} = \boldsymbol{s}$ and a ciphertext $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1) \in R_{Q_\ell}^2$, output a plaintext $\boldsymbol{m} = \boldsymbol{c}_0 + \boldsymbol{c}_1\boldsymbol{s} \pmod{Q_0}$.

We note that the CKKS scheme achieves IND-CPA security under the computational hardness of $\mathsf{RLWE}_{Q_L,\chi_s,\chi_e}$, assuming circular security [69]. In practice, we usually use a sparse ternary distribution $\mathsf{HWT}(h)$ for a secret key distribution χ_s, where each coefficient is in $\{-1, 0, 1\}$ and the Hamming weight is bounded by h, since it provides smaller noise growth after each homomorphic operation and efficient bootstrapping performance [28]. For an error distribution χ_e, we typically use a discrete Gaussian distribution.

Homomorphic Operations. The CKKS scheme supports homomorphic addition, multiplication, automorphism, and rounding operations over R in an approximate manner. For automorphism operations, there are two types: rotation and conjugation, which evaluate the automorphisms $\varphi : X \mapsto X^5$ and $\psi : X \mapsto X^{-1}$, respectively. In the CKKS scheme, one can emulate arithmetic over $\mathbb{R}[X]/(X^N+1)$ in fixed-point arithmetic over $R = \mathbb{Z}[X]/(X^N+1)$ thanks to the homomorphic rounding operations, which is a distinctive feature compared to other HE schemes [14, 15, 32, 41].

Definition 6 (External Product). *Let $Q_\ell = \prod_{i=0}^{\ell} q_i$ for $0 \leq \ell < L$. An external product is a binary operation $\boxdot : R_{Q_\ell} \times R_{Q_L}^L \to R_{Q_\ell}$, defined as follows for $\boldsymbol{a} \in R_{Q_\ell}$ and $\vec{\boldsymbol{u}} = (\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{L-1}) \in R_{Q_L}^L$.*

$$\boldsymbol{a} \boxdot \vec{\boldsymbol{u}} = \left\lfloor \frac{1}{q_L} \Big(\sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} \cdot \boldsymbol{u}_i \Big) \right\rceil \pmod{Q_\ell}$$

By an abuse of notation, we write $\boldsymbol{a} \boxdot (\vec{\boldsymbol{u}}_0, \vec{\boldsymbol{u}}_1) = (\boldsymbol{a} \boxdot \vec{\boldsymbol{u}}_0, \boldsymbol{a} \boxdot \vec{\boldsymbol{u}}_1)$ for $\vec{\boldsymbol{u}}_0, \vec{\boldsymbol{u}}_1 \in R_{Q_L}^L$.

- $\underline{\texttt{Add}}(\mathsf{ct}, \mathsf{ct}') \to \mathsf{ct}_{add}$: Given ciphertexts $\mathsf{ct}, \mathsf{ct}' \in R^2_{Q_\ell}$, output a ciphertext $\mathsf{ct}_{add} = \mathsf{ct} + \mathsf{ct}' \in R^2_{Q_\ell}$.
- $\underline{\texttt{Mul}_{\mathsf{rlk}}}(\mathsf{ct}, \mathsf{ct}') \to \mathsf{ct}_{mul}$: Given ciphertexts $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1), \mathsf{ct}' = (\boldsymbol{c}'_0, \boldsymbol{c}'_1) \in R^2_{Q_\ell}$ for $\ell < L$, let $\boldsymbol{d}_0 = \boldsymbol{c}_0\boldsymbol{c}'_0$, $\boldsymbol{d}_1 = \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1$, and $\boldsymbol{d}_2 = \boldsymbol{c}_1\boldsymbol{c}'_1$. Output a ciphertext $\mathsf{ct}_{mul} = (\boldsymbol{d}_0, \boldsymbol{d}_1) + \boldsymbol{d}_2 \boxdot \mathsf{rlk} \in R^2_{Q_\ell}$
- $\underline{\texttt{Rot}_{\mathsf{rtk}}}(\mathsf{ct}) \to \mathsf{ct}'$: Given ciphertexts $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1) \in R^2_{Q_\ell}$ for $\ell < L$, output a ciphertext $\mathsf{ct}' = (\varphi(\boldsymbol{c}_0), 0) + \varphi(\boldsymbol{c}_1) \boxdot \mathsf{rtk} \in R^2_{Q_\ell}$.
- $\underline{\texttt{Conj}_{\mathsf{cjk}}}(\mathsf{ct}) \to \mathsf{ct}'$: Given ciphertexts $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1) \in R^2_{Q_\ell}$ for $\ell < L$, output a ciphertext $\mathsf{ct}' = (\psi(\boldsymbol{c}_0), 0) + \psi(\boldsymbol{c}_1) \boxdot \mathsf{cjk} \in R^2_{Q_\ell}$.
- $\underline{\texttt{Round}}(\mathsf{ct}) \to \mathsf{ct}'$: Given a ciphertext $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1) \in R^2_{Q_\ell}$ for $\ell \geq 1$, output a ciphertext $\mathsf{ct}' = (\lfloor \boldsymbol{c}_0/q_\ell \rceil, \lfloor \boldsymbol{c}_1/q_\ell \rceil) \in R^2_{Q_{\ell-1}}$.

Since homomorphic operations in CKKS work in an approximate manner, small errors are introduced after each homomorphic operation. To quantify these errors, we use the following notation.

Definition 7 (Ciphertext Error). *For a CKKS ciphertext* $\mathsf{ct} \in R^2_{Q_\ell}$*, a plaintext* $\boldsymbol{m} \in R$*, and a secret key* sk*, the ciphertext error of* $(\mathsf{ct}, \boldsymbol{m}, \mathsf{sk})$ *is defined as follows.*

$$\texttt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) = \|\texttt{Dec}_{\mathsf{sk}}(\mathsf{ct}) - \boldsymbol{m}\|_\infty$$

In Appendix A, we provide a detailed analysis of error growth for each homomorphic operation. These analyses imply that, given the upper bounds for the Hamming weight of the secret key sk, the errors used in generating the public keys pk, and the errors and plaintexts of the input ciphertexts, we can estimate an error bound for the final output ciphertext when homomorphically evaluating a complicated circuit in CKKS, as described below.

Theorem 2 (Error Bound Estimation). *Let* $\mathtt{C} : R^k \to R$ *be a* k*-ary admissible circuit for the CKKS scheme, i.e., a circuit that can be evaluated using addition, multiplication, automorphism, and rounding operations over* R*. Let* $\mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1}$ *and* $\boldsymbol{m}_0, \ldots, \boldsymbol{m}_{k-1}$ *be such that* $\texttt{Err}_{\mathsf{sk}}(\mathsf{ct}_i, \boldsymbol{m}_i) \leq B_e$*, and* $\|\boldsymbol{m}_i\|_\infty \leq B_m$*. Furthermore, let* $\|\vec{\boldsymbol{e}}_{\mathsf{rlk}}\|_\infty \leq B_{\mathsf{rlk}}$*,* $\|\vec{\boldsymbol{e}}_{\mathsf{rtk}}\|_\infty \leq B_{\mathsf{rtk}}$*,* $\|\vec{\boldsymbol{e}}_{\mathsf{cjk}}\|_\infty \leq B_{\mathsf{cjk}}$*, and* $\|\boldsymbol{s}\|_\infty \leq h$ *for a public key* $\mathsf{pk} = (\mathsf{ek}, \mathsf{rlk}, \mathsf{rtk}, \mathsf{cjk})$ *and a secret key* $\mathsf{sk} = \boldsymbol{s}$*. Then, there exists an algorithm* $\texttt{Estim}$ *such that the following holds.*

$$\texttt{Err}\Big(\texttt{Eval}_{\mathsf{pk}}(\mathtt{C}; \mathsf{ct}_0, \cdots, \mathsf{ct}_{k-1}), \mathtt{C}(\boldsymbol{m}_0, \ldots, \boldsymbol{m}_{k-1})\Big) \leq \texttt{Estim}(\mathtt{C}; h, B_{\mathsf{rlk}}, B_{\mathsf{rtk}}, B_{\mathsf{cjk}}, B_e, B_m)$$

Proof. We defer the proof to Appendix A.

Packing Method. The CKKS scheme supports the message space $\mathbb{C}^{N/2}$ through an isomorphism $\iota : \mathbb{R}[X]/(X^N + 1) \to \mathbb{C}^{N/2}$, called the canonical embedding. Combined with fixed-point emulation of $\mathbb{R}[X]/(X^N + 1)$ in R, the CKKS scheme supports SIMD operations in the message space $\mathbb{C}^{N/2}$ in an approximate manner. The following algorithms describe how to convert a message in $\mathbb{C}^{N/2}$ to a plaintext in $\mathbb{R}$ and vice versa.

- $\underline{\mathtt{Pack}}(\vec{m}) \to \boldsymbol{m}$: Given a message vector $\vec{m} \in \mathbb{C}^{N/2}$, output a plaintext $\boldsymbol{m} = \lfloor \Delta \cdot \iota^{-1}(\vec{m}) \rceil \in R$
- $\underline{\mathtt{Unpack}}(\boldsymbol{m}) \to \vec{m}$: Given a plaintext $\boldsymbol{m} \in R$, output a message vector $\vec{m} = \frac{1}{\Delta} \cdot \iota(\boldsymbol{m}) \in \mathbb{C}^{N/2}$

2.6 Interactive Argument of Knowledge

We define an interactive argument of knowledge with the honest verifier zero-knowledge (HVZK) property as follows.

Definition 8 (Interactive Argument of Knowledge). *Let* $\mathtt{\Pi} = (\mathtt{Setup}, \mathtt{P}, \mathtt{V})$ *be an interactive protocol between a prover* $\mathtt{P}$ *and a verifier* $\mathtt{V}$*.* $\mathtt{\Pi}$ *is called an argument of knowledge for a relation* R *if it satisfies the following properties.*

Completeness. *For all PPT adversary* $\mathtt{A}$*, the following holds.*

$$\Pr\left[\begin{array}{r|c} \langle \mathtt{P}(\mathsf{pp},\mathsf{x},\mathsf{w}),\mathtt{V}(\mathsf{pp},\mathsf{x})\rangle = 1\vee & \mathsf{pp} \leftarrow \mathtt{Setup}(1^{\lambda}) \\ (\mathsf{x},\mathsf{w}) \notin \mathsf{R} & (\mathsf{x},\mathsf{w}) \leftarrow \mathtt{A}(\mathsf{pp}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Soundness. *For every PPT adversary* $\mathtt{A} = (\mathtt{A}_1, \mathtt{A}_2)$*, the following holds.*

$$\Pr\left[\begin{array}{r|c} \langle \mathtt{A}_2(\mathsf{pp},\mathsf{st},\mathsf{x}),\mathtt{V}(\mathsf{pp},\mathsf{x})\rangle = 1\wedge & \mathsf{pp} \leftarrow \mathtt{Setup}(1^{\lambda}) \\ \mathsf{x} \notin \mathsf{L}(\mathsf{R}) & (\mathsf{st},\mathsf{x}) \leftarrow \mathtt{A}_1(\mathsf{pp}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

Knowledge Soundness. *For every PPT adversary* $\mathtt{A} = (\mathtt{A}_1, \mathtt{A}_2)$*, there exists a PPT extractor* $\mathtt{E}$ *such that, given oracle access to* $\mathtt{A}$*, the following holds.*

$$\Pr\left[\begin{array}{r|c} & \mathsf{pp} \leftarrow \mathtt{Setup}(1^{\lambda}) \\ \langle \mathtt{A}_2(\mathsf{pp},\mathsf{st},\mathsf{x}),\mathtt{V}(\mathsf{pp},\mathsf{x})\rangle = 1\wedge & (\mathsf{st},\mathsf{x}) \leftarrow \mathtt{A}_1(\mathsf{pp}) \\ (\mathsf{x},\mathsf{w}) \notin \mathsf{R} & \mathsf{w} \leftarrow \mathtt{E}^{\mathtt{A}}(\mathsf{pp},\mathsf{x}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

$\mathtt{\Pi}$ *is called honest verifier zero-knowledge (HVZK) if the following holds.*

Honest Verifier Zero-knowledge. *For every PPT adversary* $\mathtt{A} = (\mathtt{A}_1, \mathtt{A}_2)$*, there exists a PPT simulator* $\mathtt{S}$ *such that the following holds, where* $\mathtt{View}$ *outputs the verifier's view.*

$$\begin{aligned} &\left| \Pr\left[\begin{array}{r|c} \mathtt{A}_2(\mathsf{ck},\mathsf{view}) = 1\wedge & \mathsf{pp} \leftarrow \mathtt{Setup}(1^{\lambda}) \\ & (\mathsf{x},\mathsf{w}) \leftarrow \mathtt{A}_1(\mathsf{pp}) \\ (\mathsf{x},\mathsf{w}) \in \mathsf{R} & \mathsf{view} \leftarrow \mathtt{S}(\mathsf{pp},\mathsf{x}) \end{array}\right] \right. \\ &\left. - \Pr\left[\begin{array}{r|r} \mathtt{A}_2(\mathsf{ck},\mathsf{view}) = 1\wedge & \mathsf{pp} \leftarrow \mathtt{Setup}(1^{\lambda}) \\ & (\mathsf{x},\mathsf{w}) \leftarrow \mathtt{A}_1(\mathsf{pp}) \\ (\mathsf{x},\mathsf{w}) \in \mathsf{R} & \mathsf{view} \leftarrow \mathtt{View}\big(\mathtt{P}(\mathsf{pp},\mathsf{x},\mathsf{w}),\mathtt{V}(\mathsf{pp},\mathsf{x})\big) \end{array}\right] \right| \leq \mathsf{negl}(\lambda) \end{aligned}$$

Additionally, $\mathtt{\Pi}$ *is called public coin if all messages from the honest verifier can be computed as a deterministic function of a random public input.*

Next, we review the definition of a polynomial interactive oracle proof (PIOP) in [17, 31], which is a special class of interactive arguments of knowledge. In the following definition, we restrict polynomials to be univariate, but it can be generalized to the multivariate case, as defined in [24].

Definition 9 (Polynomial Interactive Oracle Proof). *Let* $\Pi = (\mathtt{Setup}, \mathtt{P}, \mathtt{V})$ *be an interactive public coin argument of knowledge for a relation* R. Π *is called a polynomial interactive oracle proof (PIOP), which satisfies the followings.*

- *Every message from the prover is a polynomial oracle* $([\![\boldsymbol{f}]\!], d)$, *where* $\boldsymbol{f} \in \mathbb{Z}_p[X]$ *of degree* $\leq d$.
- *Every message from the verifier is a random challenge.*
- *The verifier has oracle access to evaluations of the prover's polynomials at arbitrary points.*

Π *is called a honest verifier zero-knowledge PIOP if it is an HVZK argument of knowledge, where* $\mathtt{View}$ *outputs the messages from the verifier and the responses to polynomial evaluation queries.*

For PIOPs, proving the soundness property is sufficient to prove their knowledge soundness, as stated below.

Lemma 1 (Lemma 2.3 in [24]). *If a PIOP satisfies the soundness property, then it also satisfies knowledge soundness with an extractor that runs in time* $O(|\mathsf{w}|)$.

A PIOP can be compiled into a succinct non-interactive argument of knowledge (SNARK) with the aid of a polynomial commitment scheme [31] and the Fiat-Shamir heuristic [43]. This framework allows a modular approach in designing proof systems: one first conceptually designs a proof system in PIOP form, then concretely instantiates it with a polynomial commitment scheme. We leave the details of PIOP compilation in Appendix B.

3 PIOP Toolbox

In this section, we introduce several PIOPs for handling relations over the vector space $\mathbb{Z}_p^N$, where $\mathbb{Z}_p$ is a prime field. These PIOPs are later used to verify the well-formedness of CKKS public keys and ciphertexts in our secure homomorphic evaluation protocol.

Prior works [9, 52] have proposed efficient PIOPs for linear relations, arithmetic satisfiability, and L^∞-norm bounds in $\mathbb{Z}_p^N$. However, these protocols do not suffice for proving the sparsity of the secret key, which is essential for the CKKS scheme. To address this, we design a new PIOP that proves an L^2-norm bound for witnesses in $\mathbb{Z}_p^N$ by generalizing the univariate sumcheck protocol from [9].

3.1 Polynomial Encoding for Vectors

To commit to a vector in $\mathbb{Z}_p^N$ in univariate PIOPs, we use the following encoding method, which transforms a vector into a polynomial, allowing the prover to construct a polynomial oracle. For polynomial interpolation, we use a multiplicative subgroup $\mathbb{H} = \{h_0, \dots, h_{N-1}\} \subset \mathbb{Z}_p^\times$ of order N.

- $\underline{\mathtt{Ecd}}(\vec{w}) \to \boldsymbol{w}$: Given a vector $\vec{w} \in \mathbb{Z}_p^N$, output the polynomial $\boldsymbol{w} \in \mathbb{Z}_p^{<N}[X]$ such that $\boldsymbol{w}(h_i) = w_i$ for $0 \leq i < N$.
- $\underline{\mathtt{REcd}}(\vec{w}) \to \hat{\boldsymbol{w}}$: Given a vector $\vec{w} \in \mathbb{Z}_p^N$, uniformly sample a polynomial $\hat{\boldsymbol{w}} \in \mathbb{Z}_p^{<2N}[X]$ such that $\hat{\boldsymbol{w}}(h_i) = w_i$ for $0 \leq i < N$.
- $\underline{\mathtt{Dcd}}(\boldsymbol{w}) \to \vec{w}$: Given a polynomial $\boldsymbol{w} \in \mathbb{Z}_p[X]$, output a vector $\vec{w} = (\boldsymbol{w}(h_0), \dots, \boldsymbol{w}(h_{N-1}))$.

For $\vec{w} \in \mathbb{Z}_p^N$, there exists a unique polynomial $\boldsymbol{w} \in \mathbb{Z}_p^{<N}[X]$ corresponding to $\mathtt{Ecd}(\vec{w})$. However, evaluating $\boldsymbol{w}$ at any point reveals information about $\vec{w}$, potentially compromising the zero-knowledge property in PIOPs. In contrast, for $\hat{\boldsymbol{w}}$, multiple valid candidates exist, allowing us to select one randomly. As a result, for $\alpha \notin \mathbb{H}$, the evaluation $\hat{\boldsymbol{w}}(\alpha)$ remains independent of $\vec{w}$ for up to $N-1$ evaluations due to bounded independence. For further details, see [9].

Once we encode vectors into polynomials using the above encoding method, we can utilize the following properties of polynomial encodings to prove statements about witness vectors.

Lemma 2 (Univariate Sum Check [9]). *Let $\boldsymbol{f} \in \mathbb{Z}_p[X]$, $\mathbb{H} \subseteq \mathbb{Z}_p^\times$ be a multiplicative subgroup of order N, and $\boldsymbol{z}_\mathbb{H} = \prod_{h\in\mathbb{H}}(X-h) = (X^N - 1)$. Then, $\sum_{h\in\mathbb{H}} \boldsymbol{f}(h) = \mu$ holds if and only if there exist polynomials $\boldsymbol{q} \in \mathbb{Z}_p[X]$ and $\boldsymbol{r} \in \mathbb{Z}_p^{<N-1}[X]$ such that $\boldsymbol{f} = \boldsymbol{q} \cdot \boldsymbol{z}_\mathbb{H} + \boldsymbol{r} \cdot X + N^{-1} \cdot \mu \pmod p$ holds.*

Lemma 3 (Univariate Zero Test [9]). *Let $\boldsymbol{f} \in \mathbb{Z}_p[X]$, $\mathbb{H} \subseteq \mathbb{Z}_p^\times$ be a multiplicative subgroup of order N, and $\boldsymbol{z}_\mathbb{H} = \prod_{h\in\mathbb{H}}(X-h) = (X^N - 1)$. Then, $\boldsymbol{f}(h) = 0$ for all $h \in \mathbb{H}$ if and only if there exists a polynomial $\boldsymbol{q} \in \mathbb{Z}_p[X]$ such that $\boldsymbol{f} = \boldsymbol{q} \cdot \boldsymbol{z}_\mathbb{H} \pmod p$ holds.*

3.2 PIOPs in [9, 52]

We briefly review the PIOPs for relations over the vector space $\mathbb{Z}_p^N$ from [9, 52]. For simplicity, we outline only their functionality here, while full protocol descriptions are provided in Appendix C.

PIOP for Linear Relatinos. For witness vectors $\vec{a}_i, \vec{b}_i \in \mathbb{Z}_p^N$ satisfying the linear relations $\vec{b}_i = M_i \vec{a}_i$ for public matrices $M_i \in \mathbb{Z}_p^{N\times N}$, where $0 \leq i < k$, their validity can be proven using the following PIOP.

$$\Pi_{\texttt{Lin}}(M_0, \ldots, M_{k-1}; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!]; [\![\hat{\boldsymbol{b}}_0]\!], \ldots, [\![\hat{\boldsymbol{b}}_{k-1}]\!])$$

Public input: matrices $M_i \in \mathbb{Z}_p^{N \times N}$ for $0 \leq i < k$.
Witness: vectors $\vec{a}_i = \texttt{Dcd}(\hat{\boldsymbol{a}}_i)$ and $\vec{b}_i = \texttt{Dcd}(\hat{\boldsymbol{b}}_i)$ for $0 \leq i < k$, where $\hat{\boldsymbol{a}}_i, \hat{\boldsymbol{b}}_i \in \mathbb{Z}_p^{<2N}[X]$.
Statement: $\vec{b}_i = M_i \vec{a}_i$ for $0 \leq i < k$.

Fig. 1: Functionality of the PIOP for batched linear relations

PIOP for Arithmetic Constraints. An arithmetic constraint over $\mathbb{Z}_p^N$, where addition and multiplication are component-wise, can be represented as a multivariate polynomial $\vec{\texttt{C}}$ over the product ring $\mathbb{Z}_p^N$. The satisfiability of $\vec{\texttt{C}}$ for witness vectors in $\mathbb{Z}_p^N$ can be verified using the following PIOP.

$$\Pi_{\texttt{AC}}(\vec{\texttt{C}}; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!])$$

Public input: k-ary circuit $\vec{\texttt{C}} \in (\mathbb{Z}_p^N)[\vec{X}_0, \ldots, \vec{X}_{k-1}]$ of degree d.
Witness: vectors $\vec{a}_i = \texttt{Dcd}(\boldsymbol{a}_i)$ for $0 \leq i < k$, where $\hat{\boldsymbol{a}}_i \in \mathbb{Z}_p^{<2N}[X]$.
Statement: $\vec{\texttt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}) = 0$.

Fig. 2: Functionality of the PIOP for arithmetic constraints

PIOP for L^∞-Norm Constraints. For witness vectors $\vec{a}_0, \ldots, \vec{a}_{k-1} \in \mathbb{Z}_p^N$, one can prove their L^∞-norm upper bound in a batched manner using the following PIOP.

$$\Pi_{L^\infty}(B; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!])$$

Public input: norm bound B.
Witness: vectors $\vec{a}_i = \texttt{Dcd}(\hat{\boldsymbol{a}}_i)$, where $\hat{\boldsymbol{a}}_i \in \mathbb{Z}_p^{<2N}[X]$ for $0 \leq i < k$.
Statement: $\|\vec{a}_i\|_\infty \leq B$ for $0 \leq i < k$.

Fig. 3: Functionality of the PIOP for L^∞-norm constraints

3.3 Our PIOP for L^2-Norm Constraints

We begin by generalizing the univariate sumcheck protocol from [9], which verifies the relation $\sum_{h \in \mathbb{H}} \boldsymbol{a}(h) = 0$ for a committed polynomial $\boldsymbol{a}$. In the context of polynomial encoding, this condition is equivalent to the inner product relation $\left\langle \vec{a}, \vec{1} \right\rangle = 0$, where $\vec{a} = \texttt{Dcd}(\boldsymbol{a})$.

However, to prove upper bounds on the L^2-norm, we require a more general form of this inner product relation. Specifically, instead of $\vec{a}$, we consider $\vec{\texttt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1})$, where $\vec{\texttt{C}}$ is a multivariate polynomial over the product ring $\mathbb{Z}_p^N$, and $\vec{a}_0, \ldots, \vec{a}_{k-1} \in \mathbb{Z}_p^N$. Thanks to polynomial encoding, this generalized relation can still be verified using Lemma 2. Below, we extend the univariate sumcheck protocol from [9] to prove the inner product relation $\left\langle \vec{\mathtt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}), \vec{1} \right\rangle = 0$ for witness vectors $\vec{a}_0, \ldots, \vec{a}_{k-1}$.

Theorem 3. *Let $\hat{\boldsymbol{a}}_i \leftarrow \mathtt{REcd}(\vec{a}_i)$ for $0 \leq i < k$, where $\left\langle \vec{\mathtt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}), \vec{1} \right\rangle = 0$. Then, an interactive protocol $\Pi_{\mathtt{IP}}$ described in Fig. 4 is an HVZK PIOP with a soundness error of $\frac{O(dN)}{p-N}$.*

$$\Pi_{\mathtt{IP}}(\vec{\mathtt{C}}; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!])$$

Instance: k-ary circuit $\vec{\mathtt{C}} \in (\mathbb{Z}_p^N)[\vec{X}_0, \ldots, \vec{X}_{k-1}]$ of degree d.
Witness: vectors $\vec{a}_i = \mathtt{Dcd}(\hat{\boldsymbol{a}}_i)$ for $0 \leq i < k$, where $\hat{\boldsymbol{a}}_i \in \mathbb{Z}_p^{<2N}[X]$.
Statement: $\left\langle \vec{\mathtt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}), \vec{1} \right\rangle = 0$.

1. Both $\mathtt{P}$ and $\mathtt{V}$ compute $\boldsymbol{\mathtt{C}} \in (\mathbb{Z}_p[X])[\boldsymbol{X}_0, \ldots, \boldsymbol{X}_{k-1}]$ of degree d, which is obtained by applying $\mathtt{Ecd}$ to each coefficient of $\vec{\mathtt{C}}$.
2. The prover $\mathtt{P}$ samples a random polynomial $\boldsymbol{g}$ of degree $\leq d(2N-1)+N-1$, computes the summation $\mu = \sum_{h \in \mathbb{H}} \boldsymbol{g}(h)$, and sends polynomial oracles $([\![\boldsymbol{g}]\!], d(2N-1)+N-1)$ and μ to the verifier $\mathtt{V}$.
3. The verifier $\mathtt{V}$ sends a random point $\beta \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$.
4. The prover $\mathtt{P}$ computes $\boldsymbol{q}$ and $\boldsymbol{r}$, which satisfy the following.
$$\boldsymbol{q} \cdot \boldsymbol{z}_{\mathbb{H}} + \boldsymbol{r} \cdot X + N^{-1} \cdot \mu = \boldsymbol{g} + \beta \cdot \boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})$$
Then, $\mathtt{P}$ sends polynomial oracles $([\![\boldsymbol{q}]\!], d(2N-1)-1)$ and $([\![\boldsymbol{r}]\!], N-2)$ to the verifier $\mathtt{V}$.
5. The verifier $\mathtt{V}$ gets evaluations $\hat{\boldsymbol{a}}_i(\alpha)$, $\boldsymbol{g}(\alpha)$, $\boldsymbol{q}(\alpha)$, and $\boldsymbol{r}(\alpha)$ by accessing polynomial oracles $[\![\hat{\boldsymbol{a}}_i]\!]$, $[\![\boldsymbol{g}]\!]$, $[\![\boldsymbol{q}]\!]$, and $[\![\boldsymbol{r}]\!]$ at a random point $\alpha \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$. Then, $\mathtt{V}$ checks whether the following holds.
$$\boldsymbol{q}(\alpha) \cdot \boldsymbol{z}_{\mathbb{H}}(\alpha) + \boldsymbol{r}(\alpha) \cdot \alpha + N^{-1} \cdot \mu = \boldsymbol{g}(\alpha) + \beta \cdot \big(\boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})\big)(\alpha)$$

Fig. 4: PIOP for an inner product relation

Proof. We first note that the statement $\left\langle \vec{\mathtt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}), \vec{1} \right\rangle = 0$ is equivalent to $\sum_{h \in \mathbb{H}} \big(\boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})\big)(h) = 0$ due to polynomial encoding.

Completeness. If $\sum_{h \in \mathbb{H}} \big(\boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})\big)(h) = 0$, then by Lemma 2, there exist polynomials $\boldsymbol{q}$ and $\boldsymbol{r}$ such that $\boldsymbol{q} \cdot \boldsymbol{z}_{\mathbb{H}} + \boldsymbol{r} \cdot X + N^{-1} \cdot \mu = \boldsymbol{g} + \beta \cdot \boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})$. Thus, the protocol is perfectly complete for valid witnesses.

Soundness. By Schwart-Zippel lemma, $\boldsymbol{q}(\alpha) \cdot \boldsymbol{z}_{\mathbb{H}}(\alpha) + \boldsymbol{r}(\alpha) \cdot \alpha + N^{-1} \cdot \mu = \boldsymbol{g}(\alpha) + \beta \cdot \big(\boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})\big)(\alpha)$ holds implies $\boldsymbol{q}$ and $\boldsymbol{r}$ such that $\boldsymbol{q} \cdot \boldsymbol{z}_{\mathbb{H}} + \boldsymbol{r} \cdot X + N^{-1} \cdot \mu = \boldsymbol{g} + \beta \cdot \boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})$, except with probability at most $O(dN)/(p-N)$. Thus, soundness holds with a soundness error of at most $\leq O(dN)/(p-N)$.

HVZK. It suffices to simulate the values of α, β, μ, $\hat{\boldsymbol{a}}_i(\alpha)$, $\{\boldsymbol{g}(\alpha)\}_{0\le i<k}$, $\boldsymbol{q}(\alpha)$, and $\boldsymbol{r}(\alpha)$ for $0 \le i < k$. The simulator operates as follows.

1. Sample a random polynomial $\boldsymbol{g}'$ of degree $\le d(2N-1)+N-1$, and decompose it into $\boldsymbol{q}\cdot \boldsymbol{z}_{\mathbb{H}} + \boldsymbol{r}\cdot X + N^{-1}\cdot \mu$.
2. Sample random polynomials $\hat{\boldsymbol{a}}_i$ of degree $< 2N$ for $0 \le i < k$, and $\alpha, \beta \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$.
3. Set $\boldsymbol{g}(\alpha) = \boldsymbol{q}(\alpha)\cdot \boldsymbol{z}_{\mathbb{H}}(\alpha) + \boldsymbol{r}(\alpha)\cdot\alpha + N^{-1}\cdot\mu - \beta\cdot\big(\mathtt{C}(\hat{\boldsymbol{a}}_0,\dots,\hat{\boldsymbol{a}}_{k-1})\big)(\alpha)$.
4. Output α, β, μ, $\{\hat{\boldsymbol{a}}_i(\alpha)\}_{0\le i<k}$, $\boldsymbol{g}(\alpha)$, $\boldsymbol{q}(\alpha)$, and $\boldsymbol{r}(\alpha)$.

The distribution of the above simulator is identical to the honest verifier's view, because the distributions of $\boldsymbol{q}, \boldsymbol{r}$, μ follow the same distribution as in the actual protocol due to random masking by $\boldsymbol{g}$, $\{\hat{\boldsymbol{a}}_i(\alpha)\}_{0\le i<k}$ appear uniformly random due to bounded independence, and $\boldsymbol{g}(\alpha)$ is fully determined by these values. □

Based on our protocol $\Pi_{\mathtt{IP}}$ for inner product relations, we now present our PIOP for proving upper bounds on the L^2-norm of witness vectors in $\mathbb{Z}_p^N$. We start with the observation that $\|\vec{a}\|_2^2 = \langle \vec{a}, \vec{a}\rangle$, so it suffices to prove the range of $\langle \vec{a}, \vec{a}\rangle$. However, since arithmetic is performed in $\mathbb{Z}_p$, what we can actually prove is the range of $[\langle \vec{a}, \vec{a}\rangle]_p$. To address this, we use Π_{L^∞}, which proves the bound of $\|\vec{a}\|_\infty$. Specifically, $\|\vec{a}\|_\infty \le B_\infty$ for some $B_\infty > 0$ implies $\|\vec{a}\|_2 \le B_\infty^2 N$. Therefore, unless $B_\infty^2 N > p$, we can ensure that $[\langle \vec{a}, \vec{a}\rangle]_p = \langle \vec{a}, \vec{a}\rangle$. Below, we describe our PIOP that proves the range of L^2-norm modulo p, which can handle multiple witness vectors in a batched manner.

Theorem 4. *Let $\hat{\boldsymbol{a}}_i \leftarrow \mathtt{REcd}(\vec{a}_i)$, where $0 \le [\|\vec{a}_i\|_2^2]_p \le B$ for $0 \le i < k$. Then, an interactive protocol Π_{L^2} described in Fig. 5 is an HVZK PIOP with a soundness error of $\frac{O(k+N)}{p-N}$.*

Proof. We first show that the statement $0 \le [\|\vec{a}_i\|_2^2]_p \le B$ is equivalent to $\left\langle \vec{a}_i \odot \vec{a}_i - \vec{b}\odot \vec{u}_i, \vec{1}\right\rangle = 0 \wedge \vec{u}\odot(\vec{u}_i - \vec{c}) = \vec{0}$ for some $\vec{u}_i \in \mathbb{Z}_p^N$ for $0 \le i < k$. Suppose $0 \le [\|\vec{a}_i\|_2^2]_p \le B$ holds, which implies $\langle \vec{a}_i, \vec{a}_i\rangle = U_i \pmod p$ for some $0 \le U_i \le B$. For such U_i, there exist $u_{i,0},\dots,u_{i,\ell-1} \in \{0,1\}$ such that $U_i = \sum_{j=0}^{\ell-1} u_{i,j}B_j$. Then, for $\vec{u}_i = (u_{i,0},\dots,u_{i,\ell-1},0,\dots,0) \in \mathbb{Z}_p^N$, it holds that $\langle \vec{a}_i, \vec{a}_i\rangle = \left\langle \vec{u}_i, \vec{b}\right\rangle \pmod p \wedge \vec{u}_i\odot(\vec{u}_i - \vec{c}) = \vec{0}$, and note that $\langle \vec{a}_i, \vec{a}_i\rangle = \left\langle \vec{u}_i, \vec{b}\right\rangle \iff \left\langle \vec{a}_i\odot\vec{a}_i, \vec{1}\right\rangle = \left\langle \vec{u}_i\odot\vec{b}, \vec{1}\right\rangle \iff \left\langle \vec{a}_i\odot\vec{a}_i - \vec{u}_i\odot\vec{b}, \vec{1}\right\rangle$.

Conversely, suppose there exists $\vec{u}_i \in \mathbb{Z}_p^N$ such that $\left\langle \vec{a}_i\odot\vec{a}_i - \vec{b}\odot\vec{u}_i, \vec{1}\right\rangle = 0$ $\wedge\ \vec{u}_i\odot(\vec{u}_i - \vec{c}) = \vec{0}$. Then, $\vec{u}_i$ is of the form $(u_{i,0},\dots,u_{i,\ell-1},0,\dots,0) \in \mathbb{Z}_p^N$, and $\langle \vec{a}_i, \vec{a}_i\rangle = \sum_{j=0}^{\ell-1} u_{i,j}B_j \pmod p$. Note that $0 \le \sum_{j=0}^{\ell-1} u_{i,j}B_j \le B$ holds, so we have $0 \le [\|\vec{a}_i\|_2^2]_p \le B$.

Finally, by the Schwartz-Zippel lemma, $\sum_{i=0}^{k-1}\gamma^i\cdot\left\langle \vec{a}_i\odot\vec{a}_i - \vec{b}\odot\vec{u}_i, \vec{1}\right\rangle = 0 \wedge \sum_{i=0}^{k-1}\gamma^i\cdot\vec{u}_i\odot(\vec{u}_i - \vec{c}) = \vec{0}$ implies $\left\langle \vec{a}_i\odot\vec{a}_i - \vec{b}\odot\vec{u}_i, \vec{1}\right\rangle = 0 \wedge \vec{u}_i\odot(\vec{u}_i - \vec{c}) = \vec{0}$ for

$$\Pi_{L^2}(B; [\![\hat{\boldsymbol{a}}_0]\!], \cdots, [\![\hat{\boldsymbol{a}}_{k-1}]\!])$$

Public input: norm bound B
Witness: vectors $\vec{a}_i = \mathtt{Dcd}(\boldsymbol{a}_i)$, where $\boldsymbol{a}_i \in \mathbb{Z}_p^{<2N}[X]$ for $0 \le i < k$
Statement: $0 \le \left[\|\vec{a}_i\|_2^2\right]_p \le B$ for $0 \le i < k$.

1. The prover $\mathtt{P}$ and the verifier $\mathtt{V}$ decompose B into $B_0 = \lceil \frac{B}{2} \rceil$, $B_1 = \lceil \frac{B-B_0}{2} \rceil$, $B_2 = \lceil \frac{B-B_0-B_1}{2} \rceil, \ldots, B_{\ell-1} = 1$, where $\ell = \lfloor \log B \rfloor + 1$, and set $\vec{b} = (B_0, \ldots, B_{\ell-1}, 0, \ldots, 0)$ and $\vec{c} = (1, \ldots, 1, 0, \ldots, 0)$, where the first ℓ elements are nonzero.
2. The prover $\mathtt{P}$ computes $u_{i,0}, \ldots, u_{i,\ell-1} \in \{0,1\}$ such that $\langle \vec{a}_i, \vec{a}_i \rangle = \sum_{j=0}^{\ell-1} u_{i,j} \cdot B_i \pmod p$ for $0 \le i < k$. Then, it samples $\hat{\boldsymbol{u}}_i \leftarrow \mathtt{REcd}(u_{i,0}, \ldots, u_{i,\ell-1}, 0, \ldots, 0)$, and sends the polynomial oracle $([\![\hat{\boldsymbol{u}}_i]\!], 2N-1)$ to the verifier $\mathtt{V}$ for $0 \le i < k$.
3. The verifier sends a random point $\gamma \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$
4. The prover $\mathtt{P}$ and the verifier $\mathtt{V}$ invoke the following PIOP.

$$\Pi_{\mathtt{AC}}\left(\sum_{i=0}^{k-1} \gamma^i \cdot \vec{X}_i \odot (\vec{X}_i - \vec{c}); \{[\![\hat{\boldsymbol{u}}_i]\!]\}_{0\le i<k}\right)$$

$$\Pi_{\mathtt{IP}}\left(\sum_{i=0}^{k-1} \gamma^i \cdot (\vec{X}_i^2 - \vec{b} \odot \vec{X}_{i+k}); \{[\![\hat{\boldsymbol{a}}_i]\!]\}_{0\le i<k}, \{[\![\hat{\boldsymbol{u}}]\!]\}_{0\le i<k}\right)$$

Fig. 5: PIOP for L^2-norm constraints

$0 \le i < k$ except probability at most $k/(p-N)$. We note that the completeness, knowledge soundness, and HVZK properties directly follow from Theorems 3 and 14. □

4 Homomorphic Evaluation Protocol via CKKS

In this section, we discuss the security and privacy of homomorphic evaluation protocols based on the CKKS scheme. We primarily focus on the case of a semi-honest sender and a possibly malicious receiver, as the only information the sender receives throughout the protocol is the receiver's ciphertexts and public key in the first round, leaving no opportunity to compromise the receiver's privacy using only this information.

As mentioned earlier, we introduce a new security definition for CKKS-based homomorphic evaluation protocols, where the sender's privacy condition is relaxed from computational indistinguishability to differential privacy (DP). We justify this relaxation by noting that most real-world applications of CKKS are secure inference or training protocols [10, 54, 57, 61, 73], which can be viewed as privacy-enhanced versions of the machine learning as a service (MLaaS) model. However, even in a standard MLaaS scenario, vulnerabilities remain, as a receiver may attempt to reconstruct the sender's private data by aggregating multiple inference results [19, 20, 21, 76]. To mitigate such attacks, DP techniques are commonly applied [1, 60].

Reckoning that differential privacy (DP) is the final privacy notion in most applications, we analyze the effect of our security relaxation and derive the result that the output of the real execution of the protocol still satisfies DP. This holds if we model the target ideal functionality as a DP mechanism and the protocol satisfies the new security definition, making it well-suited for existing MLaaS scenarios to be naturally extended to homomorphic evaluation protocols, where the sender ensures their privacy within the framework of a differential privacy.

In the rest of this section, we first introduce our new security definition, which we call differentially private homomorphic evaluation protocols, along with its useful properties and implications. Then, we present a general compilation method that achieves sender privacy from plain CKKS homomorphic evaluation protocols, assuming the ideal functionality of a zero-knowledge argument of knowledge (ZKAoK) for CKKS ciphertexts and public keys, as well as the Laplace differential privacy mechanism. Finally, we address how to instantiate ZKAoK for CKKS using polynomial interactive oracle proofs (PIOPs), introduced in Section 3, in conjunction with CKKS homomorphic evaluation algorithms.

4.1 Differentially Private Homomorphic Evaluation Protocol

We begin by presenting the basic structure of homomorphic evaluation protocols for CKKS, which is defined as follows.

Definition 10 (HE-Protocol). *Let $(\mathtt{P}_1, \mathtt{P}_2)$ be a sender-receiver protocol, and let $\mathtt{f} : \Theta \times \mathcal{X}^k \to \mathcal{Y}$ be a deterministic algorithm, where $\mathcal{X}, \mathcal{Y} \subseteq R$. We say that $(\mathtt{P}_1, \mathtt{P}_2)$ is a homomorphic evaluation protocol for $\mathtt{f}$ if it has the following structure:*

Inputs. *The receiver $\mathtt{P}_1$ has a message $\vec{\boldsymbol{x}} \in \mathcal{X}^k$ as a private input. The sender $\mathtt{P}_2$ has a parameter $\theta \in \Theta$ as a private input. A public parameter $\mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda)$ is public to both parties.*

The protocol.

1. *The receiver $\mathtt{P}_1$ sends $(\mathsf{pk}, \mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1})$ to the sender $\mathtt{P}_2$, where pk is a public key, and $\{\mathsf{ct}_i\}_{0 \leq i < k}$ are ciphertext.*
2. *The receiver $\mathtt{P}_1$ and the sender $\mathtt{P}_2$ invoke auxiliary subprotocols that are independent of θ.*
3. *The sender $\mathtt{P}_2$ computes a ciphertext $\mathsf{ct}_{\mathsf{out}}$, and sends it to the receiver $\mathtt{P}_1$.*
4. *The receiver $\mathtt{P}_1$ outputs $\boldsymbol{y} = \mathtt{Dec}_{\mathsf{sk}}(\mathsf{ct}_{\mathsf{out}}) \in \mathcal{Y}$.*

In defining the security notion for HE protocols, we aim to extend the concept of differentially private prediction defined in [37], which precisely captures how inference results affect the privacy of the sender's data within the differential privacy framework. Specifically, it models the sender's algorithm as an ensemble parameterized by some $\theta \in \Theta$, which typically represents the sender's data used to construct the algorithm, such as training data. It then discusses how to achieve differential privacy for the sender's algorithm with respect to the parameter θ in the presence of inference results. Below, we provide its precise definition, with a slight relaxation for computationally bounded adversaries.

Definition 11 (DP-Prediction [37]). *Let* $\mathtt{M} : \mathit{\Theta} \times \mathcal{X}^k \to \mathcal{Y}$ *be a randomized algorithm. We say that* $\mathtt{M}$ *is an* ε*-(C)DP prediction algorithm if, for every* $\vec{\boldsymbol{x}} \in \mathcal{X}^k$*, the output* $\mathtt{M}(\theta, \vec{\boldsymbol{x}})$ *is* ε*-(C)DP with respect to* θ>

Based on the definition of DP-prediction, we define our security notion for HE protocols, where we model the ideal functionality of the protocol as a DP-prediction.

Definition 12 (DPHE-Protocol). *Let* $\mathtt{f} : \mathit{\Theta} \times \mathcal{X}^k \to \mathcal{Y}$ *be a deterministic algorithm, where* $\mathcal{X}, \mathcal{Y} \subseteq R$*, let* $\mathtt{M} : \mathit{\Theta} \times \mathcal{X}^k \to \mathcal{Y}$ *be a randomized algorithm, and let* $\Pi = (\mathtt{P}_1, \mathtt{P}_2)$ *be a homomorphic evaluation protocol. We say that* Π *is an* ε*-differentially private homomorphic evaluation (DPHE) protocol for* $\mathtt{f}$ *with* (η, δ)*-usefulness if the following holds.*

Usefulness. *For all* $\vec{\boldsymbol{x}} \in \mathcal{X}^k$ *and* $\theta \in \mathit{\Theta}$*, the receiver's output* $\boldsymbol{y}$ *from the correct execution of* Π *with inputs* $\vec{\boldsymbol{x}}$ *and* θ *satisfies the following condition.*

$$\Pr\left[\|\boldsymbol{y} - \mathtt{f}(\theta, \vec{\boldsymbol{x}})\|_\infty > \eta\right] \le \delta$$

Receiver's privacy. *For all sender's input* $\theta \in \mathit{\Theta}$*, the views of the honest sender* $\mathtt{P}_2$ *for all receiver's inputs* $\vec{\boldsymbol{x}}, \vec{\boldsymbol{x}}' \in \mathcal{X}^k$ *are computationally indistinguishable. i.e.,* $\mathtt{View}^{\Pi}_{\mathtt{P}_2}(\vec{\boldsymbol{x}}, \theta) \approx_c \mathtt{View}^{\Pi}_{\mathtt{P}_2}(\vec{\boldsymbol{x}}', \theta)$*.*

Sender's privacy. *For all PPT adversaries* $\mathtt{A}$ *that only manipulate the receiver, there exists a PPT simulator* $\mathtt{S}$ *such that for all PPT environments* $\mathtt{Z}$ *and PPT algorithms* $\mathtt{D}$*, the following holds for all* $\theta \in \mathit{\Theta}$*, where* Π_θ *is a restriction of* Π *with the sender's input* θ*, and* $\mathtt{F}_{\mathtt{DPHE},\theta}$ *is the ideal functionality defined in Fig. 6.*

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

$\mathtt{F}_{\mathtt{DPHE},\theta}$

Inputs: The receiver $\mathtt{P}_1$ sends $\vec{\boldsymbol{x}} \in \mathcal{X}^k$.
Outputs: The receiver $\mathtt{P}_1$ obtains $\boldsymbol{y} \leftarrow \mathtt{M}(\theta, \vec{\boldsymbol{x}})$.

Fig. 6: Ideal functionality for differentially private homomorphic evaluation

Since the ideal functionality in our definition is a randomized algorithm, we relax the correctness condition to usefulness, describing how far the output may deviate from the target. For the receiver's privacy, we use the standard computational indistinguishability definition against a semi-honest sender. However, for the sender's privacy, considering the possibility of a malicious receiver, we incorporate the universal composability framework [18] alongside the computational differential privacy (CDP) notion, as much of the HE protocol relies on the IND-CPA security of the CKKS scheme.

Next, we present the main implication of our DPHE protocol, which states that if the ideal functionality is a CDP-prediction and a protocol satisfies the sender's privacy condition, then the output from the real protocol execution remains a DP-prediction, as described below.

Theorem 5 (Composition of CDP and DPHE). *Let* $\mathtt{f} : \Theta \times \mathcal{X}^k \to \mathcal{Y}$ *be a deterministic algorithm, and let* Π *be a* ε_1*-DPHE-protocol for* $\mathtt{f}$*. Suppose* $\mathtt{M}$ *is a* ε_2*-CDP prediction with respect to* θ*. Then,* $\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]$ *is a* $(2\varepsilon_1 + \varepsilon_2)$*-CDP prediction with respect to* θ *for all PPT environments* $\mathtt{Z}$ *and all PPT adversaries* $\mathtt{A}$ *that manipulate the receiver.*

Proof. Let $\mathtt{A}$ be a PPT adversary that manipulate the receiver and $\mathtt{S}$ be a PPT simulator for $\mathtt{A}$. Then, for all PPT environments $\mathtt{Z}$, $\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]$ is an ε_2-CDP prediction with respect to θ, since in the distribution of $\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]$, only the output $\boldsymbol{y} \leftarrow \mathtt{M}(\theta, \vec{\boldsymbol{x}})$ obtained from $\mathtt{F}_{\mathtt{DPHE},\theta}$ varies with the choice of θ, and $\mathtt{M}(\theta, \vec{\boldsymbol{x}})$ is an ε_2-CDP prediction.

Now, let $\theta, \theta' \in \Theta$ be adjacent, then we obtain the followings for all PPT algorithm $\mathtt{D}$.

$$\begin{aligned}\Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] &\leq e^{\varepsilon_1} \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\\ &\leq e^{\varepsilon_1+\varepsilon_2} \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta'}, \mathtt{S}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\\ &\leq e^{2\varepsilon_1+\varepsilon_2} \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_{\theta'}, \mathtt{A}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\end{aligned}$$

Therefore, $\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]$ is a $(2\varepsilon_1 + \varepsilon_2)$-CDP prediction with respect to θ. □

Based on the above theorem, existing differential privacy mechanisms can be naturally extended to DPHE protocols, provided we prove the sender's privacy condition for the target mechanism. However, since the sender's privacy is defined within the UC framework, we need to examine whether the commonly used proof techniques for computational indistinguishability in the UC framework are well adapted to our differential privacy setting. First, we show that the completeness result for the dummy adversary is well suited to our setting, which allows us to consider only the dummy adversary when proving the security of the protocol in the UC framework. Next, we prove that our definition also supports subprotocol composition, enabling us to establish the security of the protocol in the hybrid execution model. Below, we present the adaptations of these proof techniques within our security definition.

Theorem 6 (Completeness of the dummy adversary). *Let* Π *be an HE-protocol such that the sender's privacy holds for the dummy adversary* $\mathtt{A}^*$*, whose only job is to forward messages between an environment and the other protocol parties, with a parameter* ε *for all PPT environments. Then,* Π *satisfies the sender's privacy with the parameter* ε*.*

Proof. Let $\mathtt{A}$ be a PPT adversary that manipulates the receiver in $\mathtt{\Pi}$ and $\mathtt{Z}$ be a PPT environment. Then, it holds that $\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}] \approx_c \mathtt{EXEC}[\Pi_\theta, \mathtt{A}^*, \mathtt{A}|\mathtt{Z}]$, where $\mathtt{A}|\mathtt{Z}$ is an environment that both runs $\mathtt{A}$ and $\mathtt{Z}$. By assumption, there exists a simulator $\mathtt{S}^*$ for the dummy adversary where the following holds for all PPT algorithms $\mathtt{D}$.

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

Additionally, it holds that $\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*, \mathtt{A}|\mathtt{Z}] \approx_c \mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*|\mathtt{A}, \mathtt{Z}]$, where $\mathtt{S}^*|\mathtt{A}$ is a PPT algorithm that runs both $\mathtt{S}^*$ and $\mathtt{A}$. Then, we have the followings for all PPT algorithms $\mathtt{D}$.

$$\begin{aligned}\Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] &\le \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\\ &\le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\\ &\le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*|\mathtt{A}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\end{aligned}$$

$$\begin{aligned}\Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*|\mathtt{A}, \mathtt{Z}]) = 1\right] &\le \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\\ &\le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}^*, \mathtt{A}|\mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\\ &\le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)\end{aligned}$$

Therefore, $\mathtt{\Pi}$ satisfies sender's privacy with the parameter ε. □

Theorem 7 (Subprotocol composition). *Let $\mathtt{\Pi}$ be an HE-protocol that uses $\mathtt{\Pi}'$ as a subprotocol, which securely implements an ideal functionality $\mathtt{F}'$, and let $\Pi_{\mathtt{F}'}$ be the hybrid protocol, which uses the ideal functionality $\mathtt{F}'$ in place of $\mathtt{\Pi}'$. If $\Pi_{\mathtt{F}'}$ satisfies the sender's privacy with a parameter ε, then $\mathtt{\Pi}$ also satisfies the sender's privacy with the parameter ε.*

Proof. Let $\mathtt{A}$ be a PPT adversary that manipulates the receiver in Π_θ for $\theta \in \Theta$. Since $\mathtt{\Pi}'$ securely implements $\mathtt{F}'$, there exists a PPT adversary $\mathtt{A}'$ such that $\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}] \approx_c \mathtt{EXEC}[\Pi_{\mathtt{F}',\theta}, \mathtt{A}', \mathtt{Z}]$ for all PPT environments $\mathtt{Z}$ by the universal composition theorem. Also, there exists a simulator $\mathtt{S}$ such that the following holds for all PPT algorithms $\mathtt{D}$ since $\Pi_{\mathtt{F}'}$ satisfies the sender's privacy with a parameter ε.

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_{\mathtt{F}',\theta}, \mathtt{A}', \mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_{\mathtt{F}',\theta}, \mathtt{A}', \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

Therefore, we obtain the following since $\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}] \approx_c \mathtt{EXEC}[\Pi_{\mathtt{F}',\theta}, \mathtt{A}', \mathtt{Z}]$, which shows that $\mathtt{\Pi}$ also satisfies the sender's privacy with the parameter ε.

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

$$\Pr\left[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\right] \le e^\varepsilon \cdot \Pr\left[\mathtt{D}(\mathtt{EXEC}[\Pi_\theta, \mathtt{A}, \mathtt{Z}]) = 1\right] + \mathsf{negl}(\lambda)$$

□

4.2 DPHE Compiler from ZKAoK for CKKS

In this subsection, we present a general compilation method to transform a plain HE protocol into a DPHE protocol. Our approach is inspired by [3], which provides a general compilation method for the BGV scheme by proving indistinguishability with an ideal functionality for zero-knowledge arguments of knowledge (ZKAoK) for BGV and leveraging a noise flooding technique.

We adapt their approach to align with our differential privacy-based sender privacy framework. Specifically, we first define the ideal functionality of ZKAoK for CKKS. Then, instead of using noise flooding, we employ the Laplace mechanism, which also introduces additional noise during randomization but allows for moderate noise levels, whereas noise flooding requires exponentially large noise.

As an additional optimization, we achieve ciphertext rerandomization based on the Hint-MLWE problem, which removes sender-specific information embedded in the ciphertext. In previous work, this was typically achieved using noise flooding, which introduced significant parameter overhead. However, our method avoids this overhead while maintaining security. Below, we describe our compilation method in detail

$\mathrm{F}_{\mathtt{ZKAoK}}$

Inputs: The receiver $\mathtt{P}_1$ sends $(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{w}_{\mathtt{CKKS}})$, where $\mathsf{x}_{\mathtt{CKKS}} = (\mathsf{pk}, \mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1})$, $\mathsf{w}_{\mathtt{CKKS}} = (\boldsymbol{s}, \boldsymbol{e}_{\mathsf{ek}}, \vec{\boldsymbol{e}}_{\mathsf{rlk}}, \vec{\boldsymbol{e}}_{\mathsf{rtk}}, \vec{\boldsymbol{e}}_{\mathsf{cjk}}, \vec{\boldsymbol{e}}, \vec{\boldsymbol{x}})$, $\mathsf{ek} = (\mathsf{ek}_0, \mathsf{ek}_1)$, $\mathsf{rlk} = (\mathsf{rlk}_0, \mathsf{rlk}_1)$, $\mathsf{rtk} = (\mathsf{rtk}_0, \mathsf{rtk}_1)$, $\mathsf{cjk} = (\mathsf{cjk}_0, \mathsf{cjk}_1)$, $\vec{\boldsymbol{e}} = (\boldsymbol{e}_0, \ldots, \boldsymbol{e}_{k-1})$, $\vec{\boldsymbol{x}} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{k-1})$, and $\mathsf{ct}_i = (\mathsf{ct}_{i,0}, \mathsf{ct}_{i,1})$ for $0 \leq i < k$, and the sender $\mathtt{P}_2$ sends $\mathsf{x}_{\mathtt{CKKS}}$.

Outputs: The sender $\mathtt{P}_2$ obtains 1 if $(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{w}_{\mathtt{CKKS}}) \in \mathsf{R}_{\mathtt{CKKS}}$, where

$$\mathsf{R}_{\mathtt{CKKS}} = \left\{ (\mathsf{x}_{\mathtt{CKKS}}, \mathsf{w}_{\mathtt{CKKS}}) \;\middle|\; \begin{array}{r} \|\boldsymbol{s}\|_\infty \leq 1 \wedge \|\boldsymbol{s}\|_1 \leq h \wedge \\ \|\boldsymbol{e}_{\mathsf{ek}}\|_\infty \leq B_{\mathsf{ek}} \wedge \|\vec{\boldsymbol{e}}_{\mathsf{rlk}}\|_\infty \leq B_{\mathsf{rlk}} \wedge \|\vec{\boldsymbol{e}}_{\mathsf{rtk}}\|_\infty \leq B_{\mathsf{rtk}} \wedge \\ \|\vec{\boldsymbol{e}}_{\mathsf{cjk}}\|_\infty \leq B_{\mathsf{cjk}} \wedge \|\vec{\boldsymbol{e}}\|_\infty \leq B_e \wedge \\ \boldsymbol{x}_0, \ldots, \boldsymbol{x}_{k-1} \in \mathcal{X} \wedge \\ \mathsf{ek}_0 + \boldsymbol{s} \cdot \mathsf{ek}_1 = \boldsymbol{e}_{\mathsf{ek}} \wedge \\ \mathsf{rlk}_0 + \boldsymbol{s} \cdot \mathsf{rlk}_1 = \boldsymbol{s}^2 \cdot \vec{g} + \vec{\boldsymbol{e}}_{\mathsf{rtk}} \wedge \\ \mathsf{rtk}_0 + \boldsymbol{s} \cdot \mathsf{rtk}_1 = \varphi(\boldsymbol{s}) \cdot \vec{g} + \vec{\boldsymbol{e}}_{\mathsf{rtk}} \wedge \\ \mathsf{cjk}_0 + \boldsymbol{s} \cdot \mathsf{cjk}_1 = \psi(\boldsymbol{s}) \cdot \vec{g} + \vec{\boldsymbol{e}}_{\mathsf{cjk}} \wedge \\ \mathsf{ct}_{i,0} + \boldsymbol{s} \cdot \mathsf{ct}_{i,1} = \boldsymbol{x}_i + \boldsymbol{e}_i \text{ for } 0 \leq i < k \end{array} \right\}$$

Otherwise, it obatins 0.

Fig. 7: Ideal functionality for zero-knowledge argument of knowledge for CKKS public keys and ciphertexts

Theorem 8. *Let* $\mathtt{f} : \Theta \times \mathcal{X}^k \to \mathcal{Y}^k$ *be a deterministic algorithm, where* $\mathtt{f}(\theta, \cdot)$ *is admissible in CKKS for all* $\theta \in \Theta$*. Let* $\Pi_{\mathtt{CKKS}}$ *be the HE-protocol described in*

$\Pi_{\texttt{CKKS}}$

Inputs: The receiver $\texttt{P}_1$ has a message $\vec{\boldsymbol{x}} \in \mathcal{X}^k$ as a private input. The sender $\texttt{P}_2$ has a parameter $\theta \in \Theta$ as a private input. A public parameter $\mathsf{pp} \leftarrow \texttt{Setup}(1^\lambda)$ is public to both parties.

The protocol:

1. The receiver $\texttt{P}_1$ generates the followings, and sends $(\mathsf{pk}, \mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1})$ to the sender $\texttt{P}_2$.
$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \texttt{KeyGen}(\mathsf{pp}),\ \mathsf{ct}_i \leftarrow \texttt{Enc}_{\mathsf{sk}}(\boldsymbol{x}_i)$$
2. The parties invoke $\texttt{F}_{\texttt{ZKAoK}}$, and the sender $\texttt{P}_2$ aborts if its output is 0.
3. The sender $\texttt{P}_2$ computes the following.
$$(\boldsymbol{c}_0, \boldsymbol{c}_1) \leftarrow \left[\texttt{Eval}_{\mathsf{pk}}(\texttt{f}(\theta; \cdot); \mathsf{pk}, \mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1})\right]_{Q_0}$$
$$\tau = \max_{\theta \in \Theta} \Big(\texttt{Estim}\big(\texttt{f}(\theta, \cdot); h, B_{\mathsf{rlk}}, B_{\mathsf{rtk}}, B_{\mathsf{cjk}}, B_e\big)\Big)$$
The sender $\texttt{P}_2$ sends an output ciphertext $\mathsf{ct}_{\mathsf{out}}$ as follows, and send $\mathsf{ct}_{\mathsf{out}}$ to the receiver $\texttt{P}_2$.
$$\boldsymbol{r}_0 \leftarrow \texttt{DG}(\sigma)^N, \boldsymbol{r}_1 \leftarrow \texttt{DG}(B_{\mathsf{ek}} N \cdot \sigma)^N, \boldsymbol{r}_2 \leftarrow \texttt{DG}(h\sigma)^N, \boldsymbol{r}_3 \leftarrow \texttt{DG}(\sigma)^N, \boldsymbol{t} \leftarrow \texttt{Lap}(N\tau/\varepsilon)^N$$
$$\mathsf{ct}_{\mathsf{out}} = \boldsymbol{r}_0 \cdot \mathsf{ek} + (\boldsymbol{r}_1 + \boldsymbol{r}_2, \boldsymbol{r}_3) + q_1 \cdot (\boldsymbol{c}_0 + \lfloor \boldsymbol{t} \rceil, \boldsymbol{c}_1) \pmod{Q_1}$$
4. The receiver obtains an output $\boldsymbol{y} = \texttt{Dec}_{\mathsf{sk}}(\mathsf{ct}_{\mathsf{out}})$.

Fig. 8: Protocol for differentially private homomorphic evaluation using CKKS

Fig. 8, and let $\texttt{M}(\theta, \vec{\boldsymbol{x}}) = \texttt{f}(\theta, \vec{\boldsymbol{x}}) + \lfloor \boldsymbol{t} \rceil$ *for* $\boldsymbol{t} \leftarrow \texttt{Lap}(N\tau/\varepsilon)^N$. *Then* $\Pi_{\texttt{CKKS}}$ *is an* ε*-DPHE-protocol for* $\texttt{f}$ *in the* $\texttt{F}_{\texttt{ZKAoK}}$*-hybrid model, which is described in Fig. 7, if we assume* $\mathsf{RLWE}_{Q_1, \sigma/2, \sigma/2}$ *is computationally hard, and* $\sigma \geq 2\sqrt{2} \cdot \eta_{(1/2^\lambda)}(\mathbb{Z}^N)$.

Proof. Below, we demonstrate the usefulness, receiver's privacy, and sender's privacy of $\Pi_{\texttt{CKKS}}$.

Usefulness. Let $\vec{\boldsymbol{x}} \in \mathcal{X}^k$ be a receiver's input, $\theta \in \Theta$ be a sender's input, and $\boldsymbol{y} \in R$ be a receiver's output. Then, the following holds for some $\boldsymbol{e}_{\texttt{Eval}}$ such that $\|\boldsymbol{e}_{\texttt{Eval}}\|_\infty \leq \tau$, and $\eta > \tau + 1/2$ since $\boldsymbol{t} \leftarrow \texttt{Lap}(N\tau/\varepsilon)^N$.

$$\begin{aligned}\Pr[\|\boldsymbol{y} - \texttt{f}(\theta, \vec{\boldsymbol{x}})\|_\infty > \eta] &= \Pr[\|\boldsymbol{e}_{\texttt{Eval}} + \lfloor \boldsymbol{t} \rceil\|_\infty > \eta] \\ &\leq \Pr[\|\lfloor \boldsymbol{t} \rceil\|_\infty > \eta - \tau] \leq \Pr[\|\boldsymbol{t}\|_\infty > \eta - \tau - 1/2] \\ &\leq \frac{1}{2} \cdot \exp\Big(-\frac{\varepsilon}{N\tau}(\eta - \tau - 1/2)\Big)\end{aligned}$$

Receiver's privacy. Let $\Pi_{\texttt{ZKAoK}}$ be a protocol that securely implements the functionality $\texttt{F}_{\texttt{ZKAoK}}$. Then, there exists a simulator $\texttt{S}_{\texttt{ZKAoK}}$ such that $\texttt{S}_{\texttt{ZKAoK}}(\texttt{x}_{\texttt{CKKS}}) \approx_c \texttt{VIEW}_{\texttt{P}_2}^{\Pi_{\texttt{ZKAoK}}}(\texttt{x}_{\texttt{CKKS}}, \texttt{w}_{\texttt{CKKS}})$ for all $\vec{\boldsymbol{x}} \in \mathcal{X}$. Now, suppose there exists a PPT algorithm $\texttt{D}$ that distinguishes $\texttt{VIEW}_{\texttt{P}_2}^{\Pi_{\texttt{CKKS}}}(\vec{\boldsymbol{x}}, \theta)$ and $\texttt{VIEW}_{\texttt{P}_2}^{\Pi_{\texttt{CKKS}}}(\vec{\boldsymbol{x}}', \theta)$ with non-negligible probability for some $\vec{\boldsymbol{x}}, \vec{\boldsymbol{x}}' \in \mathcal{X}$. Then, the following holds for $\mathtt{VIEW}_{\mathtt{P}_2}^{\Pi_{\mathtt{CKKS}}}(\vec{\boldsymbol{x}}, \theta)$ and $\mathtt{VIEW}_{\mathtt{P}_2}^{\Pi_{\mathtt{CKKS}}}(\vec{\boldsymbol{x}}', \theta)$.

$$\mathtt{VIEW}_{\mathtt{P}_2}^{\Pi_{\mathtt{CKKS}}}(\vec{\boldsymbol{x}}, \theta) = \Big(\mathsf{x}_{\mathtt{CKKS}}, \mathtt{VIEW}_{\mathtt{P}_2}^{\Pi_{\mathtt{ZKAoK}}}(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{w}_{\mathtt{CKKS}})\Big) \approx_c \Big(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{S}_{\mathtt{ZKAoK}}(\mathsf{x}_{\mathtt{CKKS}})\Big)$$

$$\mathtt{VIEW}_{\mathtt{P}_2}^{\Pi_{\mathtt{CKKS}}}(\vec{\boldsymbol{x}}', \theta) = \Big(\mathsf{x}'_{\mathtt{CKKS}}, \mathtt{VIEW}_{\mathtt{P}_2}^{\Pi_{\mathtt{ZKAoK}}}(\mathsf{x}'_{\mathtt{CKKS}}, \mathsf{w}'_{\mathtt{CKKS}})\Big) \approx_c \Big(\mathsf{x}'_{\mathtt{CKKS}}, \mathsf{S}_{\mathtt{ZKAoK}}(\mathsf{x}'_{\mathtt{CKKS}})\Big)$$

Then, $\mathtt{D}$ also distinguishes $\Big(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{S}_{\mathtt{ZKAoK}}(\mathsf{x}_{\mathtt{CKKS}})\Big)$ and $\Big(\mathsf{x}'_{\mathtt{CKKS}}, \mathsf{S}_{\mathtt{ZKAoK}}(\mathsf{x}'_{\mathtt{CKKS}})\Big)$ with non-negligible probability, which contradicts the IND-CPA security of the CKKS scheme. Therefore, $\Pi_{\mathtt{CKKS}}$ achieves the receiver's privacy in the $\mathtt{F}_{\mathtt{ZKAoK}}$-hybrid model since no such algorithm $\mathtt{D}$ exists.

Sender's privacy. By Theorem 6, it suffices to consider only the dummy adversary $\mathtt{A}^*$ for all PPT environments. For a PPT environment $\mathtt{Z}$, We build a simulator $\mathtt{S}$ for the dummy adversary $\mathtt{A}^*$ as follows.

1. When $\mathtt{Z}$ sends a message $(\mathsf{pk}, \{\mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1}\})$ that is supposed to be forwarded by the dummy adversary to the honest sender, the simulator $\mathtt{S}$ retrieves it.
2. When $\mathtt{Z}$ sends a message $(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{w}_{\mathtt{CKKS}})$ that is supposed to be forwarded to the ideal functionality $\mathtt{F}_{\mathtt{ZKAoK}}$, the simulator $\mathtt{S}$ retrieves it.
3. If $(\mathsf{x}_{\mathtt{CKKS}}, \mathsf{w}_{\mathtt{CKKS}}) \in \mathsf{R}_{\mathtt{CKKS}}$, the simulator $\mathtt{S}$ returns 1 to the environment $\mathtt{Z}$, otherwise it returns 0.
4. The simulator $\mathtt{S}$ sends $\vec{\boldsymbol{x}}$ to the functionality $\mathtt{F}_{\mathtt{DPHE},\theta}$, and receives an output $\boldsymbol{y}'$ from the functionality.
5. The simulator $\mathtt{S}$ samples $\boldsymbol{a}' \leftarrow \mathcal{U}(R_{Q_1})$, and $\boldsymbol{r}_0 \leftarrow \mathtt{DG}(\sigma)^N, \boldsymbol{r}_1 \leftarrow \mathtt{DG}(B_{\mathsf{ek}}N \cdot \sigma)^N, \boldsymbol{r}_2 \leftarrow \mathtt{DG}(h\sigma)^N, \boldsymbol{r}_3 \leftarrow \mathtt{DG}(\sigma)^N$. Then, it returns $\mathsf{ct}'_{\mathsf{out}}$ to the environment $\mathtt{Z}$, which is computed as follows.

$$\mathsf{ct}'_{\mathsf{out}} = \Big(-\boldsymbol{a}' \cdot \boldsymbol{s} + (\boldsymbol{e}_{\mathsf{ek}} \cdot \boldsymbol{r}'_0 + \boldsymbol{r}'_1) + (\boldsymbol{r}'_2 - \boldsymbol{s} \cdot \boldsymbol{r}'_3) + q_1 \cdot \boldsymbol{y}', \boldsymbol{a}'\Big) \in R_{Q_1}^2$$

Now, we show that for all PPT algorithms $\mathtt{D}$, the following holds, which implies the sender's privacy with the parameter ε.

$$\Pr\big[\mathtt{D}(\mathtt{EXEC}[\Pi_{\mathtt{CKKS}}, \mathtt{A}^*, \mathtt{Z}]) = 1\big] \le e^{\varepsilon} \cdot \Pr\big[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\big] + \mathsf{negl}(\lambda)$$

$$\Pr\big[\mathtt{D}(\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]) = 1\big] \le e^{\varepsilon} \cdot \Pr\big[\mathtt{D}(\mathtt{EXEC}[\Pi_{\mathtt{CKKS}}, \mathtt{A}^*, \mathtt{Z}]) = 1\big] + \mathsf{negl}(\lambda)$$

We note that the only difference between $\mathtt{EXEC}[\Pi_{\mathtt{CKKS}}, \mathtt{A}^*, \mathtt{Z}]$ and $\mathtt{EXEC}[\mathtt{F}_{\mathtt{DPHE},\theta}, \mathtt{S}, \mathtt{Z}]$ is the distribution of $\mathsf{ct}_{\mathsf{out}}$ and $\mathsf{ct}'_{\mathsf{out}}$. Thus, it suffices to prove that for all PPT algorithms $\mathtt{D}$, the following holds.

$$\Pr\big[\mathtt{D}(\mathsf{ct}_{\mathsf{out}}, \mathsf{w}_{\mathtt{CKKS}}) = 1\big] \le e^{\varepsilon} \cdot \Pr\big[\mathtt{D}(\mathsf{ct}'_{\mathsf{out}}, \mathsf{w}_{\mathtt{CKKS}}) = 1\big] + \mathsf{negl}(\lambda) \qquad (1)$$

$$\Pr\big[\mathtt{D}(\mathsf{ct}'_{\mathsf{out}}, \mathsf{w}_{\mathtt{CKKS}}) = 1\big] \le e^{\varepsilon} \cdot \Pr\big[\mathtt{D}(\mathsf{ct}_{\mathsf{out}}, \mathsf{w}_{\mathtt{CKKS}}) = 1\big] + \mathsf{negl}(\lambda) \qquad (2)$$

For the environment $\mathtt{Z}$, $\mathsf{ct}_{\mathsf{out}}$ has the following form, where $\boldsymbol{a} = q_1 \cdot \boldsymbol{c}_1 + (\mathsf{ek}_0 \cdot \boldsymbol{r}_0 + \boldsymbol{r}_3)$.

$$\mathsf{ct}_{\mathsf{out}} = \Big(-\boldsymbol{a} \cdot \boldsymbol{s} + (\boldsymbol{e}_{\mathsf{ek}} \cdot \boldsymbol{r}_0 + \boldsymbol{r}_1) + (\boldsymbol{r}_2 - \boldsymbol{s} \cdot \boldsymbol{r}_3) + q_1 \cdot \boldsymbol{y}, \boldsymbol{a} \Big) \in R_{Q_1}^2$$

Let $\mathsf{ct}''_{\mathsf{out}}$ be a hybrid distribution defined as follows, where $\boldsymbol{a}' \leftarrow \mathcal{U}(R_{Q_1})$.

$$\mathsf{ct}''_{\mathsf{out}} = \Big(-\boldsymbol{a}' \cdot \boldsymbol{s} + (\boldsymbol{e}_{\mathsf{ek}} \cdot \boldsymbol{r}_0 + \boldsymbol{r}_1) + (\boldsymbol{r}_2 - \boldsymbol{s} \cdot \boldsymbol{r}_3) + q_1 \cdot \boldsymbol{y}, \boldsymbol{a}' \Big) \in R_{Q_1}^2$$

Then, $\mathsf{ct}_{\mathsf{out}} \approx_c \mathsf{ct}''_{\mathsf{out}}$ assuming that $\mathsf{HintRLWE}_{Q_1,\sigma,\sigma}^{\boldsymbol{e}_{\mathsf{ek}},\boldsymbol{s},B_{\mathsf{ek}}N\sigma,h\sigma}$ is computationally hard, which can be reduced from $\mathsf{RLWE}_{Q_1,\sigma/2,\sigma/2}$ with an advantage of at most $\mathsf{negl}(\lambda)$ by Theorem 1.

For $\mathsf{ct}''_{\mathsf{out}}$ and $\mathsf{ct}'_{\mathsf{out}}$, they only differ in the distribution of $\boldsymbol{y}$ and $\boldsymbol{y}'$. We note that $\boldsymbol{y} = \mathtt{f}(\theta, \vec{\boldsymbol{x}}) + \boldsymbol{e}_{\mathtt{Eval}} + \lfloor \boldsymbol{t} \rceil$ and $\boldsymbol{y}' = \mathtt{f}(\theta, \vec{\boldsymbol{x}}) + \lfloor \boldsymbol{t}' \rceil$ hold for some $\boldsymbol{e}_{\mathtt{Eval}}$, where $\boldsymbol{t}, \boldsymbol{t}' \leftarrow \mathtt{Lap}(N\tau/\varepsilon)^N$ and $\|\boldsymbol{e}_{\mathtt{Eval}}\|_\infty \le \tau$, which implies $\|\boldsymbol{e}_{\mathtt{Eval}}\|_1 \le N\tau$. Thus, $D_\infty(\boldsymbol{y} \| \boldsymbol{y}') \le \varepsilon$ holds due to the Laplacian mechanism, which implies $D_\infty(\mathsf{ct}''_{\mathsf{out}} \| \mathsf{ct}'_{\mathsf{out}}) \le \varepsilon$, where D_∞ denotes the max-divergence. Therefore, Eqs. (1) and (2) holds since $\mathsf{ct}_{\mathsf{out}} \approx_c \mathsf{ct}''_{\mathsf{out}}$. □

Next, we discuss which algorithm $\mathtt{f}$ is suitable for DPHE protocols. In the context of DP-prediction [37], the following condition is typically considered, which can be viewed as a generalization of the sensitivity notion in DP.

Definition 13 (Uniformly RO-Stable Prediction [37]). *An algorithm* $\mathtt{f} : \Theta \times \mathcal{X}^K \to \mathcal{Y}$ *is called a uniformly stable replace-one prediction with rate* ξ *if, for all adjacent* $\theta, \theta' \in \Theta$ *and any* $\vec{\boldsymbol{x}} \in \mathcal{X}^k$, $\|\mathtt{f}(\theta, \vec{\boldsymbol{x}}) - \mathtt{f}(\theta', \vec{\boldsymbol{x}})\|_\infty \le \xi$ *holds.*

If an algorithm $\mathtt{f}$ satisfies the above definition, we can easily design a DP-prediction using the Laplace mechanism. Moreover, [37, 74] show that several classes of training algorithms yield inference algorithms $\mathtt{f}$ that are uniformly RO-stable with respect to the training data. Below, we describe the consequence of a DPHE protocol when $\mathtt{f}$ is uniformly RO-stable.

Theorem 9. *Let* $\mathtt{f} : \Theta \times \mathcal{X}^k \to \mathcal{Y}$ *be a uniformly RO-stable prediction with rate* ξ. *Then,* $\mathtt{EXEC}[\Pi_{\mathtt{CKKS},\theta}, \mathtt{A}, \mathtt{Z}]$ *is a* $(2 + \xi/\tau)\varepsilon$*-CDP prediction with respect to* θ *for all PPT environments* $\mathtt{Z}$ *and all PPT adversaries* $\mathtt{A}$ *that manipulate the receiver.*

Proof. If $\mathtt{f}$ is uniformly stable with rate ξ, then $\mathtt{M}$ is a $(\xi/\tau)\varepsilon$-CDP prediction due to the Laplace mechanism. Therefore, by Theorem 5, $\mathtt{EXEC}[\Pi_{\mathtt{CKKS},\theta}, \mathtt{A}, \mathtt{Z}]$ is a $(2 + \xi/\tau)\varepsilon$-CDP prediction with respect to θ. □

4.3 Construction of ZKAoK for CKKS via PIOP

We present how to construct a ZKAoK for the relation $\mathsf{R}_{\mathtt{CKKS}}$ using PIOPs. When building a PIOP for HE, we first need to address two key problems: ciphertext modulus and the representation of polynomials. This is because the PIOPs presented in Section 3 can only handle vectors over a prime field, whereas the ciphertext modulus in HE is typically a composite of distinct primes, and the validity of ciphertexts is represented using polynomial arithmetic.

In [52], the first issue is addressed using the modulus switching technique. This solution first generates ciphertexts and public keys in a prime modulus $p > Q_L$, generates a proof modulo p, and then switches back to modulus Q_L by rescaling after the verification is complete. For the second issue, the following two vector representations are used.

- $\underline{\mathtt{Coeff}}(\boldsymbol{a}) \to \vec{a}$: Given a polynomial $\boldsymbol{a} = \sum_{i=0}^{N-1} a_i X^i$, outputs a vector $\vec{a} = (a_0, \ldots, a_{N-1})$.
- $\underline{\mathtt{NTT}}(\boldsymbol{a}) \to \underline{\vec{a}}$: Given a polynomial $\boldsymbol{a} = \sum_{i=0}^{N-1} a_i X^i$, outputs a vector $\underline{\vec{a}} = \big(\boldsymbol{a}(\xi), \boldsymbol{a}(\xi^3), \ldots, \boldsymbol{a}(\xi^{2N-1})\big)$, where ξ is a $2N$-th root of unity in $\mathbb{Z}_p$.

If we assume $p = 1 \pmod{2N}$, then $R_p \cong \mathbb{Z}_p^N$ via the isomorphism $\mathtt{NTT}$, thus we can prove polynomial arithmetic over component-wise arithmetic over $\mathbb{Z}_p^N$. Additionally, there exist matrices $T, P_\varphi, P_\psi \in \mathbb{Z}_p^{N\times N}$, which satisfy $\mathtt{NTT}(\boldsymbol{a}) = T \cdot \mathtt{Coeff}(\boldsymbol{a}), \mathtt{NTT}(\varphi(\boldsymbol{a})) = P_\varphi \cdot \mathtt{NTT}(\boldsymbol{a}), \mathtt{NTT}(\psi(\boldsymbol{a})) = P_\psi \cdot \mathtt{NTT}(\boldsymbol{a})$ for $\boldsymbol{a} \in R_p$. Based on these solutions, [52] successfully constructs PIOPs for the BFV scheme.

However, these solutions do not address CKKS-specific issues: proof for the sparsity of the secret key and validity of plaintext. For the sparsity condition, we prove this by incorporating our newly designed PIOP Π_{L^2} for L^2-norm bound. We note that under the condition $\|\vec{s}\|_\infty \le 1$, $[\|\vec{s}\|_2^2]_p \le h$ implies $\|\vec{s}\|_1 = h$, thus executing Π_{L^2} in conjunction with Π_{L^∞} allows us to prove the Hamming weight bound of the secret key.

To prove the validity of plaintext, using PIOP alone is hard to provide a solution, as we need to prove relations over $\mathbb{C}^{N/2}$. Here, we focus on the most frequently used setting where we assume input plaintexts $\vec{\boldsymbol{x}} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{k-1})$ satisfies $\|\mathtt{Unpack}(\boldsymbol{x}_i)\|_\infty \le 1$ for $0 \le i < k$. To prove this condition, we incorporate a homomorphic operation called the coeff-to-slot operation, which is a main building block for CKKS bootstrapping [28]. Its basic functionality is as follows.

- $\underline{\mathtt{CtoS}}(\mathsf{ct}) \to \mathsf{ct}', \mathsf{ct}''$: Given a CKKS ciphertext $\mathsf{ct} \in R_{Q_\ell}^2$, whose plaintext is $\mathtt{Dec}_{\mathsf{sk}}(\mathsf{ct}) = \boldsymbol{m} = \sum_{i=0}^{N-1} m_i X^i$, it outputs two ciphertexts $\mathsf{ct}', \mathsf{ct}'' \in R_{Q_{\ell-\ell_{\mathtt{CtoS}}}}^2$, whose plaintexts are $\boldsymbol{m}', \boldsymbol{m}'' \in R$, where $\iota(\boldsymbol{m}') = (m_0, \ldots, m_{N/2-1})$, and $\iota(\boldsymbol{m}'') = (m_{N/2}, \ldots, m_{N-1})$.

We do not specify the level consumption $\ell_{\mathtt{CtoS}}$ and error bound for the coeff-to-slot operation here, as it depends on which algorithm is used for implementation [5,25,50,51]. Based on the coeff-to-slot transformation, we modify the ciphertext generation process as follows. Given input messages $\vec{m}_0, \ldots, \vec{m}_{k-1} \in \mathbb{R}^{N/2}$, which are intended to be transformed into plaintexts via $\mathtt{Pack}$, we instead generate plaintexts $\boldsymbol{m}_i$, where $\mathtt{Coeff}(\boldsymbol{m}_i) = \lfloor \varDelta \cdot (\vec{m}_{2i} \| \vec{m}_{2i+1}) \rceil$ for $0 \le i < k/2$, and subsequently generate based on them. During the ZKAoK, we prove that $\|\boldsymbol{m}_i\|_\infty \le \varDelta$ using Π_{L^∞}. Finally, performing $\mathtt{CtoS}$ on the verified ciphertexts results in CKKS ciphertexts whose plaintexts satisfy the previously described condition. Below, we present our ZKAoK for CKKS in detail.

$\Pi_{\texttt{CT\&PK}}$

Instance: $\mathsf{pk}', \mathsf{ct}'_0, \ldots, \mathsf{ct}'_{k/2-1}$
Witness: $\boldsymbol{s}, \boldsymbol{e}'_{\mathsf{ek}} \in R_P$, $\vec{\boldsymbol{e}}'_{\mathsf{rlk}}, \vec{\boldsymbol{e}}'_{\mathsf{rtk}}, \vec{\boldsymbol{e}}'_{\mathsf{cjk}} \in R_P^L$, $\vec{\boldsymbol{e}}', \vec{\boldsymbol{m}}' \in R_P^{k/2}$

1. The prover $\texttt{P}$ samples the followings.

$$\begin{aligned}
&\hat{\boldsymbol{s}} \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{s})),\ \underline{\hat{\boldsymbol{s}}} \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{s}))\\
&\underline{\hat{\boldsymbol{s}}}_{\mathsf{rtk}} \leftarrow \texttt{REcd}(\texttt{NTT}(\varphi(\boldsymbol{s}))),\ \underline{\hat{\boldsymbol{s}}}_{\mathsf{cjk}} \leftarrow \texttt{REcd}(\texttt{NTT}(\psi(\boldsymbol{s})))\\
&\hat{\boldsymbol{e}}_{\mathsf{ek}} \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{e}'_{\mathsf{ek}})),\ \underline{\hat{\boldsymbol{e}}}_{\mathsf{ek}} \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{e}'_{\mathsf{ek}}))\\
&\hat{\boldsymbol{e}}_{\mathsf{rlk},j} \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{e}'_{\mathsf{rlk},j})),\ \underline{\hat{\boldsymbol{e}}}_{\mathsf{rlk},j} \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{e}'_{\mathsf{rlk},j})) \text{ for } 0 \le j < L\\
&\hat{\boldsymbol{e}}_{\mathsf{rtk},j} \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{e}'_{\mathsf{rtk},j})),\ \underline{\hat{\boldsymbol{e}}}_{\mathsf{rtk},j} \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{e}'_{\mathsf{rtk},j})) \text{ for } 0 \le j < L\\
&\hat{\boldsymbol{e}}_{\mathsf{cjk},j} \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{e}'_{\mathsf{cjk},j})),\ \underline{\hat{\boldsymbol{e}}}_{\mathsf{cjk},j} \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{e}'_{\mathsf{cjk},j})) \text{ for } 0 \le j < L\\
&\hat{\boldsymbol{e}}_i \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{e}'_i)),\ \underline{\hat{\boldsymbol{e}}}_i \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{e}'_i)) \text{ for } 0 \le i < k/2\\
&\hat{\boldsymbol{m}}_i \leftarrow \texttt{REcd}(\texttt{Coeff}(\boldsymbol{m}'_i)),\ \underline{\hat{\boldsymbol{m}}}_i \leftarrow \texttt{REcd}(\texttt{NTT}(\boldsymbol{m}'_i)) \text{ for } 0 \le i < k/2
\end{aligned}$$

   Then, the prover sends polynomial oracles $\llbracket \hat{\boldsymbol{s}} \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket$, $\llbracket \hat{\boldsymbol{e}}_{\mathsf{ek}} \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{ek}} \rrbracket$, $\big\{\llbracket \hat{\boldsymbol{e}}_{\mathsf{rlk},j} \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{rlk},j} \rrbracket$, $\llbracket \hat{\boldsymbol{e}}_{\mathsf{rtk},j} \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{rtk},j} \rrbracket$, $\llbracket \hat{\boldsymbol{e}}_{\mathsf{cjk},j} \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{cjk},j} \rrbracket\big\}_{0 \le j < L}$, $\big\{\llbracket \hat{\boldsymbol{e}}_i \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{e}}}_i \rrbracket$, $\llbracket \hat{\boldsymbol{m}}_i \rrbracket$, $\llbracket \underline{\hat{\boldsymbol{m}}}_i \rrbracket\big\}_{0 \le i < k/2}$ of degree $\le 2N - 1$ to the verifier $\texttt{V}$.

2. The verifier sends a random point $\delta \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$.

3. The prover $\texttt{P}$ and the verifier $\texttt{V}$ invoke the following PIOPs.

$$\Pi_{L^\infty}(1; \llbracket \hat{\boldsymbol{s}} \rrbracket),\ \Pi_{L^2}(h; \llbracket \hat{\boldsymbol{s}} \rrbracket),\ \Pi_{\texttt{Lin}}(P_\varphi, P_\psi; \llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket; \llbracket \underline{\hat{\boldsymbol{s}}}_{\mathsf{rtk}} \rrbracket, \llbracket \underline{\hat{\boldsymbol{s}}}_{\mathsf{cjk}} \rrbracket) \tag{3}$$

$$\Pi_{L^\infty}(\lfloor p/Q_L \cdot \Delta \rceil ; \{\llbracket \hat{\boldsymbol{m}}_i \rrbracket\}_{0 \le i < k/2}) \tag{4}$$

$$\Pi_{L^\infty}(B'; \llbracket \hat{\boldsymbol{e}}_{\mathsf{ek}} \rrbracket, \{\llbracket \hat{\boldsymbol{e}}_{\mathsf{rlk},j} \rrbracket, \llbracket \hat{\boldsymbol{e}}_{\mathsf{rtk},j} \rrbracket, \llbracket \hat{\boldsymbol{e}}_{\mathsf{cjk},j} \rrbracket\}_{0 \le j < L}, \{\llbracket \hat{\boldsymbol{e}}_i \rrbracket\}_{0 \le i < k/2}) \tag{5}$$

$$\begin{gathered}\Pi_{\texttt{Lin}}\Big(T; \llbracket \hat{\boldsymbol{s}} \rrbracket, \llbracket \hat{\boldsymbol{e}}_{\mathsf{ek}} \rrbracket, \{\llbracket \hat{\boldsymbol{e}}_{\mathsf{rlk},j} \rrbracket, \llbracket \hat{\boldsymbol{e}}_{\mathsf{rtk},j} \rrbracket, \llbracket \hat{\boldsymbol{e}}_{\mathsf{cjk},j} \rrbracket\}_{0 \le j < L}, \{\llbracket \hat{\boldsymbol{e}}_i \rrbracket, \llbracket \hat{\boldsymbol{m}}_i \rrbracket\}_{0 \le i < k/2} ;\\
\llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{ek}} \rrbracket, \{\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{rlk},j} \rrbracket, \llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{rtk},j} \rrbracket, \llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{cjk},j} \rrbracket\}_{0 \le j < L}, \{\llbracket \underline{\hat{\boldsymbol{e}}}_i \rrbracket, \llbracket \underline{\hat{\boldsymbol{m}}}_i \rrbracket\}_{0 \le i < k/2}\Big)\end{gathered} \tag{6}$$

$$\Pi_{\texttt{AC}}(\texttt{NTT}(\mathsf{ek}'_0) + \texttt{NTT}(\mathsf{ek}'_1) \odot \vec{X}_0 - \vec{X}_1; \llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{ek}} \rrbracket) \tag{7}$$

$$\begin{gathered}\Pi_{\texttt{AC}}\Big(\sum_{j=0}^{L-1} \delta^j \cdot \Big(\texttt{NTT}(\mathsf{rlk}'_{0,j}) + \texttt{NTT}(\mathsf{rlk}'_{1,j}) \odot \vec{X}_0 - \lfloor p/q_i \rceil \cdot \vec{X}_0^2 - \vec{X}_{j+1}\Big);\\
\llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \{\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{rlk},j} \rrbracket\}_{0 \le j < L}\Big)\end{gathered} \tag{8}$$

$$\begin{gathered}\Pi_{\texttt{AC}}\Big(\sum_{j=0}^{L-1} \delta^j \cdot \Big(\texttt{NTT}(\mathsf{rtk}'_{0,j}) + \texttt{NTT}(\mathsf{rtk}'_{1,j}) \odot \vec{X}_0 - \lfloor p/q_i \rceil \cdot \vec{X}_1 - \vec{X}_{j+2}\Big);\\
\llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \llbracket \underline{\hat{\boldsymbol{s}}}_{\mathsf{rtk}} \rrbracket, \{\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{rtk},j} \rrbracket\}_{0 \le j < L}\Big)\end{gathered} \tag{9}$$

$$\begin{gathered}\Pi_{\texttt{AC}}\Big(\sum_{j=0}^{L-1} \delta^j \cdot \Big(\texttt{NTT}(\mathsf{cjk}'_{0,j}) + \texttt{NTT}(\mathsf{cjk}'_{1,j}) \odot \vec{X}_0 - \lfloor p/q_i \rceil \cdot \vec{X}_1 - \vec{X}_{j+2}\Big);\\
\llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \llbracket \underline{\hat{\boldsymbol{s}}}_{\mathsf{cjk}} \rrbracket, \{\llbracket \underline{\hat{\boldsymbol{e}}}_{\mathsf{cjk},j} \rrbracket\}_{0 \le j < L}\Big)\end{gathered} \tag{10}$$

$$\begin{gathered}\Pi_{\texttt{AC}}\Big(\sum_{i=0}^{k/2-1} \delta^i \cdot \Big(\texttt{NTT}(\mathsf{ct}'_{0,i}) + \texttt{NTT}(\mathsf{ct}'_{1,i}) \odot \vec{X}_0 - \vec{X}_{2i+1} - \vec{X}_{2i+2}\Big);\\
\llbracket \underline{\hat{\boldsymbol{s}}} \rrbracket, \{\llbracket \underline{\hat{\boldsymbol{e}}}_i \rrbracket, \llbracket \underline{\hat{\boldsymbol{m}}}_i \rrbracket\}_{0 \le i < k/2}\Big)\end{gathered} \tag{11}$$

Fig. 9: PIOP for CKKS ciphertexts and public key

Theorem 10. *An interactive protocol* $\Pi_{\mathtt{CT\&PK}}$ *described in Fig. 9 is an HVZK PIOP for a relation* $\mathsf{R}'_{\mathtt{CKK}}$ *defined below with a soundness error of* $\frac{O(k+L+N)}{p-N}$*, where* T*,* P_φ*, and* P_ψ *are matrices in* $\mathbb{Z}_p^{N\times N}$ *corresponding to* $\mathtt{NTT}$*,* φ*, and* ψ*, respectively, for the vector representation of coefficients.*

$$\mathsf{R}'_{\mathtt{CKKS}} = \left\{ (\mathsf{x}'_{\mathtt{CKKS}}, \mathsf{w}'_{\mathtt{CKKS}}) \;\middle|\; \begin{array}{r} \|\boldsymbol{s}\|_\infty \le 1 \wedge \|\boldsymbol{s}\|_1 \le h \wedge \\ \|\boldsymbol{e}'_{\mathsf{ek}}\|_\infty, \|\vec{\boldsymbol{e}}'_{\mathsf{rlk}}\|_\infty, \|\vec{\boldsymbol{e}}'_{\mathsf{rtk}}\|_\infty, \|\vec{\boldsymbol{e}}'_{\mathsf{cjk}}\|_\infty, \|\vec{\boldsymbol{e}}'\|_\infty \le B' \wedge \\ \|\boldsymbol{m}'_0\|_\infty, \ldots, \|\boldsymbol{m}'_{k/2-1}\|_\infty \le \lfloor p/Q_L \cdot \Delta \rceil \wedge \\ \mathsf{ek}'_0 + \boldsymbol{s} \cdot \mathsf{ek}'_1 = \boldsymbol{e}'_{\mathsf{ek}} \wedge \\ \mathsf{rlk}'_0 + \boldsymbol{s} \cdot \mathsf{rlk}'_1 = \boldsymbol{s}^2 \cdot \lfloor p/Q_L \cdot \vec{g} \rceil + \vec{\boldsymbol{e}}'_{\mathsf{rtk}} \wedge \\ \mathsf{rtk}'_0 + \boldsymbol{s} \cdot \mathsf{rtk}'_1 = \varphi(\boldsymbol{s}) \cdot \lfloor p/Q_L \cdot \vec{g} \rceil + \vec{\boldsymbol{e}}'_{\mathsf{rtk}} \wedge \\ \mathsf{cjk}'_0 + \boldsymbol{s} \cdot \mathsf{cjk}'_1 = \psi(\boldsymbol{s}) \cdot \lfloor p/Q_L \cdot \vec{g} \rceil + \vec{\boldsymbol{e}}_{\mathsf{cjk}} \wedge \\ \mathsf{ct}'_{i,0} + \boldsymbol{s} \cdot \mathsf{ct}'_{i,1} = \boldsymbol{m}'_i + \boldsymbol{e}'_i \text{ for } 0 \le i < k/2 \end{array} \right\}$$

Proof. Since $R_p \cong \mathbb{Z}_p^N$ via an isomorphism $\mathtt{NTT}$, we can prove the well-formedness of public keys and ciphertexts in the NTT representation, and norm constraints for the witness in the coefficient representation. Thus, the conditions for $\mathsf{R}'_{\mathtt{CKKS}}$ are directly translated from Eq. (4) to Eq. (11), except for the conditions for the secret key $\boldsymbol{s}$, where we need to prove $\|\boldsymbol{s}\|_\infty \le 1 \wedge \|\boldsymbol{s}\|_1 \le h$.

To be precise, in Eq. (3), we prove that $\|\boldsymbol{s}\|_\infty \le 1 \wedge [\|\boldsymbol{s}\|_2^2]_p \le h$. However, this statement is equivalent to $\|\boldsymbol{s}\|_\infty \le 1 \wedge \|\boldsymbol{s}\|_1 \le h$ since $[\|\boldsymbol{s}\|_2^2]_p = \|\boldsymbol{s}\|_1$ under the condition $\|\boldsymbol{s}\|_\infty \le 1$. Therefore, $\Pi_{\mathtt{CT\&PK}}$ proves the conditions in $\mathsf{R}'_{\mathtt{CKKS}}$. We note that the completeness, knowledge soundness, and zero-knowledge properties directly follow from Theorems 4 and 13 to 15. □

Theorem 11. *Let* $(\mathsf{pk}', \mathsf{ct}'_0, \ldots, \mathsf{ct}'_{k-1}) \in \mathsf{L}(\mathsf{R}'_{\mathtt{CKKS}})$*, and let* $\mathsf{pk} = \left\lfloor \frac{Q_L}{p} \cdot \mathsf{pk}' \right\rceil$*, and* $(\mathsf{ct}_{2i}, \mathsf{ct}_{2i+1}) \leftarrow \mathtt{CtoS}_{\mathsf{pk}}\big(\left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_i \right\rceil \big)$ *for* $0 \le i < k/2$*. Then, it holds that* $(\mathsf{pk}, \mathsf{ct}_0, \ldots, \mathsf{ct}_{k-1}) \in \mathsf{L}(\mathsf{R}_{\mathtt{CKKS}})$ *for* $B_{\mathsf{ek}} = B' + \frac{h+1}{2}$*,* $B_{\mathsf{rlk}} = B' + \frac{h^2+h+1}{2}$*,* $B_{\mathsf{rtk}} = B' + \frac{2h+1}{2}$*,* $B_{\mathsf{cjk}} = B' + \frac{2h+1}{2}$*,* $B_e = \mathtt{Estim}(\mathtt{CtoS}; h, B_{\mathsf{rlk}}, B_{\mathsf{rtk}}, B_{\mathsf{cjk}}, B' + \frac{h+3}{2}, \Delta)$*, and* $\mathcal{X} = \{\boldsymbol{x} \in R \mid \|\mathtt{Unpack}(\boldsymbol{x})\|_\infty \le 1\}$*.*

Proof. See Appendix D. □

5 Evaluation

In this section, we provide concrete parameters and benchmark results. To compile our PIOPs into SNARKs, we use the HSS scheme [53] as the underlying polynomial commitment scheme (PCS), as it supports large, NTT-friendly prime fields with fast proof generation speeds. Moreover, it offers a transparent setup with plausibly post-quantum security based on lattice cryptography. Our source code is available at `https://github.com/SNUCP/ckks-piop`.

5.1 Parameter Setting

For the CKKS scheme parameters, we set $N = 2^{14}$ and $\log Q_L = 404$, with $L + 1 = 9$ distinct primes, where $q_0, \ldots, q_{L-1}$ are 43 bits each, and q_L is 60 bits in size. For the secret key and error distributions, we use a sparse ternary distribution with Hamming weight 256 and a uniform ternary distribution, respectively. Overall, the parameters we selected satisfy 128 bits of security, which was verified using the lattice estimator [2].

For the HSS scheme parameters, we select values that minimize the proof size. As the base modulus p, we choose an NTT-friendly prime of similar size to q. In the HSS scheme, the modulus p takes the form of $b^r + 1$. To ensure p is NTT-friendly, it suffices to set b as an even number and $r \geq \log N$. Thus, we set $b = 10792$ and $r = 32$, so that $p \approx q$. For other parameters of the HSS scheme, we use $\lceil \log \mathsf{q} \rceil = 100$, $\mathsf{n} = 2^{13}$, $\mathsf{d} = 2^{11}$, $\mathsf{\mu} = 1$, and $\mathsf{\nu} = 2$. For a detailed explanation of these parameters, we refer to the parameter setting section in [53].

5.2 Benchmark Results

We implement $\Pi_{\mathtt{CT\&PK}}$ and present the benchmark results. Our implementation is written in Rust, building on the implementation of the HSS scheme available at `https://github.com/SNUCP/celpc`. All experiments were conducted on a machine equipped with an Intel(R) Xeon(R) Platinum 8268 CPU running at 2.90GHz, using a single thread. We measure performance by varying the number of ciphertexts $k = 2, 4, 8$ and scaling factor $\log \Delta = 16, 32$. The results are summarized in Table 1.

k	$\log \Delta$	PK Size	CT Size	Proof Size	Prover Time	Verifier Time
2	16	39.5 MB	1.57 MB	17.9 MB	324.35s	50.88s
4	16		3.14 MB	18.9 MB	365.46s	56.08s
8	16		6.28 MB	21.0 MB	442.86s	67.13s
2	32		1.57 MB	18.7 MB	356.90s	54.65s
4	32		3.14 MB	20.4 MB	425.26s	64.33s
8	32		6.28 MB	24.0 MB	561.28s	83.63s

Table 1: Proof sizes and benchmark results of $\Pi_{\mathtt{CT\&PK}}$.

We note that our proof size is smaller than the total size of the CKKS ciphertext and public keys, as the HSS scheme results in a proof size proportional to the square root of the witness size. The performance change is relatively moderate with respect to the number of ciphertexts, as a large portion of the proof generation process is dedicated to handling public keys, whose size is much larger than the ciphertexts size.

Since our ZKAoK is the first efficient construction for the CKKS scheme, we omit performance comparisons with other works. However, we note that the proof generation speed is comparable to other ZKAoKs for HE [22, 52] and PIOP-based vector range proofs [46]. This demonstrates the practicality of our construction, as proofs are generated by the receiver, which typically has lower computing power than the sender in homomorphic evaluation protocols. We also note that our PIOP construction is modular, allowing it to be compiled into different polynomial commitment schemes depending on the scenario. For instance, if the goal is to minimize the proof size further, compiling with the KZG [55] polynomial commitment scheme would yield proof sizes at the kilobyte level, at the cost of a trusted setup and longer proof generation times.

6 Conclusion

In this work, we analyze the security of a CKKS-based homomorphic evaluation protocol in a server-client setting. To address challenges in security analysis using indistinguishability-based notions, we introduce the concept of a differentially private homomorphic evaluation protocol, whose security is analyzed within the framework of differential privacy. To handle malicious client scenarios, we propose a general compilation method based on ZKAoK and present a concrete instantiation using PIOP. Finally, to demonstrate the practicality of our approach, we implement PIOP for CKKS by compiling it with the HSS polynomial commitment scheme [53].

We expect that our approach can be extended to multi-party variants of CKKS [26, 58, 59], where public keys are generated in a distributed manner under the common reference string model, inputs from multiple parties are shared as CKKS ciphertexts, and homomorphic computations can be performed on ciphertexts from different parties. In multi-party CKKS-based protocols, achieving indistinguishability-based security is challenging for similar reasons as in CKKS-based homomorphic evaluation protocols. Adopting our differential privacy-based analysis could lead to more practical parameter settings and a more robust security analysis in both the semi-honest and malicious settings.

References

1. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. pp. 308–318 (2016)
2. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015)
3. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold fhe. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 483–501. Springer (2012)
4. Attema, T., Lyubashevsky, V., Seiler, G.: Practical product proofs for lattice commitments. In: Annual International Cryptology Conference. pp. 470–499. Springer (2020)
5. Bae, Y., Cheon, J.H., Hanrot, G., Park, J.H., Stehlé, D.: Plaintext-ciphertext matrix multiplication and fhe bootstrapping: Fast and fused. In: Annual International Cryptology Conference. pp. 387–421. Springer (2024)
6. Bae, Y., Kim, J., Stehlé, D., Suvanto, E.: Bootstrapping small integers with ckks. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 330–360 (2024)
7. Beimel, A., Nissim, K., Omri, E.: Distributed private data analysis: Simultaneously solving how and what. In: Advances in Cryptology–CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings 28. pp. 451–468. Springer (2008)
8. Bell, J., Gascón, A., Lepoint, T., Li, B., Meiklejohn, S., Raykova, M., Yun, C.: {ACORN}: input validation for secure aggregation. In: 32nd USENIX Security Symposium (USENIX Security 23). pp. 4805–4822 (2023)
9. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for r1cs. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 103–128. Springer (2019)
10. Boemer, F., Costache, A., Cammarota, R., Wierzynski, C.: ngraph-he2: A high-throughput framework for neural network inference on encrypted data. In: Proceedings of the 7th ACM workshop on encrypted computing & applied homomorphic cryptography. pp. 45–56 (2019)
11. Bois, A., Cascudo, I., Fiore, D., Kim, D.: Flexible and efficient verifiable computation on encrypted data. In: IACR International Conference on Public-Key Cryptography. pp. 528–558. Springer (2021)
12. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical secure aggregation for privacy-preserving machine learning. In: proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1175–1191 (2017)
13. Boschini, C., Camenisch, J., Ovsiankin, M., Spooner, N.: Efficient post-quantum snarks for rsis and rlwe and their applications to privacy. In: Post-Quantum Cryptography: 11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings 11. pp. 247–267. Springer (2020)
14. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: Annual cryptology conference. pp. 868–886. Springer (2012)

15. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. pp. 309–325 (2012)
16. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE symposium on security and privacy (SP). pp. 315–334. IEEE (2018)
17. Bünz, B., Fisch, B., Szepieniec, A.: Transparent snarks from dark compilers. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 677–706 (2020)
18. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: Proceedings 42nd IEEE Symposium on Foundations of Computer Science. pp. 136–145. IEEE (2001)
19. Carlini, N., Jagielski, M., Mironov, I.: Cryptanalytic extraction of neural network models. In: Annual international cryptology conference. pp. 189–218. Springer (2020)
20. Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., Song, D.: The secret sharer: Evaluating and testing unintended memorization in neural networks. In: 28th USENIX security symposium (USENIX security 19). pp. 267–284 (2019)
21. Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al.: Extracting training data from large language models. In: 30th USENIX Security Symposium (USENIX Security 21). pp. 2633–2650 (2021)
22. Chatel, S., Mouchet, C., Sahin, A.U., Pyrgelis, A., Troncoso, C., Hubaux, J.P.: Pelta-shielding multiparty-fhe against malicious adversaries. In: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security. pp. 711–725 (2023)
23. Checri, M., Sirdey, R., Boudguiga, A., Bultel, J.P.: On the practical cpa d security of "exact" and threshold fhe schemes and libraries. In: Annual International Cryptology Conference. pp. 3–33. Springer (2024)
24. Chen, B., Bünz, B., Boneh, D., Zhang, Z.: Hyperplonk: Plonk with linear-time prover and high-degree custom gates. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 499–530. Springer (2023)
25. Chen, H., Chillotti, I., Song, Y.: Improved bootstrapping for approximate homomorphic encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 34–54. Springer (2019)
26. Chen, H., Dai, W., Kim, M., Song, Y.: Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference. In: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security. pp. 395–412 (2019)
27. Cheon, J.H., Choe, H., Passelègue, A., Stehlé, D., Suvanto, E.: Attacks against the ind-cpad security of exact fhe schemes. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. pp. 2505–2519 (2024)
28. Cheon, J.H., Han, K., Kim, A., Kim, M., Song, Y.: Bootstrapping for approximate homomorphic encryption. Advances in Cryptology–EUROCRYPT 2018 pp. 360–384 (2018)
29. Cheon, J.H., Han, K., Kim, A., Kim, M., Song, Y.: A full rns variant of approximate homomorphic encryption. In: International Conference on Selected Areas in Cryptography. pp. 347–368 (2018)

30. Cheon, J.H., Kim, A., Kim, M., Song, Y.: 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)
31. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: Preprocessing zksnarks with universal and updatable srs. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 738–768 (2020)
32. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Tfhe: fast fully homomorphic encryption over the torus. Journal of Cryptology **33**(1), 34–91 (2020)
33. Couteau, G., Goudarzi, D., Klooß, M., Reichle, M.: Sharp: Short relaxed range proofs. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. pp. 609–622 (2022)
34. Couteau, G., Klooß, M., Lin, H., Reichle, M.: Efficient range proofs with transparent setup from bounded integer commitments. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 247–277. Springer (2021)
35. Dahl, M., Demmler, D., El Kazdadi, S., Meyre, A., Orfila, J.B., Rotaru, D., Smart, N.P., Tap, S., Walter, M.: Noah's ark: Efficient threshold-fhe using noise flooding. In: Proceedings of the 11th Workshop on Encrypted Computing & Applied Homomorphic Cryptography. pp. 35–46 (2023)
36. Del Pino, R., Katsumata, S., Maller, M., Mouhartem, F., Prest, T., Saarinen, M.J.: Threshold raccoon: Practical threshold signatures from standard lattice assumptions. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 219–248. Springer (2024)
37. Dwork, C., Feldman, V.: Privacy-preserving prediction. In: Conference On Learning Theory. pp. 1693–1702. PMLR (2018)
38. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. Theory of Cryptography pp. 265–284 (2006)
39. Esgin, M.F., Espitau, T., Niot, G., Prest, T., Sakzad, A., Steinfeld, R.: Plover: Masking-friendly hash-and-sign lattice signatures. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 316–345. Springer (2024)
40. Esgin, M.F., Nguyen, N.K., Seiler, G.: Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 259–288. Springer (2020)
41. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive (2012)
42. Feigenbaum, J., Ishai, Y., Malkin, T., Nissim, K., Strauss, M.J., Wright, R.N.: Secure multiparty computation of approximations. In: Automata, Languages and Programming: 28th International Colloquium, ICALP 2001 Crete, Greece, July 8–12, 2001 Proceedings 28. pp. 927–938. Springer (2001)
43. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Conference on the theory and application of cryptographic techniques. pp. 186–194. Springer (1986)
44. Fiore, D., Nitulescu, A., Pointcheval, D.: Boosting verifiable computation on encrypted data. In: Public-Key Cryptography–PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part II 23. pp. 124–154. Springer (2020)
45. Ganesh, C., Nitulescu, A., Soria-Vazquez, E.: Rinocchio: Snarks for ring arithmetic. Journal of Cryptology **36**(4), 41 (2023)

46. Gao, R., Wan, Z., Hu, Y., Wang, H.: A succinct range proof for polynomial-based vector commitment. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. pp. 3152–3166 (2024)
47. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing. pp. 169–178. ACM (2009)
48. Gentry, C., Halevi, S., Lyubashevsky, V.: Practical non-interactive publicly verifiable secret sharing with thousands of parties. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 458–487. Springer (2022)
49. Guo, Q., Nabokov, D., Suvanto, E., Johansson, T.: Key recovery attacks on approximate homomorphic encryption with non-worst-case noise flooding countermeasures. In: Usenix Security (2024)
50. Halevi, S., Shoup, V.: Bootstrapping for helib. In: Annual International conference on the theory and applications of cryptographic techniques. pp. 641–670. Springer (2015)
51. Halevi, S., Shoup, V.: Faster homomorphic linear transformations in HElib. In: Annual International Cryptology Conference. pp. 93–120. Springer (2018)
52. Hwang, I., Lee, H., Seo, J., Song, Y.: Practical zero-knowledge PIOP for public key and ciphertext generation in (multi-group) homomorphic encryption. Cryptology ePrint Archive, Paper 2024/1879 (2024), `https://eprint.iacr.org/2024/1879`
53. Hwang, I., Seo, J., Song, Y.: Concretely efficient lattice-based polynomial commitment from standard assumptions. In: Annual International Cryptology Conference. pp. 414–448. Springer (2024)
54. Ju, J.H., Park, J., Kim, J., Kang, M., Kim, D., Cheon, J.H., Ahn, J.H.: Neujeans: Private neural network inference with joint optimization of convolution and bootstrapping. In: ACM CCS (2024)
55. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16. pp. 177–194. Springer (2010)
56. Kim, D., Lee, D., Seo, J., Song, Y.: Toward practical lattice-based proof of knowledge from hint-mlwe. In: Annual International Cryptology Conference. pp. 549–580. Springer (2023)
57. Kim, M., Song, Y., Wang, S., Xia, Y., Jiang, X., et al.: Secure logistic regression based on homomorphic encryption: Design and evaluation. JMIR medical informatics **6**(2), e8805 (2018)
58. Kim, T., Kwak, H., Lee, D., Seo, J., Song, Y.: Asymptotically faster multi-key homomorphic encryption from homomorphic gadget decomposition. In: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security. pp. 726–740 (2023)
59. Kwak, H., Lee, D., Song, Y., Wagh, S.: A general framework of homomorphic encryption for multiple parties with non-interactive key-aggregation. In: International Conference on Applied Cryptography and Network Security. pp. 403–430. Springer (2024)
60. Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., Jana, S.: Certified robustness to adversarial examples with differential privacy. In: 2019 IEEE symposium on security and privacy (SP). pp. 656–672. IEEE (2019)

61. Lee, E., Lee, J.W., Lee, J., Kim, Y.S., Kim, Y., No, J.S., Choi, W.: Low-complexity deep convolutional neural networks on fully homomorphic encryption using multiplexed parallel convolutions. In: International Conference on Machine Learning. pp. 12403–12422. PMLR (2022)
62. Li, B., Micciancio, D.: On the security of homomorphic encryption on approximate numbers. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 648–677. Springer (2021)
63. Li, B., Micciancio, D., Schultz-Wu, M., Sorrell, J.: Securing approximate homomorphic encryption using differential privacy. In: Annual International Cryptology Conference. pp. 560–589. Springer (2022)
64. López-Alt, A., Tromer, E., Vaikuntanathan, V.: 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)
65. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 738–755. Springer (2012)
66. Lyubashevsky, V., Nguyen, N.K., Seiler, G.: Practical lattice-based zero-knowledge proofs for integer relations. In: Proceedings of the 2020 ACM SIGSAC conference on computer and communications security. pp. 1051–1070 (2020)
67. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29. pp. 1–23. Springer (2010)
68. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing **37**(1), 267–302 (2007)
69. Micciancio, D., Vaikuntanathan, V.: Sok: Learning with errors, circular security, and fully homomorphic encryption. In: IACR International Conference on Public-Key Cryptography. pp. 291–321. Springer (2024)
70. Mironov, I., Pandey, O., Reingold, O., Vadhan, S.: Computational differential privacy. In: Annual International Cryptology Conference. pp. 126–142. Springer (2009)
71. Naor, M., Pinkas, B.: Oblivious polynomial evaluation. SIAM Journal on Computing **35**(5), 1254–1281 (2006)
72. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) **56**(6), 1–40 (2009)
73. Sav, S., Pyrgelis, A., Troncoso-Pastoriza, J.R., Froelicher, D., Bossuat, J., Sousa, J.S., Hubaux, J.: POSEIDON: privacy-preserving federated neural network learning. In: 28th Annual Network and Distributed System Security Symposium, NDSS. The Internet Society (2021)
74. Shalev-Shwartz, S., Shamir, O., Srebro, N., Sridharan, K.: Learnability, stability and uniform convergence. The Journal of Machine Learning Research **11**, 2635–2670 (2010)
75. Shi, E., Chan, H., Rieffel, E., Chow, R., Song, D.: Privacy-preserving aggregation of time-series data. In: Annual Network & Distributed System Security Symposium (NDSS). Internet Society. (2011)
76. Shokri, R., Stronati, M., Song, C., Shmatikov, V.: Membership inference attacks against machine learning models. In: 2017 IEEE symposium on security and privacy (SP). pp. 3–18. IEEE (2017)

A CKKS Error Analysis

This section provides an error analysis for each homomorphic operation. In the analysis below, we assume that our chosen set of prime modulus $Q_L = q_1 \dots q_L$ satisfies the condition $q_L > q_i LN$ for all q_i $(1 \le i < L)$. It is important to note that selecting q_L in this manner is a common practice in practical parameter settings, as it helps reduce noise introduction during homomorphic computations. Additionally, for simplicity, we will use the same notation as in the algorithm description within the proof.

Lemma 4 (External Product). *Suppose that* $\vec{\boldsymbol{u}}_0 + \vec{\boldsymbol{u}}_1 \boldsymbol{s} = \boldsymbol{\mu} \cdot \vec{g} + \vec{\boldsymbol{e}} \pmod{Q_L}$ *for some vectors* $\vec{\boldsymbol{u}}_0 = (\boldsymbol{u}_{0,0}, \dots, \boldsymbol{u}_{0,L}), \vec{\boldsymbol{u}}_1 = (\boldsymbol{u}_{1,0}, \dots, \boldsymbol{u}_{1,L}) \in R_{Q_L}^L$ *and ring element* $\boldsymbol{\mu} \in R_{Q_{L-1}}$. *Then, the following equation holds for any* $\boldsymbol{a} \in R_{Q_\ell}$ *and* $\vec{\boldsymbol{e}} = (\boldsymbol{e}_0, \dots, \boldsymbol{e}_L) \in R_{Q_L}^L$ *where* $0 \le \ell < L$.

$$\mathtt{Err}_{\mathsf{sk}}\left(\boldsymbol{a} \boxdot (\vec{\boldsymbol{u}}_0, \vec{\boldsymbol{u}}_1), \boldsymbol{\mu} \cdot \boldsymbol{a}\right) \le \frac{\|\vec{\boldsymbol{e}}\|_\infty + h + 1}{2}.$$

Proof. By definition, we have

$$\begin{aligned}
&\underline{\mathtt{Dec}_{\mathsf{sk}}}(\boldsymbol{a} \boxdot (\vec{\boldsymbol{u}}_0, \vec{\boldsymbol{u}}_1)) = \boldsymbol{a} \boxdot \vec{\boldsymbol{u}}_0 + (\boldsymbol{a} \boxdot \vec{\boldsymbol{u}}_1) \cdot \boldsymbol{s} \\
&= \frac{1}{q_L} \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} (\boldsymbol{u}_{0,i} + \boldsymbol{u}_{1,i} \cdot \boldsymbol{s}) - \boldsymbol{r}_0 - \boldsymbol{r}_1 \cdot \boldsymbol{s} \pmod{Q_\ell}
\end{aligned}$$

where

$$\boldsymbol{r}_i = \frac{1}{q_L} \sum_{j=0}^{\ell-1} \left[(Q_{L-1}/q_j)^{-1} \cdot \boldsymbol{a}\right]_{q_j} \boldsymbol{u}_{i,j} - \left\lfloor \frac{1}{q_L} \sum_{j=0}^{\ell-1} \left[(Q_{L-1}/q_j)^{-1} \cdot \boldsymbol{a}\right]_{q_j} \boldsymbol{u}_{i,j} \right\rceil$$

for $i = 0, 1$.

Note that $\boldsymbol{u}_{0,i} + \boldsymbol{u}_{1,i} \cdot \boldsymbol{s} = \boldsymbol{\mu} \cdot Q_L / q_i + \boldsymbol{e}_i \pmod{Q_\ell \cdot q_L}$ since $Q_\ell \cdot q_L \mid Q_L$, we have the following by definition.

$$\begin{aligned}
&\sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} (\boldsymbol{u}_{0,i} + \boldsymbol{u}_{1,i} \cdot \boldsymbol{s}) \\
&= \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} (\boldsymbol{\mu} \cdot Q_L/q_i + \boldsymbol{e}_i) \\
&= q_L \cdot \boldsymbol{\mu} \cdot \boldsymbol{a} + \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} \cdot \boldsymbol{e}_i \pmod{Q_\ell \cdot q_L}
\end{aligned}$$

Note that the final equality comes from the Chinese Remainder Theorem. (Consider modulo q_i for $i = 0, \dots, \ell$ and L.) Therefore, we can conclude the following inequality:

$$\begin{aligned}
&\mathtt{Err}_{\mathsf{sk}}(\boldsymbol{a} \boxdot (\vec{\boldsymbol{u}}_0, \vec{\boldsymbol{u}}_1), \boldsymbol{\mu} \cdot \boldsymbol{a}) \\
&= \left\| \frac{1}{q_L} \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} (\boldsymbol{u}_{0,i} + \boldsymbol{u}_{1,i} \cdot \boldsymbol{s}) - \boldsymbol{r}_0 - \boldsymbol{r}_1 \cdot \boldsymbol{s} - \boldsymbol{\mu} \cdot \boldsymbol{a} \pmod{Q_\ell} \right\|_\infty \\
&= \left\| \frac{1}{q_L} \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} \cdot \boldsymbol{e}_i - \boldsymbol{r}_0 - \boldsymbol{r}_1 \cdot \boldsymbol{s} \right\|_\infty \\
&\leq \frac{1}{q_L} \left\| \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} \cdot \boldsymbol{e}_i \right\|_\infty + \|\boldsymbol{r}_0 + \boldsymbol{r}_1 \cdot \boldsymbol{s}\|_\infty
\end{aligned}$$

where the second equality is derived from the smallness of the term which does not incur the wraparound. Now, since $\left\| [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} \right\|_\infty \leq \frac{q_i}{2} < \frac{q_L}{2LN}$ by the assumption, the first term is bounded by

$$\frac{1}{q_L} \left\| \sum_{i=0}^{\ell-1} [(Q_{L-1}/q_i)^{-1} \cdot \boldsymbol{a}]_{q_i} \cdot \boldsymbol{e}_i \right\|_\infty < \frac{1}{q_L} \cdot \ell \cdot \frac{q_L}{2LN} \cdot N \|\vec{\boldsymbol{e}}\|_\infty \leq \frac{\|\vec{\boldsymbol{e}}\|_\infty}{2}.$$

On the other hand, since $\|\boldsymbol{r}_i\|_\infty \leq 1/2$, it is easy to show that $\|\boldsymbol{r}_0 + \boldsymbol{r}_1 \cdot \boldsymbol{s}\|_\infty \leq \frac{h+1}{2}$ from the fact that $\boldsymbol{s}$ is a ternary polynomial with Hamming weight at most h. Hence, we can obtain the desired noise bound $(\|\vec{\boldsymbol{e}}\|_\infty + h + 1)/2$. □

Lemma 5 (Homomorphic Addition). *Given CKKS ciphertexts* $\mathsf{ct}, \mathsf{ct}' \in R_{Q_\ell}^2$ *for* $0 \leq \ell < L$, *let* $\mathsf{ct}_{add} \leftarrow \underline{\mathtt{Add}}(\mathsf{ct}, \mathsf{ct}')$. *Then, it satisfies that*

$$\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}_{add}, \boldsymbol{m} + \boldsymbol{m}') \leq \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) + \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \boldsymbol{m}').$$

Proof. Note that $\mathsf{ct}_{add} = (\boldsymbol{c}_0 + \boldsymbol{c}_0', \boldsymbol{c}_1 + \boldsymbol{c}_1') \in R_{Q_\ell}$. By definition, we obtain

$$\begin{aligned}
\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}_{add}, \boldsymbol{m} + \boldsymbol{m}') &= \|\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}_{add}) - \boldsymbol{m} - \boldsymbol{m}'\|_\infty \\
&= \|[\boldsymbol{c}_0 + \boldsymbol{c}_0' + (\boldsymbol{c}_1 + \boldsymbol{c}_1')\boldsymbol{s}]_{Q_\ell} - \boldsymbol{m} - \boldsymbol{m}'\|_\infty \\
&\leq \|[\boldsymbol{c}_0 + \boldsymbol{c}_1\boldsymbol{s} - \boldsymbol{m}]_{Q_\ell}\|_\infty + \|[\boldsymbol{c}_0' + \boldsymbol{c}_1'\boldsymbol{s} - \boldsymbol{m}']_{Q_\ell}\|_\infty \\
&= \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) + \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \boldsymbol{m}'),
\end{aligned}$$

where the inequality is derived from the smallness of the errors and the triangle inequality. □

Lemma 6 (Homomorphic Multiplication). *Given CKKS ciphertexts* $\mathsf{ct}, \mathsf{ct}' \in R_{Q_\ell}^2$ *for* $0 \leq \ell < L$, *let* $\mathsf{ct}_{mul} \leftarrow \underline{\mathtt{Mul}_{\mathsf{rlk}}}(\mathsf{ct}, \mathsf{ct}')$. *Then, it satisfies that*

$$\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}_{mul}, \boldsymbol{m}\boldsymbol{m}') \leq N\|\boldsymbol{m}'\|_\infty E + N\|\boldsymbol{m}\|_\infty E' + NEE' + \frac{B_{\mathsf{rlk}} + h + 1}{2}$$

for $E = \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m})$, $E' = \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \boldsymbol{m}')$ *and* B_{rlk} *is the infinity-norm bound of the relinearization key error.*

Proof. Let us parse $\boldsymbol{d}_2 \boxdot \mathsf{rlk} = (\boldsymbol{f}_0, \boldsymbol{f}_1) \in R_{Q_\ell}$. Then, by Lemma 4, we have

$$\left\| \left[\boldsymbol{f}_0 + \boldsymbol{f}_1\boldsymbol{s} - \boldsymbol{d}_2 \cdot \boldsymbol{s}^2\right]_{Q_\ell} \right\|_\infty = \left\| \left[\boldsymbol{f}_0 + \boldsymbol{f}_1\boldsymbol{s} - \boldsymbol{c}_1\boldsymbol{c}_1' \cdot \boldsymbol{s}^2\right]_{Q_\ell} \right\|_\infty \leq \frac{B_{\mathsf{rlk}} + h + 1}{2}.$$

Next, let $\boldsymbol{e}$ and $\boldsymbol{e}'$ be the 'noise' of the ciphertexts ct and ct' which encrypt $\boldsymbol{m}$ and $\boldsymbol{m'}$, respectively. In other words, $\underline{\mathtt{Dec}_{\mathsf{sk}}}(\mathsf{ct}) = \boldsymbol{m} + \boldsymbol{e}$ and $\underline{\mathtt{Dec}_{\mathsf{sk}}}(\mathsf{ct}') = \boldsymbol{m'} + \boldsymbol{e'}$. Then, we can deduce that

$$\begin{aligned}
&\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}_{mul}, \boldsymbol{mm'}) \\
&= \left\| [\boldsymbol{c}_0\boldsymbol{c}_0' + (\boldsymbol{c}_0\boldsymbol{c}_1' + \boldsymbol{c}_0'\boldsymbol{c}_1)\boldsymbol{s} + (\boldsymbol{f}_0 + \boldsymbol{f}_1\boldsymbol{s})]_{Q_\ell} - \boldsymbol{mm'} \right\|_\infty \\
&\leq \left\| \left[\boldsymbol{c}_0\boldsymbol{c}_0' + (\boldsymbol{c}_0\boldsymbol{c}_1' + \boldsymbol{c}_0'\boldsymbol{c}_1)\boldsymbol{s} + \boldsymbol{c}_1\boldsymbol{c}_1'\boldsymbol{s}^2\right]_{Q_\ell} - \boldsymbol{mm'} \right\|_\infty + \left\| \left[\boldsymbol{f}_0 + \boldsymbol{f}_1\boldsymbol{s} - \boldsymbol{c}_1\boldsymbol{c}_1' \cdot \boldsymbol{s}^2\right]_{Q_\ell} \right\|_\infty \\
&\leq \left\| \left[\boldsymbol{c}_0\boldsymbol{c}_0' + (\boldsymbol{c}_0\boldsymbol{c}_1' + \boldsymbol{c}_0'\boldsymbol{c}_1)\boldsymbol{s} + \boldsymbol{c}_1\boldsymbol{c}_1'\boldsymbol{s}^2\right]_{Q_\ell} - \boldsymbol{mm'} \right\|_\infty + \frac{B_{\mathsf{rlk}} + h + 1}{2} \\
&= \left\| (\boldsymbol{m} + \boldsymbol{e})(\boldsymbol{m}' + \boldsymbol{e}') - \boldsymbol{mm'} \right\|_\infty + \frac{B_{\mathsf{rlk}} + h + 1}{2} \\
&\leq N\|\boldsymbol{m}\|_\infty\|\boldsymbol{e}'\|_\infty + N\|\boldsymbol{m}'\|_\infty\|\boldsymbol{e}\|_\infty + N\|\boldsymbol{e}\|_\infty\|\boldsymbol{e}'\|_\infty + \frac{B_{\mathsf{rlk}} + h + 1}{2}.
\end{aligned}$$

Since $\|\boldsymbol{e}\|_\infty \leq \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m})$ and $\|\boldsymbol{e}'\|_\infty \leq \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \boldsymbol{m}')$ by definition, we proved our claim. □

Lemma 7 (Homomorphic Rotation). *Given CKKS ciphertext* $\mathsf{ct} \in R_{Q_\ell}^2$ *for* $0 \leq \ell < L$*, let* $\mathsf{ct}' \leftarrow \underline{\mathtt{Rot}_{\mathsf{rtk}}}(\mathsf{ct})$*. Then, it satisfies that*

$$\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \varphi(\boldsymbol{m})) \leq \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) + \frac{B_{\mathsf{rtk}} + h + 1}{2}$$

where B_{rtk} *is the infinity norm bound of the rotation key error.*

Proof. By Lemma 4, we have

$$\begin{aligned}
&\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \varphi(\boldsymbol{m})) \\
&= \left\| [\varphi(\boldsymbol{c}_0) + \varphi(\boldsymbol{c}_1) \boxdot \mathsf{rtk}]_{Q_\ell} - \varphi(\boldsymbol{m}) \right\|_\infty \\
&\leq \left\| [\varphi(\boldsymbol{c}_0) + \varphi(\boldsymbol{c}_1) \cdot \varphi(\boldsymbol{s})]_{Q_\ell} - \varphi(\boldsymbol{m}) \right\|_\infty + \left\| [\varphi(\boldsymbol{c}_1) \boxdot \mathsf{rtk} - \varphi(\boldsymbol{c}_1) \cdot \varphi(\boldsymbol{s})]_{Q_\ell} \right\|_\infty \\
&\leq \left\| [\varphi(\boldsymbol{c}_0 + \boldsymbol{c}_1 \cdot \boldsymbol{s} - \boldsymbol{m})]_{Q_\ell} \right\|_\infty + \frac{B_{\mathsf{rtk}} + h + 1}{2} = \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) + \frac{B_{\mathsf{rtk}} + h + 1}{2}
\end{aligned}$$

where the last equality is derived from the fact that the automorphism $X \mapsto X^5$ only rearranges the coefficients up to sign. □

Lemma 8 (Homomorphic Conjugation). *Given CKKS ciphertext* $\mathsf{ct} \in R_{Q_\ell}^2$ *for* $0 \leq \ell < L$*, let* $\mathsf{ct}' \leftarrow \underline{\mathtt{Conj}_{\mathsf{cjk}}}(\mathsf{ct})$*. Then it satisfies that*

$$\mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}', \psi(\boldsymbol{m})) \leq \mathtt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) + \frac{B_{\mathsf{cjk}} + h + 1}{2}$$

where B_{cjk} *is the infinity norm bound of the rotation key.*

Proof. The proof is essentially identical to the proof of Lemma 7. □

Lemma 9 (Rounding Operation). *Given CKKS ciphertext* $\mathsf{ct} \in R_{Q_\ell}^2$ *for* $0 \leq \ell < L$, *let* $\mathsf{ct}' \leftarrow \underline{\texttt{Round}}(\mathsf{ct})$. *Then, it satisfies that*

$$\texttt{Err}_{\mathsf{sk}}(\mathsf{ct}', \boldsymbol{m}/q_\ell) \leq \frac{h+1}{2}.$$

Proof. Let $\underline{\texttt{Dec}_{\mathsf{sk}}}(\mathsf{ct}) = \boldsymbol{m} + \boldsymbol{e}$ for some small noise $\boldsymbol{e}$. Then,

$$\begin{aligned} [\lfloor \boldsymbol{c}_0/q_\ell \rceil + \lfloor \boldsymbol{c}_1/q_\ell \rceil \cdot \boldsymbol{s}]_{Q_{\ell-1}} &= \frac{1}{q_\ell} [\boldsymbol{c}_0 + \boldsymbol{c}_1 \cdot \boldsymbol{s}]_{Q_\ell} - \boldsymbol{r}_0 - \boldsymbol{r}_1 \boldsymbol{s} \\ &= \frac{1}{q_\ell} (\boldsymbol{m} + \boldsymbol{e}) - \boldsymbol{r}_0 - \boldsymbol{r}_1 \boldsymbol{s} \end{aligned}$$

where

$$\boldsymbol{r}_i = \frac{1}{q_\ell} \boldsymbol{c}_i - \left\lfloor \frac{1}{q_\ell} \boldsymbol{c}_i \right\rceil$$

for $i = 0, 1$. Analogous to the proof of Lemma 4, we can deduce that $\|\boldsymbol{r}_i\|_\infty \leq 1/2$. Therefore, from the definition,

$$\begin{aligned} \texttt{Err}_{\mathsf{sk}}(\mathsf{ct}', \boldsymbol{m}/q_\ell) &= \left\| [\lfloor \boldsymbol{c}_0/q_\ell \rceil + \lfloor \boldsymbol{c}_1/q_\ell \rceil \cdot \boldsymbol{s}]_{Q_{\ell-1}} - \boldsymbol{m}/q_\ell \right\|_\infty \\ &= \left\| \frac{1}{q_\ell} \boldsymbol{e} - \boldsymbol{r}_0 - \boldsymbol{r}_1 \boldsymbol{s} \right\|_\infty \\ &\leq \frac{1}{q_\ell} \|\boldsymbol{e}\|_\infty + \frac{h+1}{2} = \frac{1}{q_\ell} \texttt{Err}_{\mathsf{sk}}(\mathsf{ct}, \boldsymbol{m}) + \frac{h+1}{2} \end{aligned}$$

as $\boldsymbol{s}$ is a ternary vector with Hamming weight h. □

Proof (Proof of Theorem 2.). In Lemmas 5 to 9, we provided specific formulas to calculate the error bounds following each homomorphic operation. These calculations rely on several parameters: the error and message bounds of the input ciphertext(s) of the operation, the error bounds of the public keys, and the Hamming weight bound of the secret key. With these parameters, and given the circuit, one can deduce the error in the output ciphertext by applying these formulas systematically through the circuit. In other words, there is an algorithm $\texttt{Estim}(\texttt{C}; h, B_{\mathsf{rlk}}, B_{\mathsf{rtk}}, B_{\mathsf{cjk}}, B_e, B_m)$ that calculates the upper limit of the error bound for the output ciphertext of the specified circuit $\texttt{C}$. □

B PIOP Compilation

B.1 Polynomial Commitment Scheme

A polynomial commitment scheme (PCS) is a class of commitment schemes that takes polynomials as messages and allows the evaluation of committed polynomials. Below, we define a polynomial commitment scheme for univariate polynomials, adapted from [31].

Definition 14 (Polynomial Commitment). *A polynomial commitment* $\mathtt{PC}$ *consists of the following PPT algorithms.*

- $\underline{\mathtt{PC.Setup}}(1^\lambda, D) \to \mathsf{ck}$*: Given a security parameter* λ *and a global polynomial degree upper bound* D*, it generates a commitment key* ck*.*
- $\underline{\mathtt{PC.Com}}(\mathsf{ck}, d, \boldsymbol{f}) \to (c, \delta)$*: Given a polynomial* $\boldsymbol{f} \in \mathbb{Z}_p[X]$ *with degree* $< d$*, it generates a commitment* c *and an opening hint* δ*.*
- $\underline{\mathtt{PC.Open}}(\mathsf{ck}, c, d, \boldsymbol{f}, \delta) \to b$*: Given a commitment* c*, a polynomial* $\boldsymbol{f}$ *with degree* $< d$*, and an opening hint* δ*, it outputs* 0 *or* 1*.*
- $\underline{\mathtt{PC.Eval}}(\mathsf{ck}, x, d, \boldsymbol{f}, \delta) \to (y, \rho)$*: Given an evaluation point* $x \in \mathbb{Z}_p$ *and an opening hint* δ*, it returns an evaluation result* y*, and an evaluation proof* ρ*.*
- $\underline{\mathtt{PC.Check}}(\mathsf{ck}, c, d, x, y, \rho) \to b$*: Given a commitment* c*, a degree upper bound* d*, an evaluation point* x*, an evaluation result* y*, and an evaluation proof* ρ*, it outputs* 0 *or* 1*.*

$\mathtt{PC}$ *is called a polynomial commitment scheme if it satisfies the following properties.*

Correctness. *For every polynomial* $\boldsymbol{f} \in \mathbb{Z}_p[X]$ *with a degree upper bound* $d \leq D$ *and every point* $x \in \mathbb{Z}_p$*, the following holds.*

$$\Pr\left[\begin{array}{r|c} \mathtt{PC.Open}(\mathsf{ck}, c, d, \boldsymbol{f}, \delta) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, D) \\ \mathtt{PC.Check}(\mathsf{ck}, c, d, x, \boldsymbol{f}(x), \rho) = 1 & (c, \delta) \leftarrow \mathtt{PC.Com}(\mathsf{ck}, d, \boldsymbol{f}) \\ & (y, \rho) \leftarrow \mathtt{PC.Eval}(x, \delta) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Extractability. *For every PPT adversary* $\mathtt{A}$*, there exists a PPT extractor* $\mathtt{E}$ *such that for all randomness* r*, the following holds.*

$$\Pr\left[\begin{array}{r|r} \mathtt{PC.Check}(\mathsf{ck}, c, d, x, y, \rho) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, D) \\ \big(\mathtt{PC.Open}(\mathsf{ck}, c, d, \boldsymbol{f}, \delta) = 0 \vee & (c, d, x, y, \rho) \leftarrow \mathtt{A}(\mathsf{ck}, \mathsf{r}) \\ y \neq \boldsymbol{f}(x)\big) & (\boldsymbol{f}, \delta) \leftarrow \mathtt{E}(\mathsf{ck}, \mathsf{r}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

Binding. *For every PPT adversary* $\mathtt{A}$*, the following holds.*

$$\Pr\left[\begin{array}{r|c} \mathtt{PC.Open}(\mathsf{ck}, c, d, \boldsymbol{f}, \delta) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, D) \\ \mathtt{PC.Open}(\mathsf{ck}, c, d, \boldsymbol{f}', \delta') = 1 \wedge & (c, d, \boldsymbol{f}, \boldsymbol{f}', \delta, \delta') \leftarrow \mathtt{A}(\mathsf{ck}) \\ \boldsymbol{f} \neq \boldsymbol{f}' & \end{array}\right] \leq \mathsf{negl}(\lambda)$$

$\mathtt{PC}$ *is called hiding if the following property holds.*

Hiding. *For every PPT adversary* $\mathtt{A} = (\mathtt{A}_1, \mathtt{A}_2)$*, there exists a PPT simulator* $\mathtt{S}$ *such that the following holds.*

$$\left| \Pr\left[\begin{array}{r} \mathtt{A}_2(\mathsf{ck}, c, \rho) = 1 \wedge \\ \mathtt{PC.Check}(\mathsf{ck}, c, d, x, \boldsymbol{f}(x), \rho) = 1 \end{array} \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, D) \\ (d, \boldsymbol{f}, x) \leftarrow \mathtt{A}_1(\mathsf{ck}) \\ (c, \rho) \leftarrow \mathtt{S}(\mathsf{ck}, x, \boldsymbol{f}(x)) \end{array} \right] \right.$$
$$\left. - \Pr\left[\begin{array}{r} \mathtt{A}_2(\mathsf{ck}, c, \rho) = 1 \wedge \\ \mathtt{PC.Check}(\mathsf{ck}, c, d, x, \boldsymbol{f}(x), \rho) = 1 \end{array} \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, D) \\ (d, \boldsymbol{f}, x) \leftarrow \mathtt{A}_1(\mathsf{ck}) \\ (c, \delta) \leftarrow \mathtt{PC.Com}(\mathsf{ck}, d, \boldsymbol{f}) \\ (y, \rho) \leftarrow \mathtt{PC.Eval}(x, \delta) \end{array} \right] \right| \leq \mathsf{negl}(\lambda)$$

In [31], Chiesa et al. formalize how a PIOP can be compiled into an argument of knowledge using a PCS. In short, the compilation is done by replacing all the oracle polynomials in the PIOP with commitments from the PCS, and then attaching evaluation proofs from the PCS for each polynomial query in the PIOP. The complexity of the resulting argument of knowledge can be described as follows.

Theorem 12 (Theorem 8.1 in [31]). *Let* Π *be a PIOP for a relation* R *and* $\mathtt{PC}$ *be a polynomial commitment scheme. Then, there exists a public coin argument of knowledge* Π' *for* R *with the following complexity.*

Prover complexity. *The sum of the runtime of the PIOP prover, the time to commit polynomials in* $\mathtt{PC}$*, and the time to produce evaluation proofs for oracle queries in* $\mathtt{PC}$*.*

Verifier complexity. *The sum of the runtime of the PIOP verifier, the time to verify evaluation proofs in* $\mathtt{PC}$*.*

Proof size. *The sum of the messages from the PIOP verifier, commitments size in* $\mathtt{PC}$*, and evaluation proof size in* $\mathtt{PC}$*. Additionally, if* Π *is HVZK and* $\mathtt{PC}$ *is hiding, then* Π' *is HVZK.*

C Deferred PIOPs

Theorem 13. *Let* $\hat{\boldsymbol{a}}_i \leftarrow \mathtt{REcd}(\vec{a}_i)$ *and* $\hat{\boldsymbol{b}}_i \leftarrow \mathtt{REcd}(\vec{b}_i)$ *for* $0 \leq i < k$*, where* $\vec{b}_i = M_i \vec{a}_i$*. Then, an interactive protocol* $\Pi_{\mathtt{Lin}}$ *described in Fig. 10 is an HVZK PIOP with a soundness error of* $\frac{O(k+N)}{p-N}$*.*

$$\Pi_{\mathtt{Lin}}(M_0, \ldots, M_{k-1}; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!]; [\![\hat{\boldsymbol{b}}_0]\!], \ldots, [\![\hat{\boldsymbol{b}}_{k-1}]\!])$$

Public input: matrices $M_i \in \mathbb{Z}_p^{N \times N}$ for $0 \le i < k$.
Witness: vectors $\vec{a}_i = \mathtt{Dcd}(\hat{\boldsymbol{a}}_i)$ and $\vec{b}_i = \mathtt{Dcd}(\hat{\boldsymbol{b}}_i)$ for $0 \le i < k$, where $\hat{\boldsymbol{a}}_i, \hat{\boldsymbol{b}}_i \in \mathbb{Z}_p^{<2N}[X]$.
Statement: $\vec{b}_i = M_i \vec{a}_i$ for $0 \le i < k$.

1. The verifier $\mathtt{V}$ sends random point $v, \gamma \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$.
2. The prover $\mathtt{P}$ and the verifier $\mathtt{V}$ invoke the following PIOP, where $\vec{v} = (1, v, \ldots, v^{N-1})$ and $\vec{w}_i = M_i^\top \vec{v}$.

$$\Pi_{\mathtt{IP}}\left(\sum_{i=0}^{k-1} \gamma^i \cdot (\vec{w}_i \odot \vec{X}_i - \vec{v} \odot \vec{X}_{i+k}); [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!], [\![\hat{\boldsymbol{b}}_0]\!], \ldots, [\![\hat{\boldsymbol{b}}_{k-1}]\!]\right)$$

Fig. 10: PIOP for batched linear relations

Proof. We refer to Theorem 6.2 in [9]. □

Theorem 14. *Let $\hat{\boldsymbol{a}}_i \leftarrow \mathtt{REcd}(\vec{a}_i)$ for $0 \le i < k$, where $\vec{\mathtt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}) = \vec{0}$. Then, an interactive protocol $\Pi_{\mathtt{AC}}$ described in Fig. 11 is an HVZK PIOP with a soundness error of $\frac{O(dN)}{p-N}$.*

$$\Pi_{\mathtt{AC}}(\vec{\mathtt{C}}; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!])$$

Public input: k-ary circuit $\vec{\mathtt{C}} \in (\mathbb{Z}_p^N)[\vec{X}_0, \ldots, \vec{X}_{k-1}]$ of degree d.
Witness: vectors $\vec{a}_i = \mathtt{Dcd}(\boldsymbol{a}_i)$ for $0 \le i < k$, where $\hat{\boldsymbol{a}}_i \in \mathbb{Z}_p^{<2N}[X]$.
Statement: $\vec{\mathtt{C}}(\vec{a}_0, \ldots, \vec{a}_{k-1}) = 0$.

1. Both $\mathtt{P}$ and $\mathtt{V}$ compute $\boldsymbol{\mathtt{C}} \in (\mathbb{Z}_p[X])[\boldsymbol{X}_0, \ldots, \boldsymbol{X}_{k-1}]$ of degree d, which is obtained by applying $\mathtt{Ecd}$ to each coefficient of $\vec{\mathtt{C}}$.
2. The prover $\mathtt{P}$ computes the quotient polynomial $\boldsymbol{q} = \boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1}) / \boldsymbol{z}_{\mathbb{H}}$, and sends a polynomial oracle $([\![\boldsymbol{q}]\!], d(2N-1)-1)$ to $\mathtt{V}$.
3. The verifier $\mathtt{V}$ gets evaluations $\boldsymbol{a}_i(\alpha)$ and $\boldsymbol{q}(\alpha)$ by accessing polynomial oracles $[\![\hat{\boldsymbol{a}}_i]\!]$, $[\![\boldsymbol{q}]\!]$ at a random point $\alpha \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$. Then, $\mathtt{V}$ checks whether the following holds.

$$\boldsymbol{q}(\alpha) \cdot \boldsymbol{z}_{\mathbb{H}}(\alpha) = \big(\boldsymbol{\mathtt{C}}(\hat{\boldsymbol{a}}_0, \ldots, \hat{\boldsymbol{a}}_{k-1})\big)(\alpha)$$

Fig. 11: PIOP for arithmetic constraints

Proof. We refer to Theorem 4 in [52]. □

Theorem 15. *Let $\hat{\boldsymbol{a}} \leftarrow \mathtt{REcd}(\vec{a})$, where $\|\vec{a}\|_\infty \le B$. Then, an interactive protocol Π_{L^∞} described in Fig. 12 is an HVZK PIOP with a soundness error of $\frac{O(k+\ell+N)}{p-N}$.*

Proof. We refer to Theorem 5 in [52]. □

$$\Pi_{L^\infty}(B; [\![\hat{\boldsymbol{a}}_0]\!], \ldots, [\![\hat{\boldsymbol{a}}_{k-1}]\!])$$

Public input: norm bound B.
Witness: vectors $\vec{a}_i = \mathtt{Dcd}(\hat{\boldsymbol{a}}_i)$, where $\hat{\boldsymbol{a}}_i \in \mathbb{Z}_p^{<2N}[X]$ for $0 \leq i < k$.
Statement: $\|\vec{a}_i\|_\infty \leq B$ for $0 \leq i < k$.

1. The prover $\mathtt{P}$ and the verifier $\mathtt{V}$ decompose B into $B_0 = \lceil \frac{B}{2} \rceil$, $B_1 = \lceil \frac{B-B_0}{2} \rceil$, $B_2 = \lceil \frac{B-B_0-B_1}{2} \rceil, \ldots, B_{\ell-1} = 1$, where $\ell = \lfloor \log B \rfloor + 1$.
2. The prover $\mathtt{P}$ decompose $\vec{a}_i$ into $\vec{a}_{i,0}, \ldots, \vec{a}_{i,\ell-1} \in R_p$ such that $\vec{a}_i = \sum_{j=0}^{\ell-1} B_i \cdot \vec{a}_{i,j}$ and $\|\vec{a}_{i,j}\|_\infty \leq 1$, and samples polynomial encodings $\hat{\boldsymbol{a}}_{i,j} \leftarrow \mathtt{REcd}(\vec{a}_{i,j})$. Then, $\mathtt{P}$ sends polynomial oracles $([\![\hat{\boldsymbol{a}}_{i,j}]\!], 2N-1)$ to $\mathtt{V}$ for $0 \leq i < k$ and $0 \leq j < \ell$.
3. The verifier $\mathtt{V}$ sends random points $\beta, \gamma \leftarrow \mathcal{U}(\mathbb{Z}_p \setminus \mathbb{H})$.
4. The prover $\mathtt{P}$ and the verifier $\mathtt{V}$ invoke the following PIOPs.

$$\Pi_{\mathtt{AC}}\Bigg(\sum_{i=0}^{k-1} \gamma^i \cdot \Big(\vec{X}_{i(\ell+1)} - \sum_{j=0}^{\ell-1} B_j \cdot \vec{X}_{i(\ell+1)+j+1} + \sum_{j=0}^{\ell-1} \beta^{j+1} \cdot (\vec{X}^3_{i(\ell+1)+j+1} - \vec{X}_{i(\ell+1)+j+1})\Big);$$
$$[\![\boldsymbol{a}_0]\!], [\![\boldsymbol{a}_{0,0}]\!], \ldots, [\![\boldsymbol{a}_{0,\ell-1}]\!], \ldots, [\![\boldsymbol{a}_{k-1}]\!], [\![\boldsymbol{a}_{k-1,0}]\!], \ldots, [\![\boldsymbol{a}_{k-1,\ell-1}]\!]\Bigg)$$

Fig. 12: PIOP for L^∞-norm constraints

D Deferred Proofs

Proof of Theorem 11

Proof. We show how the norm bound changes for each type of public key and ciphertext after modulus switching.

For the relinearzation key, it holds that $\mathsf{rlk} = \left(\left\lfloor \frac{Q_L}{p} \cdot \mathsf{rlk}'_0 \right\rceil, \left\lfloor \frac{Q_L}{p} \cdot \mathsf{rlk}'_1 \right\rceil\right)$. Then, we have the followings for rounding errors $\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0}, \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1}, \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},2} \in \mathbb{R}[X]/(X^N+1)$.

$$\begin{aligned}
\left\lfloor \frac{Q_L}{p} \cdot \mathsf{rlk}'_0 \right\rceil + \left\lfloor \frac{Q_L}{p} \cdot \mathsf{rlk}'_1 \right\rceil \cdot \boldsymbol{s} &= \left(\frac{Q_L}{p} \cdot \mathsf{rlk}'_0 + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0}\right) + \left(\frac{Q_L}{p} \cdot \mathsf{rlk}'_1 + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1}\right) \cdot \boldsymbol{s} \\
&= \frac{Q_L}{p} \cdot (\boldsymbol{s}^2 \cdot \lfloor p/Q_L \cdot \vec{g} \rceil + \vec{\boldsymbol{e}}'_{\mathsf{rlk}}) + (\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1} \cdot \boldsymbol{s}) \\
&= \frac{Q_L}{p} \cdot \left(\boldsymbol{s}^2 \cdot (p/Q_L \cdot \vec{g} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},2}) + \vec{\boldsymbol{e}}'_{\mathsf{rlk}}\right) + (\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1} \cdot \boldsymbol{s}) \\
&= \boldsymbol{s}^2 \cdot \vec{g} + \frac{Q_L}{p} \cdot \vec{\boldsymbol{e}}'_{\mathsf{rlk}} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1} \cdot \boldsymbol{s} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1} \cdot \boldsymbol{s}^2 \pmod{Q_L}
\end{aligned}$$

Then, for $\vec{\boldsymbol{e}}_{\mathsf{rlk}} = \frac{Q_L}{p} \cdot \vec{\boldsymbol{e}}'_{\mathsf{rlk}} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1} \cdot \boldsymbol{s} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1} \cdot \boldsymbol{s}^2$, it holds that $\|\vec{\boldsymbol{e}}_{\mathsf{rlk}}\|_\infty \leq B' + \frac{h^2+h+1}{2}$ since $Q_L < p$, $\|\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0}\|_\infty, \|\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1}\|_\infty, \|\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},2}\|_\infty \leq 1/2$ and $\|\boldsymbol{s}\|_1 \leq h$.

For the rotation key, it holds that $\mathsf{rtk} = \left(\left\lfloor \frac{Q_L}{p} \cdot \mathsf{rtk}'_0 \right\rceil, \left\lfloor \frac{Q_L}{p} \cdot \mathsf{rtk}'_1 \right\rceil\right)$. Then, we have the followings for rounding errors $\vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},0}, \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1}, \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},2} \in \mathbb{R}[X]/(X^N + 1)$.

$$\begin{aligned}
\left\lfloor \frac{Q_L}{p} \cdot \mathsf{rtk}'_0 \right\rceil + \left\lfloor \frac{Q_L}{p} \cdot \mathsf{rtk}'_1 \right\rceil \cdot \boldsymbol{s} &= \left(\frac{Q_L}{p} \cdot \mathsf{rtk}'_0 + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},0}\right) + \left(\frac{Q_L}{p} \cdot \mathsf{rtk}'_1 + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1}\right) \cdot \boldsymbol{s} \\
&= \frac{Q_L}{p} \cdot (\varphi(\boldsymbol{s}) \cdot \lfloor p/Q_L \cdot \vec{g} \rceil + \vec{\boldsymbol{e}}'_{\mathsf{rtk}}) + (\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1} \cdot \boldsymbol{s}) \\
&= \frac{Q_L}{p} \cdot \Big(\varphi(\boldsymbol{s}) \cdot (p/Q_L \cdot \vec{g} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},2}) + \vec{\boldsymbol{e}}'_{\mathsf{rtk}}\Big) + (\vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1} \cdot \boldsymbol{s}) \\
&= \boldsymbol{s}^2 \cdot \vec{g} + \frac{Q_L}{p} \cdot \vec{\boldsymbol{e}}'_{\mathsf{rtk}} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1} \cdot \boldsymbol{s} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1} \cdot \varphi(\boldsymbol{s}) \pmod{Q_L}
\end{aligned}$$

Then, for $\vec{\boldsymbol{e}}_{\mathsf{rtk}} = \frac{Q_L}{p} \cdot \vec{\boldsymbol{e}}'_{\mathsf{rlk}} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},0} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1} \cdot \boldsymbol{s} + \vec{\boldsymbol{\varepsilon}}_{\mathsf{rtk},1} \cdot \varphi s$, it holds that $\|\vec{\boldsymbol{e}}_{\mathsf{rtk}}\|_\infty \leq B' + \frac{2h+1}{2}$ since $Q_L < p$, $\|\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},0}\|_\infty, \|\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},1}\|_\infty, \|\vec{\boldsymbol{\varepsilon}}_{\mathsf{rlk},2}\|_\infty \leq 1/2$, and $\|\varphi(\boldsymbol{s})\|_1 = \|\boldsymbol{s}\|_1 \leq h$. We note that for the conjugation key, the overall process is identical to the rotation key case since $\|\psi(\boldsymbol{s})\|_1 = \|\boldsymbol{s}\|_1$ holds, so we omit the details.

For the ciphertexts, we first analyze $\left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_i \right\rceil = \left(\left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_{i,0} \right\rceil, \left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_{i,1} \right\rceil\right)$. Then, we have the following for rounding errors $\boldsymbol{\varepsilon}_{i,0}, \boldsymbol{\varepsilon}_{i,1}, \boldsymbol{\varepsilon}_{i,2} \in \mathbb{R}[X]/(X^N + 1)$, where we represent $\boldsymbol{m}'_i = \frac{p}{Q_L}\boldsymbol{m}_i + \boldsymbol{\varepsilon}_{i,2}$ for $\|\boldsymbol{m}_i\|_\infty \leq \varDelta$.

$$\begin{aligned}
\left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_{i,0} \right\rceil + \left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_{i,1} \right\rceil \cdot \boldsymbol{s} &= \left(\frac{Q_L}{p} \cdot \mathsf{ct}'_{i,0} + \boldsymbol{\varepsilon}_{i,0}\right) + \left(\frac{Q_L}{p} \cdot \mathsf{ct}'_{i,1} + \boldsymbol{\varepsilon}_{i,1}\right) \cdot \boldsymbol{s} \\
&= \frac{Q_L}{p} \cdot (\boldsymbol{m}'_i + \boldsymbol{e}'_i) + (\boldsymbol{\varepsilon}_{i,0} + \boldsymbol{\varepsilon}_{i,1} \cdot \boldsymbol{s}) \\
&= \boldsymbol{m}_i + \frac{Q_L}{p} \cdot \boldsymbol{\varepsilon}_{i,2} + \frac{Q_L}{p} \cdot \boldsymbol{e}'_i + (\boldsymbol{\varepsilon}_{i,0} + \boldsymbol{\varepsilon}_{i,1} \cdot \boldsymbol{s}) \pmod{Q_L}
\end{aligned}$$

Therefore, $\left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_i \right\rceil \in R^2_{Q_L}$ is a ciphertext, which encrypts $\boldsymbol{m}_i$ with an error bound $\leq B' + \frac{h+3}{2}$. Now, consider $(\mathsf{ct}_{2i}, \mathsf{ct}_{2i+1}) \leftarrow \mathtt{CtoS}_{\mathsf{pk}}\big(\left\lfloor \frac{Q_L}{p} \cdot \mathsf{ct}'_i \right\rceil\big)$ for $0 \leq i < k/2$. Then, it holds the following.

$$\begin{aligned}
\mathsf{ct}_{2i,0} + \mathsf{ct}_{2i,1} \cdot \boldsymbol{s} &= \widetilde{\boldsymbol{m}}_{2i} + \widetilde{\boldsymbol{e}}_{2i} \pmod{Q_L} \\
\mathsf{ct}_{2i+1,0} + \mathsf{ct}_{2i+1,1} \cdot \boldsymbol{s} &= \widetilde{\boldsymbol{m}}_{2i+1} + \widetilde{\boldsymbol{e}}_{2i+1} \pmod{Q_L}
\end{aligned}$$

where $\iota(\widetilde{\boldsymbol{m}}_{2i}) = (m_{i,0}, \ldots, m_{i,N/2-1})$ and $\iota(\widetilde{\boldsymbol{m}}_{2i+1}) = (m_{i,N/2}, \ldots, m_{i,N-1})$ for $(m_{i,0}, \ldots, m_{i,N-1}) = \mathtt{Coeff}(\boldsymbol{m}_i)$, and $\|\widetilde{\boldsymbol{e}}_{2i}\|_\infty, \|\widetilde{\boldsymbol{e}}_{2i+1}\|_\infty \leq B_e$. Therefore, we have $\widetilde{\boldsymbol{m}}_{2i}, \widetilde{\boldsymbol{m}}_{2i+1} \in \mathcal{X}$ since $\|\mathtt{Unpack}(\widetilde{\boldsymbol{m}}_{2i})\|_\infty, \|\mathtt{Unpack}(\widetilde{\boldsymbol{m}}_{2i+1})\|_\infty \leq \varDelta^{-1} \cdot \|\boldsymbol{m}\|_\infty = 1$, and $\mathtt{Unpack}(\cdot) = \varDelta^{-1} \cdot \iota(\cdot)$.