(Fine-Grained) Unbounded Inner-Product Functional Encryption from LWE

Valerio Cini[1] and Erkan Tairi[2⋆]

[1] Bocconi University
valerio.cini@unibocconi.it

[2] UC Berkeley
erkan.tairi@berkeley.edu

Abstract. Inner-product functional encryption (IPFE), introduced by Abdalla-Bourse-De Caro-Pointcheval (PKC'15), is a public-key primitive that allows to decrypt an encrypted vector $\mathbf{x}$ with a secret key associated to a vector $\mathbf{y}$ such that only their inner-product $\langle \mathbf{x}, \mathbf{y} \rangle$ is revealed. The initial definition and constructions all required the length of such vectors to be *bounded* at setup, and therefore, be fixed in the public parameters.
In order to overcome this drawback, Dufour-Sans-Pointcheval (ACNS'19) and Tomida-Takashima (AC'18) introduced the notion of *unbounded* IPFE, where the length of vectors does not need to be fixed during the setup phase, and gave constructions from pairing-based assumptions.
In this paper, we make progress and provide the *first* unbounded IPFE constructions that i) are based on the Learning With Errors (LWE) assumption and proven secure in the standard model, ii) achieve adaptive security, iii) provide fine-grained access control, i.e., are identity- and attribute-based, and iv) rely only on black-box access to cryptographic and lattice algorithms. Hence, our constructions are also plausibly post-quantum secure.

1 Introduction

Functional encryption (FE) [BSW11, O'N10] is an advanced encryption primitive that allows fine-grained access control over the encrypted data. In contrast to conventional encryption schemes, which are all-or-nothing, in (public-key) FE there is a master secret key msk that allows to generate constrained functional secret keys, and a master public key mpk that is used for encrypting messages. More precisely, every functional secret key sk_f is associated with a function f, and given a ciphertext ct_x of some message x (under the master public key mpk), the decryption with sk_f only reveals $f(x)$, and nothing more about x.

Since its introduction, FE has been subject to intense study which resulted in FE schemes for general functionalities [GVW12, GKP+13, GGH+13, Wat15, AV19]. However, these schemes are either not collusion-resistant (i.e., they only allow the adversary to learn bounded number of functional secret keys) or require heavy-machinery, such as indistinguishability obfuscation (iO) [GGH+13, JLS21, JLS22]. On the positive side, we have practical FE schemes for limited classes of functions, e.g., (function-hiding) inner-product FE (IPFE) [ABDP15, BJK15, ALS16, Lin17, Tom19, ALMT20], compact FE for quadratic functions (QFE) [BCFG17, Lin17, AS17, Gay20, Tom23], and FE for attribute-weighted sums (AWS) [AGW20, DP21, DPT22]. Furthermore, these classes of FE have been extended to multi-input [AGRW17, ACF+18, AGT21a, AGT22, ATY23], (decentralized) multi-client [CDG+18, ABKW19, LT19, ABG19, AGT21b], and identity/attribute-based [ACGU20, LLW21, PD21, CRS+24] settings.

Inner-Product Functional Encryption. A particularly important class from the aforementioned is IPFE, which was initially introduced by Abdalla et al. [ABDP15]. In IPFE, given a functional secret key $\mathsf{sk}_{\mathbf{y}}$ for a vector $\mathbf{y}$ and a ciphertext $\mathsf{ct}_{\mathbf{x}}$ of a vector $\mathbf{x}$, decryption reveals only the inner product $\langle \mathbf{x}, \mathbf{y} \rangle$. IPFE is a simple functionality, yet it enables numerous applications, such as privacy-preserving machine learning [MSH+19], money-laundering detection [dPP22], search in encrypted data streams [BCS23], video data analytics[3] and

⋆ Work done while the author was at ENS Paris.

[3] https://research.kudelskisecurity.com/2021/02/02/benchmarking-privacy-preserving-motion-detection

data marketplace [KPC+20]. Furthermore, it even allows evaluation of constant-degree polynomials via relinearization.

However, practical applications in general require the capability to encrypt arbitrarily long vectors. Nevertheless, nearly all of the existing IPFE construction require the data being processed to have a fixed length [ABDP15, BJK15, ALS16, Lin17, Tom19, ALMT20]. An exception to these is the *unbounded IPFE (UIPFE)*, concurrently and independently introduced by Tomida and Takashima [TT18] and Dufour-Sans and Pointcheval [DP19]. Unlike in regular IPFE, in UIPFE we do not fix the length of the vectors at the setup time, and instead allow computing variable length inner products. Though, both [TT18] and [DP19] are based on pairings, and hence, they are not post-quantum secure. Prior to these works, Goyal, Koppula and Waters [GKW16] presented a generic transformation for "bundling" a family of functionalities under one set of public parameters, which in turn can be used to construct an unbounded IPFE. Instantiating their construction with the LWE-based (bounded) IPFE from Agrawal, Libert, and Stehlé [ALS16] yields a plausibly post-quantum UIPFE construction. However, this transformation suffers from various drawbacks: it incurs a high computation overhead as it requires non-black-box use of cryptographic primitives (in particular, it requires to evaluate a garbled circuit that depends on the bitlength of the master public key of the underlying bounded IPFE), it only supports a strict form of UIPFE functionality, and achieves semi-adaptive security.[4]

Given this state of affairs[5], we ask the following question:

Can we construct black-box and adaptively secure lattice-based unbounded IPFE in the standard model?

In the literature, the term unbounded IPFE typically refers to the so called *strict* variant [DP19], meaning that functional keys only work correctly on ciphertexts encrypting messages whose length matches that of the functional key.

However, many real-world datasets not only do not have entries with a fixed length but the length of entries might change over time as the dataset evolves. For example, suppose you have a growing list of features describing transactions, images, or sensor readings over which authorized users want to perform statistical analysis. Under a strict UIPFE scheme, a user who obtains a functional key for a certain dimension cannot decrypt inner-products from ciphertexts encoding a longer message, because the domains do not match exactly. As a result, one must either regenerate ciphertexts or reissue functional keys each time new features are added, creating significant overhead.

By contrast, the so-called *permissive* variant [TT18, DP19] of UIPFE enables functional keys for smaller domains to decrypt inner-products from ciphertexts covering larger domains, that is as long as the smaller domain is contained within the larger one.This design eliminates the need for new keys whenever additional data features appear, making it especially suited for real-time and continuously evolving datasets.

Taking this into account, we pose the following additional question:

Can we extend a strict UIPFE construction to a permissive one?

An additional feature that is useful for practical applications is to provide some form of fine-grained access control to the function evaluations, such as what is provided by *identity-based* or *attribute-based* IPFE [ACGU20]. Nevertheless, from the existing UIPFE constructions, only [DP19] provided identity-based UIPFE (IB-UIPFE) constructions, and neither [DP19] nor [TT18] achieve attribute-based UIPFE (AB-UIPFE). Only recently, Datta and Pal [DP23] provided the first AB-UIPFE construction by extending the work of [DP19]. Given that all these prior works are pairing-based, and hence, are not post-quantum secure, we ask the following:

Can we construct lattice-based fine-grained UIPFE?

[4] Though, while this was not discussed in [GKW16], by starting with an adaptively secure bounded IPFE one can potentially get an adaptively secure unbounded IPFE.

[5] There are lower bounds for lattice-based (bounded input) function-hiding IPFE [Üna20] and compact QFE [TÜ24], and hence, we do not focus on these directions in this work.

1.1 Our Contributions

This work answers all the above questions affirmatively. More concretely, we construct three adaptively secure and one selectively secure lattice-based unbounded IPFE schemes, all in the standard model. Each construction is based on the (plain) LWE assumption and relies only on black-box access[6] to cryptographic and lattice algorithms.

1. **Strict UIPFE:** Our first scheme is a *strict* UIPFE (Section 3) that allows decryption only when the domain of the ciphertext matches that of the functional secret key. The size of the functional secret keys and ciphertexts are linear (i.e., linear in the inputs vectors), whereas the size of the master public key and master secret key are constant.
2. **Permissive UIPFE:** Our second scheme is a *permissive* UIPFE (Section 4) that allows decryption when the support of the functional secret key is included in the domain of the ciphertext. It has the same efficiency as the strict construction, with the exception that the ciphertexts are of quadratic size (i.e., quadratic in the input vector).
3. **Identity-based UIPFE:** Our third scheme is an *identity-based* UIPFE (Section 5) that associates identities to functional secret keys and ciphertexts, such that the decryption only reveals the functional evaluation if these identities match. Our IB-UIPFE can be instantiated in both strict and permissive settings and inherits the efficiency guarantees of the underlying (plain) UIPFE scheme.
4. **Attribute-based UIPFE:** Our fourth scheme is an *attribute-based* UIPFE (Section 6) that supports access policies realizable by bounded-depth circuits and can be instantiated in both strict and permissive settings. Unlike the previous constructions, it achieves selective and $(1, \mathsf{poly})$-security[7] (but can be extended to (Q, poly)-security as described in Remark 3), with bounded 1-key restriction being inherited from the underlying bounded AB-IPFE scheme given in [LLW21].

We refer the reader to Section 2.3 for a formal definition of strict and permissive (fine-grained) UIPFE. We summarize the features of our constructions and compare them with previous constructions of unbounded IPFE in Table 1.

1.2 Related Work

Tomida and Takashima [TT18] and Dufour-Sans and Pointcheval [DP19] concurrently and independently proposed the first UIPFE constructions. Tomida and Takashima [TT18] presented two constructions, where the first one is in a secret-key setting and achieves full function-hiding security and the second one is in a public-key setting and achieves adaptive indistinguishability security. Both of these constructions are proven secure under the SXDH assumption. On the other hand, Dufour-Sans and Pointcheval [DP19] presented a public-key construction with succinct functional secret keys, which is selectively secure under the DBDH assumption in the random oracle model (ROM). They also lifted their construction to the identity-based setting in order to provide access control. Datta, Pal and Takashima [DPT22] provided an FE for AWS that supports Logspace Turing machines, hence, obtaining an unbounded FE for AWS. Their construction is proven adaptively simulation secure under the SXDH assumption. Viewing IPFE as a special case of FE for AWS[10], their construction also achieves an adaptively simulation secure UIPFE under the SXDH assumption.

[6] Following [RTV04], by "black-box access" we mean using a cryptographic primitive or an algorithm only via its input/output interface (i.e., its oracle behavior), without relying on its internal code or structure.

[7] Recall that (Q, poly)-security means that the adversary is allowed to query Q many 1-keys and arbitrary polynomial number of 0-keys. We call a secret key $\mathsf{sk}_{C,\mathbf{y}}$ a 1-key with respect to a ciphertext $\mathsf{ct}_{\boldsymbol{\alpha},\mathbf{x}}$, if $C(\boldsymbol{\alpha}) = 0$, and otherwise, we call it a 0-key.

[9] More concretely, the construction of [GKW16] requires an identity-based encryption scheme, a garbled circuit scheme and a PRF to turn a bounded IPFE scheme into an unbounded one.

[10] FE for AWS implies AB-IPFE, however, we note that only *unbounded* FE for AWS gives unbounded (AB-)IPFE, and the only generic way we know to go from any bounded FE to unbounded FE is to use the previously mentioned non-black-box transformation from [GKW16].

Scheme	Functionality	IB	AB[†]	Security	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ct}\rvert$	Assumption	SM	BB
[DP19]	strict / permissive	✓	✗	selective	$O(1)$	$O(\ell)$	DBDH	✗	✓
[DP23]	strict / permissive	✓	✓	selective	$O(\ell')$	$O(\ell)$	DBDH	✗	✓
[TT18]	strict / permissive	✗	✗	adaptive	$O(\ell')$	$O(\ell)$	SXDH	✓	✓
[GKW16][*]	strict	✓	✓	semi-adaptive	$O(\lvert\mathsf{mpk}\rvert)$	$O(\lvert\mathsf{mpk}\rvert)$	IPFE + OWF	✓	✗
Section 3	strict	✗	✗	adaptive	$O(\ell')$	$O(\ell)$	LWE	✓	✓
Section 4	permissive	✗	✗	adaptive	$O(\ell')$	$O(\ell^2)$	LWE	✓	✓
Section 5	strict / permissive	✓	✗	adaptive	$O(\ell')$	$O(\ell)$ / $O(\ell^2)$	LWE	✓	✓
Section 6	strict / permissive	✓	✓	selective	$O(\ell')$	$O(\ell)$ / $O(\ell^2)$	LWE	✓	✓

Table 1. Comparison of the existing unbounded IPFE constructions, where ℓ and ℓ' denote the length of the vectors input to the encryption and key generation algorithms respectively, and we ignore factors polynomial in the security parameter λ. IB, AB, SM and BB denote identity-based, attribute-based, standard model, and black-box, respectively. All these schemes (including ours) have constant size mpk and msk (ignoring $\mathsf{poly}(\lambda)$ factors), hence we ignore them here.

[*][GKW16] is a generic construction that can be instantiated in a post-quantum regime by combining the LWE-based (bounded) IPFE from [ALS16] or (bounded) IB/AB-IPFE from [LLW21] with a lattice-based OWF.[9] The secret key and ciphertext sizes of the resulting construction are linear in the bitlength of the master public key of the underlying bounded IPFE. For example, by using the parameters of the LWE-based bounded IPFE construction over the integers given in [ALS16, Section 4.1], we would have $|\mathsf{mpk}| = mn \log q + \ell n \log q$, for some $n \geq \widetilde{\Omega}(\lambda \log K)$, $m = \Theta(n \log q)$ and $q = \Omega(\sqrt{n}/\alpha')$, where $\alpha' = 1/((n \log q)^{O(1)} \cdot K^2)$ and K denotes an upper bound on the computable inner product values.

[†]We remark that any AB-UIPFE scheme is by default also an IB-UIPFE. We also note that the AB-UIPFE construction of [DP23] can only support access policies realizable by linear secret sharing schemes (LSSS), whereas our construction from Section 6 can support family $\mathcal{C}$ of circuits with depth-d.

Tomida [Tom23] gave the first unbounded QFE construction, which achieves semi-adaptive indistinguishability security under the MDDH assumption in the ROM. The same work also provided attribute-based access control for the unbounded QFE via arithmetic branching programs (ABPs). We note that ABPs are a type of non-uniform model of computation, hence, the length of the attributes in [Tom23] is bounded. Recently, Dowerah et al. [DDM+23] constructed unbounded predicate IPFE, where both length of the attributes and message vectors are unbounded, and additionally, the attributes are hidden. They provided constructions for both the zero predicate and non-zero predicate[11]. For the zero predicate case, their construction achieves full attribute-hiding security in the indistinguishability-based semi-adaptive model under the SXDH assumption. For the non-zero predicate case, they provided two constructions that only achieve weak attribute-hiding: the first is in the public-key setting and achieves indistinguishability-based semi-adaptive security, whereas the second construction is in the secret-key setting and offers succinct functional secret keys, and it achieves simulation security under the bilateral k-Lin assumption in the ROM. In another recent work, Datta and Pal [DP23] constructed (multi-authority) attribute-based UIPFE by extending the techniques from [DP19]. Their construction is proven selectively secure under the DBDH assumption in the ROM.

We remark that all of the these works consider both the strict and permissive cases for the unbounded inputs, however, all of them are in the pairing-based setting, and consequently, they are not post-quantum se-

[11] In a predicate IPFE scheme of [DDM+23], a message $\mathbf{x}$ is encrypted under an attribute $\mathbf{w}$ and functional secret key is generated for a pair $(\mathbf{y}, \mathbf{v})$. Then, for the *zero predicate* case, the inner product $\langle \mathbf{x}, \mathbf{y} \rangle$ is recovered if $\langle \mathbf{w}, \mathbf{v} \rangle = 0$, whereas for the *non-zero predicate* case $\langle \mathbf{x}, \mathbf{y} \rangle$ is recovered if $\langle \mathbf{w}, \mathbf{v} \rangle \neq 0$.

cure. In the lattice-based setting, all of the existing IPFE constructions [ALS16, ACGU20, LLW21, MKMS22, CRS+24, ET25] support only bounded length vectors.[12]

Apart from the previously mentioned works on unbounded FE for restricted classes of functions, there exist also FE constructions for uniform models of computation, such as Turing machines or RAM programs, which also allow for unbounded inputs. However, the existing works on FE for Turing machines [KNTY19, AMVY21, AKM+22] and RAM programs [ACFQ22, JLL23] either handle only bounded number of collusions or require collusion-resistant (public-key) FE for circuits, which in turn can be based on iO constructions [JLS21, JLS22], rendering them quite inefficient and not post-quantum secure.

1.3 Open Problems

As we describe in Section 1.4, extending our strict UIPFE to permissive setting is highly non-trivial, and we resorted to a "semi-generic" approach that incurs quadratic blowup in ciphertext size. Hence, we leave it as an interesting open problem to improve the efficiency of our permissive construction given in Section 4, i.e., reducing the ciphertext size from quadratic to linear.

Additionally, in this work we follow the UIPFE model given in [DP19], which defines only *strict* and *permissive* settings. Nevertheless, we note that for some practical applications it might be plausible to consider the opposite of the permissive setting, i.e., where keys are for the larger domain and ciphertexts cover the smaller domain, as in [TT18]. We leave it as an open problem to construct lattice-based UIPFE in this setting.

1.4 Technical Overview

We proceed to provide a technical overview of our constructions. We start by describing our construction of a strict unbounded IPFE and its security proof, then discuss how to modify it to obtain a permissive UIPFE, and finally address the fine-grained UIPFE constructions.

ALS16 IPFE. Let us start by recalling the $\mathsf{ALS}[\ell]$ construction [ALS16], where ℓ denotes the length of messages, as slightly simplified in subsequent works [WFL19, ACGU20] (and also fully specified in Appendix A). In what follows, we omit error terms in the ciphertext for ease of exposition. For length ℓ, message and keys space $\{0,\ldots,V-1\}^\ell$ and $\{0,\ldots,P-1\}^\ell$, respectively, the scheme is as follows, where $K_\ell \coloneqq \ell PV$:

- $\mathsf{Setup}(1^\lambda, 1^\ell, P, V)$: Sample

$$\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m},$$
$$\mathbf{T} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^{m\times\ell},\tau}.$$

  Set $\mathbf{D} = \mathbf{A}\cdot\mathbf{T} \in \mathbb{Z}_q^{n\times\ell}$, $\mathsf{mpk} = (\mathbf{A},\mathbf{D})$, and $\mathsf{msk} = \mathbf{T}$, and return $(\mathsf{mpk},\mathsf{msk})$.
- $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}\in\{0,\ldots,V-1\}^\ell)$:

$$\mathbf{s}\leftarrow\mathbb{Z}_q^n, \mathbf{e}_0\leftarrow\mathcal{D}_{\mathbb{Z}^m,\chi}, \mathbf{e}_1\leftarrow\mathcal{D}_{\mathbb{Z}^\ell,\chi}.$$

  Compute

$$\mathbf{c}_0 \coloneqq \mathbf{s}\cdot\mathbf{A}+\mathbf{e}_0 \bmod q,$$
$$\mathbf{c}_1 \coloneqq \mathbf{s}\cdot\mathbf{D}+\mathbf{e}_1+\left\lfloor\frac{q}{K}\right\rfloor\cdot\mathbf{x} \bmod q.$$

  Output $\mathsf{ct}_\mathbf{x} \coloneqq (\mathbf{c}_0,\mathbf{c}_1)$.

[12] We note that the ROM-based constructions given in [ACGU20, ET25] could be modified to support vectors of unbounded length by exploiting the random oracle. In fact, a lattice-based UIPFE in the ROM was recently given in a concurrent and independent work by Roy and Dutta [RD25].

– $\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{y} \in \{0, \ldots, P-1\}^\ell)$: Let

$$\mathbf{t}_\mathbf{y}^\top = \mathbf{T} \cdot \mathbf{y}^\top$$

Output $\mathsf{sk}_\mathbf{y} \coloneqq (\mathbf{t}_\mathbf{y}, \mathbf{y})$.

– $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_\mathbf{y}, \mathsf{ct}_\mathbf{x})$: Parse $\mathsf{ct}_\mathbf{x} = (\mathbf{c}_0, \mathbf{c}_1)$, $\mathsf{sk}_\mathbf{y} = (\mathbf{t}_\mathbf{y}, \mathbf{y})$. Let

$$\mu = \mathsf{ct}_1 \cdot \mathbf{y}^\top - \mathsf{ct}_0 \cdot \mathbf{t}_\mathbf{y}^\top,$$

and output

$$\mu' = \underset{\mu' \in \{-K_\ell+1, \ldots, K-1\}}{\arg\min} \left| \mu - \left\lfloor \frac{q}{K} \right\rfloor \cdot \mu' \right|.$$

To modify this construction and make it an unbounded IPFE scheme, one needs to ensure that the output of the Setup algorithm is independent of the length ℓ.

Starting from [ALS16], [WFL19] and [ACGU20] showed that the parameters can be chosen so that only dependencies on ℓ, apart from the parameter ℓ itself, come from the modulus q and Gaussian parameter χ, which both have a polynomial dependency on ℓ (see Appendix A for concrete parameters), and the scheme is correct and can be proven secure under $\mathsf{LWE}_{n,m,q,\alpha}$.vcnoteremove alpha here. Check again if this paragraph reads clearly This means that, by setting the modulus q and the Gaussian parameter χ to be superpolynomial in the security parameter λ, all parameters, except clearly ℓ, can be chosen to be independent of the length ℓ of messages and keys.

Therefore, the remaining dependencies of Setup's output on the length ℓ come from the master public key, $\mathbf{D} \in \mathbb{Z}_q^{n \times \ell}$, and the master secret key, $\mathbf{T} \in \mathbb{Z}^{m \times \ell}$. We will describe now how we address both of these issues.

Master Secret Key. To overcome this first source of dependency, we modify the Setup algorithm and let it sample $\mathbf{A}$ together with an associate trapdoor $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m \times m}$, and sample $\mathbf{D} \in \mathbb{Z}_q^{n \times \ell}$ uniformly at random. Using the trapdoor $\mathbf{T}_\mathbf{A}$, which will be part of the new master secret key, it is now possible to sample a Gaussian preimage $\mathbf{T}$ of $\mathbf{D}$ under $\mathbf{A}$, i.e, $\mathbf{A} \cdot \mathbf{T} = \mathbf{D} \bmod q$. Using such $\mathbf{T}$, the functional secret key associated to vector $\mathbf{y} \in \{0, \ldots, P-1\}^\ell$ is produced as before as $\mathbf{k}_\mathbf{y}^\top = \mathbf{T} \cdot \mathbf{y}^\top$. Notice that in the ALS scheme the functional secret key is deterministically generated. In order to guarantee the same behavior, which is also crucial for the security of the construction, we derandomize the sampling of $\mathbf{T}$ using a pseudorandom function (PRF) F. The PRF key is sampled during setup and added to the master secret key. To summarize, Setup and KeyGen are modified as follows[13]:

– $\mathsf{Setup}(1^\lambda, 1^\ell, P, V)$: Sample

$$(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{D} \leftarrow\!\$\ \mathbb{Z}_q^{n \times \ell},$$
$$\mathsf{k} \leftarrow\!\$\ \{0,1\}^\lambda.$$

Set $\mathsf{mpk} = (\mathbf{A}, \mathbf{D})$, and $\mathsf{msk} = (\mathbf{T}_\mathbf{A}, \mathsf{k})$, and output $(\mathsf{mpk}, \mathsf{msk})$.

– $\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{y} \in \{0, \ldots, P-1\}^\ell)$: Sample

$$\mathbf{T} \leftarrow \mathsf{SamplePre}\left(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{D}, \tau; \mathsf{F}(\mathsf{k}, 1)\right).$$

Set $\mathbf{t}_\mathbf{y}^\top = \mathbf{T} \cdot \mathbf{y}^\top$, and output $\mathsf{sk}_\mathbf{y} \coloneqq (\mathbf{t}_\mathbf{y}, \mathbf{y})$.

Master Public Key. The only constraint left on ℓ in the Setup is now the matrix $\mathbf{D}$, which is part of the master public key mpk. In order to solve this second source of dependency we rely on recent techniques introduced by Cini and Wee in the context of attribute-based encryption [CW24], that allow to "delay the sampling of $\mathbf{D}$" to key generation. Specifically, we further modify the scheme as follows:

[13] We refer the reader to Lemma 3 for definition of $\mathsf{TrapGen}$ and $\mathsf{SamplePre}$.

– mpk now contains random matrices $\mathbf{B}, \mathbf{W}, \mathbf{Y} \in \mathbb{Z}_q^{n \times m}$, and $\mathbf{v} \in \mathbb{Z}_q^n$, in addition to $\mathbf{A}$, and msk contains the trapdoors of $\mathbf{A}$ and $\mathbf{B}$, as well as the PRF key k;
– the ciphertext contains

$$\left\{\mathbf{s}_j \cdot \mathbf{v}^\top + x_j \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor, \quad \mathbf{s}_j \cdot \mathbf{B}, \quad \mathbf{s}_j \cdot (\mathbf{W} + j \cdot \mathbf{G}) - \mathbf{s} \cdot \mathbf{G}, \quad \mathbf{s}_j \cdot (\mathbf{Y} + \ell \cdot \mathbf{G})\right\}_{j \in [\ell]},$$

where $\mathbf{s}, \mathbf{s}_j \leftarrow\!\!\$\ \mathbb{Z}_q^n$ are sampled during encryption;
– during decryption, we compute

$$\mathbf{s} \cdot \mathbf{d}_j^\top + x_j \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor \approx \left(\mathbf{s}_j \cdot \mathbf{v}^\top + x_j \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor\right) - \Big(\mathbf{s}_j \cdot (\mathbf{W} + j \cdot \mathbf{G}) - \mathbf{s} \cdot \mathbf{G}\Big) \cdot \mathbf{G}^{-1}(\mathbf{d}_j^\top) - \Big(\mathbf{s}_j \cdot (\mathbf{Y} + \ell \cdot \mathbf{G})\Big) \cdot \mathbf{p}_j^\top - \mathbf{s}_j \cdot \mathbf{B} \cdot \mathbf{z}_j^\top$$

where $\mathbf{p}_j \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m, \tau}$, $\mathbf{d}_j \leftarrow\!\!\$\ \mathbb{Z}_q^n$, and $\mathbf{z}_j \leftarrow \mathbf{B}^{-1}\Big(\mathbf{v}^\top - (\mathbf{W} + j \cdot \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{d}_j^\top) - (\mathbf{Y} + \ell \cdot \mathbf{G}) \cdot \mathbf{p}_j^\top\Big)$ are provided in the secret key;
– key generation for $\mathbf{y} \in \{0, 1 \ldots, P-1\}^\ell$ return the same $\mathbf{p}_j, \mathbf{d}_j, \mathbf{z}_j$ across all secret keys (for vectors of the same length ℓ), generated again using the PRF key k in msk such that we do not need to maintain state across different key queries, as well as an ALS secret key for vector $\mathbf{y}$.

This yields the following UIPFE scheme:

$\mathsf{mpk} = \mathbf{A}, \mathbf{B}, \mathbf{W}, \mathbf{Y} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{v} \leftarrow \mathbb{Z}_q^n$.

$\mathsf{ct} = \Big(\overbrace{\mathbf{s} \cdot \mathbf{A}}^{\mathbf{c}_0}, \overbrace{\{\mathbf{s}_j \cdot \mathbf{v}^\top + x_j \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor}^{\mathbf{c}_{1,j}}, \overbrace{\mathbf{s}_j \cdot \mathbf{B}}^{\mathbf{c}_{2,j}}, \overbrace{\mathbf{s}_j \cdot (\mathbf{W} + j \cdot \mathbf{G}) - \mathbf{s} \cdot \mathbf{G}}^{\mathbf{c}_{3,j}}, \overbrace{\mathbf{s}_j \cdot (\mathbf{Y} + \ell \cdot \mathbf{G})}^{\mathbf{c}_{4,j}}\}_{j \in [\ell]}, \Big), \mathbf{s}, \mathbf{s}_j \leftarrow \mathbb{Z}_q^n$.

$\mathsf{sk} = \big(\{\mathbf{z}_j, \mathbf{r}_j, \mathbf{p}_j\}_{j \in [\ell]}, \mathbf{t}_\mathbf{y}\big)$, where $\mathbf{y}_j, \mathbf{r}_j, \mathbf{p}_j$ are fixed across all keys (for vectors of same length ℓ)

$$\begin{bmatrix} \mathbf{z}_j^\top \\ \mathbf{r}_j^\top \\ \mathbf{p}_j^\top \end{bmatrix} \leftarrow \mathcal{D}_{\mathbb{Z}^{3m \times m}, \tau} \text{ s.t. } \big[\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell \cdot \mathbf{G}\big] \cdot \begin{bmatrix} \mathbf{z}_j^\top \\ \mathbf{r}_j^\top \\ \mathbf{p}_j^\top \end{bmatrix} = \mathbf{v}^\top,$$

$\mathbf{d}_j^\top = \mathbf{G} \cdot \mathbf{r}_j^\top$

$\mathbf{t}_\mathbf{y}^\top = \mathbf{T} \cdot \mathbf{y}^\top$ where $\mathbf{T} \leftarrow \mathcal{D}_{\mathbb{Z}^{m \times \ell}, \tau}$ s.t. $\mathbf{A} \cdot \mathbf{T} = \mathbf{D} \bmod q$ for $\mathbf{D} = [\mathbf{d}_1^\top \mid \ldots \mid \mathbf{d}_\ell^\top]$.

Decryption first uses

$$\mathbf{c}_{1,j} - \mathbf{c}_{2,j} \cdot \mathbf{z}_j^\top - \mathbf{c}_{3,j} \cdot \mathbf{r}_j^\top - \mathbf{c}_{4,j}^\top \cdot \mathbf{p}_j^\top \approx \mathbf{s} \cdot \mathbf{d}_j^\top + x_j \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor \tag{1}$$

to recover, together with $\mathbf{c}_0$, an ALS ciphertext, and then proceeds as in ALS decryption. In the final construction there will actually be a different matrix $\mathbf{D}_\ell$ for each possible vector length ℓ (and this is where the ciphertext component $\mathbf{Y} + \ell \cdot \mathbf{G}$ will become important), i.e., the secret key for vector $\mathbf{y}$ of length ℓ will actually be

$$\big(\{\mathbf{z}_{j,\ell}, \mathbf{r}_{j,\ell}, \mathbf{p}_{j,\ell}\}_{j \in [\ell]}, \mathbf{t}_\mathbf{y}\big),$$

where $\mathbf{z}_{j,\ell}, \mathbf{r}_{j,\ell}, \mathbf{p}_{j,\ell}$ are fixed across all keys for length-ℓ vectors, and where

$$\begin{bmatrix} \mathbf{z}_{j,\ell}^\top \\ \mathbf{r}_{j,\ell}^\top \\ \mathbf{p}_{j,\ell}^\top \end{bmatrix} \leftarrow \mathcal{D}_{\mathbb{Z}^{3m}, \tau} \quad \text{s.t.} \quad \big[\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell \cdot \mathbf{G}\big] \cdot \begin{bmatrix} \mathbf{z}_{j,\ell}^\top \\ \mathbf{r}_{j,\ell}^\top \\ \mathbf{p}_{j,\ell}^\top \end{bmatrix} = \mathbf{v}^\top,$$

and $\mathbf{D}_\ell = [\mathbf{d}_{1,\ell}^\top \mid \ldots \mid \mathbf{d}_{\ell,\ell}^\top]$, for $\mathbf{d}_{j,\ell}^\top = \mathbf{G} \cdot \mathbf{r}_{j,\ell}^\top$, and $\mathbf{t}_\mathbf{y}^\top = \mathbf{T}_\ell \cdot \mathbf{y}^\top$, where $\mathbf{A} \cdot \mathbf{T}_\ell = \mathbf{D}_\ell \bmod q$.

Proof Overview. The proof proceeds in two steps:

Step 1. Let ℓ^* be the length of the challenge vectors. For $k = 1, 2, \ldots, \ell^*$, we rely on pseudorandomness of $\mathbf{s}_k \cdot [\mathbf{B} \mid \mathbf{W} + k \cdot \mathbf{G} \mid \mathbf{Y} + \ell^* \cdot \mathbf{G}]$ to replace $\mathbf{c}_{2,k}, \mathbf{c}_{3,k}, \mathbf{c}_{4,k}$ with random and rewrite $\mathbf{c}_{1,k}$ in terms of $\mathbf{s}_k \cdot \mathbf{v}^\top + x_k \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor$ using (1). Concretely,

- we program $\mathbf{W} \coloneqq \mathbf{B} \cdot \tilde{\mathbf{W}}_k - k \cdot \mathbf{G}$ and $\mathbf{Y} \coloneqq \mathbf{B} \cdot \tilde{\mathbf{Y}}_k - \ell^* \cdot \mathbf{G}$ for random low-norm $\tilde{\mathbf{W}}_k, \tilde{\mathbf{Y}}_k$;
- we sample random Gaussians $\mathbf{z}_{k,\ell^*}, \mathbf{r}_{k,\ell^*}, \mathbf{p}_{k,\ell^*}$ and program $\mathbf{v}$ accordingly, i.e.,

$$\begin{aligned}
\mathbf{v}^\top &\coloneqq [\mathbf{B} \mid \mathbf{W} + k \cdot \mathbf{G} \mid \mathbf{Y} + \ell^* \cdot \mathbf{G}] \cdot \begin{bmatrix} \mathbf{z}_{k,\ell^*}^\top \\ \mathbf{r}_{k,\ell^*}^\top \\ \mathbf{p}_{k,\ell^*}^\top \end{bmatrix} \\
&= [\mathbf{B} \mid \mathbf{B} \cdot \tilde{\mathbf{W}}_k \mid \mathbf{B} \cdot \tilde{\mathbf{Y}}_k] \cdot \begin{bmatrix} \mathbf{z}_{k,\ell^*}^\top \\ \mathbf{r}_{k,\ell^*}^\top \\ \mathbf{p}_{k,\ell^*}^\top \end{bmatrix} \\
&= \mathbf{B} \cdot \left(\mathbf{z}_{k,\ell^*}^\top + \tilde{\mathbf{W}}_k \cdot \mathbf{r}_{k,\ell^*}^\top + \tilde{\mathbf{Y}}_k \cdot \mathbf{p}_{k,\ell^*}^\top \right).
\end{aligned}$$

- for all $\ell \neq \ell^*, j \in [\ell]$ we sample $\mathbf{z}_{j,\ell}, \mathbf{r}_{j,\ell}, \mathbf{p}_{j,\ell}$ using the trapdoor for

$$[\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell \cdot \mathbf{G}] = [\mathbf{B} \mid \mathbf{B} \cdot \tilde{\mathbf{W}} + (j-k) \cdot \mathbf{G} \mid \mathbf{B} \cdot \tilde{\mathbf{Y}} + \underbrace{(\ell - \ell^*)}_{\neq 0} \cdot \mathbf{G}].$$

  Similarly, for $\ell = \ell^*, j \neq k$, we sample $\mathbf{z}_{j,\ell}, \mathbf{r}_{j,\ell}, \mathbf{p}_{j,\ell}$ using the trapdoor for

$$[\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell \cdot \mathbf{G}] = [\mathbf{B} \mid \mathbf{B} \cdot \tilde{\mathbf{W}} + \underbrace{(j-k)}_{\neq 0} \cdot \mathbf{G} \mid \mathbf{B} \cdot \tilde{\mathbf{Y}} + (\ell - \ell^*) \cdot \mathbf{G}];$$

- use the LWE assumption to replace $\mathbf{s}_j \cdot \mathbf{B}$ with random.

Step 2. Reduce to security of the ALS construction for message length ℓ^*. This step knows $\mathbf{T}_\ell$ for all $\ell \neq \ell^*$, which are used to produce functional secret keys for vectors of length $\ell \neq \ell^*$. The remaining functional secret keys, i.e. those for vector of length exactly equal to ℓ^*, are queried directly to the ALS challenger. As far as the challenge ciphertext is concerned, once the adversary outputs the challenge messages $\mathbf{x}_0, \mathbf{x}_1$ of length ℓ^*, these are forwarded to the ALS challenger, and the obtained challenge ciphertext ct^* is forwarded back to the adversary. The reduction is depicted pictorially in Figure 1.

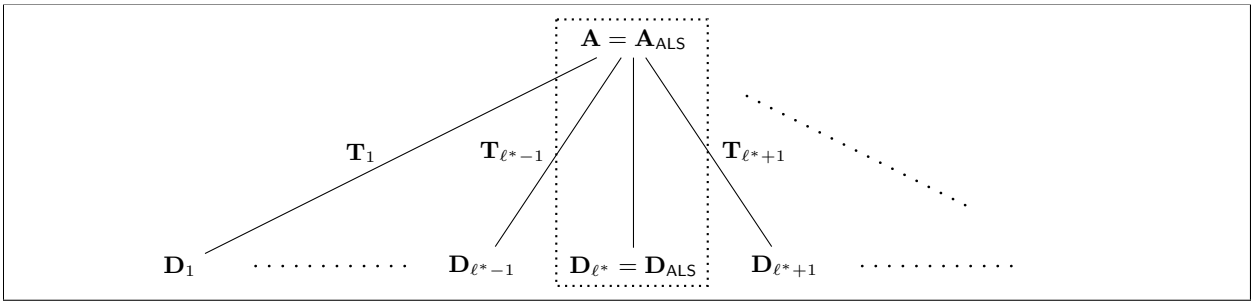

Fig. 1. Graphic representation of reduction to security of ALS construction for the strict UIPFE scheme.

Getting to Permissive UIPFE. Recall that, in a permissive UIPFE scheme, we need to allow both an honest party and an adversary to ask for keys for vectors $\mathbf{y}$ with different lengths ℓ', such that these keys can be used on any ciphertext encrypting a vector of length ℓ, with $\ell \geq \ell'$, to recover $\langle \mathbf{x}|_{[\ell']}, \mathbf{y} \rangle$. The previous scheme clearly does not support this functionality.

The obvious approach of combining multiple strict UIPFE, one for each possible message length, clearly fails as there is no a-priori upper-bound on the message length. Similarly, starting from a strict UIPFE and trying to produce during key generation, on input some vector $\mathbf{y}$ of length ℓ', functional secret keys for $(\mathbf{y}||\mathbf{0}_{\ell-\ell'})$ such that $\langle \mathbf{x}|_{[\ell']}, \mathbf{y} \rangle = \langle \mathbf{x}, (\mathbf{y}||\mathbf{0}_{\ell-\ell'}) \rangle$ can be used to recover the appropriate value also fails for the same reason: KeyGen does not have an a-priori bound on ℓ. Hence, extending a strict UIPFE scheme to permissive UIPFE seems to be a non-trivial task and no compiler from strict to permissive is currently known.

We overcome the aforementioned challenges by appropriately modifying our previous construction: informally to construct the permissive UIPFE, we start from the strict UIPFE construction described above. To encrypt a plaintext $\mathbf{x}$ of length ℓ, we apply the strict UIPFE encryption algorithm to each prefix $\mathbf{x}|_{[\ell']}$ of $\mathbf{x}$ for $\ell' \in [\ell]$,[14] with one key modification: these encryptions are not entirely *independent* – they share the same randomness $\mathbf{s}$ and ciphertext component $\mathbf{c}_0$. Key generation remains unchanged and during decryption, the relevant components of the ciphertext, together with a functional secret key for a vector $\mathbf{y}$ of length $\ell' \leq \ell$, can be used as before to recover $\langle \mathbf{x}|_{[\ell']}, \mathbf{y} \rangle$.

The most challenging task is to prove security of this modified construction. Indeed, the approach used in the strict UIPFE construction no longer works: if we try to reduce the security to that of $\mathsf{ALS}[\ell^*]$ as done before,[15] this means that to simulate 1) the challenge ciphertext and 2) functional keys for vectors $\mathbf{y}$ of some length $\ell' \leq \ell^*$ the reduction would have to query the $\mathsf{ALS}[\ell^*]$ challenger on input 1) the adversarial challenge message $(\mathbf{x}_0, \mathbf{x}_1)$ and 2) the vector $(\mathbf{y}||\mathbf{0}_{\ell^*-\ell'})$ for functional secret keys respectively (since these keys can be used to recover information from the challenge ciphertext). In particular, this implies that

1. the vector $\mathbf{d}_j$ obtained in Equation (1) for the challenge ciphertext would only depend on the index j, and not on the length ℓ^* of the challenge message,
2. for any $\ell \leq \ell^*$ the matrices $\mathbf{T}_\ell, \mathbf{T}_{\ell^*}$ will have to satisfy $\mathbf{T}_{\ell^*} = [\mathbf{T}_\ell||\mathbf{t}_{\ell+1}^\top|| \dots ||\mathbf{t}_{\ell^*}^\top]$ for some vectors $\mathbf{t}_{\ell+1}^\top, \dots, \mathbf{t}_{\ell^*}^\top$.

Since by functionality the secret keys for vectors of length ℓ can be used on ciphertexts encrypting messages of length $> \ell$, this implies that Item 2 holds for any ℓ' (and not just for $\ell' = \ell^*$).

From these facts it follows that the reduction must fail. Concretely, an adversary could query to the KeyGen oracle a vector $\mathbf{y} = (\tilde{\mathbf{y}}||\mathbf{0})$ such that $|\tilde{\mathbf{y}}| = \ell^*$ and $\langle \mathbf{x}_0, \tilde{\mathbf{y}} \rangle \neq \langle \mathbf{x}_1, \tilde{\mathbf{y}} \rangle$. Such an adversary is valid since there is no restriction on what functional secret keys for vectors of length $> \ell^*$ the adversary queries. However, by the above facts we also know that such secret key can actually be used as valid secret key for vector $\tilde{\mathbf{y}}$, which by the choice of the vector $\tilde{\mathbf{y}}$ allows to distinguish whether the challenge ciphertext contains $\mathbf{x}_0$ or $\mathbf{x}_1$.

In order to overcome these challenges we modify the way we reduce security of the UIPFE construction to that of the ALS scheme. More precisely, instead of reducing the security to that of $\mathsf{ALS}[\ell^*]$ we reduce to that of $\mathsf{ALS}[\ell^*(\ell^*+1)/2]$. In this way we can "embed" each possible message length into a different orthogonal subspaces of $\mathbb{Z}^{\ell^*(\ell^*+1)/2} = \mathbb{Z}^1 \times \mathbb{Z}^2 \times \cdots \times \mathbb{Z}^{\ell^*}$.

To explain the exact strategy we first need to describe the exact modifications to the scheme. We modify how encryption works and, in particular, the ciphertext now has the form

$$\mathsf{ct} = \Big(\overbrace{\mathbf{s} \cdot \mathbf{A}}^{\mathbf{c}_0}, \overbrace{\{\mathbf{s}_{i,j} \cdot \mathbf{v}^\top + x_j \cdot \left\lfloor \frac{q}{K_i} \right\rfloor}^{\mathbf{c}_{1,i,j}}, \overbrace{\mathbf{s}_{i,j} \cdot \mathbf{B}}^{\mathbf{c}_{2,i,j}}, \overbrace{\mathbf{s}_{i,j} \cdot (\mathbf{W} + j \cdot \mathbf{G}) - \mathbf{s} \cdot \mathbf{G}}^{\mathbf{c}_{3,i,j}}, \overbrace{\mathbf{s}_{i,j} \cdot (\mathbf{Y} + i \cdot \mathbf{G})}^{\mathbf{c}_{4,i,j}} \}_{i \in [\ell], j \in [i],} \Big), \ \mathbf{s}, \mathbf{s}_{i,j} \leftarrow \mathbb{Z}_q^n.$$

[14] If $\mathbf{x} = (x_1, \dots, x_\ell)$, then $\mathbf{x}|_{[\ell']} = (x_1, \dots, x_{\ell'})$.

[15] We are not claiming here that no security reduction to the security of $\mathsf{ALS}[\ell^*]$ is possible, we merely note that the one used before would not work here.

That is, for each $i \in [\ell]$, the components

$$\mathbf{c}_0, \{\mathbf{c}_{1,i,j}, \mathbf{c}_{2,i,j}, \mathbf{c}_{3,i,j}, \mathbf{c}_{4,i,j}\}_{j\in[i]},$$

correspond to a "strict UIPFE encryption" of the message $\mathbf{x}|_{[i]}$. Notice that all of them share the same LWE secret $\mathbf{s}$ and ciphertext component $\mathbf{c}_0$.

When $\ell \geq \ell'$, the component corresponding to $i = \ell'$ can be used to run the same decryption algorithm from before and recover $\langle \mathbf{x}|_{[\ell']}, \mathbf{y}\rangle$ as required. Next we describe the changes to the security proof from before.

Step 1. Let ℓ^* be the length of the challenge vectors. For $h = 1, 2, \ldots, \ell^*$, and $k = 1, 2, \ldots, h$, we rely on pseudorandomness of $\mathbf{s}_{h,k} \cdot [\mathbf{B} \mid \mathbf{W} + k \cdot \mathbf{G} \mid \mathbf{Y} + h \cdot \mathbf{G}]$ to replace $\mathbf{c}_{2,h,k}, \mathbf{c}_{3,h,k}, \mathbf{c}_{4,h,k}$ with random and rewrite $\mathbf{c}_{1,h,k}$ in terms of $\mathbf{s}_{h,k} \cdot \mathbf{v}^\top + x_k \cdot \left\lfloor \frac{q}{K_\ell} \right\rfloor$ using (1). This is the same as before except that we in addition run over all possible $h \in [\ell^*]$ (whereas previously one only had $h = \ell^*$).

Step 2. Reduce to security of the ALS construction for message length $\ell^*(\ell^*+1)/2$. This step now knows $\mathbf{T}_{\ell'}$ for all $\ell' > \ell^*$, which are used to produce functional secret keys for vectors of length $\ell' > \ell^*$. The remaining functional secret keys are queried (in)directly to the ALS challenger for length $\ell^*(\ell^*+1)/2$. In particular, for a key query for a vector $\mathbf{y}$ of length $\ell' \leq \ell^*$, the reduction will prepare

$$\tilde{\mathbf{y}} = \left[\mathbf{0}_1 \mid \ldots \mid \mathbf{0}_{\ell'-1} \mid \mathbf{y} \mid \mathbf{0}_{\ell'+1} \mid \ldots \mid \mathbf{0}_{\ell^*}\right],$$

forward it to the ALS challenger, receive back the corresponding key, and send this to the adversary. As for the challeng ciphertext, once the adversary outputs the challenge messages $\mathbf{x}_0, \mathbf{x}_1$ of length ℓ^*, the reduction will prepare

$$\tilde{\mathbf{x}}_b = \left[\mathbf{x}_b|_{[1]} \;\middle|\; \ldots \;\middle|\; \mathbf{x}_b|_{[\ell']} \;\middle|\; \ldots \;\middle|\; \mathbf{x}_b|_{[\ell^*]}\right] \qquad \text{for } b \in \{0,1\}.$$

These are forwarded to the ALS challenger and the challenge ciphertext received back is sent to the adversary. Observe that

- on the one hand, correctness is preserved since

$$\langle \tilde{\mathbf{x}}_b, \tilde{\mathbf{y}}\rangle = \langle \mathbf{x}_b|_{[\ell']}, \mathbf{y}\rangle,$$

- on the other hand, the reduction goes through since the set of key queries made by the adversary, say $(\mathbf{y}_j)_{j\in[Q]}$, will induce a set of key queries $(\tilde{\mathbf{y}}_j)_{j\in[Q]}$ for the $\mathsf{ALS}[\ell^*(\ell^*+1)/2]$ challenger that is valid for messages $\tilde{\mathbf{x}}_0, \tilde{\mathbf{x}}_1$, as long as $(\mathbf{y}_j)_{j\in[Q]}$ is a set of valid queries for messages $\mathbf{x}_0, \mathbf{x}_1$.

This allows us to reduce the security to that of $\mathsf{ALS}[\ell^*(\ell^*+1)/2]$, as depicted in Figure 2, and conclude the proof.

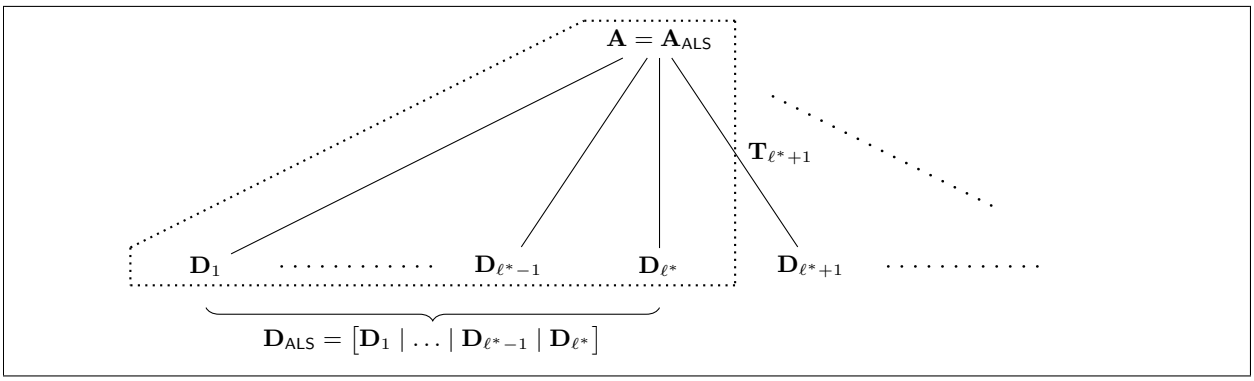

Fig. 2. Graphic representation of reduction to security of ALS construction for the permissive UIPFE scheme.

Fine-Grained UIPFE. The security proofs of the previous constructions follow a modular approach, reducing the security of our constructions to that of the ALS IPFE. This reduction framework provides an interesting and useful feature: it allows for the substitution of the "underlying" IPFE with a "richer" primitive while maintaining security almost for free. In particular, by replacing the ALS IPFE with identity-based IPFE (IB-IPFE) or attribute-based IPFE (AB-IPFE) from [LLW21], we show that we can obtain unbounded versions of these primitives. This is done in Section 5 and Section 6, respectively. This means that the same proof techniques extend naturally to these more expressive functional encryption schemes, allowing for fine-grained access control while preserving the efficiency and unbounded nature of the original constructions.

2 Preliminaries

Notation. We denote the security parameter by $\lambda \in \mathbb{N}$, by which each scheme and adversary is parameterized. For $n \in \mathbb{N}$, set $[n] = \{1, 2, \ldots, n\}$. We denote by $x \leftarrow\!\!\$\ \mathcal{X}$ the uniform sampling of the variable x from a set or distribution $\mathcal{X}$. We write $x \leftarrow \mathsf{A}(y)$ to denote that a probabilistic polynomial time (PPT) algorithm A on input y, outputs x. When we want to make the randomness r used by A explicit, we write it as $x \leftarrow \mathsf{A}(y; r)$. If A has oracle access to a procedure O, we represent this by superscript A^{O}. We define *polynomial* functions as $\mathsf{poly}(\lambda) = \bigcup_{d\in\mathbb{N}} O(\lambda^d)$ and *negligible* functions as $\mathsf{negl}(\lambda) = \bigcap_{d\in\mathbb{N}} o(\lambda^{-d})$.

We use boldface lower case for row vectors (e.g., $\mathbf{r}$) and boldface upper case for matrices (e.g., $\mathbf{R}$). For integral vectors and matrices (i.e., those over $\mathbb{Z}$), we use the notation $|\mathbf{r}|$ to denote the dimension of $\mathbf{r}$, and $||\mathbf{r}||$, $||\mathbf{R}||$ to denote the maximum absolute value over all the entries. We use $\approx_s$ and $\approx_c$ as the abbreviation for statistically close and computationally indistinguishable. We denoted by $\mathcal{D}_{\mathbb{Z}^m,\chi}$ the (centered) discrete Gaussian distribution over $\mathbb{Z}^m$ with parameter χ, i.e., the distribution over $\mathbb{Z}^m$ where for all $\mathbf{x}$, $\Pr[\mathbf{x}] \propto e^{-\pi\cdot(x_1^2+\cdots+x_m^2)/\chi^2}$.

2.1 Pseudorandom Functions

A pseudorandom function (PRF) is a family of functions $\{\mathsf{F}(\mathsf{k},\cdot) : \{0,1\}^{m(\lambda)} \to \{0,1\}^{\ell(\lambda)}\}_{\lambda\in\mathbb{N},\mathsf{k}\in\{0,1\}^\lambda}$ such that:

- *efficiency*: one can compute $\mathsf{F}(\mathsf{k}, x)$ in $\mathsf{poly}(\lambda)$-time given x and k,
- *security*: for any PPT adversary $\mathcal{A}$ let

$$\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{A},\mathsf{F}}(\lambda) := \left|\Pr\left[\mathcal{A}^{\mathsf{F}(\mathsf{k},\cdot)}(1^\lambda) = 1\right] - \Pr\left[\mathcal{A}^{R(\cdot)}(1^\lambda) = 1\right]\right|,$$

  where $\mathsf{k} \leftarrow\!\!\$\ \{0,1\}^\lambda$ and $R \leftarrow\!\!\$\ \mathcal{F}(\{0,1\}^{m(\lambda)} \to \{0,1\}^{\ell(\lambda)})$, with $\mathcal{F}(\{0,1\}^{m(\lambda)} \to \{0,1\}^{\ell(\lambda)})$ denoting the set of all functions mapping $m(\lambda)$ bits to $\ell(\lambda)$ bits. A PRF F is secure if for all PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{A},\mathsf{F}}(\lambda)$ is a negligible function in λ.

2.2 Functional Encryption

We recall the definition of functional encryption as originally defined in [BSW11, O'N10].

Definition 1. *A functionality F defined over $(\mathcal{K}, \mathcal{X})$ is a function $F\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Z} \cup \{\perp\}$, where $\mathcal{K}$ is a key space, $\mathcal{X}$ is a message space and $\mathcal{Z}$ is an output space, which does not contain the special symbol $\perp$.*

Definition 2 (Functional Encryption). *A functional encryption (FE) scheme for a functionality $F\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Z} \cup \{\perp\}$ is a tuple of algorithms $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows:*

$\mathsf{Setup}(1^\lambda)$**:** *takes as input a security parameter 1^λ, and outputs a master public/secret keypair $(\mathsf{mpk}, \mathsf{msk})$. (We assume that mpk is given as an implicit input to all other algorithms.)*

$\mathsf{KeyGen}(\mathsf{msk}, k)$**:** *takes as input the master secret key msk and a key (i.e., a function description) $k \in \mathcal{K}$, and outputs a functional secret key sk_k.*

$\mathsf{Enc}(\mathsf{mpk}, x)$**:** *takes as input the master public key* mpk *and a message* $x \in \mathcal{X}$*, and outputs a ciphertext* c*.*

$\mathsf{Dec}(\mathsf{sk}_k, c)$**:** *takes as input a functional secret key* sk_k *and a ciphertext* c*, and returns an output* $z \in \mathcal{Z} \cup \{\perp\}$*, where* $\perp$ *is a special rejection symbol.*

Correctness. The correctness property states that for all $\lambda \in \mathbb{N}$, for any $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, for any key $k \in \mathcal{K}$, any message $x \in \mathcal{X}$, if $\mathsf{sk}_k \leftarrow \mathsf{KeyGen}(\mathsf{msk}, k)$ and $c \leftarrow \mathsf{Enc}(\mathsf{mpk}, x)$, then we have that $\mathsf{Dec}(\mathsf{sk}_k, c) = F(k, x)$.

Security. For any stateful PPT adversary $\mathcal{A}$ and any functional encryption scheme FE, we define the advantage function

$$\mathsf{Adv}^{\mathsf{FE}}_{\mathcal{A}}(\lambda) := \Pr\left[\, b = b' \;\middle|\; \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ (x_0, x_1) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk}, \cdot)}(\mathsf{mpk}) \\ b \leftarrow\!\!\$\ \{0,1\} \\ c \leftarrow \mathsf{Enc}(\mathsf{mpk}, x_b) \\ b' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk}, \cdot)}(c) \end{array} \right] - \frac{1}{2},$$

with the restriction that all queries k that $\mathcal{A}$ sent to $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$ satisfy $F(k, x_0) = F(k, x_1)$. We say that an FE scheme is *adaptively* IND-CPA *secure* if for all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{FE}}_{\mathcal{A}}(\lambda)$ is a negligible function in λ.

2.3 Unbounded and Fine-Grained Inner-Product Functionality

Here we give two definitions of IPFE for unbounded vectors adapted from [DP19] to cover the fine-grained setting. The definitions differ solely in their requirement on the domains of the ciphertexts and the keys for encryption to be successful. Let $R = \{R_\lambda\}_{\lambda \in \mathbb{N}}$ be a sequence of relations, where $R \colon \mathcal{C} \times \mathcal{W} \to \{0,1\}$is defined as $R(C, x) = C(x)$.

Definition 3 (Strict Fine-Grained Unbounded IPFE).

- $\mathcal{K} = \{\epsilon\} \cup \{(C, (y_i)_{i \in \mathcal{D}'}) \mid C \in \mathcal{C}, \mathcal{D}' \subset \mathbb{N}^*$ *finite and* $y_i \in \mathbb{Z}, \forall i \in \mathcal{D}'\}$
- $\mathcal{X} = \{(w, \mathbf{x} = (x_i)_{i \in \mathcal{D}}) \mid w \in \mathcal{W}, \mathcal{D} \subset \mathbb{N}^*$ *finite and* $x_i \in \mathbb{Z}, \forall i \in \mathcal{D}\}$
- $\mathcal{Z} = \mathbb{Z}$
- $F(\epsilon, (w, (x_i)_{i \in \mathcal{D}})) = (w, \mathcal{D})$ *and*

$$F((C, (y_i)_{i \in \mathcal{D}'}), (w, (x_i)_{i \in \mathcal{D}})) = \begin{cases} \langle \mathbf{x}, \mathbf{y} \rangle & \text{if } \mathcal{D}' = \mathcal{D} \wedge C(w) = 0^{16}, \\ \perp & \text{otherwise.} \end{cases}$$

Definition 4 (Permissive Fine-Grained Unbounded IPFE).

- $\mathcal{K} = \{\epsilon\} \cup \{(C, (y_i)_{i \in \mathcal{D}'}) \mid C \in \mathcal{C}, \mathcal{D}' \subset \mathbb{N}^*$ *finite and* $y_i \in \mathbb{Z}, \forall i \in \mathcal{D}'\}$
- $\mathcal{X} = \{(w, \mathbf{x} = (x_i)_{i \in \mathcal{D}}) \mid w \in \mathcal{W}, \mathcal{D} \subset \mathbb{N}^*$ *finite and* $x_i \in \mathbb{Z}, \forall i \in \mathcal{D}\}$
- $\mathcal{Z} = \mathbb{Z}$
- $F(\epsilon, (w, (x_i)_{i \in \mathcal{D}})) = (w, \mathcal{D})$ *and*

$$F((C, (y_i)_{i \in \mathcal{D}'}), (w, (x_i)_{i \in \mathcal{D}})) = \begin{cases} \langle \mathbf{x}, \mathbf{y}|_{\mathcal{D}'} \rangle & \text{if } \mathcal{D}' \subset \mathcal{D} \wedge C(w) = 0, \\ \perp & \text{otherwise.} \end{cases}$$

In the plain setting, we have $\mathcal{C} = \mathcal{W} = \emptyset$and R is the null relation, and in the identity-based setting we have $\mathcal{C} = \mathcal{W} = \{0,1\}^\lambda$(i.e., the identity space) and R becomes equality check on identities. For ease of exposition, in Sections 3 and 4 we drop explicitly mentioning the domains $\mathcal{D}$ and $\mathcal{D}'$, and instead we consider the *consecutive* setting, i.e., each element of a vector is automatically indexed to its position when input to the encryption or key generation algorithm, e.g., for a vector (a, b, c), a's index is set to 1, b's index to 2 and c's index to 3.

[16] We follow here the convention that $C(w) = 0$ signifies authorization, which is a standard convention used in lattice-based ABE schemes.

2.4 Lattice Preliminaries

Learning with Errors. Given $n, m, q, \chi_e \in \mathbb{N}$, the $\mathsf{LWE}_{n,m,q,\chi_e}$ assumption states that

$$(\mathbf{A}, \mathbf{s} \cdot \mathbf{A} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{c}),$$

where

$$\mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_q^n, \mathbf{e} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m, \chi_e}, \mathbf{c} \leftarrow\!\!\$\ \mathbb{Z}_q^m.$$

Lemma 1 (Noise Flooding [GKPV10]). *Let $n \in \mathbb{N}$. For any real $\chi > \omega(\sqrt{\log n})$, and any $\mathbf{c} \in \mathbb{Z}^n$, it holds $\mathsf{SD}(\mathcal{D}_{\mathbb{Z}^n, \chi}, \mathcal{D}_{\mathbb{Z}^n, \chi} + \mathbf{c}) \leq ||\mathbf{c}||/\chi$. In particular, if $\chi \geq \lambda^{\omega(1)} \cdot ||\mathbf{c}||$, one has $\mathcal{D}_{\mathbb{Z}^n, \chi} \approx_s \mathcal{D}_{\mathbb{Z}^n, \chi} + \mathbf{c}$.*

Leftover Hash Lemma and Generalizations. A result that we will use is the so-called leftover hash lemma (LHL) [HILL99], which states that for $m \geq (n+1) \cdot \log q + 2 \cdot \lambda$ the distribution of $(\mathbf{A}, \mathbf{u} = \mathbf{A} \cdot \mathbf{x})$ for uniform and independent $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{x} \leftarrow \{1, -1\}^m$ is statistically indistinguishable from uniformly random.

Lemma 2 (Generalized Leftover Hash Lemma [DRS04, ABB10]). *Suppose that $m > (n+1) \log q + \omega(\log n)$ and that $q > 2$ is prime. Let $\mathbf{R}$ be an $m \times k$ matrix chosen uniformly in $\{1, -1\}^{m \times k} \mod q$ where $k = k(n)$ is polynomial in n. Let $\mathbf{A}$ and $\mathbf{B}$ be matrices chosen uniformly in $\mathbb{Z}_q^{n \times m}$ and $\mathbb{Z}_q^{n \times k}$ respectively. Then, for all vectors $\mathbf{w}$ in $\mathbb{Z}_q^m$, the distribution $(\mathbf{A}, \mathbf{A} \cdot \mathbf{R}, \mathbf{w}^\top \cdot \mathbf{R})$ is statistically close to the distribution $(\mathbf{A}, \mathbf{B}, \mathbf{w}^\top \cdot \mathbf{R})$.*

Trapdoor and Preimage Sampling. Let $n, q \in \mathbb{Z}$,

$$\mathbf{g}_q = (1, 2, 4, \ldots, 2^{\lceil \log q \rceil - 1}) \in \mathbb{Z}^{\lceil \log q \rceil}.$$

The gadget matrix $\mathbf{G}_{n,q}$ is defined as the diagonal concatenation of $\mathbf{g}_q$ n times. Formally, $\mathbf{G}_{n,q} = \mathbf{g}_q \otimes \mathbf{I}_n \in \mathbb{Z}^{n \times n \cdot \lceil \log q \rceil}$. For any $t \in \mathbb{Z}$, the function $\mathbf{G}_{n,q}^{-1} : \mathbb{Z}_q^{n \times t} \to \{0,1\}^{n \cdot \lceil \log q \rceil \times t}$ expands each entry $a \in \mathbb{Z}_q$ of the input matrix into a column of size $\lceil \log q \rceil$ consisting of the bit-representation of a. For any matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$ it holds that $\mathbf{G}_{n,q} \cdot \mathbf{G}_{n,q}^{-1}(\mathbf{A}) = \mathbf{A} \mod q$. We refer to the gadget matrix simply as $\mathbf{G}$ when parameters n and q are clear from the context.

Let $n, m, q \in \mathbb{N}$ and consider a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. For all $\mathbf{V} \in \mathbb{Z}_q^{n \times m'}$ we let $\mathbf{A}^{-1}(\mathbf{V}, \tau)$ denote the random variable whose distribution is the discrete Gaussian $\mathcal{D}_{\mathbb{Z}^{m \times m'}, \tau}$ conditioned on $\mathbf{A} \cdot \mathbf{A}^{-1}(\mathbf{V}, \tau) = \mathbf{V} \mod q$. If $\mathbf{Y} \leftarrow \mathbf{A}^{-1}(\mathbf{V}, \tau)$ then $||\mathbf{Y}|| \leq k \cdot \tau \cdot \sqrt{m \cdot m'}$ with probability at least $1 - e^{-\Omega(k^2)}$. A matrix $\mathbf{T} \in \mathbb{Z}^{m \times m}$ such that $\mathbf{A} \cdot \mathbf{T} = \mathbf{H} \cdot \mathbf{G}$, for some invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$ is called a τ-trapdoor for $\mathbf{A}$, for $\tau \geq 2 \cdot m \cdot \sqrt{n \cdot \log q} \cdot ||\mathbf{T}||$. The following properties have been established in a long sequence of works.

Lemma 3 (Trapdoor Generation and Sampling [Ajt96, GPV08, MP12]). *There exists a pair of probabilistic polynomial-time algorithms:*

- $\mathsf{TrapGen}(1^n, 1^m, q)$ *that for all $m \geq m_0 = m_0(n, q) = O(n \log q)$, outputs $(\mathbf{A}, \mathbf{T}_\mathbf{A})$ s.t. $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is within statistical distance 2^{-n} from uniform and $\mathbf{T}_\mathbf{A}$ is a τ-trapdoor for $\mathbf{A}$ where $\tau = O(\sqrt{n} \cdot \log q \cdot \log n)$.*
- $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{V}, \tau)$ *that given $\mathbf{A}$ and any τ-trapdoor $\mathbf{T}$ of $\mathbf{A}$, outputs a sample from $\mathbf{A}^{-1}(\mathbf{V}, \tau)$.*

Moreover

1. *for $\mathbf{x} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m, \tau}$, the marginal distribution of $\mathbf{y} = \mathbf{A} \cdot \mathbf{x} \in \mathbb{Z}_q^n$ is uniform (up to $\mathsf{negl}(n)$ statistical distance), and the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ is $\mathbf{A}^{-1}(\mathbf{y}, \tau)$.*

Lemma 4 (Trapdoor Extension [ABB10, CHKP10]). *Given $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, with a τ-trapdoor $\mathbf{T}$, it is efficient to sample from $[\mathbf{A}|\mathbf{B}]^{-1}(\cdot, \tau)$ for all $\mathbf{B} \in \mathbb{Z}_q^{n \times k}$. Moreover, for any $\mathbf{V} \in \mathbb{Z}_q^{n \times m'}$, the following two distributions are statistically close*

– $\mathbf{U} \in \mathbb{Z}^{m+k\times m'}$, *where* $\mathbf{U} \leftarrow\!\$\ [\mathbf{A}|\mathbf{B}]^{-1}(\mathbf{V}, \tau)$,
– $\begin{bmatrix}\mathbf{U}_0\\ \mathbf{U}_1\end{bmatrix} \in \mathbb{Z}^{m+k\times m'}$, *where* $\mathbf{U}_1 \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^{k\times m'},\tau}$ *and* $\mathbf{U}_0 \leftarrow\!\$\ \mathbf{A}^{-1}(\mathbf{V} - \mathbf{B}\mathbf{U}_1, \tau)$.

Homomorphic Computation on Matrices. We recall the basic homomorphic computation on matrices from [BGG+14]:

Theorem 1 ([GSW13, BGG+14]). *There exist efficient deterministic algorithms* EvalC *and* EvalCX *such that for all* $n, q, \ell \in \mathbb{N}$, *and for any sequence of matrices* $\mathbf{A} := (\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \in (\mathbb{Z}^{n\times n\cdot\lceil\log q\rceil})^\ell$, *for any depth-d Boolean circuit* $C\colon \{0,1\}^\ell \to \{0,1\}$ *and for every* $\mathbf{x} := (x_1, \ldots, x_\ell) \in zo^\ell$, *the following properties hold:*

– *The outputs* $\mathbf{A}_C := \mathsf{EvalC}(\mathbf{A}, C)$ *and* $\mathbf{H}_{\mathbf{A},C,\mathbf{x}} := \mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x})$ *are matrices in* $\mathbb{Z}_q^{n\times(n\cdot\lceil\log q\rceil)}$ *and* $\mathbb{Z}^{(\ell\cdot n\cdot\lceil\log q\rceil)\times(n\cdot\lceil\log q\rceil)}$.
– *It holds that* $||\mathbf{H}_{\mathbf{A},C,\mathbf{x}}|| \le (n\cdot\log q)^{O(d)}$.
– *It holds that*

$$(\mathbf{A} - \mathbf{x}\otimes\mathbf{G}_{n,q})\cdot\mathbf{H}_{\mathbf{A},C,\mathbf{x}} = \mathbf{A}_C - C(\mathbf{x})\cdot\mathbf{G}_{n,q} \bmod q.$$

For a proof of this theorem, we refer the reader to [BCTW16, Fact 3.4].

3 Strict Unbounded IPFE

3.1 Construction

We consider a message space $\mathcal{V} = \{0, \ldots, V-1\}^*$, for some integer V, secret keys are associated with vectors in $\mathcal{P} = \{0, \ldots, P-1\}^*$, for some integer P, and inner products are evaluated over $\mathbb{Z}$. We can efficiently decrypt inner product values within an interval: here the inner product between the plaintext and key vectors of length ℓ belongs to $\{0, \ldots, K_\ell - 1\}$, with $K_\ell = \ell\cdot P\cdot V$.

Let $\Pi_1 = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ be an UIPFE scheme defined as follows:

– $\mathsf{Setup}(1^\lambda, \mathcal{P}, \mathcal{V})$: Sample

$$(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), (\mathbf{B}, \mathbf{T}_\mathbf{B}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{W}, \mathbf{Y} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}, \mathbf{v} \leftarrow\!\$\ \mathbb{Z}_q^n$$
$$\mathsf{k} \leftarrow\!\$\ \{0,1\}^\lambda.$$

Set $\mathsf{mpk} = (\mathbf{A}, \mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v})$, and $\mathsf{msk} = (\mathbf{T}_\mathbf{A}, \mathbf{T}_\mathbf{B}, \mathsf{k})$. Return $(\mathsf{mpk}, \mathsf{msk})$.
– $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x} \in \mathcal{V})$: Let $\ell = |\mathbf{x}|$. Sample

$$\mathbf{s} \leftarrow\!\$\ \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi''},$$
$$\mathbf{s}_j \leftarrow\!\$\ \mathbb{Z}_q^n, e_{1,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi''}, \mathbf{e}_{2,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi}, \mathbf{e}_{3,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi'}, \mathbf{e}_{4,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi'} \quad \text{for all } j \in [\ell].$$

Compute

$$\begin{aligned}
\mathbf{c}_0 &:= \mathbf{s}\cdot\mathbf{A} + \mathbf{e}_0 \mod q,\\
\mathbf{c}_{1,j} &:= \mathbf{s}_j\cdot\mathbf{v}^\top + x_j\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + e_{1,j} \mod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{2,j} &:= \mathbf{s}_j\cdot\mathbf{B} + \mathbf{e}_{2,j} \bmod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{3,j} &:= \mathbf{s}_j\cdot(\mathbf{W} + j\cdot\mathbf{G}) - \mathbf{s}\cdot\mathbf{G} + \mathbf{e}_{3,j} \mod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{4,j} &:= \mathbf{s}_j\cdot(\mathbf{Y} + \ell\cdot\mathbf{G}) + \mathbf{e}_{4,j} \mod q \quad \text{for all } j\in[\ell].
\end{aligned}$$

Return $\mathsf{ct}_\mathbf{x} := \left(\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j\in[\ell]}, \{\mathbf{c}_{2,j}\}_{j\in[\ell]}, \{\mathbf{c}_{3,j}\}_{j\in[\ell]}, \{\mathbf{c}_{4,j}\}_{j\in[\ell]}\right)$.

– $\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y} \in \mathcal{P})$: Let $\ell' = |\mathbf{y}|$. For all $j \in [\ell']$, sample

$$\mathbf{k}_{j,\ell'}^\top \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell' \cdot \mathbf{G}], \begin{bmatrix} \mathbf{T_B} \\ \mathbf{0}_{m\times m} \\ \mathbf{0}_{m\times m} \end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k}, j, \ell', 1)\right).$$

Parse

$$\mathbf{k}_{j,\ell'}^\top = \begin{bmatrix} \mathbf{z}_{j,\ell'}^\top \\ \mathbf{r}_{j,\ell'}^\top \\ \mathbf{p}_{j,\ell'}^\top \end{bmatrix},$$

and let $\mathbf{d}_{j,\ell'}^\top \coloneqq \mathbf{G} \cdot \mathbf{r}_{j,\ell'}^\top \bmod q$. Let $\mathbf{D}_{\ell'} \coloneqq [\mathbf{d}_{1,\ell'}^\top \mid \ldots \mid \mathbf{d}_{\ell',\ell'}^\top]$. Sample

$$\mathbf{T}_{\ell'} \leftarrow \mathsf{SamplePre}\Big(\mathbf{A}, \mathbf{T_A}, \mathbf{D}_{\ell'}, \tau_2; \mathsf{F}(\mathsf{k}, \ell', 2)\Big),$$

Set $\mathbf{t}_\mathbf{y}^\top = \mathbf{T}_{\ell'} \cdot \mathbf{y}^\top$. Return $\mathsf{sk}_\mathbf{y} \coloneqq \big(\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y}\big)$.

– $\mathsf{Dec}(\mathsf{sk}_\mathbf{y}, \mathbf{y}, \mathsf{ct}_\mathbf{x})$: Parse $\mathsf{ct}_\mathbf{x} = \big(\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j\in[\ell]}, \{\mathbf{c}_{2,j}\}_{j\in[\ell]}, \{\mathbf{c}_{3,j}\}_{j\in[\ell]}, \{\mathbf{c}_{4,j}\}_{j\in[\ell]}\big)$, $\mathsf{sk}_\mathbf{y} = \big(\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y}\big)$. If $\ell \neq \ell'$, return $\perp$. Else, compute

$$\mu' \coloneqq \left(\Big[\mathbf{c}_{1,1} \,\Big|\, \ldots \,\Big|\, \mathbf{c}_{1,\ell}\Big] - \Big[\big[\mathbf{c}_{2,1} \,|\, \mathbf{c}_{3,1} \,|\, \mathbf{c}_{4,1}\big] \cdot \mathbf{k}_{1,\ell}^\top \,\Big|\, \ldots \,\Big|\, \big[\mathbf{c}_{2,\ell} \,|\, \mathbf{c}_{3,\ell} \,|\, \mathbf{c}_{4,\ell}\big] \cdot \mathbf{k}_{\ell,\ell}^\top\Big]\right) \cdot \mathbf{y}^\top - \mathbf{c}_0 \cdot \mathbf{t}_\mathbf{y}^\top \mod q$$

and return the value $\mu \in \{-K_\ell + 1, \ldots, K_\ell - 1\}$ that minimizes $|\lfloor \frac{q}{K_\ell} \rfloor \cdot \mu - \mu'|$.

Parameters. We have three Gaussian parameters:

$$\overbrace{\chi}^{\approx||\mathbf{e}_{2,j}||} \leq \overbrace{\chi'}^{\approx||\mathbf{e}_{3,j}||,||\mathbf{e}_{4,j}||} \leq \overbrace{\chi''}^{\approx||\mathbf{e}_0||,||\mathbf{e}_{1,j}||}.$$

The parameters requirements can be compactly specified as:

$m \geq O(n \log q)$	trapdoor generation (Lemma 3)
$2^{n^\delta} \geq q/\chi, \qquad \chi \geq O(n + \lambda)$	$\mathsf{LWE}_{n,m,\chi,q}$ hardness ($\mathsf{H}_{3,k,6} \approx_c \mathsf{H}_{3,k,7}$)
$\chi' \geq m \cdot \chi \cdot \lambda^{\omega(1)}$	noise flooding ($\mathsf{H}_{3,k,5} \approx_s \mathsf{H}_{3,k,6}$)
$\chi'' \geq 3 \cdot m \cdot \chi' \cdot \tau_1 \cdot \lambda^{\omega(1)}$	noise flooding ($\mathsf{H}_{3,k,4} \approx_s \mathsf{H}_{3,k,5}$)
$m \geq (n+1) \cdot \log q + 2\lambda + \omega(\log n)$	(G)LHL ($\mathsf{H}_{3,k,0} \approx_s \mathsf{H}_{3,k,1}, \mathsf{H}_{3,k,7} \approx_s \mathsf{H}_{3,l,8}$)
$\tau_1 \geq O(m^2)$	trapdoor generation ($\mathsf{H}_{3,k,2} \approx_s \mathsf{H}_{3,k,3}$)
$\tau_2 \geq O(\sqrt{n} \cdot \log q \cdot \log n)$	trapdoor generation (ALS)
$q \geq \lambda^{\omega(1)} \cdot \mathsf{poly}(\lambda, m, \ell) \cdot \tau_1 \cdot \tau_2 \cdot K_\ell \cdot P \cdot (\chi'' + \chi + \chi')$	correctness (Theorem 2)

Fix $\lambda_1 = \lambda^{\omega(1)}$. Bounding $\mathsf{poly}(\lambda, m, \ell)$ factors by λ_1, we can set

$$\begin{aligned} &n = \mathsf{poly}(\lambda) && m = O(n \log q), \\ &\chi = \lambda_1, && \chi' = \lambda_1^3, && \chi'' = \lambda_1^6, \\ &\tau_1 = \lambda_1, && \tau_2 = \lambda_1, \\ &q = K_\ell \cdot P \cdot \lambda_1^{10} = V \cdot P^2 \cdot \lambda_1^{11}. \end{aligned} \tag{2}$$

Theorem 2 (Correctness). *Let Π_1 be the UIPFE construction described in Section 3, with parameters as in Equation (2). Then, Π_1 is correct.*

Hybrid	mpk	ct	$\mathsf{sk}_{\mathbf{y}}$	justification
H_0	$(\mathbf{A},\mathbf{T}_{\mathbf{A}}) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^n,1^m,q)$ $(\mathbf{B},\mathbf{T}_{\mathbf{B}}) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^n,1^m,q)$ $\mathbf{W} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{Y} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{v} \leftarrow\!\!\$\ \mathbb{Z}_q^{n}$ $\mathsf{k} \leftarrow \mathcal{K}$	$\mathbf{c}_0 \approx \mathbf{s}\cdot\mathbf{A}$ $\mathbf{c}_{1,j} \approx \mathbf{s}_j\cdot\mathbf{v} + x_j^*\cdot\lfloor\frac{q}{K}\rfloor$ $\mathbf{c}_{2,j} \approx \mathbf{s}_j\cdot\mathbf{B}$ $\mathbf{c}_{3,j} \approx \mathbf{s}_j\cdot(\mathbf{W}+j\cdot\mathbf{G}) - \mathbf{s}\cdot\mathbf{G}$ $\mathbf{c}_{4,j} \approx \mathbf{s}_j\cdot(\mathbf{Y}+\ell\cdot\mathbf{G})$	$\mathbf{T}_{\mathbf{A}},\mathbf{T}_{\mathbf{B}},\mathsf{k}$ $\mathbf{k}_{j,\ell}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}\mathbf{B}\mid\mathbf{W}+j\cdot\mathbf{G}\mid\mathbf{Y}+\ell\cdot\mathbf{G}\end{bmatrix}, \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k},j,\ell,1)\right)$ $\mathbf{D}_\ell = [\mathbf{G}\cdot\mathbf{r}_{1,\ell}^\top \mid \ldots \mid \mathbf{G}\cdot\mathbf{r}_{\ell,\ell}^\top]$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left(\mathbf{A},\mathbf{T}_{\mathbf{A}},\mathbf{D}_\ell,\tau_2;\mathsf{F}(\mathsf{k},\ell,2)\right)$ $\mathbf{t}_{\mathbf{y}}^\top = \mathbf{T}_\ell\cdot\mathbf{y}^\top$	
H_1	$\downarrow$	$\downarrow$	$\downarrow$	guess $\lvert\mathbf{x}^*\rvert = \ell^*$
H_2	$\downarrow$	$\downarrow$	$\boldsymbol{\rho} \leftarrow\!\!\$\ \{0,1\}^{\mathsf{poly}(\lambda)}$ $\mathbf{k}_{j,\ell}^\top \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}\mathbf{B}\mid\mathbf{W}+j\cdot\mathbf{G}\mid\mathbf{Y}+\ell\cdot\mathbf{G}\end{bmatrix}, \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell}\right)$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left(\mathbf{A},\mathbf{T}_{\mathbf{A}},\mathbf{D}_\ell,\tau_2;\boldsymbol{\rho}_{2,\ell}\right)$	PRF security
H_3	$\downarrow$	$\mathbf{c}_{2,j},\mathbf{c}_{3,j},\mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m$ $\mathbf{c}_{1,j} \approx [\mathbf{c}_{2,j}\mid\mathbf{c}_{3,j}\mid\mathbf{c}_{4,j}]\cdot\mathbf{k}_{j,\ell^*}^\top + \mathbf{s}\cdot\mathbf{d}_{j,\ell^*}^\top + x_j^*\cdot\lfloor\frac{q}{K_{\ell^*}}\rfloor$	$\downarrow$	$\mathsf{LWE}_{n,m,\chi'',q}$ (Figure 4)
H_4	$\downarrow$	$\downarrow$	$\mathbf{r}_{j,\ell} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m,\tau_1}$ $\begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}\mathbf{B}\mid\mathbf{Y}+\ell\cdot\mathbf{G}\end{bmatrix}, \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}-[\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\right)$ $\mathbf{k}_{j,\ell}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix}$	Lemma 4
H_5	$\downarrow$	$\downarrow$	$\mathbf{D}_\ell = [\mathbf{d}_{1,\ell}^\top \mid \ldots \mid \mathbf{d}_{\ell,\ell}^\top] \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times\ell}$ $\mathbf{r}_{j,\ell}^\top = \mathsf{SamplePre}(\mathbf{G},\mathbf{I},\mathbf{d}_{j,\ell}^\top,\tau_1;\boldsymbol{\rho}_{1,j,\ell,2})$ $\begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}\mathbf{B}\mid\mathbf{Y}+\ell\cdot\mathbf{G}\end{bmatrix}, \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top-[\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\right)$ $\mathbf{k}_{j,\ell}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix}$	Lemma 3 (Item 1)
H_6	$\downarrow$	$\downarrow$	$\mathbf{T}_\ell \leftarrow \mathcal{D}_{\mathbb{Z}^{m\times\ell},\tau_2}$ $\mathbf{D}_\ell = \mathbf{A}\cdot\mathbf{T}_\ell$	Lemma 3 (Item 1)
H_7	$\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$	$\downarrow$	$\downarrow$	Lemma 3 TrapGen
$\mathsf{ALS}[\ell^*]$ reduction	$\mathbf{A} = \mathbf{A}_{\mathsf{ALS}}$ $\mathbf{D}_{\ell^*} = \mathbf{D}_{\mathsf{ALS}}$	$\mathbf{c}_0 = \mathbf{c}_{0,\mathsf{ALS}}$ $\mathbf{c}_{1,j} = [\mathbf{c}_{2,j}\mid\mathbf{c}_{3,j}\mid\mathbf{c}_{4,j}]\cdot\mathbf{k}_{j,\ell^*}^\top + \mathbf{c}_{1,j,\mathsf{ALS}}$	$\mathbf{t}_{\mathbf{y}}^\top$ using $\begin{cases}\mathsf{sk}_{\mathbf{y},\mathsf{ALS}} & \text{if } \ell=\ell^*\\ \mathbf{T}_\ell\cdot\mathbf{y}^\top & \text{if } \ell\neq\ell^*\end{cases}$	ALS security

Fig. 3. Summary of the security proof of strict construction. $\downarrow$ denotes the same as the previous hybrid.

Proof. Fix $\mathbf{x}\in\{0,\ldots,V\}^\ell, \mathbf{y}\in\{0,\ldots,P\}^{\ell'}$, with $\ell=\ell'$. The bulk of the proof lies in showing that

$$\left(\left[\mathbf{c}_{1,1}\mid\ldots\mid\mathbf{c}_{1,\ell}\right] - \left[\left[\mathbf{c}_{2,1}\mid\mathbf{c}_{3,1}\mid\mathbf{c}_{4,1}\right]\cdot\mathbf{k}_{1,\ell}^\top\mid\ldots\mid\left[\mathbf{c}_{2,\ell}\mid\mathbf{c}_{3,\ell}\mid\mathbf{c}_{4,\ell}\right]\cdot\mathbf{k}_{\ell,\ell}^\top\right]\right)$$
$$= \mathbf{s}\cdot\mathbf{D}_\ell + \mathbf{x}\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + \tilde{\mathbf{e}} \bmod q, \tag{3}$$

where $||\tilde{\mathbf{e}}||$ is small. Correctness then follows readily from the fact that

$$\left(\mathbf{s}\cdot\mathbf{D}_\ell + \mathbf{x}\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + \tilde{\mathbf{e}}\right)\cdot\mathbf{y}^\top - \mathbf{c}_0\cdot\mathbf{t}_{\mathbf{y}}^\top = \left(\mathbf{s}\cdot\mathbf{D}_\ell + \mathbf{x}\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + \tilde{\mathbf{e}}\right)\cdot\mathbf{y}^\top - (\mathbf{s}\cdot\mathbf{A}+\mathbf{e}_0)\cdot\mathbf{t}_{\mathbf{y}}^\top$$
$$= \left\lfloor\frac{q}{K_\ell}\right\rfloor\cdot\langle\mathbf{x},\mathbf{y}\rangle + \boxed{\tilde{\mathbf{e}}\cdot\mathbf{y}^\top - \mathbf{e}_0\cdot\mathbf{t}_{\mathbf{y}}^\top} \mod q.$$

To prove Equation (3):

Hybrid	mpk	ct	$\mathsf{sk}_{\mathbf{y}}$	justification
$\mathsf{H}_{3,k,0}$	$\downarrow$	$\downarrow$	$\downarrow$	$\mathsf{H}_{3,1,0} = \mathsf{H}_2$, $\mathsf{H}_{3,k,0} = \mathsf{H}_{3,k-1,9}, k \in [2, \ell^*]$
$\mathsf{H}_{3,k,1}$	$\tilde{\mathbf{W}}_k, \tilde{\mathbf{Y}}_{\ell^*} \leftarrow \{1,-1\}^{m\times m}$ $\mathbf{W} = \mathbf{B}\cdot\tilde{\mathbf{W}}_k - k\cdot\mathbf{G}$ $\mathbf{Y} = \mathbf{B}\cdot\tilde{\mathbf{Y}}_{\ell^*} - \ell^*\cdot\mathbf{G}$	$\downarrow$	$\downarrow$	Lemma 2 (plain) LHL
$\mathsf{H}_{3,k,2}$	$\mathbf{v}^\top = [\mathbf{B} \mid \mathbf{W} + k\cdot\mathbf{G} \mid \mathbf{Y} + \ell^*\cdot\mathbf{G}]\cdot\mathbf{k}_{k,\ell^*}^\top$	$\downarrow$	$\mathbf{k}_{k,\ell^*} \leftarrow \mathcal{D}_{\mathbb{Z}^{3m},\tau_1}$	Lemma 3 (Item 1)
$\mathsf{H}_{3,k,3}$	$\downarrow$	$\downarrow$	$\mathbf{k}_{j,\ell}$ using $\begin{cases} \begin{bmatrix}\tilde{\mathbf{Y}}_{\ell^*}\\ \mathbf{0}_{m\times m}\\ -\mathbf{I}_m\end{bmatrix} & \text{if } \ell \neq \ell^* \\ \begin{bmatrix}\tilde{\mathbf{W}}_k\\ -\mathbf{I}_m\\ \mathbf{0}_{m\times m}\end{bmatrix} & \text{if } \ell = \ell^*, j \neq k \end{cases}$	Lemma 3 $\mathsf{SamplePre}$
$\mathsf{H}_{3,k,4}$	$\mathbf{B} \leftarrow \mathbb{Z}_q^{n\times m}$	$\downarrow$	$\downarrow$	Lemma 3 $\mathsf{TrapGen}$
$\mathsf{H}_{3,k,5}$	$\downarrow$	$\mathbf{c}_{1,k} = [\mathbf{c}_{2,k} \mid \mathbf{c}_{3,k} \mid \mathbf{c}_{4,k}]\cdot\mathbf{k}_{k,\ell^*}^\top$ $+\mathbf{s}\cdot\mathbf{d}_{k,\ell^*} + x_k^*\cdot\lfloor\frac{q}{K_{\ell^*}}\rfloor + \tilde{e}_{1,k}$	$\downarrow$	Lemma 1 Noise Flooding
$\mathsf{H}_{3,k,6}$	$\downarrow$	$\mathbf{c}_{3,k} = \mathbf{c}_{2,k}\cdot\tilde{\mathbf{W}}_k - \mathbf{s}\cdot\mathbf{G} + \tilde{\mathbf{e}}_{3,k}$ $\mathbf{c}_{4,k} = \mathbf{c}_{2,k}\cdot\tilde{\mathbf{Y}}_{\ell^*} + \tilde{\mathbf{e}}_{4,k}$	$\downarrow$	Lemma 1 Noise Flooding
$\mathsf{H}_{3,k,7}$	$\downarrow$	$\mathbf{c}_{2,k} \leftarrow \mathbb{Z}_q^m$	$\downarrow$	$\mathsf{LWE}_{n,m,\chi'',q}$
$\mathsf{H}_{3,k,8}$	$\downarrow$	$\mathbf{c}_{3,k}, \mathbf{c}_{4,k} \leftarrow \mathbb{Z}_q^m$	$\downarrow$	Lemma 2 GLHL
$\mathsf{H}_{3,k,9}$	same as H_2	$\downarrow$	same as H_2	$\mathsf{H}_{3,k,4} \approx_s \mathsf{H}_{3,k,0}$

Fig. 4. Summary for $\mathsf{H}_2 \approx_c \mathsf{H}_3$. The sequence of hybrid is repeated for all $k \in [\ell^*]$. That is, $\mathsf{H}_2 = \mathsf{H}_{3,1,0} \approx \cdots \approx \mathsf{H}_{3,1,9} = \mathsf{H}_{3,2,0} \approx \cdots \approx \mathsf{H}_{3,\ell^*,9} = \mathsf{H}_3$.

– First, for any $j \in [\ell]$, we have

$$\begin{aligned}\left[\mathbf{c}_{2,j} \,|\, \mathbf{c}_{3,j} \,|\, \mathbf{c}_{4,j}\right]\cdot\mathbf{k}_{j,\ell}^\top &= \left(\mathbf{s}_j\cdot\left[\mathbf{B}_0 \,|\, \mathbf{W} + j\cdot\mathbf{G} \,|\, \mathbf{Y} + \ell\cdot\mathbf{G}\right] - \mathbf{s}\cdot\left[\mathbf{0}_{n\times m} \,|\, \mathbf{G} \,|\, \mathbf{0}_{n\times m}\right] + \left[\mathbf{e}_{2,j} \,|\, \mathbf{e}_{3,j} \,|\, \mathbf{e}_{4,j}\right]\right)\cdot\mathbf{k}_{j,\ell}^\top \\ &= \mathbf{s}_j\cdot\left[\mathbf{B}_0 \,|\, \mathbf{W} + j\cdot\mathbf{G} \,|\, \mathbf{Y} + \ell\cdot\mathbf{G}\right]\cdot\mathbf{k}_{j,\ell}^\top - \mathbf{s}\cdot\left[\mathbf{0}_{n\times m} \,|\, \mathbf{G} \,|\, \mathbf{0}_{n\times m}\right]\cdot\mathbf{k}_{j,\ell}^\top + \boxed{\left[\mathbf{e}_{2,j} \,|\, \mathbf{e}_{3,j} \,|\, \mathbf{e}_{4,j}\right]\cdot\mathbf{k}_{j,\ell}^\top} \\ &\approx \mathbf{s}_j\cdot\mathbf{v}^\top - \mathbf{s}\cdot\mathbf{d}_{j,\ell}^\top \bmod q.\end{aligned}$$

– Further, we have

$$\begin{aligned}\mathbf{c}_{1,j} - \left(\mathbf{s}_j\cdot\mathbf{v}^\top - \mathbf{s}\cdot\mathbf{d}_{j,\ell}^\top\right) &= \mathbf{s}_j\cdot\mathbf{v}^\top + x_j\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + \boxed{e_{1,j}} - \mathbf{s}_j\cdot\mathbf{v}^\top + \mathbf{s}\cdot\mathbf{d}_{j,\ell}^\top \\ &\approx \mathbf{s}\cdot\mathbf{d}_{j,\ell}^\top + x_j\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor \bmod q.\end{aligned}$$

– Putting everything together, we obtain

$$\left(\left[\mathbf{c}_{1,1} \,\middle|\, \ldots \,\middle|\, \mathbf{c}_{1,\ell}\right] - \left[\left[\mathbf{c}_{2,1} \,|\, \mathbf{c}_{3,1} \,|\, \mathbf{c}_{4,1}\right]\cdot\mathbf{k}_{1,\ell}^\top \,\middle|\, \ldots \,\middle|\, \left[\mathbf{c}_{2,\ell} \,|\, \mathbf{c}_{3,\ell} \,|\, \mathbf{c}_{4,\ell}\right]\cdot\mathbf{k}_{\ell,\ell}^\top\right]\right) = \mathbf{s}\cdot\mathbf{D}_\ell + \mathbf{x}\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + \tilde{\mathbf{e}} \bmod q,$$

where $\mathbf{D}_\ell = \left[\mathbf{d}_{1,\ell}^\top \,|\, \ldots \,|\, \mathbf{d}_{\ell,\ell}^\top\right]$ and

$$\tilde{\mathbf{e}} \coloneqq [e_{1,1} \,|\, \ldots \,|\, e_{1,\ell}] - \left[\left[\mathbf{e}_{2,1} \,|\, \mathbf{e}_{3,1} \,|\, \mathbf{e}_{4,1}\right]\cdot\mathbf{k}_{1,\ell}^\top \,|\, \ldots \,|\, \left[\mathbf{e}_{2,\ell} \,|\, \mathbf{e}_{3,\ell} \,|\, \mathbf{e}_{4,\ell}\right]\cdot\mathbf{k}_{\ell,\ell}^\top\right],$$

with

$$||\tilde{\mathbf{e}}|| \leq \lambda\cdot\chi'' + 3\cdot m\cdot\lambda^2\cdot(\chi + \chi')\cdot\tau_1.$$

Finally, using

$$\mathbf{A}\cdot\mathbf{T}_\ell = \mathbf{D}_\ell \bmod q,$$

as $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\Big(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{D}_\ell, \tau_2; \mathsf{F}(\mathsf{k}, \ell, 2)\Big)$, and that $\mathbf{t}_\mathbf{y}^\top = \mathbf{T}_\ell \cdot \mathbf{y}^\top$, we have

$$\begin{aligned}\mathbf{c}_0 \cdot \mathbf{t}_\mathbf{y}^\top &= (\mathbf{s} \cdot \mathbf{A} + \mathbf{e}_0) \cdot \mathbf{T}_\ell \cdot \mathbf{y}^\top \\ &= \mathbf{s} \cdot \mathbf{A} \cdot \mathbf{T}_\ell \cdot \mathbf{y}^\top + \boxed{\mathbf{e}_0 \cdot \mathbf{T}_\ell \cdot \mathbf{y}^\top} \\ &\approx \mathbf{s} \cdot \mathbf{D}_\ell \cdot \mathbf{y}^\top \mod q.\end{aligned}$$

In particular, the norm of the final error term is, with overwhelming probability in λ, bounded by

$$\left|\left|\mathbf{e}_0 \cdot \mathbf{t}_\mathbf{y}^\top\right|\right| + \left|\left|\tilde{\mathbf{e}} \cdot \mathbf{y}^\top\right|\right| \leq \lambda^2 \cdot \ell \cdot m \cdot \chi'' \cdot \tau_2 \cdot P + \ell \cdot P \cdot \Big(\lambda \cdot \chi'' + 3 \cdot m \cdot \lambda^2 \cdot (\chi + \chi') \cdot \tau_1\Big),$$

where we have used the fact that $||\mathbf{T}_\ell|| \leq \lambda \cdot \tau_2$ and that $\mathbf{y}$ is a vector of length ℓ with norm bounded by P. Since

$$q \geq \lambda^{\omega(1)} \cdot \mathsf{poly}(\lambda, m, \ell) \cdot \tau_1 \cdot \tau_2 \cdot K_\ell \cdot P \cdot (\chi'' + \chi + \chi') \geq 4 \cdot K_\ell \cdot \left(\left|\left|\mathbf{e}_0 \cdot \mathbf{t}_\mathbf{y}^\top\right|\right| + \left|\left|\tilde{\mathbf{e}} \cdot \mathbf{y}^\top\right|\right|\right) \tag{4}$$

the theorem follows. □

3.2 Security

Theorem 3 (Security). *Let Π_1 be the UIPFE construction described in Section 3, with parameters set as in Equation (2), and F be a PRF. Then, for any adversary $\mathcal{A}$ that runs is time $T = T(\lambda)$, there exist adversaries $\mathcal{B}_0$, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ against PRF-security, $\mathsf{LWE}_{n,m,\chi'',q}$, $\mathsf{LWE}_{n,m+\lambda,\chi'',q}$, and $\mathsf{IND\text{-}CPA}_{\mathsf{ALS}[\ell^*]}$, respectively, such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\Pi_1}(\lambda) \leq T \cdot \Big(\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{B}_0,\mathsf{F}}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m,\chi'',q}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m+\lambda,\chi'',q}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{B}_3,\mathsf{ALS}[\ell^*]}(\lambda) + \mathsf{negl}(\lambda)\Big).$$

Proof. Consider the following sequence of hybrids, summarized in Figures 3 and 4. Let $\mathsf{Adv}_i(\mathcal{A})$ denote the advantage of $\mathcal{A}$ in hybrid H_i. Notice that we can bound the length ℓ of any function $\mathbf{y}$ queried to the KeyGen oracle by T, i.e., an adversary $\mathcal{A}$ running in time T will never obtain $\mathbf{k}_{j,\ell}$ or $\mathbf{D}_\ell$ for $\ell > T$.

- H_0: This is identical to the real security game. Therefore,

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\Pi_1}(\lambda) = \mathsf{Adv}_0(\mathcal{A}).$$

- H_1: This is identical to H_0, except for the fact that the reduction guesses $|\mathbf{x}^*| = \ell^*$ before generating the public parameters. If the guess is not correct, the reduction aborts. Since $\mathcal{A}$ runs in time T, one has that $\ell^* \leq T$, so the reduction can guess ℓ^* and incur a security loss of T. Hence, we have that

$$\mathsf{Adv}_0(\mathcal{A}) \leq T \cdot \mathsf{Adv}_1(\mathcal{A}).$$

- H_2: This is identical to H_1, except for the following modification to KeyGen:
  - for all $j \in [T]$, sample once and for all a random string $\boldsymbol{\rho}_j \leftarrow\!\!\$\ \{0,1\}^{\mathsf{poly}(\lambda)}$,
  - parse $\boldsymbol{\rho}_j = [\boldsymbol{\rho}_{1,j} \mid \boldsymbol{\rho}_{2,j}]$,
  - use $\boldsymbol{\rho}_{1,j,\ell}$ as randomness to sample $\mathbf{k}_{j,\ell}$, i.e.,

$$\mathbf{k}_{j,\ell}^\top \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell \cdot \mathbf{G}], \begin{bmatrix}\mathbf{T}_\mathbf{B} \\ \mathbf{0}_{m\times m} \\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell}\right).$$

  and $\boldsymbol{\rho}_{2,\ell}$ as randomness to sample $\mathbf{T}_\ell$, i.e.,

$$\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{D}_\ell, \tau_2; \boldsymbol{\rho}_{2,\ell}\right).$$

To show that $\mathsf{H}_1 \approx_c \mathsf{H}_2$, we rely on the PRF security of F. The reduction works as follows:

- it samples $\mathbf{A}, \mathbf{B}, \mathbf{W}, \mathbf{v}$ and $\mathbf{Y}$ as in H_1,
- it obtains $\mathbf{x}^*$ from the adversary $\mathcal{A}$,
- it answers KeyGen queries as in H_1 but using the outputs of PRF oracles $\mathcal{O}(j, \ell, 1)$ and $\mathcal{O}(\ell, 2)$ as the randomness tosample $\mathbf{k}_{j,\ell}$ and $\mathbf{T}_\ell$, respectively,
- whenever the adversary $\mathcal{A}$ produces $(\mathbf{x}_0, \mathbf{x}_1)$, it produces the challenge ciphertext ct as in H_1.

Observe that
- if $\mathcal{O}(\cdot) = \mathsf{F}(\mathsf{k}, \cdot)$ is a pseudorandom function instance, the view of the adversary $\mathcal{A}$ is identical to H_1;
- if $\mathcal{O}(\cdot) = F(\cdot)$ is a truly random function instance, the view of $\mathcal{A}$ is identical to H_2.

We conclude that

$$\mathsf{Adv}_1(\mathcal{A}) \leq \mathsf{Adv}_2(\mathcal{A}) + \mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{B}_0,\mathsf{F}}(\lambda),$$

and in particular, that $\mathsf{H}_1 \approx_c \mathsf{H}_2$.

For $k \in [\ell^*]$:

– $\mathsf{H}_{3,k,1}$: This is the same as previous hybrid, except for the following modification to $\mathbf{W}$ and $\mathbf{Y}$ in mpk:
- sample $\tilde{\mathbf{W}}_k \leftarrow \{1,-1\}^{m\times m}$ and $\tilde{\mathbf{Y}}_{\ell^*} \leftarrow \{1,-1\}^{m\times m}$,
- set $\mathbf{W} \coloneqq \mathbf{B}\cdot\tilde{\mathbf{W}}_k - k\cdot\mathbf{G}$ and $\mathbf{Y} \coloneqq \mathbf{B}\cdot\tilde{\mathbf{Y}}_{\ell^*} - \ell^*\cdot\mathbf{G}$.

Since $\tilde{\mathbf{W}}_k$ and $\tilde{\mathbf{Y}}_{\ell^*}$ are sampled uniformly and $m \geq (n+1)\cdot\log q + 2\cdot\lambda + \omega(\log n)$, statistical indistinguishability of $\mathsf{H}_{3,k,1}$ from previous hybrid follows from the (plain) leftover hash lemma (Lemma 2, without "leakages"). Notice that, for all $j, \ell \in [T]$, we can now rewrite

$$\mathbf{W} + j\cdot\mathbf{G} = \mathbf{B}\cdot\tilde{\mathbf{W}}_k + (j-k)\cdot\mathbf{G},$$
$$\mathbf{Y} + \ell\cdot\mathbf{G} = \mathbf{B}\cdot\tilde{\mathbf{Y}}_{\ell^*} + (\ell-\ell^*)\cdot\mathbf{G}.$$

– $\mathsf{H}_{3,k,2}$ This is the same as $\mathsf{H}_{3,k,1}$, except for the following modification to $\mathbf{v}$ in mpk and to $\mathbf{k}_{k,\ell^*}$ in KeyGen queries:
- Pre-sample $\mathbf{k}_{k,\ell^*} \leftarrow \mathcal{D}_{\mathbb{Z}^{3m},\tau_1}$ using the random coins $\boldsymbol{\rho}_{1,k,\ell^*}$,
- set $\mathbf{v}^\top \coloneqq [\mathbf{B} \mid \mathbf{W} + k\cdot\mathbf{G} \mid \mathbf{Y} + \ell^*\cdot\mathbf{G}]\cdot\mathbf{k}^\top_{k,\ell^*}$.

By the properties ofthe $\mathsf{SamplePre}$ algorithm (Lemma 3, Step 1), the distributions of $\mathbf{v}$ and $\mathbf{k}_{k,\ell^*}$ are statistically indistinguishable between $\mathsf{H}_{3,k,1}$ and $\mathsf{H}_{3,k,2}$. Therefore,

$$\mathsf{Adv}_{3,k,1}(\mathcal{A}) \leq \mathsf{Adv}_{3,k,2}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– $\mathsf{H}_{3,k,3}$: This is the same as $\mathsf{H}_{3,k,2}$, except for the following modification to KeyGen queries when $\ell \neq \ell^*$ or when $j \neq k$:
- compute $\mathbf{T}_{\mathsf{length}} \coloneqq \begin{bmatrix}\tilde{\mathbf{Y}}_{\ell^*}\\ \mathbf{0}_{m\times m}\\ -\mathbf{I}_m\end{bmatrix}$ and $\mathbf{T}_{\mathsf{index}} \coloneqq \begin{bmatrix}\tilde{\mathbf{W}}_k\\ -\mathbf{I}_m\\ \mathbf{0}_{m\times m}\end{bmatrix}$. Observe that

$$[\mathbf{B} \mid \mathbf{W} + j\cdot\mathbf{G} \mid \mathbf{Y} + \ell\cdot\mathbf{G}]\cdot\mathbf{T}_{\mathsf{length}} = [\mathbf{B} \mid \mathbf{B}\cdot\tilde{\mathbf{W}}_k + (j-k)\cdot\mathbf{G} \mid \mathbf{Y} + (\ell-\ell^*)\cdot\mathbf{G}]\cdot\begin{bmatrix}\tilde{\mathbf{Y}}_{\ell^*}\\ \mathbf{0}_{m\times m}\\ -\mathbf{I}_m\end{bmatrix} = (\ell-\ell^*)\cdot\mathbf{G},$$

$$[\mathbf{B} \mid \mathbf{W} + j\cdot\mathbf{G} \mid \mathbf{Y} + \ell\cdot\mathbf{G}]\cdot\mathbf{T}_{\mathsf{index}} = [\mathbf{B} \mid \mathbf{B}\cdot\tilde{\mathbf{W}}_k + (j-k)\cdot\mathbf{G} \mid \mathbf{Y} + (\ell-\ell^*)\cdot\mathbf{G}]\cdot\begin{bmatrix}\tilde{\mathbf{W}}_k\\ -\mathbf{I}_m\\ \mathbf{0}_{m\times m}\end{bmatrix} = (j-k)\cdot\mathbf{G}.$$

- for $\ell \neq \ell^*$, compute

$$\mathbf{k}^\top_{j,\ell} \leftarrow \mathsf{SamplePre}([\mathbf{B} \mid \mathbf{W} + j\cdot\mathbf{G} \mid \mathbf{Y} + \ell\cdot\mathbf{G}], \mathbf{T}_{\mathsf{length}}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j\ell}),$$

and for $\ell = \ell^*$ and $j \neq k$, compute

$$\mathbf{k}^\top_{j,\ell} \leftarrow \mathsf{SamplePre}([\mathbf{B} \mid \mathbf{W} + j\cdot\mathbf{G} \mid \mathbf{Y} + \ell\cdot\mathbf{G}], \mathbf{T}_{\mathsf{index}}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j\ell})$$

to answer KeyGen queries. This works as long as

$$\tau_1 \geq O(m^2) \geq O\left(m^2\cdot\left\|\tilde{\mathbf{W}}\right\|\right), O\left(m^2\cdot\left\|\tilde{\mathbf{Y}}\right\|\right). \tag{5}$$

Hence, since τ_1 satisfies such constraint by our choice of parameters, we have

$$\mathsf{Adv}_{3,k,2}(\mathcal{A}) \leq \mathsf{Adv}_{3,k,3}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

Notice that the reduction does not use $\mathbf{T_B}$ anymore.

– $\mathsf{H}_{3,k,4}$: This is the same as $\mathsf{H}_{3,k,3}$, except that we modify $\mathbf{B}$ in mpk as:
  • sample $\mathbf{B} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times m}$ instead of $(\mathbf{B}, \mathbf{T_B}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.

By the properties of the $\mathsf{TrapGen}$ algorithm (Lemma 3), the distribution of $\mathbf{B}$ is statistically indistinguishable between $\mathsf{H}_{3,k,3}$ and $\mathsf{H}_{3,k,4}$. Therefore,

$$\mathsf{Adv}_{3,k,3}(\mathcal{A}) \leq \mathsf{Adv}_{3,k,4}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– $\mathsf{H}_{3,k,5}$: This is the same as $\mathsf{H}_{3,k,4}$ except for the following modification to $\mathbf{c}_{1,k}$ in the challenge ciphertext:
  • Set

$$\mathbf{c}_{1,k} \coloneqq [\mathbf{c}_{2,k} \mid \mathbf{c}_{3,k} \mid \mathbf{c}_{4,k}] \cdot \mathbf{k}_{k,\ell^*} + \mathbf{s} \cdot \mathbf{d}_{k,\ell^*} + x_k^* \cdot \left\lfloor \frac{q}{K} \right\rfloor + \tilde{e}_{1,k}$$

  for $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_q^n$, $\tilde{e}_{1,k} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\chi''}$.

First, recall that in $\mathsf{H}_{3,k,4}$, we have

$$\begin{aligned}
\mathbf{c}_{1,k} &= \mathbf{s} \cdot \mathbf{v}^\top + x_k^* \cdot \left\lfloor \frac{q}{K} \right\rfloor + e_{1,k} \\
&= \underbrace{\mathbf{s}_k \cdot [\mathbf{B} \mid \mathbf{W} + k \cdot \mathbf{G} \mid \mathbf{Y} + \ell^* \cdot \mathbf{G}]}_{[\mathbf{c}_{2,k}|\mathbf{c}_{3,k}|\mathbf{c}_{4,k}] + [\mathbf{0}|\mathbf{s}\cdot\mathbf{G}|\mathbf{0}] - [\mathbf{e}_{2,k}|\mathbf{e}_{3,k}|\mathbf{e}_{4,k}]} \cdot \underbrace{\begin{bmatrix} \mathbf{z}_{k,\ell^*}^\top \\ \mathbf{r}_{k,\ell^*}^\top \\ \mathbf{p}_{k,\ell^*}^\top \end{bmatrix}}_{\mathbf{k}_{k,\ell^*}^\top} + x_k^* \cdot \left\lfloor \frac{q}{K} \right\rfloor + e_{1,k} \\
&= [\mathbf{c}_{2,k} \mid \mathbf{c}_{3,k} \mid \mathbf{c}_{4,k}] \cdot \mathbf{k}_{k,\ell^*}^\top + \mathbf{s} \cdot \mathbf{d}_{k,\ell^*}^\top + x_k^* \cdot \left\lfloor \frac{q}{K} \right\rfloor + \boxed{e_{1,k} - [\mathbf{e}_{2,k} \mid \mathbf{e}_{3,k} \mid \mathbf{e}_{4,k}] \cdot \mathbf{k}_{k,\ell^*}^\top} \bmod q
\end{aligned}$$

where in the last equality we have used that $\mathbf{d}_{k,\ell^*}^\top = \mathbf{G} \cdot \mathbf{r}_{k,\ell^*}^\top$ and the boxed term is the term in $\mathsf{H}_{3,k,4}$ that will be modified in $\mathsf{H}_{3,k,5}$. By noise flooding (Lemma 1), we have

$$\left([\mathbf{e}_{2,k} \mid \mathbf{e}_{3,k} \mid \mathbf{e}_{4,k}], \mathbf{k}_{k,\ell^*}^\top, \boxed{e_{1,k} - [\mathbf{e}_{2,k} \mid \mathbf{e}_{3,k} \mid \mathbf{e}_{4,k}] \cdot \mathbf{k}_{k,\ell^*}^\top}\right) \approx_s \left([\mathbf{e}_{2,k} \mid \mathbf{e}_{3,k} \mid \mathbf{e}_{4,k}], \mathbf{k}_{k,\ell^*}^\top, \tilde{e}_{1,k}\right),$$

as long as

$$\begin{aligned}
\chi'' &\geq 3 \cdot m \cdot \max\{\chi, \chi'\} \cdot \tau_1 \cdot \lambda^{\omega(1)}, \\
&\geq \left|\left|[\mathbf{e}_{2,k} \mid \mathbf{e}_{3,k} \mid \mathbf{e}_{4,k}] \cdot \mathbf{k}_{k,\ell^*}^\top\right|\right| \cdot \lambda^{\omega(1)}.
\end{aligned}$$

Hence, we conclude that

$$\mathsf{Adv}_{3,k,4}(\mathcal{A}) \leq \mathsf{Adv}_{3,k,5}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– $\mathsf{H}_{3,k,6}$: This is the same as $\mathsf{H}_{3,k,5}$, except for the following modification to $\mathbf{c}_{3,k}$ and $\mathbf{c}_{4,k}$ in the challenge ciphertext:
  • set

$$\begin{aligned}
\mathbf{c}_{3,k} &\coloneqq \mathbf{c}_{2,k} \cdot \tilde{\mathbf{W}}_k - \mathbf{s} \cdot \mathbf{G} + \tilde{\mathbf{e}}_{3,k}, \\
\mathbf{c}_{4,k} &\coloneqq \mathbf{c}_{2,k} \cdot \tilde{\mathbf{Y}}_{\ell^*} + \tilde{\mathbf{e}}_{4,k},
\end{aligned}$$

  for $\tilde{\mathbf{e}}_{3,k} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi'} \tilde{\mathbf{e}}_{4,k} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi'}$ and $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_q^n$.

First, recall that in $\mathsf{H}_{3,k,5}$, we have

$$\begin{aligned}\mathbf{c}_{3,k} &= \mathbf{s}_k \cdot (\mathbf{W} + k \cdot \mathbf{G}) - \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,k} \\ &= \mathbf{s}_k \cdot (\mathbf{B} \cdot \tilde{\mathbf{W}}_k) - \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,k} \\ &= \underbrace{(\mathbf{s}_k \cdot \mathbf{B} + \mathbf{e}_{2,k})}_{\mathbf{c}_{2,k}} \cdot \tilde{\mathbf{W}}_k - \mathbf{s} \cdot \mathbf{G} + \boxed{\mathbf{e}_{3,k} - \mathbf{e}_{2,k} \cdot \tilde{\mathbf{W}}_k} \bmod q\end{aligned}$$

and

$$\begin{aligned}\mathbf{c}_{4,k} &= \mathbf{s}_k \cdot (\mathbf{Y} + \ell^* \cdot \mathbf{G}) + \mathbf{e}_{4,k} \\ &= \mathbf{s}_k \cdot (\mathbf{B} \cdot \tilde{\mathbf{Y}}_{\ell^*}) + \mathbf{e}_{4,k} \\ &= \underbrace{(\mathbf{s}_k \cdot \mathbf{B} + \mathbf{e}_{2,k})}_{\mathbf{c}_{2,k}} \cdot \tilde{\mathbf{Y}}_{\ell^*} + \boxed{\mathbf{e}_{4,k} - \mathbf{e}_{2,k} \cdot \tilde{\mathbf{Y}}_{\ell^*}} \bmod q\end{aligned}$$

where the boxed terms is the term in $\mathsf{H}_{3,k,5}$ that will be modified in $\mathsf{H}_{3,k,6}$. By noise flooding (Lemma 1), we have

$$\left(\mathbf{e}_{2,k}, \tilde{\mathbf{W}}_k, \tilde{\mathbf{Y}}_{\ell^*}, \boxed{\mathbf{e}_{3,k} - \mathbf{e}_{2,k} \cdot \tilde{\mathbf{W}}_k}, \boxed{\mathbf{e}_{4,k} - \mathbf{e}_{2,k} \cdot \tilde{\mathbf{Y}}_{\ell^*}}\right) \approx_s \left(\mathbf{e}_{2,k}, \tilde{\mathbf{W}}_k, \tilde{\mathbf{Y}}_{\ell^*}, \tilde{\mathbf{e}}_{3,k}, \tilde{\mathbf{e}}_{4,k}\right),$$

as long as

$$\begin{aligned}\chi' &\geq m \cdot \chi \cdot \lambda^{\omega(1)} \\ &\geq \left|\left|\mathbf{e}_{2,k} \cdot \tilde{\mathbf{W}}_k\right|\right| \cdot \lambda^{\omega(1)}, \left|\left|\mathbf{e}_{2,k} \cdot \tilde{\mathbf{Y}}_{\ell^*}\right|\right| \cdot \lambda^{\omega(1)}.\end{aligned}$$

We conclude that

$$\mathsf{Adv}_{3,k,5}(\mathcal{A}) \leq \mathsf{Adv}_{3,k,6}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– $\mathsf{H}_{3,k,7}$: This is the same as $\mathsf{H}_{3,k,6}$, except for the following modification to $\mathbf{c}_{2,k}$ in the challenge ciphertext:
  - sample

$$\mathbf{c}_{2,k} \leftarrow\!\!\$\ \mathbb{Z}_q^m.$$

Recall that in $\mathsf{H}_{3,k,6}$, we have

$$\mathbf{c}_{2,k} = \mathbf{s}_k \cdot \mathbf{B} + \mathbf{e}_{2,k},$$

where $\mathbf{s}_k \leftarrow \mathbb{Z}_q^n$, $\mathbf{e}_{2,k} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi}$. To show that $\mathsf{H}_{3,k,6} \approx_c \mathsf{H}_{3,k,7}$, we rely on $\mathsf{LWE}_{n,m,\chi,q}$. The reduction works as follows:

- it parses $\mathbf{B} = \mathbf{B} \in \mathbb{Z}_q^{n \times m}$ and $\tilde{\mathbf{c}} = \mathbf{c}_{2,k} \in \mathbb{Z}_q^m$ from the $\mathsf{LWE}_{n,m,\chi,q}$ instance,
- it samples $\tilde{\mathbf{W}}_k, \tilde{\mathbf{Y}}_{\ell^*} \leftarrow\!\!\$\ \{1,-1\}^{m \times m}$ and computes $\mathbf{A}, \mathbf{W}, \mathbf{Y}, \mathbf{v}$ as in $\mathsf{H}_{3,k,6}$, while using $\mathbf{B}$ obtained from the LWE instance,
- it answers KeyGen queries using $\mathbf{T}_{\mathsf{length}}$ or $\mathbf{T}_{\mathsf{index}}$ (which can be computed from $\tilde{\mathbf{W}}_k$ and $\tilde{\mathbf{Y}}_{\ell^*}$ respectively) as in $\mathsf{H}_{3,k,6}$,
- whenever the adversary $\mathcal{A}$ outputs $(\mathbf{x}_0, \mathbf{x}_1)$, it samples

$$\begin{gathered}b \leftarrow \{0,1\}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi''}, \mathbf{e}_4 \leftarrow \mathcal{D}_{\mathbb{Z}^\lambda,\chi'}, \mathbf{e}_5 \leftarrow \mathcal{D}_{\mathbb{Z}^\lambda,\chi_5} \\ \mathbf{c}_{2,j}, \mathbf{c}_{3,j}, \mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m \quad \text{for } j \in [k-1], \\ \tilde{e}_{1,k} \leftarrow \mathcal{D}_{\mathbb{Z},\chi''}, \tilde{\mathbf{e}}_{3,k} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi'}, \tilde{\mathbf{e}}_{4,k} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi'} \text{ and} \\ \mathbf{s}_j \leftarrow \mathbb{Z}_q^n, e_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi''}, \mathbf{e}_{2,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi}, \mathbf{e}_{3,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi'}, \mathbf{e}_{4,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi'} \text{ for } j \in [k+1:\ell^*]\end{gathered}$$

and outputs

$$\mathsf{ct} = \left(\begin{array}{c} \mathbf{s}\cdot\mathbf{A}+\mathbf{e}_0 \\ \{[\mathbf{c}_{2,j} \mid \mathbf{c}_{3,j} \mid \mathbf{c}_{4,j}]\cdot\mathbf{k}_{j,\ell^*}^\top - \mathbf{s}\cdot\mathbf{d}_{j,\ell^*}^\top + x_{j,b}^*\lfloor\frac{q}{K}\rfloor + \tilde{e}_{1,j}\}_{j\in[k]}, \quad \{\mathbf{s}_j\cdot\mathbf{v}^\top + x_{j,b}^*\lfloor\frac{q}{K}\rfloor + e_{1,j}\}_{j\in[k+1:\ell^*]} \\ \{\mathbf{c}_{2,j}\}_{j\in[k]}, \quad \{\mathbf{s}_j\cdot\mathbf{B}+\mathbf{e}_{2,j}\}_{j\in[k+1:\ell^*]} \\ \{\mathbf{c}_{3,j}\}_{j\in[k-1]}, \quad \mathbf{c}_{2,k}\cdot\tilde{\mathbf{W}}_k + k\cdot\mathbf{s}\mathbf{G}+\tilde{\mathbf{e}}_{3,k}, \quad \{\mathbf{s}_j\cdot(\mathbf{W}+j\cdot\mathbf{G})-\mathbf{s}\cdot\mathbf{G}+\mathbf{e}_{3,j}\}_{j\in[k+1:\ell^*]} \\ \{\mathbf{c}_{4,j}\}_{j\in[k-1]}, \quad \mathbf{c}_{2,k}\cdot\tilde{\mathbf{Y}}_{\ell^*}+\tilde{\mathbf{e}}_{4,k}, \quad \{\mathbf{s}_j\cdot(\mathbf{Y}+\ell^*\cdot\mathbf{G})+\mathbf{e}_{4,j}\}_{j\in[k+1:\ell^*]} \end{array} \right)$$

Observe that

- if $(\mathbf{B},\tilde{\mathbf{c}})$ is a structured $\mathsf{LWE}_{n,m,\chi,q}$ instance, the view of the adversary $\mathcal{A}$ is identical to $\mathsf{H}_{3,k,6}$;
- if $(\mathbf{B},\tilde{\mathbf{c}})$ is a uniform random instance, the view of $\mathcal{A}$ is identical to $\mathsf{H}_{3,k,7}$.

We conclude that

$$\mathsf{Adv}_{3,k,6}(\mathcal{A}) \le \mathsf{Adv}_{3,k,7}(\mathcal{A}) + \mathsf{Adv}^{\mathsf{LWE}_{n,m,\chi,q}}_{\mathcal{B}_1}(\lambda).$$

– $\mathsf{H}_{3,k,8}$: This is the same as $\mathsf{H}_{3,k,7}$, except for the following modification to $\mathbf{c}_{3,k}$ and $\mathbf{c}_{4,k}$ in the challenge ciphertext:

- sample

$$\mathbf{c}_{3,k},\mathbf{c}_{4,k} \leftarrow\!\$\ \mathbb{Z}_q^m.$$

Recall that in $\mathsf{H}_{3,k,7}$, we have

$$\mathbf{c}_{3,k} = \mathbf{c}_{2,k}\cdot\tilde{\mathbf{W}}_k - \mathbf{s}\cdot\mathbf{G}+\tilde{\mathbf{e}}_{3,k},$$
$$\mathbf{c}_{4,k} = \mathbf{c}_{2,k}\cdot\tilde{\mathbf{Y}}_{\ell^*}+\tilde{\mathbf{e}}_{4,k}.$$

Since $\mathbf{c}_{2,k}$ is uniform random in $\mathbb{Z}_q^m$ in $\mathsf{H}_{3,k,7}$, $\tilde{\mathbf{W}}_k$ and $\tilde{\mathbf{Y}}_{\ell^*}$ are sampled uniformly in $\{1,-1\}^{m\times m}$, and $m \ge (n+1)\cdot\log q + 2\cdot\lambda + \omega(\log n)$, indistinguishability ($\mathsf{H}_{3,k,7} \approx_s \mathsf{H}_{3,k,8}$) follows from the generalized leftover hash lemma (the adversary's view includes $\mathbf{W} = \mathbf{B}\cdot\tilde{\mathbf{W}}_k - k\cdot\mathbf{G}$ and $\mathbf{Y} = \mathbf{B}\cdot\tilde{\mathbf{Y}}_{\ell^*} - \ell^*\cdot\mathbf{G}$), that is

$$\mathsf{Adv}_{3,k,7}(\mathcal{A}) \le \mathsf{Adv}_{3,k,8}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– $\mathsf{H}_{3,k,9}$: This is the same as $\mathsf{H}_{3,k,8}$, except for the following modification to $\mathbf{B}$ in mpk and to the KeyGen queries:

- sample $(\mathbf{B},\mathbf{T}_\mathbf{B}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n,1^m,q)$, instead of $\mathbf{B} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$,
- compute

$$\mathbf{k}_{j,\ell}^\top \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{W}+j\cdot\mathbf{G} \mid \mathbf{Y}+\ell\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T}_\mathbf{B}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell}\right),$$

to answer KeyGen queries.

By the properties ofthe $\mathsf{TrapGen}$ algorithm (Lemma 3) and that of the $\mathsf{SamplePre}$ algorithm, the distribution of $\mathbf{B}$ and that of answers to the KeyGen queries are statistically indistinguishable between $\mathsf{H}_{3,k,8}$ and $\mathsf{H}_{3,k,9}$. Therefore,

$$\mathsf{Adv}_{3,k,8}(\mathcal{A}) \le \mathsf{Adv}_{3,k,9}(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– H_4: This is the same as $\mathsf{H}_{3,\ell^*,9}$, except for the following modification to $\mathbf{k}_{j,\ell}$ for all $j,\ell\in[T]$, and to the relative KeyGen queries:

- sample, once and for all key queries, $\mathbf{r}_{j,\ell} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\tau_1}$,
- compute

$$\begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{Y}+\ell\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T}_\mathbf{B}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top - [\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\right),$$

- set $\mathbf{k}_{j,\ell}^\top := \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix}$.

By the properties ofthe $\mathsf{SamplePre}$ algorithm and Lemma 4, the distribution of $\{\mathbf{k}_{j,\ell}\}_{j,\ell\in[T]}$ is statistically indistinguishable between $\mathsf{H}_{3,\ell^*,9}$ and H_4. Therefore,

$$\mathsf{Adv}_{3,\ell^*,9}(\mathcal{A}) \leq \mathsf{Adv}_4(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– H_5: This is the same as H_4, except for the following modification to $\mathbf{D}_\ell, \mathbf{k}_{j,\ell}$ for all $j, \ell \in [T]$, and to the relative KeyGen queries:
- sample $\mathbf{D}_\ell = [\mathbf{d}_{1,\ell}^\top \mid \ldots \mid \mathbf{d}_{\ell,\ell}^\top] \leftarrow\!\$\ \mathbb{Z}_q^{n\times\ell}$,
- set $\mathbf{r}_{j,\ell}^\top = \mathsf{SamplePre}(\mathbf{G}, \mathbf{I}, \mathbf{d}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,2})$,
- compute

$$\begin{bmatrix}\mathbf{z}_{j,\ell}^\top \\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{Y} + \ell\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T}_\mathbf{B} \\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top - [\mathbf{W} + j\cdot\mathbf{G}]\cdot\mathbf{r}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\right),$$

- set $\mathbf{k}_{j,\ell}^\top := \begin{bmatrix}\mathbf{z}_{j,\ell}^\top \\ \mathbf{r}_{j,\ell}^\top \\ \mathbf{p}_{j,\ell}^\top\end{bmatrix}$.

By the properties ofthe $\mathsf{SamplePre}$ algorithm (Lemma 3, Step 1), the distribution of $\{\mathbf{D}_\ell, \mathbf{k}_{j,\ell}\}_{j,\ell\in[T]}$ is statistically indistinguishable between H_4 and H_5. Therefore,

$$\mathsf{Adv}_4(\mathcal{A}) \leq \mathsf{Adv}_5(\mathcal{A}) + \mathsf{negl}(\lambda).$$

– H_6: This is the same as H_5, except for the following modification to $\mathbf{T}_\ell$ and $\mathbf{D}_\ell$ for all $j \in [T]$:
- sample $\mathbf{T}_\ell \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^{m\times\ell},\tau_2}$,
- set $\mathbf{D}_\ell = \mathbf{A}\cdot\mathbf{T}_\ell$.

By the properties ofthe $\mathsf{SamplePre}$ algorithm (Lemma 3, Step 1), the distribution of $\mathbf{D}_\ell$ and $\mathbf{T}_{\ell^*}$ is statistically indistinguishable between H_5 and H_6. Therefore,

$$\mathsf{Adv}_5(\mathcal{A}) \leq \mathsf{Adv}_6(\mathcal{A}) + \mathsf{negl}(\lambda).$$

Notice that the reduction does not use $\mathbf{T}_\mathbf{A}$ anymore.

– H_7: This is the same as H_6, except that we modify $\mathbf{A}$ in mpk as:
- sample $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ instead of $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)$.

By the properties of the $\mathsf{TrapGen}$ algorithm (Lemma 3), the distribution of $\mathbf{A}$ is statistically indistinguishable between H_6 and H_7. Therefore,

$$\mathsf{Adv}_6(\mathcal{A}) \leq \mathsf{Adv}_7(\mathcal{A}) + \mathsf{negl}(\lambda).$$

Finally, we show that

$$\mathsf{Adv}_7(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{B}_3,\mathsf{ALS}[\ell^*]}(\lambda),$$

where the ALS scheme that we reduce to uses Gaussian parameter χ''. The reduction works as follows:

– it receives $\mathsf{mpk}_\mathsf{ALS} = (\mathbf{A}_\mathsf{ALS}, \mathbf{D}_\mathsf{ALS}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{n\times\ell^*}$ from the ALS challenger,
– it samples $\mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v}$ as in H_7, and sets $\mathbf{A} := \mathbf{A}_\mathsf{ALS}$,
– it samples $\{\mathbf{T}_\ell\}_{\ell\in[T],\ell\neq\ell^*}$ and implicitly sets $\{\mathbf{D}_\ell\}_{j\in[T],\ell\neq\ell^*}$ as in H_7, while using $\mathbf{D}_{\ell^*} := \mathbf{D}_\mathsf{ALS}$ obtained from the ALS challenger,
– it samples $\mathbf{k}_{j,\ell}$ for all $j, \ell \in [T]$ as in H_7,
– it answers KeyGen queries for $\mathbf{y} \in \{0, \ldots, P-1\}^{\ell'}$ using $\mathbf{T}_{\ell'}$ when $\ell' \neq \ell^*$. For $\ell' = \ell^*$, it forwards the query to the ALS challenger, receives back $\mathsf{sk}_\mathbf{y}$, and forwards that to the adversary,

– whenever the adversary $\mathcal{A}$ outputs $(\mathbf{x}_0, \mathbf{x}_1)$, it forwards that to the ALS challenger and receives back

$$\mathbf{c}_{0,\mathsf{ALS}} \in \mathbb{Z}_q^m, \mathbf{c}_{1,\mathsf{ALS}} \in \mathbb{Z}_q^{\ell^*}.$$

It samples

$$\mathbf{c}_{2,j}, \mathbf{c}_{3,j}, \mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m \quad \text{for } j \in [\ell^*],$$

and outputs

$$\mathsf{ct} \coloneqq \begin{pmatrix} \mathbf{c}_{0,\mathsf{ALS}} \\ \{[\mathbf{c}_{2,j} \mid \mathbf{c}_{3,j} \mid \mathbf{c}_{4,j}] \cdot \mathbf{k}_{j,\ell^*}^\top + \mathbf{c}_{1,j,\mathsf{ALS}}\}_{j\in[\ell^*]}, \\ \{\mathbf{c}_{2,j}\}_{j\in[\ell^*]}, \\ \{\mathbf{c}_{3,j}\}_{j\in[\ell^*]}, \\ \{\mathbf{c}_{4,j}\}_{j\in[\ell^*]} \end{pmatrix}.$$

Whenever the adversary outputs a bit $b' \in \{0,1\}$, the reduction forwards it to the ALS challenger. Putting everything together, we obtain

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\Pi_1}(\lambda) \le T \cdot \left(\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{B}_0,\mathsf{F}}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m,\chi'',q}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m+\lambda,\chi'',q}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{B}_3,\mathsf{ALS}[\ell^*]}(\lambda) + \mathsf{negl}(\lambda)\right),$$

as claimed. □

4 Permissive Unbounded IPFE

As before, we refer to Section 1.4 for an overview of the scheme and the security proof. We highlight the changes compared to the strict construction and relative security proof from Section 3 with a blue box.

4.1 Construction

We consider that inner-products are evaluated over $\mathbb{Z}$, and message space $\mathcal{P}$, secret key vector space $\mathcal{V}$ and output space $\{0, \ldots, K_\ell - 1\}$ are defined as in Section 3.1.

Let $\Pi_2 = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ be an UIPFE scheme defined as follows:

– $\mathsf{Setup}(1^\lambda, \mathcal{P}, \mathcal{V})$: Sample

$$(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow\$ \mathsf{TrapGen}(1^n, 1^m, q), (\mathbf{B}, \mathbf{T}_\mathbf{B}) \leftarrow\$ \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{W}, \mathbf{Y} \leftarrow\$ \mathbb{Z}_q^{n\times m}, \mathbf{v} \leftarrow\$ \mathbb{Z}_q^n$$
$$\mathsf{k} \leftarrow\$ \{0,1\}^\lambda.$$

Set $\mathsf{mpk} = (\mathbf{A}, \mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v})$, and $\mathsf{msk} = (\mathbf{T}_\mathbf{A}, \mathbf{T}_\mathbf{B}, \mathsf{k})$. Return $(\mathsf{mpk}, \mathsf{msk})$.

– $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x} \in \mathcal{V})$: Let $\ell = |\mathbf{x}|$. Sample

$$\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi''},$$
$$\mathbf{s}_{i,j} \leftarrow \mathbb{Z}_q^n, e_{1,i,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi''}, \mathbf{e}_{2,i,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi}, \mathbf{e}_{3,i,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi'}, \mathbf{e}_{4,i,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\chi'} \quad \text{for all } i \in [\ell],\ j \in [i].$$

For each $i \in [\ell]$, compute

$$\begin{aligned}
\mathbf{c}_0 &\coloneqq \mathbf{s} \cdot \mathbf{A} + \mathbf{e}_0 \bmod q, \\
\mathbf{c}_{1,i,j} &\coloneqq \mathbf{s}_{i,j} \cdot \mathbf{v} + x_j \cdot \left\lfloor \frac{q}{K_i} \right\rfloor + e_{1,i,j} \bmod q \quad \text{for all } j \in [i], \\
\mathbf{c}_{2,i,j} &\coloneqq \mathbf{s}_{i,j} \cdot \mathbf{B} + \mathbf{e}_{2,i,j} \bmod q \quad \text{for all } j \in [i], \\
\mathbf{c}_{3,i,j} &\coloneqq \mathbf{s}_{i,j} \cdot (\mathbf{W} + j \cdot \mathbf{G}) - \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,i,j} \bmod q \quad \text{for all } j \in [i], \\
\mathbf{c}_{4,i,j} &\coloneqq \mathbf{s}_{i,j} \cdot (\mathbf{Y} + i \cdot \mathbf{G}) + \mathbf{e}_{4,i,j} \bmod q \quad \text{for all } j \in [i].
\end{aligned}$$

Return $\mathsf{ct}_\mathbf{x} \coloneqq (\mathbf{c}_0, \{\mathbf{c}_{1,i,j}\}_{i\in[\ell],j\in[i]}, \{\mathbf{c}_{2,i,j}\}_{i\in[\ell],j\in[i]}, \{\mathbf{c}_{3,i,j}\}_{i\in[\ell],j\in[i]}, \{\mathbf{c}_{4,i,j}\}_{i\in[\ell],j\in[i]})$.[17]

[17] Notice that, for every $\ell' \le \ell$, each $(\mathbf{c}_0, \{\mathbf{c}_{1,\ell',j}\}_{j\in[\ell']}, \{\mathbf{c}_{2,\ell',j}\}_{j\in[\ell']}, \{\mathbf{c}_{3,\ell',j}\}_{j\in[\ell']}, \{\mathbf{c}_{4,\ell',j}\}_{j\in[\ell']})$ is a Π_1 encryption of message $[x_1, \ldots, x_{\ell'}]$.

– $\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y} \in \mathcal{P})$: Let $\ell' = |\mathbf{y}|$. For all $j \in [\ell']$, sample

$$\mathbf{k}_{j,\ell'}^\top \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{W} + j \cdot \mathbf{G} \mid \mathbf{Y} + \ell' \cdot \mathbf{G}], \begin{bmatrix} \mathbf{T_B} \\ \mathbf{0}_{m\times m} \\ \mathbf{0}_{m\times m} \end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k}, j, \ell', 1)\right).$$

Parse

$$\mathbf{k}_{j,\ell'}^\top = \begin{bmatrix} \mathbf{z}_{j,\ell'}^\top \\ \mathbf{r}_{j,\ell'}^\top \\ \mathbf{p}_{j,\ell'}^\top \end{bmatrix},$$

and let $\mathbf{d}_{j,\ell'}^\top \coloneqq \mathbf{G} \cdot \mathbf{r}_{j,\ell'}^\top \bmod q$. Let $\mathbf{D}_{\ell'} \coloneqq [\mathbf{d}_{1,\ell'}^\top | \ldots | \mathbf{d}_{\ell',\ell'}^\top]$. Sample

$$\mathbf{T}_{\ell'} \leftarrow \mathsf{SamplePre}\Big(\mathbf{A}, \mathbf{T_A}, \mathbf{D}_{\ell'}, \tau_2; \mathsf{F}(\mathsf{k}, \ell', 2)\Big),$$

Set $\mathbf{t}_\mathbf{y}^\top = \mathbf{T}_{\ell'} \cdot \mathbf{y}^\top$. Return $\mathsf{sk}_\mathbf{y} \coloneqq (\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y})$.

– $\mathsf{Dec}(\mathsf{sk}_\mathbf{y}, \mathbf{y}, \mathsf{ct}_\mathbf{x})$: Parse $\mathsf{ct}_\mathbf{x} = (\mathbf{c}_0, \{\mathbf{c}_{1,i,j}\}_{i\in[\ell],j\in[i]}, \{\mathbf{c}_{2,i,j}\}_{i\in[\ell],j\in[i]}, \{\mathbf{c}_{3,i,j}\}_{i\in[\ell],j\in[i]}, \{\mathbf{c}_{4,i,j}\}_{i\in[\ell],j\in[i]})$, $\mathsf{sk}_\mathbf{y} = (\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y})$. If $\ell' > \ell$, return $\perp$. Else, compute

$$\mu' \coloneqq \left(\Big[\mathbf{c}_{1,\ell',1} \,\Big|\, \ldots \,\Big|\, \mathbf{c}_{1,\ell',\ell'}\Big] - \Big[\big[\mathbf{c}_{2,\ell',1} \,|\, \mathbf{c}_{3,\ell',1} \,|\, \mathbf{c}_{4,\ell',1}\big] \cdot \mathbf{k}_{1,\ell'}^\top \,\Big|\, \ldots \,\Big|\, \big[\mathbf{c}_{2,\ell',\ell'} \,|\, \mathbf{c}_{3,\ell',\ell'}^\top \,|\, \mathbf{c}_{4,\ell',\ell'}\big] \cdot \mathbf{k}_{\ell',\ell'}^\top\Big]\right) \cdot \mathbf{y}^\top - \Big[\mathbf{c}_0\Big] \cdot \mathbf{t}_\mathbf{y}^\top,$$

and return the value $\mu \in \{-K_\ell + 1, \ldots, K_\ell - 1\}$ that minimizes $|\lfloor \frac{q}{K} \rfloor \cdot \mu - \mu'|$.

Parameters. The parameters can be set exactly as in Equation (2).

Theorem 4 (Correctness). *Let Π_2 be the UIPFE construction described in Section 4, with parameters as in Equation (2). Then, Π_2 is correct.*

Proof (sketch). Correctness follows exactly as in Theorem 2, since each Π_2 ciphertext of some message $\mathbf{x}$ of length ℓ consists of ℓ Π_1-ciphertexts corresponding to messages $[x_1], [x_1, x_2], \ldots, [x_1, \ldots, x_\ell] = \mathbf{x}$, respectively. Moreover, decryption using some functional key $\mathsf{sk}_\mathbf{y}$ for vector $\mathbf{y}$ of length $\ell' \leq \ell$ works exactly as in Π_1 using the ciphertext component corresponding to the Π_1 encryption of $[x_1, \ldots, x_{\ell'}]$. □

4.2 Security

The security proof follows closely the proof of the strict unbounded construction (Section 3.1) given in Section 3.2, except that

– between H_2 and H_4, we run through hybrids $\mathsf{H}_{3,h,k,1}, \ldots, \mathsf{H}_{3,h,k,9}$ for $h \in [2, \ell^*]$ and $k \in [2, h]$. The variable h is used to index the run through each of the ℓ^* "Π_1-encryptions" (see Footnote 17). Comparing with to the proof of Theorem 3, there the variable h was absent, or could to be thought as fixed to ℓ^*, the length of the challenge message.
After each of these hybrids, the challenge ciphertext components $(\mathbf{c}_{2,h,k}, \mathbf{c}_{3,h,k}, \mathbf{c}_{4,h,k})$ are changed to random, and $\mathbf{c}_{1,h,k}$ appropriately modified to guaranteed correctness.
– the reduction to the ALS challenger is modified as follows:
  • the message length of the ALS scheme it reduces to is $\ell^*(\ell^* + 1)/2$ (instead of just ℓ^* as before),
  • given $\mathsf{mpk}_\mathsf{ALS} = (\mathbf{A}_\mathsf{ALS}, \mathbf{D}_\mathsf{ALS}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{n\times \ell^*(\ell^*+1)/2}$, it parses $\mathbf{D}_\mathsf{ALS} = [\mathbf{D}_{1,\mathsf{ALS}} \mid \ldots \mid \mathbf{D}_{\ell^*,\mathsf{ALS}}]$, with each $\mathbf{D}_{i,\mathsf{ALS}} \in \mathbb{Z}_q^{n\times i}$, for $i \in [\ell^*]$, and sets $\mathbf{A} \coloneqq \mathbf{A}_\mathsf{ALS}$ and $\mathbf{D}_i \coloneqq \mathbf{D}_{i,\mathsf{ALS}}$ for each $i \in [\ell^*]$,

Hybrid	mpk	ct	$\mathsf{sk}_{\mathbf{y}}$	justification
H_0	$(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)$ $(\mathbf{B}, \mathbf{T}_{\mathbf{B}}) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)$ $\mathbf{W} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{Y} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{v} \leftarrow\!\!\$\ \mathbb{Z}_q^{n}$ $\mathsf{k} \leftarrow \mathcal{K}$	$\mathbf{c}_0 \approx \mathbf{s}\cdot\mathbf{A}$ $\mathbf{c}_{1,i,j} \approx \mathbf{s}_{i,j}\cdot\mathbf{v} + x^*_j\cdot\lfloor\frac{q}{K}\rfloor$ $\mathbf{c}_{2,i,j} \approx \mathbf{s}_{i,j}\cdot\mathbf{B}$ $\mathbf{c}_{3,i,j} \approx \mathbf{s}_{i,j}\cdot(\mathbf{W}+j\cdot\mathbf{G}) - \mathbf{s}\cdot\mathbf{G}$ $\mathbf{c}_{4,i,j} \approx \mathbf{s}_{i,j}\cdot(\mathbf{Y}+i\cdot\mathbf{G})$	$\mathbf{T}_{\mathbf{A}}, \mathbf{T}_{\mathbf{B}}, \mathsf{k}$ $\mathbf{k}^\top_{j,\ell} = \begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{r}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix} \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{W}+j\cdot\mathbf{G}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\\ \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k}, j, \ell, 1)\end{bmatrix}\right)$ $\mathbf{D}_\ell = [\mathbf{G}\cdot\mathbf{r}^\top_{1,\ell}\vert\ldots\vert\mathbf{G}\cdot\mathbf{r}^\top_{\ell,\ell}]$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{D}_\ell, \tau_2; \mathsf{F}(\mathsf{k}, \ell, 2)\right)$ $\mathbf{t}^\top_{\mathbf{y}} = \mathbf{T}_\ell\cdot\mathbf{y}^\top$	
H_1	$\downarrow$	$\downarrow$	$\downarrow$	guess $\vert\mathbf{x}^*\vert = \ell^*$
H_2	$\downarrow$	$\downarrow$	$\boldsymbol{\rho} \leftarrow\!\!\$\ \{0,1\}^{\mathsf{poly}(\lambda)}$ $\mathbf{k}^\top_{j,\ell} \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{W}+j\cdot\mathbf{G}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\\ \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell}\end{bmatrix}\right)$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{D}_\ell, \tau_2; \boldsymbol{\rho}_{2,\ell}\right)$	PRF security
H_3	$\downarrow$	$\mathbf{c}_{2,i,j}, \mathbf{c}_{3,i,j}, \mathbf{c}_{4,i,j} \leftarrow \mathbb{Z}_q^m$ $\mathbf{c}_{1,i,j} \approx [\mathbf{c}_{2,i,j} \mid \mathbf{c}_{3,i,j} \mid \mathbf{c}^\top_{4,i,j}]\cdot\mathbf{k}^\top_{j,i}$ $+\mathbf{s}\cdot\mathbf{d}_{j,i} + x^*_j\cdot\lfloor\frac{q}{K}\rfloor$	$\downarrow$	$\mathsf{LWE}_{n,m,\chi'',q}$ (Figure 4)
H_4	$\downarrow$	$\downarrow$	$\mathbf{r}_{j,\ell} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m,\tau_1}$ $\begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}\mathbf{B}\vert\mathbf{Y}+\ell\cdot\mathbf{G}\end{bmatrix}, \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top - [\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}^\top_{j,\ell}, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\right)$ $\mathbf{k}^\top_{j,\ell} = \begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{r}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix}$	Lemma 4
H_5	$\downarrow$	$\downarrow$	$\mathbf{D}_\ell = [\mathbf{d}^\top_{1,\ell}\vert\ldots\vert\mathbf{d}^\top_{\ell,\ell}] \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times\ell}$ $\mathbf{r}^\top_{j,\ell} = \mathsf{SamplePre}(\mathbf{G}, \mathbf{I}, \mathbf{d}^\top_{j,\ell}, \tau_1; \boldsymbol{\rho}_{1,j,\ell,2})$ $\begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}\mathbf{B}\vert\mathbf{Y}+\ell\cdot\mathbf{G}\end{bmatrix}, \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top - [\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}^\top_{j}, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\right)$ $\mathbf{k}^\top_{j,\ell} = \begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{r}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix}$	Lemma 3 (Item 1)
H_6	$\downarrow$	$\downarrow$	$\mathbf{T}_\ell \leftarrow \mathcal{D}_{\mathbb{Z}^{m\times\ell},\tau_2}$ $\mathbf{D}_\ell = \mathbf{A}\cdot\mathbf{T}_\ell$	Lemma 2 LHL
H_7	$\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$	$\downarrow$	$\downarrow$	Lemma 3 TrapGen
$\mathsf{ALS}[\ell^*(\ell^*+1)/2]$ reduction	$\mathbf{A} = \mathbf{A}_{\mathsf{ALS}}$ $[\mathbf{D}_1\vert\ldots\vert\mathbf{D}_{\ell^*}] = \mathbf{D}_{\mathsf{ALS}}$	$\mathbf{c}_0 \approx \mathbf{c}_{0,\mathsf{ALS}}$ $\mathbf{c}_{1,i,j} \approx [\mathbf{c}_{2,i,j} \mid \mathbf{c}_{3,i,j} \mid \mathbf{c}_{4,i,j}]\cdot\mathbf{k}^\top_{j,i}$ $+\mathbf{c}_{1,i,j,\mathsf{ALS}}$	$\mathbf{t}^\top_{\mathbf{y}}$ using $\begin{cases}\mathsf{sk}_{\tilde{\mathbf{y}},\mathsf{ALS}} & \text{if } \ell' \le \ell^*\\ \mathbf{T}_{\ell'}\cdot\mathbf{y}^\top & \text{if } \ell' > \ell^*\end{cases}$ where $\tilde{\mathbf{y}} = [\mathbf{0}_1\vert\ldots\vert\mathbf{0}_{\ell'-1}\vert\mathbf{y}\vert\mathbf{0}_{\ell'+1}\vert\ldots\vert\mathbf{0}_{\ell^*}]$	ALS security

Fig. 5. Summary of the security proof of permissive construction. $\downarrow$ denotes the same as the previous hybrid.

- given messages $\mathbf{x}_0, \mathbf{x}_1$ from $\mathcal{A}$, the reduction computes

$$\tilde{\mathbf{x}}_b = [\underbrace{x_{b,1}}_{\mathbf{x}_b|_{[1]}} \mid \underbrace{x_{b,1}, x_{b,2}}_{\mathbf{x}_b|_{[2]}} \mid \ldots \mid \underbrace{x_{b,1}, x_{b,2}\ldots, x_{b,\ell^*}}_{\mathbf{x}_b|_{[\ell^*]}=\mathbf{x}_b}] \in \{0,\ldots,V-1\}^{\ell^*(\ell^*+1)/2}$$

  for $b \in \{0,1\}$, forwards $\tilde{\mathbf{x}}_0, \tilde{\mathbf{x}}_1$ to the ALS challenger, and receives back $(\mathbf{c}_{0,\mathsf{ALS}}, \mathbf{c}_{1,\mathsf{ALS}}) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^{\ell^*(\ell^*+1)/2}$
- it parses $\mathbf{c}_{1,\mathsf{ALS}} = (\mathbf{c}_{1,i,j,\mathsf{ALS}})_{i\in[\ell^*],j\in[i]}$, with $\mathbf{c}_{1,i,j,\mathsf{ALS}} \in \mathbb{Z}_q$, and programs $\mathbf{c}_{1,i,j}$ similarly as in Theorem 3,
- whenever $\mathcal{A}$ queries a functional key for vector $\mathbf{y}$ of length ℓ', the reduction replies using $\mathbf{T}_{\ell'}$, if $\ell' > \ell^*$. Otherwise, it computes

$$\tilde{\mathbf{y}} = \left[\mathbf{0}_1 \mid \ldots \mid \mathbf{0}_{\ell'-1} \mid \mathbf{y} \mid \mathbf{0}_{\ell'+1} \mid \ldots \mid \mathbf{0}_{\ell^*}\right] \in \{0,\ldots,P-1\}^{\ell^*(\ell^*+1)/2},$$

  forwards it to the ALS challenger, receives back $\mathsf{sk}_{\tilde{\mathbf{y}},\mathsf{ALS}}$ and uses it as a reply to $\mathcal{A}$'s query.

Notice that, for any $\mathbf{y}$ of length $\ell' \le \ell^*$, one has

$$\langle\tilde{\mathbf{x}}_b, \tilde{\mathbf{y}}\rangle = \langle\mathbf{x}_b\,|_{[\ell']}, \mathbf{y}\rangle.$$

Hybrid	mpk	ct	$\mathsf{sk}_{\mathbf{y}}$	justification
$\mathsf{H}_{3,h,k,0}$	$\downarrow$	$\downarrow$	$\downarrow$	$\mathsf{H}_{3,1,1,0} = \mathsf{H}_2$ $\mathsf{H}_{3,h,k,0} = \mathsf{H}_{3,h,k-1,9}, k \in [2,h]$ $\mathsf{H}_{3,h,1,0} = \mathsf{H}_{3,h-1,h-1,9}, h \in [2,\ell^*]$
$\mathsf{H}_{3,h,k,1}$	$\tilde{\mathbf{W}}_k, \tilde{\mathbf{Y}}_h \leftarrow \{1,-1\}^{m\times m}$ $\mathbf{W} = \mathbf{B}\cdot\tilde{\mathbf{W}}_k - k\cdot\mathbf{G}$ $\mathbf{Y} = \mathbf{B}\cdot\tilde{\mathbf{Y}}_h - h\cdot\mathbf{G}$	$\downarrow$	$\downarrow$	Lemma 2 (plain) LHL
$\mathsf{H}_{3,h,k,2}$	$\mathbf{v}^\top = [\mathbf{B} \mid \mathbf{W} + k\cdot\mathbf{G} \mid \mathbf{Y} + h\cdot\mathbf{G}]\cdot\mathbf{k}_{k,h}^\top$	$\downarrow$	$\mathbf{k}_{k,h} \leftarrow \mathcal{D}_{\mathbb{Z}^{3m},\tau_1}$	Lemma 3 (Item 1)
$\mathsf{H}_{3,h,k,3}$	$\downarrow$	$\downarrow$	$\mathbf{k}_{j,\ell}$ using $\begin{cases} \begin{bmatrix}\tilde{\mathbf{Y}}_h\\ \mathbf{0}_{m\times m}\\ -\mathbf{I}_m\end{bmatrix} & \text{if } \ell \neq h \\ \begin{bmatrix}\tilde{\mathbf{W}}_k\\ -\mathbf{I}_m\\ \mathbf{0}_{m\times m}\end{bmatrix} & \text{if } \ell = h, j \neq k \end{cases}$	Lemma 3 SamplePre
$\mathsf{H}_{3,h,k,4}$	$\mathbf{B} \leftarrow \mathbb{Z}_q^{n\times m}$	$\downarrow$	$\downarrow$	Lemma 3 TrapGen
$\mathsf{H}_{3,h,k,5}$	$\downarrow$	$\mathbf{c}_{1,h,k} = [\mathbf{c}_{2,h,k} \mid \mathbf{c}_{3,h,k} \mid \mathbf{c}_{4,h,k}]\cdot\mathbf{k}_{k,h} + \mathbf{s}\cdot\mathbf{d}_{k,h} + x_k^*\cdot\lfloor\frac{q}{K}\rfloor + \tilde{\mathbf{e}}_{1,h,k}$	$\downarrow$	Lemma 1 Noise Flooding
$\mathsf{H}_{3,h,k,6}$	$\downarrow$	$\mathbf{c}_{3,h,k} = \mathbf{c}_{2,h,k}\cdot\tilde{\mathbf{W}}_k - \mathbf{s}\cdot\mathbf{G} + \tilde{\mathbf{e}}_{3,h,k}$ $\mathbf{c}_{4,h,k} = \mathbf{c}_{2,h,k}\cdot\tilde{\mathbf{Y}}_h + \tilde{\mathbf{e}}_{4,h,k}$	$\downarrow$	Lemma 1 Noise Flooding
$\mathsf{H}_{3,h,k,7}$	$\downarrow$	$\mathbf{c}_{2,h,k} \leftarrow \mathbb{Z}_q^m$	$\downarrow$	$\mathsf{LWE}_{n,m,\chi'',q}$
$\mathsf{H}_{3,h,k,8}$	$\downarrow$	$\mathbf{c}_{3,h,k}, \mathbf{c}_{4,h,k} \leftarrow \mathbb{Z}_q^m$	$\downarrow$	Lemma 2 GLHL
$\mathsf{H}_{3,h,k,9}$	same as H_2	$\downarrow$	same as H_2	$\mathsf{H}_{3,h,k,4} \approx_s \mathsf{H}_{3,h,k,0}$

Fig. 6. Summary for $\mathsf{H}_2 \approx_c \mathsf{H}_3$. The sequence of hybrid is repeated for all $h \in [\ell^*], k \in [h]$. That is, $\mathsf{H}_2 = \mathsf{H}_{3,1,1,0} \approx \cdots \approx \mathsf{H}_{3,1,1,9} = \mathsf{H}_{3,2,1,0} \approx \cdots \approx \mathsf{H}_{3,2,1,9} = \mathsf{H}_{3,2,2,0} \approx \cdots \approx \mathsf{H}_{3,\ell^*,\ell^*,9} = \mathsf{H}_3$.

Therefore, correctness is guaranteed. It remains to argue that the reduction is a valid $\mathsf{ALS}[\ell^*(\ell^*+1)/2]$ adversary with respect to challenge messages $\tilde{\mathbf{x}}_0, \tilde{\mathbf{x}}_1$ for each valid adversary $\mathcal{A}$ against Π_2, with respect to challenge messages $\mathbf{x}_0, \mathbf{x}_1$. Indeed, for any key query $\mathbf{y}$ of length $\ell' \leq \ell^*$ that a valid adversary $\mathcal{A}$ against Π_2 can make, one has

$$\langle \mathbf{x}_0 \restriction_{[\ell']}, \mathbf{y}\rangle = \langle \mathbf{x}_1 \restriction_{[\ell']}, \mathbf{y}\rangle,$$

which implies

$$\langle \tilde{\mathbf{x}}_0, \tilde{\mathbf{y}}\rangle = \langle \tilde{\mathbf{x}}_1, \tilde{\mathbf{y}}\rangle,$$

as required in order to guaranteed that the reduction is a valid $\mathsf{ALS}[\ell^*(\ell^*+1)/2]$ adversary with respect to challenge messages $\tilde{\mathbf{x}}_0, \tilde{\mathbf{x}}_1$.

The various hybrids are described in Figures 5 and 6. This result is summarized in the following theorem.

Theorem 5 (Security). *Let Π_2 be the UIPFE construction described in Section 4, with parameters set as in Equation (2), and F be a PRF. Then, for any adversary $\mathcal{A}$ that runs is time $T = T(\lambda)$, there exists adversaries $\mathcal{B}_0$, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ against PRF-security, $\mathsf{LWE}_{n,m,\chi'',q}$, $\mathsf{LWE}_{n,m+\lambda,\chi'',q}$, and $\mathsf{IND\text{-}CPA}_{\mathsf{ALS}[\ell^*(\ell^*+1)/2]}$, respectively, such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\Pi_2}(\lambda) \leq T\cdot\Big(\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{B}_0,\mathsf{F}}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m,\chi'',q}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m+\lambda,\chi'',q}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{B}_3,\mathsf{ALS}[\ell^*(\ell^*+1)/2]}(\lambda) + \mathsf{negl}(\lambda)\Big).$$

5 Identity-based Unbounded IPFE

We present here an identity-based UIPFE (IB-UIPFE) scheme in the strict setting. We remark that we consider a strict construction for ease of exposition and note that the techniques presented here readily work with our permissive construction from Section 4 as well. We highlight the changes relative to the strict construction and its security proof from Section 3 with a blue box. The underlying identity-based IPFE which we reduce the security to is the construction of Lai, Liu and Wang [LLW21, Section 7.1], which we recall in Appendix B.

5.1 Construction

We consider that inner-products are evaluated over $\mathbb{Z}$, and message space $\mathcal{P}$, secret key vector space $\mathcal{V}$ and output space $\{0,\ldots,K_\ell - 1\}$ are defined as in Section 3.1. Let $\Pi_3 = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ be an identity-based UIPFE scheme defined as follows:

– $\mathsf{Setup}(1^\lambda, \mathcal{P}, \mathcal{V})$: Sample

$$(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), (\mathbf{B}, \mathbf{T_B}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{A}_i \leftarrow\$ \mathbb{Z}_q^{n\times m} \text{ for all } i \in [\lambda]$$
$$\mathbf{W}, \mathbf{Y} \leftarrow\$ \mathbb{Z}_q^{n\times m}, \mathbf{v} \leftarrow\$ \mathbb{Z}_q^n$$
$$\mathsf{k} \leftarrow\$ \{0,1\}^\lambda.$$

Set $\mathsf{mpk} = (\mathbf{A}, \{\mathbf{A}_i\}_{i\in[\lambda]}, \mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v})$, and $\mathsf{msk} = (\mathbf{T_A}, \mathbf{T_B}, \mathsf{k})$. Return $(\mathsf{mpk}, \mathsf{msk})$.

– $\mathsf{Enc}(\mathsf{mpk}, \mathsf{id} \in \{0,1\}^\lambda, \mathbf{x} \in \mathcal{V})$: Parse $\mathsf{id} := (b_1,\ldots,b_\lambda) \in \{0,1\}^\lambda$, and let $\ell = |\mathbf{x}|$. Sample

$$\mathbf{s} \leftarrow\$ \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow\$ \mathcal{D}_{\mathbb{Z}^{2m},\chi''},$$
$$\mathbf{s}_j \leftarrow\$ \mathbb{Z}_q^n, e_{1,j} \leftarrow\$ \mathcal{D}_{\mathbb{Z},\chi''}, \mathbf{e}_{2,j} \leftarrow\$ \mathcal{D}_{\mathbb{Z}^m,\chi}, \mathbf{e}_{3,j} \leftarrow\$ \mathcal{D}_{\mathbb{Z}^m,\chi'}, \mathbf{e}_{4,j} \leftarrow\$ \mathcal{D}_{\mathbb{Z}^m,\chi'} \quad \text{for all } j \in [\ell].$$

Set $\mathbf{A}_{\mathsf{id}} := \mathbf{G} + \sum_{i=1}^{\lambda}(b_i\mathbf{A}_i) \in \mathbb{Z}_q^{n\times m}$, and compute

$$\begin{aligned}
\mathbf{c}_0 &:= \mathbf{s}\cdot[\mathbf{A}|\mathbf{A}_{\mathsf{id}}] + \mathbf{e}_0 \bmod q,\\
\mathbf{c}_{1,j} &:= \mathbf{s}_j\cdot\mathbf{v}^\top + x_j\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + e_{1,j} \mod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{2,j} &:= \mathbf{s}_j\cdot\mathbf{B} + \mathbf{e}_{2,j} \bmod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{3,j} &:= \mathbf{s}_j\cdot(\mathbf{W} + j\cdot\mathbf{G}) - \mathbf{s}\cdot\mathbf{G} + \mathbf{e}_{3,j} \mod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{4,j} &:= \mathbf{s}_j\cdot(\mathbf{Y} + \ell\cdot\mathbf{G}) + \mathbf{e}_{4,j} \mod q \quad \text{for all } j\in[\ell].
\end{aligned}$$

Return $\mathsf{ct}_\mathbf{x} := \left(\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j\in[\ell]}, \{\mathbf{c}_{2,j}\}_{j\in[\ell]}, \{\mathbf{c}_{3,j}\}_{j\in[\ell]}, \{\mathbf{c}_{4,j}\}_{j\in[\ell]}\right)$.

– $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id} \in \{0,1\}^\lambda, \mathbf{y} \in \mathcal{P})$: Parse $\mathsf{id} := (b_1,\ldots,b_\lambda) \in \{0,1\}^\lambda$, and let $\ell' = |\mathbf{y}|$. For all $j \in [\ell']$, sample

$$\mathbf{k}_{j,\ell'}^\top \leftarrow \mathsf{SamplePre}\left([\mathbf{B} \mid \mathbf{W} + j\cdot\mathbf{G} \mid \mathbf{Y} + \ell'\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T_B}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k}, j, \ell', 1)\right).$$

Parse $\mathbf{k}_{j,\ell'}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell'}^\top\\ \mathbf{r}_{j,\ell'}^\top\\ \mathbf{p}_{j,\ell'}^\top\end{bmatrix}$, set $\mathbf{d}_{j,\ell'}^\top := \mathbf{G}\cdot\mathbf{r}_{j,\ell'}^\top \bmod q$ and $\mathbf{D}_{\ell'} := [\mathbf{d}_{1,\ell'}^\top | \ldots | \mathbf{d}_{\ell',\ell'}^\top]$. Next, set the identity matrix $\mathbf{A}_{\mathsf{id}} := \mathbf{G} + \sum_{i=1}^{\lambda}(b_i\mathbf{A}_i) \in \mathbb{Z}_q^{n\times m}$, and sample

$$\mathbf{T}_{\ell'} \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{A}_{\mathsf{id}}], \begin{bmatrix}\mathbf{T_A}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{D}_{\ell'}, \tau_2; \mathsf{F}(\mathsf{k}, \ell', 2)\right),$$

Set $\mathbf{t}_\mathbf{y}^\top = \mathbf{T}_\ell\cdot\mathbf{y}^\top$. Return $\mathsf{sk}_\mathbf{y} := \left(\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y}\right)$.

Hybrid	mpk	ct	$\mathsf{sk}_{\mathbf{y}}$	justification
H_0	$(\mathbf{A},\mathbf{T}_{\mathbf{A}}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n,1^m,q)$ $(\mathbf{B},\mathbf{T}_{\mathbf{B}}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n,1^m,q)$ $\mathbf{A}_i \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ for all $i\in[\lambda]$ $\mathbf{W} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{Y} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{v} \leftarrow\!\$\ \mathbb{Z}_q^{n}$ $\mathsf{k} \leftarrow \mathcal{K}$	$\mathbf{c}_0 \approx \mathbf{s}\cdot[\mathbf{A}\vert\mathbf{A}_{\mathsf{id}}]$ $\mathbf{c}_{1,j} \approx \mathbf{s}_j\cdot\mathbf{v} + x^*_j\cdot\lfloor\frac{q}{K}\rfloor$ $\mathbf{c}_{2,j} \approx \mathbf{s}_j\cdot\mathbf{B}$ $\mathbf{c}_{3,j} \approx \mathbf{s}_j\cdot(\mathbf{W}+j\cdot\mathbf{G}) - \mathbf{s}^\top\cdot\mathbf{G}$ $\mathbf{c}_{4,j} \approx \mathbf{s}_j\cdot(\mathbf{Y}+\ell\cdot\mathbf{G})$	$\mathbf{T}_{\mathbf{A}},\mathbf{T}_{\mathbf{B}},\mathsf{k}$ $\mathbf{k}^\top_{j,\ell} = \begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{r}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix} \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{W}+j\cdot\mathbf{G}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\\ \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix},\mathbf{v}^\top,\tau_1;\mathsf{F}(\mathsf{k},j,\ell,1)\end{bmatrix}\right)$ $\mathbf{D}_\ell = [\mathbf{G}\cdot\mathbf{r}^\top_{1,\ell}\vert\ldots\vert\mathbf{G}\cdot\mathbf{r}^\top_{\ell,\ell}]$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left([\mathbf{A}\vert\mathbf{A}_{\mathsf{id}}],\begin{bmatrix}\mathbf{T}_{\mathbf{A}}\\ \mathbf{0}_{m\times m}\end{bmatrix},\mathbf{D}_\ell,\tau_2;\mathsf{F}(\mathsf{k},\ell,2)\right)$ $\mathbf{t}^\top_{\mathbf{y}} = \mathbf{T}_\ell\cdot\mathbf{y}^\top$	
H_1	$\downarrow$	$\downarrow$	$\downarrow$	guess $\lvert\mathbf{x}^*\rvert = \ell^*$
H_2	$\downarrow$	$\downarrow$	$\boldsymbol{\rho} \leftarrow\!\$\ \{0,1\}^{\mathsf{poly}(\lambda)}$ $\mathbf{k}^\top_{j,\ell} \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{W}+j\cdot\mathbf{G}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\\ \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix},\mathbf{v}^\top,\tau_1;\boldsymbol{\rho}_{1,j,\ell}\end{bmatrix}\right)$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left([\mathbf{A}\vert\mathbf{A}_{\mathsf{id}}],\begin{bmatrix}\mathbf{T}_{\mathbf{A}}\\ \mathbf{0}_{m\times m}\end{bmatrix},\mathbf{D}_\ell,\tau_2;\boldsymbol{\rho}_{2,\ell}\right)$	PRF security
H_3	$\downarrow$	$\mathbf{c}_{2,j},\mathbf{c}_{3,j},\mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m$ $\mathbf{c}_{1,j} \approx [\mathbf{c}_{2,j}\mid\mathbf{c}_{3,j}\mid\mathbf{c}_{4,j}]\cdot\mathbf{k}^\top_{j,\ell^*} + \mathbf{s}\cdot\mathbf{d}^\top_{j,\ell^*} + x^*_j\cdot\lfloor\frac{q}{K_{\ell^*}}\rfloor$	$\downarrow$	$\mathsf{LWE}_{n,m,\chi'',q}$ (Figure 4)
H_4	$\downarrow$	$\downarrow$	$\mathbf{r}_{j,\ell} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\tau_1}$ $\begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix},\\ \mathbf{v}-[\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}^\top_{j,\ell},\tau_1;\boldsymbol{\rho}_{1,j,\ell,1}\end{bmatrix}\right)$ $\mathbf{k}^\top_{j,\ell} = \begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{r}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix}$	Lemma 4
H_5	$\downarrow$	$\downarrow$	$\mathbf{D}_\ell = [\mathbf{d}^\top_{1,\ell}\vert\ldots\vert\mathbf{d}^\top_{\ell,\ell}] \leftarrow\!\$\ \mathbb{Z}_q^{n\times\ell}$ $\mathbf{r}^\top_{j,\ell} = \mathsf{SamplePre}(\mathbf{G},\mathbf{I},\mathbf{d}^\top_{j,\ell},\tau_1;\boldsymbol{\rho}_{1,j,\ell,2})$ $\begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix},\\ \mathbf{v}^\top-[\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}^\top_{j,\ell},\tau_1;\boldsymbol{\rho}_{1,j,\ell,1}\end{bmatrix}\right)$ $\mathbf{k}^\top_{j,\ell} = \begin{bmatrix}\mathbf{z}^\top_{j,\ell}\\ \mathbf{r}^\top_{j,\ell}\\ \mathbf{p}^\top_{j,\ell}\end{bmatrix}$	Lemma 3 (Item 1)
H_6	$\downarrow$	$\downarrow$	$\mathbf{T}_\ell \leftarrow \mathcal{D}_{\mathbb{Z}^{2m\times\ell},\tau_2}$ $\mathbf{D}_\ell = [\mathbf{A}\vert\mathbf{A}_{\mathsf{id}}]\cdot\mathbf{T}_\ell$	Lemma 3 (Item 1)
H_7	$\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$	$\downarrow$	$\downarrow$	Lemma 3 TrapGen
LLW-IB-IPFE[ℓ^*] reduction	$\mathbf{A} = \mathbf{A}_{\mathsf{LLW}}$ $\mathbf{A}_i = \mathbf{A}_{i,\mathsf{LLW}}$ for all $i\in[\lambda]$ $\mathbf{D}_{\ell^*} = \mathbf{D}_{\mathsf{LLW}}$	$\mathbf{c}_0 = \mathbf{c}_{0,\mathsf{LLW}}$ $\mathbf{c}_{1,j} = [\mathbf{c}_{2,j}\mid\mathbf{c}_{3,j}\mid\mathbf{c}_{4,j}]\cdot\mathbf{k}^\top_{j,\ell^*} + \mathbf{c}_{1,j,\mathsf{LLW}}$	$\mathbf{t}^\top_{\mathbf{y}}$ using $\begin{cases}\mathsf{sk}_{\mathbf{y},\mathsf{LLW}} & \text{if } \ell=\ell^*\\ \mathbf{T}_\ell\cdot\mathbf{y}^\top & \text{if } \ell\neq\ell^*\end{cases}$	LLW-IB-IPFE security

Fig. 7. Summary of the security proof of IB-UIPFE construction. $\downarrow$ denotes the same as the previous hybrid.

– $\mathsf{Dec}(\mathsf{sk}_{\mathbf{y}},\mathbf{y},\mathsf{ct}_{\mathbf{x}})$: Parse $\mathsf{ct}_{\mathbf{x}} = \left(\mathbf{c}_0,\{\mathbf{c}_{1,j}\}_{j\in[\ell]},\{\mathbf{c}_{2,j}\}_{j\in[\ell]},\{\mathbf{c}_{3,j}\}_{j\in[\ell]},\{\mathbf{c}_{4,j}\}_{j\in[\ell]}\right)$,and $\mathsf{sk}_{\mathbf{y}} = \left(\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']},\mathbf{t}_{\mathbf{y}}\right)$. If $\ell\neq\ell'$, return $\perp$. Else, compute

$$\mu' := \left(\left[\mathbf{c}_{1,1}\,\middle|\,\ldots\,\middle|\,\mathbf{c}_{1,\ell}\right] - \left[\left[\mathbf{c}_{2,1}\,|\,\mathbf{c}_{3,1}\,|\,\mathbf{c}_{4,1}\right]\cdot\mathbf{k}^\top_{1,\ell}\,\middle|\,\ldots\,\middle|\,\left[\mathbf{c}_{2,\ell}\,|\,\mathbf{c}_{3,\ell}\,|\,\mathbf{c}_{4,\ell}\right]\cdot\mathbf{k}^\top_{\ell,\ell}\right]\right)\cdot\mathbf{y}^\top - \mathbf{c}_0\cdot\mathbf{t}^\top_{\mathbf{y}} \mod q$$

and return the value $\mu\in\{-K_\ell+1,\ldots,K_\ell-1\}$ that minimizes $\lvert\lfloor\frac{q}{K_\ell}\rfloor\cdot\mu-\mu'\rvert$.

Parameters. The parameters can be set exactly as in Equation (2).

Theorem 6 (Correctness). *Let Π_3 be the IB-UIPFE construction described in Section 5, with parameters as in Equation (2). Then, Π_3 is correct.*

Proof (sketch). Correctness is exactly the same as in Theorem 2, since the only change done in the ciphertext is to $\mathbf{c}_0$ component, which is now an LWE sample for the public matrix $[\mathbf{A}\vert\mathbf{A}_{\mathsf{id}}]$, where $\mathbf{A}_{\mathsf{id}} := \mathbf{G}+\sum_{i=1}^{\lambda}(b_i\mathbf{A}_i)$

for some identity $\mathsf{id} \coloneqq (b_1, \ldots, b_\lambda)$. This change is reflected also during the key generation, where now we sample $\mathbf{T}_{\ell'}$ according to $[\mathbf{A}|\mathbf{A}_{\mathsf{id}}]$ using the trapdoor $\mathbf{T}_{\mathbf{A}}$, and hence, maintain the decryption correctness, i.e., the correctness of $\mathbf{c}_0 \cdot \mathbf{t}_{\mathbf{y}}^\top$. □

5.2 Security

The security proof follows analogously to the proof of the strict unbounded construction (Section 3.1) given in Section 3.2, except that we do not reduce it at the end to the security of the Π_{ALS} scheme (Appendix A), but we reduce it to the security of the $\Pi_{\mathsf{LLW\text{-}IB\text{-}IPFE}}$ scheme described in Appendix B. Concretely, the reduction works as follows:

- it receives $\mathsf{mpk}_{\mathsf{LLW}} = (\mathbf{A}_{\mathsf{LLW}}, \{\mathbf{A}_{i,\mathsf{LLW}}\}_{i\in[\lambda]}, \mathbf{D}_{\mathsf{LLW}}) \in \mathbb{Z}_q^{n\times m} \times (\mathbb{Z}_q^{n\times m})^\lambda \times \mathbb{Z}_q^{n\times \ell^*}$ from the LLW-IB-IPFE challenger,
- it samples $\mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v}$ as in hybrid H_7 from Figure 7, while setting $\mathbf{A} \coloneqq \mathbf{A}_{\mathsf{LLW}}$ and $\mathbf{A}_i \coloneqq \mathbf{A}_{i,\mathsf{LLW}}$ for all $i \in [\lambda]$,
- it samples $\{\mathbf{T}_\ell\}_{\ell\in[T],\ell\neq\ell^*}$ and implicitly sets $\{\mathbf{D}_\ell\}_{j\in[T],\ell\neq\ell^*}$ as in hybrid H_7, while setting $\mathbf{D}_{\ell^*} \coloneqq \mathbf{D}_{\mathsf{LLW}}$ obtained from the LLW-IB-IPFE challenger (recall that T denotes the running time of $\mathcal{A}$ and we have that $\ell^* \le T$),
- it samples $\mathbf{k}_{j,\ell}$ for all $j, \ell \in [T]$ as in hybrid H_7,
- it answers KeyGen queries for $\mathbf{y} \in \{0, \ldots, P-1\}^{\ell'}$ using $\mathbf{T}_{\ell'}$ when $\ell' \neq \ell^*$. For $\ell' = \ell^*$, it forwards the query to the LLW-IB-IPFE challenger, receives back $\mathsf{sk}_{\mathbf{y}}$, and forwards that to the adversary,
- whenever the adversary $\mathcal{A}$ outputs $(\mathbf{x}_0, \mathbf{x}_1)$, it forwards that to the LLW-IB-IPFE challenger and receives back

  $$\mathbf{c}_{0,\mathsf{LLW}} \in \mathbb{Z}_q^{2m}, \mathbf{c}_{1,\mathsf{LLW}} \in \mathbb{Z}_q^{\ell^*}.$$

  It samples

  $$\mathbf{c}_{2,j}, \mathbf{c}_{3,j}, \mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m \quad \text{for } j \in [\ell^*],$$

  and outputs

  $$\mathsf{ct} \coloneqq \begin{pmatrix} \mathbf{c}_{0,\mathsf{LLW}} \\ \{[\mathbf{c}_{2,j} \mid \mathbf{c}_{3,j} \mid \mathbf{c}_{4,j}] \cdot \mathbf{k}_{j,\ell^*}^\top + \mathbf{c}_{1,j,\mathsf{LLW}}\}_{j\in[\ell^*]}, \\ \{\mathbf{c}_{2,j}\}_{j\in[\ell^*]}, \\ \{\mathbf{c}_{3,j}\}_{j\in[\ell^*]}, \\ \{\mathbf{c}_{4,j}\}_{j\in[\ell^*]}, \end{pmatrix}.$$

- whenever the adversary outputs a bit $b' \in \{0,1\}$, the reduction forwards it to the LLW-IB-IPFE challenger.

It is easy to verify that correctness is guaranteed and that the reduction is a valid $\mathsf{LLW\text{-}IB\text{-}IPFE}[\ell^*]$ adversary.

The various hybrids are described in Figures 4 and 7. This result is summarized in the following theorem.

Theorem 7 (Security). *Let Π_3 be the IB-UIPFE construction described in Section 5, with parameters set as in Equation (2), and F be a PRF. Then, for any adversary $\mathcal{A}$ that runs is time $T = T(\lambda)$, there exists adversaries $\mathcal{B}_0$, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ against PRF-security, $\mathsf{LWE}_{n,m,\chi'',q}$, $\mathsf{LWE}_{n,m+\lambda,\chi'',q}$, and $\mathsf{IND\text{-}CPA}_{\mathsf{LLW\text{-}IB\text{-}IPFE}[\ell^*]}$, respectively, such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\Pi_3}(\lambda) \le T \cdot \left(\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{B}_0,\mathsf{F}}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m,\chi'',q}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m+\lambda,\chi'',q}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{B}_3,\mathsf{LLW}[\ell^*]}(\lambda) + \mathsf{negl}(\lambda)\right).$$

6 Attribute-based Unbounded IPFE

We present here an attribute-based UIPFE (AB-UIPFE) scheme in the strict setting. Analogous to the case of IB-UIPFE (Section 5), we note that we can apply the techniques presented here to our permissive construction from Section 4 as well to obtain a permissive AB-UIPFE scheme. As before we have highlighted the changes relative to the strict construction and its security proof from Section 3 with a blue box. The underlying attribute-based IPFE which we reduce the security to is the construction of Lai, Liu and Wang [LLW21, Adapted from Section 7.2], which we recall in Appendix C.

6.1 Construction

We consider that inner-products are evaluated over $\mathbb{Z}$, and message space $\mathcal{P}$, secret key vector space $\mathcal{V}$ and output space $\{0,\ldots,K_\ell-1\}$ are defined as in Section 3.1. We assume that the attribute space is $\{0,1\}^\lambda$, and consider the family $\mathcal{C}$ of circuits of depth-d mapping $\{0,1\}^\lambda$ to $\{0,1\}$.

Let $\Pi_4 = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ be an attribute-based UIPFE scheme defined as follows:

– $\mathsf{Setup}(1^\lambda, \mathcal{C}, \mathcal{P}, \mathcal{V})$: Sample

$$(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), (\mathbf{B}, \mathbf{T_B}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{A}_i \leftarrow\!\$\ \mathbb{Z}_q^{n\times m} \text{ for all } i \in [\lambda]$$
$$\mathbf{W}, \mathbf{Y} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}, \mathbf{v} \leftarrow\!\$\ \mathbb{Z}_q^n$$
$$\mathsf{k} \leftarrow\!\$\ \{0,1\}^\lambda.$$

Set $\mathsf{mpk} = (\mathbf{A}, \{\mathbf{A}_i\}_{i\in[\lambda]}, \mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v})$, and $\mathsf{msk} = (\mathbf{T_A}, \mathbf{T_B}, \mathsf{k})$. Return $(\mathsf{mpk}, \mathsf{msk})$.

– $\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{\alpha} \in \{0,1\}^\lambda, \mathbf{x} \in \mathcal{V})$: Parse $\boldsymbol{\alpha} := (\alpha_1, \ldots, \alpha_\lambda) \in \{0,1\}^\lambda$, and let $\ell = |\mathbf{x}|$. Sample

$$\mathbf{s} \leftarrow\!\$\ \mathbb{Z}_q^n, \mathbf{e} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi''},$$
$$\mathbf{s}_j \leftarrow\!\$\ \mathbb{Z}_q^n, e_{1,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi''}, \mathbf{e}_{2,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi}, \mathbf{e}_{3,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi'}, \mathbf{e}_{4,j} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\chi'} \text{ for all } j \in [\ell],$$
$$\mathbf{R}_i \leftarrow\!\$\ \{0,1\}^{m\times m} \text{ for all } i \in [\lambda].$$

Set

$$\mathbf{A}_{\boldsymbol{\alpha}} := [\mathbf{A}_1 + \alpha_1\mathbf{G} \mid \cdots \mid \mathbf{A}_\lambda + \alpha_\lambda\mathbf{G}] \in \mathbb{Z}_q^{n\times\lambda m},$$
$$\mathbf{e}_0 := \mathbf{e}\cdot[\mathbf{I}_m \mid \mathbf{R}_1 \mid \cdots \mid \mathbf{R}_\lambda] \in \mathbb{Z}^{(\lambda+1)m},$$

and compute

$$\begin{aligned}
\mathbf{c}_0 &:= \mathbf{s}\cdot[\mathbf{A}|\mathbf{A}_{\boldsymbol{\alpha}}] + \mathbf{e}_0 \bmod q,\\
\mathbf{c}_{1,j} &:= \mathbf{s}_j\cdot\mathbf{v}^\top + x_j\cdot\left\lfloor\frac{q}{K_\ell}\right\rfloor + e_{1,j} \mod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{2,j} &:= \mathbf{s}_j\cdot\mathbf{B} + \mathbf{e}_{2,j} \bmod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{3,j} &:= \mathbf{s}_j\cdot(\mathbf{W} + j\cdot\mathbf{G}) - \mathbf{s}\cdot\mathbf{G} + \mathbf{e}_{3,j} \mod q \quad \text{for all } j\in[\ell],\\
\mathbf{c}_{4,j} &:= \mathbf{s}_j\cdot(\mathbf{Y} + \ell\cdot\mathbf{G}) + \mathbf{e}_{4,j} \mod q \quad \text{for all } j\in[\ell].
\end{aligned}$$

Return $\mathsf{ct}_{\boldsymbol{\alpha},\mathbf{x}} := \left(\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j\in[\ell]}, \{\mathbf{c}_{2,j}\}_{j\in[\ell]}, \{\mathbf{c}_{3,j}\}_{j\in[\ell]}, \{\mathbf{c}_{4,j}\}_{j\in[\ell]}\right)$.

– $\mathsf{KeyGen}(\mathsf{msk}, C \in \mathcal{C}, \mathbf{y} \in \mathcal{P})$: Let $\ell' = |\mathbf{y}|$ and set $\mathbf{A}_c := \mathsf{EvalC}\left(\{\mathbf{A}_i\}_{i\in[\lambda]}, C\right)$. For all $j \in [\ell']$, sample

$$\mathbf{k}_{j,\ell'}^\top \leftarrow \mathsf{SamplePre}\left([\mathbf{B}|\mathbf{W} + j\cdot\mathbf{G}|\mathbf{Y} + \ell'\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T_B}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k}, j, \ell', 1)\right).$$

Parse

$$\mathbf{k}_{j,\ell'}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell'}^\top\\ \mathbf{r}_{j,\ell'}^\top\\ \mathbf{p}_{j,\ell'}^\top\end{bmatrix},$$

set $\mathbf{d}_{j,\ell'}^\top := \mathbf{G} \cdot \mathbf{r}_{j,\ell'}^\top \bmod q$ and $\mathbf{D}_{\ell'} := [\mathbf{d}_{1,\ell'}^\top | \dots | \mathbf{d}_{\ell',\ell'}^\top]$. Sample

$$\mathbf{T}_{\ell'} \leftarrow \mathsf{SamplePre}\left([\mathbf{A}|\mathbf{A}_C], \begin{bmatrix}\mathbf{T_A}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{D}_{\ell'}, \tau_2; \mathsf{F}(\mathsf{k}, \ell', 2)\right),$$

Set $\mathbf{t}_\mathbf{y}^\top = \mathbf{T}_\ell \cdot \mathbf{y}^\top$. Return $\mathsf{sk}_{C,\mathbf{y}} := \left(\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y}\right)$.

– $\mathsf{Dec}(\mathsf{sk}_{C,\mathbf{y}}, \mathbf{y}, C, \mathsf{ct}_{\boldsymbol{\alpha},\mathbf{x}}, \boldsymbol{\alpha})$: Parse $\mathsf{sk}_\mathbf{y} = \left(\{\mathbf{k}_{j,\ell'}\}_{j\in[\ell']}, \mathbf{t}_\mathbf{y}\right)$ and $\mathsf{ct}_{\boldsymbol{\alpha},\mathbf{x}} = (\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j\in[\ell]}, \{\mathbf{c}_{2,j}\}_{j\in[\ell]}, \{\mathbf{c}_{3,j}\}_{j\in[\ell]}, \{\mathbf{c}_{4,j}\}_{j\in[\ell]})$. If $\ell \neq \ell'$, return $\perp$. Else, parse $\mathbf{c}_0 := (\mathbf{c}_{0,0}, \dots, \mathbf{c}_{0,\lambda}) \in \mathbb{Z}_q^{(\ell+1)m}$ and compute

$$\mathbf{H}_{\mathbf{A},C,\boldsymbol{\alpha}} := \mathsf{EvalCX}(\mathbf{A}, C, \boldsymbol{\alpha}),$$
$$\mathbf{c}_C := [\mathbf{c}_{0,1} \mid \dots \mid \mathbf{c}_{0,\lambda}] \cdot \mathbf{H}_{\mathbf{A},C,\boldsymbol{\alpha}} \in \mathbb{Z}_q^m.$$

Set $\mathbf{c}'_0 := [\mathbf{c}_{0,0} \mid \mathbf{c}_C] \in \mathbb{Z}_q^{2m}$ and compute

$$\mu' := \left(\left[\mathbf{c}_{1,1} \,\middle|\, \dots \,\middle|\, \mathbf{c}_{1,\ell}\right] - \left[\left[\mathbf{c}_{2,1} \,|\, \mathbf{c}_{3,1} \,|\, \mathbf{c}_{4,1}\right] \cdot \mathbf{k}_{1,\ell}^\top \,\middle|\, \dots \,\middle|\, \left[\mathbf{c}_{2,\ell} \,|\, \mathbf{c}_{3,\ell} \,|\, \mathbf{c}_{4,\ell}\right] \cdot \mathbf{k}_{\ell,\ell}^\top\right]\right) \cdot \mathbf{y}^\top - \mathbf{c}'_0 \cdot \mathbf{t}_\mathbf{y}^\top \mod q$$

and return the value $\mu \in \{-K_\ell + 1, \dots, K_\ell - 1\}$ that minimizes $|\lfloor \frac{q}{K_\ell} \rfloor \cdot \mu - \mu'|$.

Parameters. The parameters can be set as in Equation (2), except that we set the modulus to

$$q = \lambda_1^{O(d)}, \tag{6}$$

in order to compensate for the error growth due to the homomorphic computation procedure (see Theorem 1), where d is the depth of the circuits in $\mathcal{C}$.

Theorem 8 (Correctness). *Let Π_4 be the AB-UIPFE construction described in Section 6, with parameters as in Equation (2) and (6). Then, Π_4 is correct.*

Proof (sketch). Correctness is exactly the same as in Theorem 2, since the only change done in the ciphertext is to $\mathbf{c}_0$ component, which is now an LWE sample for the public matrix $[\mathbf{A} \mid \mathbf{A}_{\boldsymbol{\alpha}}]$, where $\mathbf{A}_{\boldsymbol{\alpha}} := [\mathbf{A}_1 + \alpha_1\mathbf{G} \mid \dots \mid \mathbf{A}_\lambda + \alpha_\lambda\mathbf{G}]$ for some attribute $\boldsymbol{\alpha} := (\alpha_1, \dots, \alpha_\lambda)$. This change is reflected also during the key generation, where now we sample $\mathbf{T}_{\ell'}$ according to $[\mathbf{A} \mid \mathbf{A}_C]$ using the trapdoor $\mathbf{T_A}$, where $\mathbf{A}_C := \mathsf{EvalC}\left(\{\mathbf{A}_i\}_{i\in[\lambda]}, C\right)$ for a circuit $C \in \mathcal{C}$. Then, during the decryption we set $\mathbf{c}'_0 := [\mathbf{c}_{0,0} \mid \mathbf{c}_C]$, where $\mathbf{c}_C := [\mathbf{c}_{0,1} \mid \dots \mid \mathbf{c}_{0,\lambda}] \cdot \mathbf{H}_{\mathbf{A},C,\boldsymbol{\alpha}}$ for $\mathbf{H}_{\mathbf{A},C,\boldsymbol{\alpha}} := \mathsf{EvalCX}(\mathbf{A}, C\boldsymbol{\alpha})$. By the properties of homomorphic computation on matrices (Theorem 1), we get that

$$\begin{aligned}\mathbf{c}'_0 &:= [\mathbf{c}_{0,0} \mid \mathbf{c}_C]\\ &\approx \mathbf{s} \cdot [\mathbf{A} \mid \mathbf{A}_C + C(\boldsymbol{\alpha}) \cdot \mathbf{G}] \bmod q\\ &\approx \mathbf{s} \cdot [\mathbf{A} \mid \mathbf{A}_C] \bmod q,\end{aligned}$$

when $C(\boldsymbol{\alpha}) = 0$, i.e., the attribute $\boldsymbol{\alpha}$ satisfies the policy C. Hence, we have that the correctness of $\mathbf{c}'_0 \cdot \mathbf{t}_\mathbf{y}^\top$, and the overall decryption procedure is maintained. □

Hybrid	mpk	ct	$\mathsf{sk}_{\mathbf{y}}$	justification
H_0	$(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)$ $(\mathbf{B}, \mathbf{T}_{\mathbf{B}}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)$ $\mathbf{A}_i \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ for all $i \in [\lambda]$ $\mathbf{W} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{Y} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}$ $\mathbf{v} \leftarrow\!\$\ \mathbb{Z}_q^{n}$ $\mathsf{k} \leftarrow \mathcal{K}$	$\mathbf{c}_0 \approx \mathbf{s}\cdot [\mathbf{A}\vert\mathbf{A}_{\boldsymbol{\alpha}}]$ $\mathbf{c}_{1,j} \approx \mathbf{s}_j \cdot \mathbf{v} + x_j^* \cdot \lfloor \frac{q}{K} \rfloor$ $\mathbf{c}_{2,j} \approx \mathbf{s}_j \cdot \mathbf{B}$ $\mathbf{c}_{3,j} \approx \mathbf{s}_j \cdot (\mathbf{W} + j\cdot\mathbf{G}) - \mathbf{s}^\top \cdot \mathbf{G}$ $\mathbf{c}_{4,j} \approx \mathbf{s}_j \cdot (\mathbf{Y} + \ell\cdot\mathbf{G})$	$\mathbf{T}_{\mathbf{A}}, \mathbf{T}_{\mathbf{B}}, \mathsf{k}$ $\mathbf{k}_{j,\ell}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{W}+j\cdot\mathbf{G}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\\ \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \mathsf{F}(\mathsf{k}, j, \ell, 1)\end{bmatrix}\right)$ $\mathbf{D}_\ell = [\mathbf{G}\cdot\mathbf{r}_{1,\ell}^\top \vert \ldots \vert \mathbf{G}\cdot\mathbf{r}_{\ell,\ell}^\top]$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left([\mathbf{A}\vert\mathbf{A}_C], \begin{bmatrix}\mathbf{T}_{\mathbf{A}}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{D}_\ell, \tau_2; \mathsf{F}(\mathsf{k}, \ell, 2)\right)$ $\mathbf{t}_{\mathbf{y}}^\top = \mathbf{T}_\ell \cdot \mathbf{y}^\top$	
H_1	$\downarrow$	$\downarrow$	$\downarrow$	guess $\vert\mathbf{x}^*\vert = \ell^*$
H_2	$\downarrow$	$\downarrow$	$\boldsymbol{\rho} \leftarrow\!\$\ \{0,1\}^{\mathsf{poly}(\lambda)}$ $\mathbf{k}_{j,\ell}^\top \leftarrow \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{W}+j\cdot\mathbf{G}\vert\mathbf{Y}+\ell\cdot\mathbf{G}],\\ \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{v}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell}\end{bmatrix}\right)$ $\mathbf{T}_\ell \leftarrow \mathsf{SamplePre}\left([\mathbf{A}\vert\mathbf{A}_C], \begin{bmatrix}\mathbf{T}_{\mathbf{A}}\\ \mathbf{0}_{m\times m}\end{bmatrix}, \mathbf{D}_\ell, \tau_2; \boldsymbol{\rho}_{2,\ell}\right)$	PRF security
H_3	$\downarrow$	$\mathbf{c}_{2,j}, \mathbf{c}_{3,j}, \mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m$ $\mathbf{c}_{1,j} \approx [\mathbf{c}_{2,j} \mid \mathbf{c}_{3,j} \mid \mathbf{c}_{4,j}] \cdot \mathbf{k}_{j,\ell^*}^\top + \mathbf{s}\cdot\mathbf{d}_{j,\ell^*}^\top + x_j^* \cdot \lfloor \frac{q}{K_{\ell^*}} \rfloor$	$\downarrow$	$\mathsf{LWE}_{n,m,\chi'',q}$ (Figure 4)
H_4	$\downarrow$	$\downarrow$	$\mathbf{r}_{j,\ell} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m,\tau_1}$ $\begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{Y}+\ell\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix},\\ \mathbf{v} - [\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\end{bmatrix}\right)$ $\mathbf{k}_{j,\ell}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix}$	Lemma 4
H_5	$\downarrow$	$\downarrow$	$\mathbf{D}_\ell = [\mathbf{d}_{1,\ell}^\top \vert \ldots \vert \mathbf{d}_{\ell,\ell}^\top] \leftarrow\!\$\ \mathbb{Z}_q^{n\times\ell}$ $\mathbf{r}_{j,\ell}^\top = \mathsf{SamplePre}(\mathbf{G}, \mathbf{I}, \mathbf{d}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,2})$ $\begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix} = \mathsf{SamplePre}\left(\begin{bmatrix}[\mathbf{B}\vert\mathbf{Y}+\ell\cdot\mathbf{G}], \begin{bmatrix}\mathbf{T}_{\mathbf{B}}\\ \mathbf{0}_{m\times m}\end{bmatrix},\\ \mathbf{v}^\top - [\mathbf{W}+j\cdot\mathbf{G}]\cdot\mathbf{r}_{j,\ell}^\top, \tau_1; \boldsymbol{\rho}_{1,j,\ell,1}\end{bmatrix}\right)$ $\mathbf{k}_{j,\ell}^\top = \begin{bmatrix}\mathbf{z}_{j,\ell}^\top\\ \mathbf{r}_{j,\ell}^\top\\ \mathbf{p}_{j,\ell}^\top\end{bmatrix}$	Lemma 3 (Item 1)
H_6	$\downarrow$	$\downarrow$	$\mathbf{T}_\ell \leftarrow \mathcal{D}_{\mathbb{Z}^{2m\times\ell},\tau_2}$ $\mathbf{D}_\ell = [\mathbf{A}\vert\mathbf{A}_C]\cdot\mathbf{T}_\ell$	Lemma 3 (Item 1)
H_7	$\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$	$\downarrow$	$\downarrow$	Lemma 3 TrapGen
LLW-AB-IPFE[ℓ^*] reduction	$\mathbf{A} = \mathbf{A}_{\mathsf{LLW}}$ $\mathbf{A}_i = \mathbf{A}_{i,\mathsf{LLW}}$ for all $i \in [\lambda]$ $\mathbf{D}_{\ell^*} = \mathbf{D}_{\mathsf{LLW}}$	$\mathbf{c}_0 = \mathbf{c}_{0,\mathsf{LLW}}$ $\mathbf{c}_{1,j} = [\mathbf{c}_{2,j} \mid \mathbf{c}_{3,j} \mid \mathbf{c}_{4,j}] \cdot \mathbf{k}_{j,\ell^*}^\top + \mathbf{c}_{1,j,\mathsf{LLW}}$	$\mathbf{t}_{\mathbf{y}}^\top$ using $\begin{cases}\mathsf{sk}_{\mathbf{y},\mathsf{LLW}} & \text{if } \ell = \ell^*\\ \mathbf{T}_\ell \cdot \mathbf{y}^\top & \text{if } \ell \neq \ell^*\end{cases}$	LLW-AB-IPFE security

Fig. 8. Summary of the security proof of AB-UIPFE construction. $\downarrow$ denotes the same as the previous hybrid.

6.2 Security

The security proof follows similarly to the proof of the strict unbounded construction (Section 3.1) given in Section 3.2, except that we do not reduce it at the end to the security of the Π_{ALS} scheme (Appendix A), but we reduce it to the security of the $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ scheme described in Appendix C. We emphasize that the $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ scheme achieves selective security with respect to the challenge attribute (i.e., the adversary has to declare the challenge attribute before seeing the mpk), however, the challenge messages are adaptive (i.e., the adversary can declare the challenge messages after seeing the mpk and making key generation queries). Hence, our AB-UIPFE scheme inherits the same "attribute-selective message-adaptive" security of $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$. Nevertheless, we note that we can upgrade the security to the semi-adaptive setting using the techniques given in [LLW21], which are explained in Remark 3.

We describe here how the reduction to $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ works:

- the reduction receives the challenge attribute $\boldsymbol{\alpha}^*$ from the adversary $\mathcal{A}$ and forwards this to the LLW-AB-IPFE challenger,
- it receives $\mathsf{mpk}_{\mathsf{LLW}} = (\mathbf{A}_{\mathsf{LLW}}, \{\mathbf{A}_{i,\mathsf{LLW}}\}_{i\in[\lambda]}, \mathbf{D}_{\mathsf{LLW}}) \in \mathbb{Z}_q^{n\times m} \times (\mathbb{Z}_q^{n\times m})^\lambda \times \mathbb{Z}_q^{n\times\ell^*}$ from the LLW-AB-IPFE challenger,

– it samples $\mathbf{B}, \mathbf{W}, \mathbf{Y}, \mathbf{v}$ as in hybrid H_7 from Figure 8, while setting $\mathbf{A} \coloneqq \mathbf{A}_{\mathsf{LLW}}$ and $\mathbf{A}_i \coloneqq \mathbf{A}_{i,\mathsf{LLW}}$ for all $i \in [\lambda]$,
– it samples $\{\mathbf{T}_\ell\}_{\ell \in [T], \ell \neq \ell^*}$ and implicitly sets $\{\mathbf{D}_\ell\}_{j \in [T], \ell \neq \ell^*}$ as in hybrid H_7, while setting $\mathbf{D}_{\ell^*} \coloneqq \mathbf{D}_{\mathsf{LLW}}$ obtained from the LLW-AB-IPFE challenger (recall that T denotes the running time of $\mathcal{A}$ and we have that $\ell^* \leq T$),
– it samples $\mathbf{k}_{j,\ell}$ for all $j, \ell \in [T]$ as in hybrid H_7,
– it answers KeyGen queries for $\mathbf{y} \in \{0, \ldots, P-1\}^{\ell'}$ using $\mathbf{T}_{\ell'}$ when $\ell' \neq \ell^*$. For $\ell' = \ell^*$, it forwards the query to the LLW-AB-IPFE challenger, receives back $\mathsf{sk}_{\mathbf{y}}$, and forwards that to the adversary,
– whenever the adversary $\mathcal{A}$ outputs $(\mathbf{x}_0, \mathbf{x}_1)$, it forwards that to the LLW-AB-IPFE challenger and receives back

$$\mathbf{c}_{0,\mathsf{LLW}} \in \mathbb{Z}_q^{(\lambda+1)m}, \mathbf{c}_{1,\mathsf{LLW}} \in \mathbb{Z}_q^{\ell^*}.$$

It samples

$$\mathbf{c}_{2,j}, \mathbf{c}_{3,j}, \mathbf{c}_{4,j} \leftarrow \mathbb{Z}_q^m \quad \text{for } j \in [\ell^*],$$

and outputs

$$\mathsf{ct} \coloneqq \begin{pmatrix} \mathbf{c}_{0,\mathsf{LLW}} \\ \{[\mathbf{c}_{2,j} \mid \mathbf{c}_{3,j} \mid \mathbf{c}_{4,j}] \cdot \mathbf{k}_{j,\ell^*}^\top + \mathbf{c}_{1,j,\mathsf{LLW}}\}_{j \in [\ell^*]}, \\ \{\mathbf{c}_{2,j}\}_{j \in [\ell^*]}, \\ \{\mathbf{c}_{3,j}\}_{j \in [\ell^*]}, \\ \{\mathbf{c}_{4,j}\}_{j \in [\ell^*]}, \end{pmatrix}.$$

– whenever the adversary outputs a bit $b' \in \{0,1\}$, the reduction forwards it to the LLW-AB-IPFE challenger.

It is easy to verify that correctness is guaranteed and that the reduction is a valid LLW-AB-IPFE$[\ell^*]$ adversary.

The various hybrids are described in Figures 4 and 8. This result is summarized in the following theorem.

Theorem 9 (Security). *Let Π_4 be the AB-UIPFE construction described in Section 6, with parameters set as in Equation (2) and (6), and F be a PRF. Then, for any adversary $\mathcal{A}$ that runs is time $T = T(\lambda)$, there exists adversaries $\mathcal{B}_0$, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ against PRF-security, $\mathsf{LWE}_{n,m,\chi'',q}$, $\mathsf{LWE}_{n,m+\lambda,\chi'',q}$, and Sel-IND-CPA$_{\mathsf{LLW\text{-}AB\text{-}IPFE}[\ell^*]}$, respectively, such that*

$$\mathsf{Adv}^{\mathsf{Sel\text{-}IND\text{-}CPA}}_{\mathcal{A},\Pi_3}(\lambda) \leq T \cdot \Big(\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{B}_0,\mathsf{F}}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m,\chi'',q}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{LWE}_{n,m+\lambda,\chi'',q}}_{\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{Sel\text{-}IND\text{-}CPA}}_{\mathcal{B}_3,\mathsf{LLW}[\ell^*]}(\lambda) + \mathsf{negl}(\lambda) \Big).$$

Acknowledgements. We thank the anonymous reviewers for their very helpful comments and suggestions. The research of Erkan Tairi is (partially) supported by the France 2030 ANR Project ANR-22-PECY-003 SecureCompute, and by the Austrian Science Fund (FWF) Project J4879-N. The research of Valerio Cini is supported by the European Research Council through an ERC Starting Grant (grant agreement No. 101077455, ObfusQation) and an MSCA Postdoctoral Fellowship (grant agreement No. 101202597, ACryL).

References

ABB10. Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, Berlin, Heidelberg, May / June 2010. 13

ABDP15. Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Berlin, Heidelberg, March / April 2015. 1, 2

ABG19. Michel Abdalla, Fabrice Benhamouda, and Romain Gay. From single-input to multi-client inner-product functional encryption. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 552–582. Springer, Cham, December 2019. 1

ABKW19. Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. Decentralizing inner-product functional encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 128–157. Springer, Cham, April 2019. 1

ACF+18. Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 597–627. Springer, Cham, August 2018. 1

ACFQ22. Prabhanjan Ananth, Kai-Min Chung, Xiong Fan, and Luowen Qian. Collusion-resistant functional encryption for RAMs. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 160–194. Springer, Cham, December 2022. 5

ACGU20. Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 467–497. Springer, Cham, December 2020. 1, 2, 5, 6, 39

AGRW17. Michel Abdalla, Romain Gay, Mariana Raykova, and Hoeteck Wee. Multi-input inner-product functional encryption from pairings. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 601–626. Springer, Cham, April / May 2017. 1

AGT21a. Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-input quadratic functional encryption from pairings. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 208–238, Virtual Event, August 2021. Springer, Cham. 1

AGT21b. Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 224–255. Springer, Cham, November 2021. 1

AGT22. Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-input quadratic functional encryption: Stronger security, broader functionality. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 711–740. Springer, Cham, November 2022. 1

AGW20. Michel Abdalla, Junqing Gong, and Hoeteck Wee. Functional encryption for attribute-weighted sums from k-Lin. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 685–716. Springer, Cham, August 2020. 1

Ajt96. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996. 13

AKM+22. Shweta Agrawal, Fuyuki Kitagawa, Anuja Modi, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Bounded functional encryption for Turing machines: Adaptive security from general assumptions. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 618–647. Springer, Cham, November 2022. 5

ALMT20. Shweta Agrawal, Benoît Libert, Monosij Maitra, and Radu Titiu. Adaptive simulation security for inner product functional encryption. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 34–64. Springer, Cham, May 2020. 1, 2

ALS16. Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Berlin, Heidelberg, August 2016. 1, 2, 4, 5, 6, 38, 39, 41, 42

AMVY21. Shweta Agrawal, Monosij Maitra, Narasimha Sai Vempati, and Shota Yamada. Functional encryption for Turing machines with dynamic bounded collusion from LWE. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 239–269, Virtual Event, August 2021. Springer, Cham. 5

AS17. Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 152–181. Springer, Cham, April / May 2017. 1

ATY23. Shweta Agrawal, Junichi Tomida, and Anshu Yadav. Attribute-based multi-input FE (and more) for attribute-weighted sums. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 464–497. Springer, Cham, August 2023. 1

AV19. Prabhanjan Ananth and Vinod Vaikuntanathan. Optimal bounded-collusion secure functional encryption. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 174–198. Springer, Cham, December 2019. 1

BCFG17. Carmen Elisabetta Zaira Baltico, Dario Catalano, Dario Fiore, and Romain Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In Jonathan Katz and

Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 67–98. Springer, Cham, August 2017. 1

BCS23. Elie Bouscatié, Guilhem Castagnos, and Olivier Sanders. Pattern matching in encrypted stream from inner product encryption. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 774–801. Springer, Cham, May 2023. 1

BCTW16. Zvika Brakerski, David Cash, Rotem Tsabary, and Hoeteck Wee. Targeted homomorphic attribute-based encryption. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 330–360. Springer, Berlin, Heidelberg, October / November 2016. 14

BGG+14. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Berlin, Heidelberg, May 2014. 14

BJK15. Allison Bishop, Abhishek Jain, and Lucas Kowalczyk. Function-hiding inner product encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 470–491. Springer, Berlin, Heidelberg, November / December 2015. 1, 2

BSW11. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Berlin, Heidelberg, March 2011. 1, 11

BV16. Zvika Brakerski and Vinod Vaikuntanathan. Circuit-ABE from LWE: Unbounded attributes and semi-adaptive security. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 363–384. Springer, Berlin, Heidelberg, August 2016. 42

CDG+18. Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Decentralized multi-client functional encryption for inner product. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 703–732. Springer, Cham, December 2018. 1

CHKP10. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, Berlin, Heidelberg, May / June 2010. 13

CRS+24. Valerio Cini, Sebastian Ramacher, Daniel Slamanig, Christoph Striecks, and Erkan Tairi. (Inner-product) functional encryption with updatable ciphertexts. *Journal of Cryptology*, 37(1):8, January 2024. 1, 5

CW24. Valerio Cini and Hoeteck Wee. Unbounded ABE for circuits from LWE, revisited. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part IV*, volume 15487 of *LNCS*, pages 238–267. Springer, Singapore, December 2024. 6

DDM+23. Uddipana Dowerah, Subhranil Dutta, Aikaterini Mitrokotsa, Sayantan Mukherjee, and Tapas Pal. Unbounded predicate inner product functional encryption from pairings. *Journal of Cryptology*, 36(3):29, July 2023. 4

DP19. Edouard Dufour Sans and David Pointcheval. Unbounded inner-product functional encryption with succinct keys. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 2019*, volume 11464 of *LNCS*, pages 426–441. Springer, Cham, June 2019. 2, 3, 4, 5, 12

DP21. Pratish Datta and Tapas Pal. (Compact) adaptively secure FE for attribute-weighted sums from k-lin. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 434–467. Springer, Cham, December 2021. 1

DP23. Pratish Datta and Tapas Pal. Decentralized multi-authority attribute-based inner-product FE: Large universe and unbounded. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 587–621. Springer, Cham, May 2023. 2, 4

dPP22. Paola de Perthuis and David Pointcheval. Two-client inner-product functional encryption with an application to money-laundering detection. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 725–737. ACM Press, November 2022. 1

DPT22. Pratish Datta, Tapas Pal, and Katsuyuki Takashima. Compact FE for unbounded attribute-weighted sums for logspace from SXDH. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 126–159. Springer, Cham, December 2022. 1, 3

DRS04. Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540. Springer, Berlin, Heidelberg, May 2004. 13

ET25. Tenma Edamura and Atsushi Takayasu. Tight adaptive simulation security for identity-based inner-product FE in the (quantum) random oracle model. In Tibor Jager and Jiaxin Pan, editors, *PKC 2025, Part III*, volume 15676 of *LNCS*, pages 3–36. Springer, Cham, May 2025. 5

Gay20. Romain Gay. A new paradigm for public-key functional encryption for degree-2 polynomials. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 95–120. Springer, Cham, May 2020. 1

GGH+13. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013. 1

GKP+13. Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 555–564. ACM Press, June 2013. 1

GKPV10. Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In Andrew Chi-Chih Yao, editor, *ICS 2010*, pages 230–240. Tsinghua University Press, January 2010. 13

GKW16. Rishab Goyal, Venkata Koppula, and Brent Waters. Semi-adaptive security and bundling functionalities made generic and easy. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 361–388. Springer, Berlin, Heidelberg, October / November 2016. 2, 3, 4

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 13

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Berlin, Heidelberg, August 2013. 14

GVW12. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 162–179. Springer, Berlin, Heidelberg, August 2012. 1

HILL99. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. 13

JLL23. Aayush Jain, Huijia Lin, and Ji Luo. On the optimal succinctness and efficiency of functional encryption and attribute-based encryption. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 479–510. Springer, Cham, April 2023. 5

JLS21. Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 60–73. ACM Press, June 2021. 1, 5

JLS22. Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 670–699. Springer, Cham, May / June 2022. 1, 5

KNTY19. Fuyuki Kitagawa, Ryo Nishimaki, Keisuke Tanaka, and Takashi Yamakawa. Adaptively secure and succinct functional encryption: Improving security and efficiency, simultaneously. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 521–551. Springer, Cham, August 2019. 5

KPC+20. Vlasis Koutsos, Dimitrios Papadopoulos, Dimitris Chatzopoulos, Sasu Tarkoma, and Pan Hui. Agora: A privacy-aware data marketplace. Cryptology ePrint Archive, Report 2020/865, 2020. 2

Lin17. Huijia Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 599–629. Springer, Cham, August 2017. 1, 2

LLW21. Qiqi Lai, Feng-Hao Liu, and Zhedong Wang. New lattice two-stage sampling technique and its applications to functional encryption - stronger security and smaller ciphertexts. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 498–527. Springer, Cham, October 2021. 1, 3, 4, 5, 11, 27, 30, 33, 39, 40, 41, 42

LT19. Benoît Libert and Radu Titiu. Multi-client functional encryption for linear functions in the standard model from LWE. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 520–551. Springer, Cham, December 2019. 1

MKMS22. Jose Maria Bermudo Mera, Angshuman Karmakar, Tilen Marc, and Azam Soleimanian. Efficient lattice-based inner-product functional encryption. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part II*, volume 13178 of *LNCS*, pages 163–193. Springer, Cham, March 2022. 5

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Berlin, Heidelberg, April 2012. 13

MSH+19. Tilen Marc, Miha Stopar, Jan Hartman, Manca Bizjak, and Jolanda Modic. Privacy-enhanced machine learning with functional encryption. In Kazue Sako, Steve Schneider, and Peter Y. A. Ryan, editors, *ESORICS 2019, Part I*, volume 11735 of *LNCS*, pages 3–21. Springer, Cham, September 2019. 1

O'N10. Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. 1, 11

PD21. Tapas Pal and Ratna Dutta. Attribute-based access control for inner product functional encryption from LWE. In Patrick Longa and Carla Ràfols, editors, *LATINCRYPT 2021*, volume 12912 of *LNCS*, pages 127–148. Springer, Cham, October 2021. 1

RD25. Suprava Roy and Ratna Dutta. Toward practical lattice-based unbounded inner product functional encryption: Construction and implementation. Cryptology ePrint Archive, Paper 2025/2232, 2025. 5

RTV04. Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 1–20. Springer, Berlin, Heidelberg, February 2004. 3

Tom19. Junichi Tomida. Tightly secure inner product functional encryption: Multi-input and function-hiding constructions. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 459–488. Springer, Cham, December 2019. 1, 2

Tom23. Junichi Tomida. Unbounded quadratic functional encryption and more from pairings. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 543–572. Springer, Cham, April 2023. 1, 4

TT18. Junichi Tomida and Katsuyuki Takashima. Unbounded inner product functional encryption from bilinear maps. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 609–639. Springer, Cham, December 2018. 2, 3, 4, 5

TÜ24. Erkan Tairi and Akin Ünal. Lower bounds for lattice-based compact functional encryption. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 249–279. Springer, Cham, May 2024. 2

Üna20. Akin Ünal. Impossibility results for lattice-based functional encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 169–199. Springer, Cham, May 2020. 2

Wat15. Brent Waters. A punctured programming approach to adaptively secure functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 678–697. Springer, Berlin, Heidelberg, August 2015. 1

WFL19. Zhedong Wang, Xiong Fan, and Feng-Hao Liu. FE for inner products and its application to decentralized ABE. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 97–127. Springer, Cham, April 2019. 5, 6, 39, 41, 42

A ALS Inner-Product Functional Encryption Scheme

We recall here the lattice-based (bounded) IPFE scheme over $\mathbb{Z}$ from [ALS16], which we denote as Π_{ALS}. We assume that the message space is $\mathcal{V} = \{0, \ldots, V-1\}^\ell$, for some integer V, and secret keys are associated with vectors in $\mathcal{P} = \{0, \ldots, P-1\}^\ell$, for some integer P. For vectors of fixed length ℓ, the inner-product between the message and key vectors belongs to $\{0, \ldots, K_\ell - 1\}$, with $K_\ell = \ell \cdot P \cdot V$.

The scheme Π_{ALS} is defined as follows:

– $\mathsf{Setup}(1^\lambda, 1^\ell, \mathcal{P}, \mathcal{V})$: Sample

$$\mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m},$$
$$\mathbf{Z} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^{m \times \ell}, \tau}.$$

Compute $\mathbf{D} \coloneqq \mathbf{A} \cdot \mathbf{Z} \in \mathbb{Z}_q^{n \times \ell}$, set $\mathsf{mpk} \coloneqq (\mathbf{A}, \mathbf{D})$, and $\mathsf{msk} \coloneqq \mathbf{Z}$, and return $(\mathsf{mpk}, \mathsf{msk})$.

– $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x} \in \mathcal{V})$: Sample

$$\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^m, \chi}, \mathbf{e}_1 \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}^\ell, \chi}.$$

Parse $\mathsf{mpk} := (\mathbf{A}, \mathbf{D})$ and compute

$$\mathbf{c}_0 := \mathbf{s} \cdot \mathbf{A} + \mathbf{e}_0 \bmod q,$$
$$\mathbf{c}_1 := \mathbf{s} \cdot \mathbf{D} + \mathbf{e}_1 + \left\lfloor \frac{q}{K} \right\rfloor \cdot \mathbf{x} \bmod q.$$

Return $\mathsf{ct}_{\mathbf{x}} := (\mathbf{c}_0, \mathbf{c}_1)$.

– $\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y} \in \mathcal{P})$: Parse $\mathsf{msk} := \mathbf{Z}$, and compute

$$\mathbf{t}_{\mathbf{y}}^{\top} = \mathbf{Z} \cdot \mathbf{y}^{\top}$$

Return $\mathsf{sk}_{\mathbf{y}} := \mathbf{t}_{\mathbf{y}}$.

– $\mathsf{Dec}(\mathsf{sk}_{\mathbf{y}}, \mathbf{y}, \mathsf{ct}_{\mathbf{x}})$: Parse $\mathsf{ct}_{\mathbf{x}} := (\mathbf{c}_0, \mathbf{c}_1)$ and $\mathsf{sk}_{\mathbf{y}} := \mathbf{t}_{\mathbf{y}}$. Compute

$$\mu := \mathbf{c}_1 \cdot \mathbf{y}^{\top} - \mathbf{c}_0 \cdot \mathbf{t}_{\mathbf{y}}^{\top},$$

and

$$\mu' := \underset{\mu' \in \{-K_\ell+1, \ldots, K_\ell-1\}}{\arg\min} \left| \mu - \left\lfloor \frac{q}{K} \right\rfloor \cdot \mu' \right|.$$

Return μ'.

Parameters and Security. We consider the following parameter set for Π_{ALS} given in [ACGU20]:

$$\begin{aligned} q &\geq 2K_\ell \ell \sqrt{\ell} P \omega(\log^2 n) & m &\geq 2n \log q, \\ \chi &\geq 2C\alpha q(\sqrt{m} + \sqrt{n} + \sqrt{\ell}) & \tau &\geq \omega(\sqrt{\log n}), \end{aligned} \tag{7}$$

where C is some universal constant and α is a Gaussian parameter of the LWE assumption. More precisely, $\mathsf{LWE}_{n,m,\alpha,q}$ is reducible to lattice problems in dimension n, when $q\alpha \geq \Omega(\sqrt{n})$. Given the choice of parameters, we obtain the following theorem:

Theorem 10 ([ALS16, WFL19, ACGU20]). *Let n be the security parameter, $m \geq 2n \log q$ and q, χ, τ be defined as in Equation (7), where $\alpha \leq \frac{\chi}{2Cq(\sqrt{m}+\sqrt{n}+\sqrt{\ell})}$. Then, the scheme Π_{ALS} is adaptively $\mathsf{IND\text{-}CPA}$ secure, assuming that $\mathsf{LWE}_{n,m,\alpha,q}$ is hard.*

B LLW Identity-based Inner-Product Functional Encryption Scheme

We recall the lattice-based (bounded) IB-IPFE scheme over $\mathbb{Z}$ from [LLW21], which we denote as Π_{LLW}. As before, we assume that the message space is $\mathcal{V} = \{0, \ldots, V-1\}^\ell$, for some integer V, and secret keys are associated with vectors in $\mathcal{P} = \{0, \ldots, P-1\}^\ell$, for some integer P. For vectors of fixed length ℓ, the inner-product between the message and key vectors belongs to $\{0, \ldots, K_\ell - 1\}$, with $K_\ell = \ell \cdot P \cdot V$. We additionally assume that the identity space is $\{0,1\}^\lambda$.

We describe here a derandomized variant (i.e., the key generation is derandomized using a PRF) of the IB-IPFE scheme given in [LLW21]:

– $\mathsf{Setup}(1^\lambda, 1^\ell, \mathcal{P}, \mathcal{V})$: Sample

$$(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{A}_i \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m} \text{ for all } i \in [\lambda],$$
$$\mathbf{D} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times \ell}$$
$$\mathsf{k} \leftarrow\!\!\$\ \{0,1\}^\lambda.$$

Set $\mathsf{mpk} := (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\lambda]}, \mathbf{D})$, and $\mathsf{msk} := (\mathbf{T}_{\mathbf{A}}, \mathsf{k})$, and return $(\mathsf{mpk}, \mathsf{msk})$.

– $\mathsf{Enc}(\mathsf{mpk}, \mathsf{id} \in \{0,1\}^\lambda, \mathbf{x} \in \mathcal{V})$: Sample

$$\mathbf{s} \leftarrow\$ \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow\$ \mathcal{D}_{\mathbb{Z}^{2^m}, 2\chi\sigma}, \mathbf{e}_1 \leftarrow\$ \mathcal{D}_{\mathbb{Z}^\ell, 2\chi\sigma+\chi}.$$

Parse $\mathsf{mpk} \coloneqq (\mathbf{A}, \{\mathbf{A}_i\}_{i\in[\lambda]}, \mathbf{D})$ and $\mathsf{id} \coloneqq (b_1, \ldots, b_\lambda) \in \{0,1\}^\lambda$, set $\mathbf{A}_{\mathsf{id}} = \mathbf{G} + \sum_{i=1}^{\lambda}(b_i \mathbf{A}_i) \in \mathbb{Z}_q^{n\times m}$, and compute

$$\mathbf{c}_0 \coloneqq \mathbf{s} \cdot [\mathbf{A} \mid \mathbf{A}_{\mathsf{id}}] + \mathbf{e}_0 \bmod q,$$
$$\mathbf{c}_1 \coloneqq \mathbf{s} \cdot \mathbf{D} + \mathbf{e}_1 + \left\lfloor \frac{q}{K} \right\rfloor \cdot \mathbf{x} \bmod q.$$

Output $\mathsf{ct}_{\mathbf{x}} \coloneqq (\mathbf{c}_0, \mathbf{c}_1)$.

– $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{id}, \mathbf{y} \in \mathcal{P})$: Parse $\mathsf{msk} \coloneqq (\mathbf{T}_{\mathbf{A}}, \mathsf{k})$ and $\mathsf{id} \coloneqq (b_1, \ldots, b_\lambda) \in \{0,1\}^\lambda$, set $\mathbf{A}_{\mathsf{id}} = \mathbf{G} + \sum_{i=1}^{\lambda}(b_i \mathbf{A}_i) \in \mathbb{Z}_q^{n\times m}$, and sample

$$\mathbf{J} \leftarrow \mathsf{Sample}(\mathcal{D}_{\mathbb{Z}^{m\times\ell}, \rho}; \mathsf{F}(\mathsf{k}, (\mathsf{id}, \mathbf{y}, 1))),$$

where the Sample algorithm samples from the input distribution $\mathcal{D}_{\mathbb{Z}^{m\times\ell}, \rho}$ using the random coins $\mathsf{F}(\mathsf{k}, (\mathsf{id}, \mathbf{y}, 1))$, i.e., we sample $\mathbf{J}$ in a deterministic way. Set $\mathbf{U} = \mathbf{D} - \mathbf{A}\mathbf{J} \bmod q$, and sample

$$\mathbf{T} \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{A}_{\mathsf{id}}], \begin{bmatrix} \mathbf{T}_{\mathbf{A}} \\ \mathbf{0}_{m\times m} \end{bmatrix}, \mathbf{U}, \tau; \mathsf{F}(\mathsf{k}, (\mathsf{id}, \mathbf{y}, 2))\right).$$

Parse $\mathbf{T} \coloneqq \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \end{bmatrix}$, set $\mathbf{T}_{\mathsf{id}} \coloneqq \begin{bmatrix} \mathbf{J} + \mathbf{T}_1 \\ \mathbf{T}_2 \end{bmatrix} \in \mathbb{Z}_q^{2m\times\ell}$ and $\mathbf{t}_{\mathbf{y}}^\top \coloneqq \mathbf{T}_{\mathsf{id}} \cdot \mathbf{y}^\top$.
Return $\mathsf{sk}_{\mathbf{y}} \coloneqq \mathbf{t}_{\mathbf{y}}$.

– $\mathsf{Dec}(\mathsf{sk}_{\mathbf{y}}, \mathbf{y}, \mathsf{ct}_{\mathbf{x}})$: Parse $\mathsf{ct}_{\mathbf{x}} \coloneqq (\mathbf{c}_0, \mathbf{c}_1)$ and $\mathsf{sk}_{\mathbf{y}} \coloneqq \mathbf{t}_{\mathbf{y}}$. Compute

$$\mu \coloneqq \mathbf{c}_1 \cdot \mathbf{y}^\top - \mathbf{c}_0 \cdot \mathbf{t}_{\mathbf{y}}^\top,$$

and

$$\mu' \coloneqq \underset{\mu' \in \{-K_\ell+1, \ldots, K_\ell-1\}}{\arg\min} \left| \mu - \left\lfloor \frac{q}{K} \right\rfloor \cdot \mu' \right|.$$

Return μ'.

Parameters and Security. We consider the following parameter set for Π_{LLW} given in [LLW21]:

$$\begin{aligned} q &\geq 8K_\ell \chi m \sigma \rho + 2K\ell P(2\chi\sigma + \chi) & m &> 6n \log q \\ \chi &\geq 2C\alpha q(\sqrt{m} + \sqrt{n} + \sqrt{\ell}) & \tau &\geq m\lambda\omega(\sqrt{\log m}) \\ \sigma &\geq \sqrt{1 + C^2(\rho^2 + \chi^2) \cdot (2\sqrt{m} + \sqrt{\ell})^2 + \lambda^2 m^2} & \rho &\geq \tau m \lambda^{\omega(1)}, \end{aligned} \tag{8}$$

where C is some universal constant and α is a Gaussian parameter of the LWE assumption. Given the choice of parameters, we obtain the following theorem:

Theorem 11 ([LLW21]). *Let Π_{ALS} be the IPFE scheme with parameters given in Appendix A. If Π_{ALS} is adaptively* IND-CPA *secure under the $\mathsf{LWE}_{n,m,\alpha,q}$ assumption, then Π_{LLW} with parameters given in Equation (8) is also adaptively* IND-CPA *secure under the $\mathsf{LWE}_{n,m,\alpha,q}$ assumption.*

Remark 1. We note that the IB-IPFE construction from [LLW21] has a randomized key-generation algorithm, however, we consider here a derandomized variant of it, which is more suitable for the security proof of our IB-UIPFE construction from Section 5. We emphasize that the security proof of our $\Pi_{\mathsf{LLW\text{-}IB\text{-}IPFE}}$ scheme given here follows analogously to that given in [LLW21, Theorem 7.2], with the exception of one additional hybrid at the beginning of the security proof, where we replace PRF generated random coins with uniformly sampled coins via the PRF security.

C LLW Attribute-based Inner-Product Functional Encryption Scheme

We describe here a slightly modified variant of the lattice-based (bounded) AB-IPFE scheme from [LLW21], which we denote as $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$. The original construction from [LLW21, Section 7.2] worked over mod p and was reduced to N-ALS given in [WFL19], whereas here we consider a variant of the scheme that works over the integers (without modular reduction) and whose security reduces to plain ALS [ALS16]. Moreover, [LLW21] considered a randomized key generation algorithm, but for our purposes we derandomize it using a PRF as in Appendix B.

Analogous to the previous sections, we assume that the message space is $\mathcal{V} = \{0, \ldots, V-1\}^\ell$, for some integer V, and secret keys are associated with vectors in $\mathcal{P} = \{0, \ldots, P-1\}^\ell$, for some integer P. For vectors of fixed length ℓ, the inner-product between the message and key vectors belongs to $\{0, \ldots, K_\ell - 1\}$, with $K_\ell = \ell \cdot P \cdot V$. We assume without loss of generality that the attribute space is $\{0,1\}^\lambda$, and consider the family $\mathcal{C}$ of circuits of depth-d mapping $\{0,1\}^\lambda$ to $\{0,1\}$.

The scheme $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ is as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell, \mathcal{C}, \mathcal{P}, \mathcal{V})$: Sample

$$(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q),$$
$$\mathbf{A}_i \leftarrow\!\$\, \mathbb{Z}_q^{n \times m} \text{ for all } i \in [\lambda],$$
$$\mathbf{D} \leftarrow\!\$\, \mathbb{Z}_q^{n \times \ell}$$
$$\mathsf{k} \leftarrow\!\$\, \{0,1\}^\lambda.$$

  Set $\mathsf{mpk} := (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\lambda]}, \mathbf{D})$, and $\mathsf{msk} := (\mathbf{T}_\mathbf{A}, \mathsf{k})$, and return $(\mathsf{mpk}, \mathsf{msk})$.

- $\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{\alpha} \in \{0,1\}^\lambda, \mathbf{x} \in \mathcal{V})$: Sample

$$\mathbf{s} \leftarrow\!\$\, \mathbb{Z}_q^n, \mathbf{e} \leftarrow\!\$\, \mathcal{D}_{\mathbb{Z}^m, \chi}, \mathbf{e}_1 \leftarrow\!\$\, \mathcal{D}_{\mathbb{Z}^\ell, 2\chi\sigma + \chi},$$
$$\mathbf{R}_i \leftarrow\!\$\, \{0,1\}^{m \times m} \text{ for all } i \in [\lambda].$$

  Parse $\mathsf{mpk} := (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\lambda]}, \mathbf{D})$ and $\boldsymbol{\alpha} := (\alpha_1, \ldots, \alpha_\lambda) \in \{0,1\}^\lambda$, set

$$\mathbf{A}_{\boldsymbol{\alpha}} := [\alpha_1 \mathbf{G} + \mathbf{A}_1 \mid \cdots \mid \alpha_\lambda \mathbf{G} + \mathbf{A}_\lambda] \in \mathbb{Z}_q^{n \times \lambda m},$$
$$\mathbf{e}_0 := \mathbf{e} \cdot [\mathbf{I}_m \mid \mathbf{R}_1 \mid \cdots \mid \mathbf{R}_\lambda] \in \mathbb{Z}^{(\lambda+1)m},$$

  and compute

$$\mathbf{c}_0 := \mathbf{s} \cdot [\mathbf{A} \mid \mathbf{A}_{\boldsymbol{\alpha}}] + \mathbf{e}_0 \bmod q,$$
$$\mathbf{c}_1 := \mathbf{s} \cdot \mathbf{D} + \mathbf{e}_1 + \left\lfloor \frac{q}{K} \right\rfloor \cdot \mathbf{x} \bmod q.$$

  Output $\mathsf{ct}_{\boldsymbol{\alpha}, \mathbf{x}} := (\mathbf{c}_0, \mathbf{c}_1)$.

- $\mathsf{KeyGen}(\mathsf{msk}, C \in \mathcal{C}, \mathbf{y} \in \mathcal{P})$: Parse $\mathsf{msk} := (\mathbf{T}_\mathbf{A}, \mathsf{k})$ and set $\mathbf{A}_C := \mathsf{EvalC}\left(\{\mathbf{A}\}_{i \in [\lambda]}, C\right)$, and sample

$$\mathbf{J} \leftarrow \mathsf{Sample}(\mathcal{D}_{\mathbb{Z}^{m \times \ell}, \rho}; \mathsf{F}(\mathsf{k}, (C, \mathbf{y}, 1))),$$

  where the Sample algorithm samples from the input distribution $\mathcal{D}_{\mathbb{Z}^{m \times \ell}, \rho}$ using the random coins $\mathsf{F}(\mathsf{k}, (C, \mathbf{y}, 1))$, i.e., we sample $\mathbf{J}$ in a deterministic way. Set $\mathbf{U} = \mathbf{D} - \mathbf{A}\mathbf{J} \bmod q$, and sample

$$\mathbf{T} \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{A}_C], \begin{bmatrix} \mathbf{T}_\mathbf{A} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{U}, \tau; \mathsf{F}(\mathsf{k}, (C, \mathbf{y}, 2))\right).$$

  Parse $\mathbf{T} := \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \end{bmatrix}$, set $\mathbf{T}_C := \begin{bmatrix} \mathbf{J} + \mathbf{T}_1 \\ \mathbf{T}_2 \end{bmatrix} \in \mathbb{Z}_q^{2m \times \ell}$ and $\mathbf{t}_\mathbf{y}^\top := \mathbf{T}_C \cdot \mathbf{y}^\top$.
  Return $\mathsf{sk}_{C, \mathbf{y}} := \mathbf{t}_\mathbf{y}$.

– $\mathsf{Dec}(\mathsf{sk}_{C,\mathbf{y}}, \mathbf{y}, C, \mathsf{ct}_{\boldsymbol{\alpha},\mathbf{x}}, \boldsymbol{\alpha})$: Parse $\mathsf{ct}_{\boldsymbol{\alpha},\mathbf{x}} \coloneqq (\mathbf{c}_0, \mathbf{c}_1)$, $\mathbf{c}_0 \coloneqq (\mathbf{c}_{0,0}, \ldots, \mathbf{c}_{0,\lambda}) \in \mathbb{Z}_q^{(\lambda+1)m}$ and $\mathsf{sk}_{C,\mathbf{y}} \coloneqq \mathbf{t}_{\mathbf{y}}$. Compute

$$\begin{aligned} \mathbf{H}_{\mathbf{A},C,\boldsymbol{\alpha}} &\coloneqq \mathsf{EvalCX}(\mathbf{A}, C, \boldsymbol{\alpha}), \\ \mathbf{c}_C &\coloneqq [\mathbf{c}_{0,1} \mid \cdots \mathbf{c}_{0,\lambda}] \cdot \mathbf{H}_{\mathbf{A},C,\boldsymbol{\alpha}} \in \mathbb{Z}_q^m. \end{aligned}$$

Set $\mathbf{c}_0' \coloneqq [\mathbf{c}_{0,0} \mid \mathbf{c}_C] \in \mathbb{Z}_q^{2m}$ and compute

$$\begin{aligned} \mu &\coloneqq \mathbf{c}_1 \cdot \mathbf{y}^\top - \mathbf{c}_0' \cdot \mathbf{t}_{\mathbf{y}}^\top, \\ \mu' &\coloneqq \operatorname*{arg\,min}_{\mu' \in \{-K_\ell+1, \ldots, K_\ell-1\}} \left| \mu - \left\lfloor \frac{q}{K} \right\rfloor \cdot \mu' \right|. \end{aligned}$$

Return μ'.

Parameters and Security. The parameters can be set exactly as in Equation (8), except that we require the modulus q to have an additional factor of $\lambda^{O(d)}$ (due to the homomorphic evaluation procedure given in Theorem 1), where d is the depth of the circuits in $\mathcal{C}$. This gives us the following theorem:

Theorem 12 ([LLW21]). *Let Π_{ALS} be the IPFE scheme with parameters given in Appendix A. If Π_{ALS} is* IND-CPA *secure under the $\mathsf{LWE}_{n,m,\alpha,q}$ assumption, then $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ with parameters given in Equation (8) is selectively* IND-CPA *secure under the $\mathsf{LWE}_{n,m,\alpha,q}$ assumption.*

Remark 2. We emphasize that the security proof of the $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ scheme given here follows analogously to that given in [LLW21, Theorem 7.7], except for the following changes:

– The proof requires one additional hybrid at the beginning, where we replace PRF generated random coins with uniformly sampled coins via the PRF security.
– After the last hybrid, instead of doing a reduction to the security of the N-ALS scheme [WFL19], we reduce it to the ALS scheme given in Appendix A.

Remark 3. The (bounded) AB-IPFE construction from [LLW21, Section 7.2] achieves (Q, poly)-selective security with respect to the attributes (i.e., the adversary has to declare the challenge attribute before seeing the mpk, but the challenge messages are adaptive) by using the cover-free sets technique [LLW21, Section 5.2] and reducing the security to N-ALS [WFL19]. The authors also show how to upgrade the security to (Q, poly)-semi-adaptive security by using a semantically secure PKE scheme, following the techniques from [BV16]. We note that the $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$ scheme presented here is a simplified variant of the scheme from [LLW21, Section 7.2], and it only achieves $(1, \mathsf{poly})$-selective security. However, we emphasize that the cover-free sets technique and semi-adaptive upgrade method can also be applied to $\Pi_{\mathsf{LLW\text{-}AB\text{-}IPFE}}$, in order to boost the security to (Q, poly)-semi-adaptive security, albeit this would require to reduce its security to N-ALS [WFL19] instead of plain ALS [ALS16].