Indistinguishability Obfuscation from SXDH on 5-Linear Maps and Locality-5 PRGs

Huijia Lin[⋆]

University of California, Santa Barbara

Abstract. Two recent works [Lin, EUROCRYPT 2016, Lin and Vaikuntanathan, FOCS 2016] showed how to construct Indistinguishability Obfuscation (IO) from constant degree multilinear maps. However, the concrete degrees of multilinear maps used in their constructions exceed 30. In this work, we reduce the degree of multilinear maps needed to 5, by giving a new construction of IO from asymmetric L-linear maps and a pseudo-random generator (PRG) with output locality L and polynomial stretch. When plugging in a candidate PRG with locality-5 (*e.g.*, [Goldreich, ECCC 2010, Mossel, Shpilka, and Trevisan, FOCS 2013, O'Donnald and Wither, CCC 2014]), we obtain a construction of IO from *5-linear maps*.

Our construction improves the state-of-the-art at two other fronts: First, it relies on "classical" multilinear maps, instead of their powerful generalization of graded encodings. Second, it comes with a security reduction to i) the SXDH assumption on algebraic multilinear maps [Boneh and Silverberg, Contemporary Mathematics, Rothblum, TCC 2013], ii) the security of PRG, and iii) sub-exponential LWE, all with sub-exponential hardness. The SXDH assumption is weaker and/or simpler than assumptions on multilinear maps underlying previous IO constructions. When noisy multilinear maps [Garg, Gentry, and Halivi, EUROCRYPT 2013] are used instead, security is based on a family of more complex assumptions that hold in the generic model.

1 Introduction

Indistinguishability obfuscation, defined first in the seminal work of Barak *et al.* [11], aims to transform programs into "unintelligible" ones while preserving functionality. IO is an extradinarily powerful object and has been used as a central tool for obtaining a plethora of new cryptographic constructions, solutions to long-standing open problems, and techniques enabling new cryptographic goals.

Unfortunately, so far, the existence of IO remain uncertain. Most known candidate IO schemes [27,17,10,49,5,33,53,7,30,46,25] are built from the so-called *graded encoding schemes* [26], a framework of complex algebraic structures that, in essence, enables evaluating *polynomial-degree* polynomials on secret encoded values and revealing whether the output is zero or not. The security of most

[⋆] `rachel.lin@cs.ucsb.edu`. Huijia Lin was partially supported by NSF grants CNS-1528178, CNS-1514526, CNS-1652849 (CAREER).

IO candidates are either analyzed in the ideal model or based on strong uber assumptions [49], with only one exception [33]. On the front of instantiating graded encodings from concrete mathematical objects, the state of affairs is even more worrisome: Vulnerabilities have been demonstrated in all instantiations proposed so far [26,21,39,31,22]. Of course, this does not mean that the resulting IO constructions are insecure. In fact, this has motivated the search for IO constructions that withstand all existing attacks [29].

The state-of-affairs motivates the following natural question.

How much can we narrow the gap between
objects and assumptions that imply IO and well-studied ones,
such as, asymmetric bilinear maps with the SXDH assumption?

Two recent works [41,44] have made significant progress towards answering the question: Lin [41] showed that to construct IO, we do not need full-fledged graded encodings that support evaluation of all polynomial-degree polynomials, instead, it suffices to start with graded encodings for only constant-degree polynomials, called *constant-degree graded encodings*. Following that, Lin and Vaikuntanathan [44] further weakened the assumption on constant-degree graded encodings from a uber assumption in [41] to the so-called joint-SXDH assumption, which is a stronger variant of the classical SXDH assumption. Besides from multilinear maps, their IO constructions additionally rely on *PRGs in* NC^0 and sub-exponential LWE.

The trajectory of recent developments points towards the holly grail of "building IO from bilinear maps". In this work, we make new strides in this direction: We give a new construction of IO from asymmetric L-linear maps and a PRG with output locality L (*i.e.*, every output bit depends on at most L input bits). When plugging in a candidate PRG with locality-5 in the literature [34,47,48], we obtain a construction of IO from 5-*linear maps*. This gets the degree of multilinear maps needed for IO much closer to the dream version of 2. In comparison, previous IO constructions [41,44] rely on multilinear maps with degree at least 30. On the other hand, no PRGs with locality 4 exist [23,47]. Thus, our approach hits a barrier and cannot base IO on multilinear maps with degree $L \leq 4$. This barrier is common to recent IO constructions [41,44] and suggests that we need new techniques circumventing the lower bound on locality of PRGs.

In addition to reducing the degree of multilinear maps, our IO construction improves the state-of-the-art at two other fronts. First, our construction uses the classical asymmetric multilinear maps introduced in [15,50], which are direct generalization of bilinear pairing groups to higher degree. Previous constructions rely on graded encodings, which are enhanced versions of multilinear maps with more powerful functionalities (such as, supporting complex label structures). Second, the security of our IO scheme is based on the sub-exponential SXDH assumption on L-linear maps, the sub-exponential security of PRGs, and sub-exponential LWE. The SXDH assumption on multilinear maps is much simpler and/or weaker than the assumption on graded encodings underlying previous IO constructions, for instance, the joint-SXDH assumption in [44] and the multilinear subgroup elimination assumption in [33].

1.1 Our Results

We start with defining the SXDH assumption on multilinear maps and then describe our results.

SXDH on Multilinear Maps Asymmetric multilinear pairing groups introduced in [15,50] generalize asymmetric bilinear pairing maps to a collection of source groups $G_1, \cdots, G_D$, whose elements can be paired to produce elements in a target group G_T via a multilinear map $e(g_1^{a_1}, \cdots, g_D^{a_D}) = g_T^{a_1 \cdots a_D}$. The degree (a.k.a. multilinearity) of the multilinear map is the number of elements that can be paired together, which equals to the number of source groups D. We say that the multilinear pairing groups have *prime order* if all source groups and the target group have the same prime order, and *composite order* if all groups have the same composite order. In this work, we consider constant-degree multilinear paring groups, and in particular 5-linear pairing groups, with either prime or composite order. We omit specifying the order of groups below.

The SXDH assumption on asymmetric multilinear pairing groups is a natural generalization of the standard symmetric external Diffie-Hellman (SXDH) assumption on asymmetric bilinear pairing groups, introduced first in [50]. In short, SXDH states that the decisional Diffie-Hellman assumption holds in every source group: It postulates that the distribution of g_d^a, g_d^b, g_d^{ab} in any source group d should be indistinguishable to that of g_d^a, g_d^b, g_d^r. Formally, for all $d \in [D]$,

$$\left\{ g_d^a,\ g_d^b \xleftarrow{\$} G_d \ :\ \{g_i\}_{i \in [D]},\ g_d^a,\ g_d^b,\ g_d^{ab} \right\}$$
$$\approx \left\{ g_d^a,\ g_d^b,\ g_d^r \xleftarrow{\$} G_d \ :\ \{g_i\}_{i \in [D]},\ g_d^a,\ g_d^b,\ g_d^r \right\} ,$$

where $\{g_i\}$ is the set of generators in all groups. When $D = 2$, this gives the SXDH assumption on bilinear pairing groups.

Multilinear Maps v.s. Graded Encodings. The interface of (asymmetric) multilinear pairing groups is much simpler than that of graded encoding schemes introduced by [26]. First, graded encoding schemes support *graded multiplication* over a collection of groups $\{G_l\}$: Graded multiplication can pair elements of two groups G_{l_1}, G_{l_2}, indexed by two labels l_1, l_2, to produce an element in the group $G_{l_1+l_2}$, indexed by label $l_1 + l_2$[1]. In particular, the output element in $G_{l_1+l_2}$ can be further paired with elements in other groups to produce elements in group $G_{l_1+l_2+l_3+\cdots}$ and so on. In contrast, multilinear map allows only "one-shot" multiplication, where the output element belongs to the target group G_T that cannot be paired anymore. Second, graded encoding schemes support the notion of "pairable groups" in the sense that only elements from groups G_{l_1}, G_{l_2} that satisfy a "pairable" relation can be paired [2].

[1] The operation is according to some well-defined addition operation over the labels; for example, if labels are integers, + is integer addition, and if labels are sets, + is set union.

[2] For instance, if labels are sets, then two groups are pairable, if their label-sets l_1, l_2 are disjoint.

The support for graded multiplication between pairable groups provides powerful capabilities. In essence, GES allows one to "engineer" the labels of a set of group elements $\{g_{l_i}^{a_i}\}$, so that, only polynomials of certain specific forms can be evaluated on values in the exponent. In contrast, the simple interface of multilinear maps does not provide such capabilities.

SXDH v.s. Joint-SXDH Lin and Vaikuntanathan introduced the joint-SXDH assumption on graded encoding schemes, and showed that IO for $\mathsf{P/poly}$ can be based on sub-exponential joint-SXDH and PRG in NC^0. Their joint-SXDH assumption strengthens the SXDH assumption as follows: It considers the joint distribution of elements $(g_l^a, g_l^b, g_l^{ab})_{l \in S}$ in a set S of groups. The intuition is that as long as *no* pairs of groups G_{l_1}, G_{l_2} in the set S are pairable, in the same spirit as SXDH, the distribution is possibly indistinguishable to the joint distribution of elements $(g_l^a, g_l^b, g_l^r)_{l \in S}$ in the same set of groups. [3] The joint-SXDH assumption is more complex and potentially stronger than the SXDH assumption.

Our Main Result: IO from SXDH on L-Linear Maps and Local-L PRG

Theorem 1 (Main Theorem). *Let L be any positive integer. Assume the sub-exponential hardness of LWE with sub-exponential modulus-to-noise ratio. Then, IO for $\mathsf{P/poly}$ is implied by the sub-exponential SXDH assumption on L-linear pairing groups, and the existence of a sub-exponentially secure locality-L PRG with polynomial $n^{1+\varepsilon}$-stretch for some $\varepsilon > 0$.*

We note that the sub-exponential hardness of SXDH and PRG required by our theorem is weaker than standard notions of sub-exponential hardness of decisional problems, in the sense that we only require the distinguishing gap to be sub-exponentially small against polynomial time adversaries, as opposed to sub-exponential time adversaries .

Our result establishes a direct and tight connection between the degree D of multilinear maps needed for constructing IO and the locality L of PRGs — *they are the same $D = L$* — assuming sub-exponential LWE. In comparison, the previous state-of-the-art [44] requires the degree of the multilinear map to be much larger, namely $D > 6L$. Thus, when plugging-in a PRG of locality-5, their construction requires at least 30-linear maps, whereas our construction relies on 5-linear maps.

Step 1: Bootstrapping IO from Locality-L PRG and Degree-L FE We follow the same two-step approach in all previous IO constructions: First, construct IO for $\mathsf{P/poly}$ from some simpler primitives — call this the *bootstrapping step* — and then instantiate the primitives needed, using graded encodings or multilinear maps. In the literature, previous bootstrapping theorems have shown that general purpose IO can be built from one of the following: *i)* IO for NC^1 [27], *or ii)* sub-exponentially secure FE for NC^1 [2,14,3,13], *or iii)* sub-exponentially

[3] Note that in both distributions, the same exponents, a, b, r, are used in all groups in S.

secure IO for constant degree computations and PRG in NC^0 [41], *or iv)* sub-exponentially secure FE for NC^0 and PRG in NC^0 [44].[4]

In this work, we strengthen the bootstrapping theorem of [44], and show how to build IO from PRGs with locality-L and FE for computing degree L polynomials in some ring $\mathcal{R}$ (which eventually corresponds to the exponent space of multilinear maps used for instantiating the FE).

Theorem 2 (Bootstrapping Theorem). *Let L be any positive integer. Assume the sub-exponential hardness of LWE with sub-exponential modulus-to-noise ratio. IO for $\mathsf{P/poly}$ is implied by the existence of sub-exponentially secure (collusion resistant) secret-key FE schemes for computing degree-L polynomials in some ring $\mathcal{R}$ with linear efficiency, and a sub-exponentially secure locality-L PRG with $n^{1+\varepsilon}$-stretch for some $\varepsilon > 0$.*

(In the case that the FE schemes are public-key, the assumption of sub-exponential LWE is not needed.)

Above, the linear efficiency of FE schemes means that encryption time is linear in the input length $N(\lambda)$, that is, $\mathsf{Time}_{\mathsf{FE.Enc}} = N(\lambda)\,\mathrm{poly}(\lambda)$. In fact, we only need the FE scheme to achieve the weaker functionality of revealing whether the output of a degree-L polynomial is zero in $\mathcal{R}$. Below, we refer to such FE schemes as degree-L FE in $\mathcal{R}$ with linear efficiency.

In comparison, with locality-L PRG, the bootstrapping theorem in [44] needs to start with FE for computing polynomials with higher degree $3L+2$. We here reduce the degree of FE to exactly L, by proposing a new pre-processing idea: At a very high-level, we let the encryptor pre-process the input to be encrypted to perform part of the degree-$(3L+2)$ computations, and encrypt the processed values, so that later, the decryptor only need to perform a degree-L computation, and hence degree-L FE suffices. An overview of our bootstrapping step is given in Section 2.1.

Step 2: Degree Preserving Construction of FE Next, we construct degree-L FE based on the SXDH assumption on L-linear maps.

Theorem 3. *Let D be any positive integer and $\mathcal{R}$ any ring. Assuming SXDH on D-linear maps over ring $\mathcal{R}$, there exist secret key FE schemes for degree-D polynomials in $\mathcal{R}$, with linear efficiency.*

This new FE scheme is our main technical contribution. Previous constructions of FE for NC^1 either relies on IO for NC^1 or high degree multilinear maps [27,28], whose degree is polynomial in the circuit-size of the computations. In [44], Lin and Vaikuntanathan constructed FE for NC^0 from constant-degree graded encodings. Their construction, however, is *not* degree-preserving: To compute NC^0 functions that can be evaluated in degree D, they require degree $2D$ graded encodings. Our FE construction is the first one that supports degree-D computations using only degree-D multilinear maps.

[4] Some bootstrapping theorems additionally assume LWE [27,41] or the existence of public key encryption [13]).

It turns out that removing a factor of 2 in the degree requires completely new techniques for constructing FE. The reason is that the factor of 2 increase in degree allows the FE construction in [44] to evaluate instead of a degree-D computation directly, an arithmetic randomized encodings of the computation. The benefit is that they can rely on the security of randomized encoding to argue the security of FE. In our case, since the degree is exactly D, we cannot afford to "embed" any cryptographic primitives in the FE construction, and must come up with ways of encoding inputs and intermediate computation values using multilinear maps that directly guarantee security. For this reason, our construction share similar flavor with constructions of inner product encryptions based on bilinear maps. See Section 2.2 and 2.3 for an overview of our degree-preserving FE construction.

Additional Contributions Along the way of designing our degree-preserving FE construction, we also construct the following primitives that are of independent interests.

Simple Function Hiding IPE Schemes from SXDH on Bilinear Maps Without using the heavy hammers of multilinear maps or IO, the state-of-the-art collusion resistant FE schemes can only compute inner products, they are called Inner Product Encryption (IPE). In the literature, Abdalla, Bourse, De Caro and Pointcheval (ABDP) [1] came up with a public key IPE scheme based on one of a variety of assumptions, such as, DDH, Paillier, and LWE.

Bishop, Jain and Kowalczyk [12] (BJK) constructed the first secret-key IPE scheme based on the SXDH assumption over asymmetric bilinear pairing groups. Their scheme achieves a stronger security notion, called *weak function-hiding*, and is improved by [24] to achieve full *function hiding*. Lin and Vaikuntanathan [44] further showed that any weakly function hiding IPE scheme can be generically transformed into a function hiding IPE scheme. Here, (weak) function hiding requires the FE scheme to hide *both* inputs and functions (revealing only outputs), and is much harder to achieve than standard security that hides only inputs.

While the ABDP public-key IPE scheme is simple, the secret-key (weak) function hiding IPE schemes [12,24] are much more complex. In this work, we give a *simple* construction of weak function hiding IPE from SXDH on bilinear maps, which can then be transformed to function hiding IPE using [44]. Our IPE scheme is built from the ABDP public-key IPE scheme in a modular way, and inherits its efficiency and simplicity: Ciphertexts and secret keys of length-N vectors consists of $(N + 2)$ group elements, and the construction and security proof of our scheme fits within 2 pages (reducing to the security of the ABDP IPE scheme). In addition, the new scheme satisfies certain special properties that are important for our construction of degree-L FE schemes, which are not satisfied by previous IPE schemes [12,24]. See Section 2.5 for an overview of our simple function hiding IPE.

High-Degree IPE We also generalize IPE to the notion of *high-degree IPE*, or *HIPE* for short. They are *multi-input* FE schemes [35] for computing, so called, *degree-D inner product* defined as

$$\left\langle \mathbf{x}^1, \cdots, \mathbf{x}^D \right\rangle = \Sigma_{i \in [N]} x_i^1 x_i^2 \cdots x_i^D \ .$$

We construct HIPE for degree-D inner products from degree-D multilinear maps, which is then used to build degree-D FE schemes. We believe that this notion is interesting on its own and may have other applications. See Section 2.3 for an overview of HIPE.

Algebraic v.s. Noisy Multilinear Maps Our results and proofs are described w.r.t. algebraic multilinear maps. However, finding algebraic multilinear maps with degree above 2 is still a major open problem. *Can our IO and FE schemes be instantiated with known noisy multilinear map candidates [26,21,39,22,31]?* The answer is nuanced: The constructions can be instantiated as-is with noisy multilinear maps and correctness holds, but the security proof fails, for 1) the SXDH assumption does not hold on known candidates, and 2) the current security reduction relies on the homomorphic scalar multiplication functionality, which is not supported by known candidates. (The latter is shared with all previous reductions that base security on a laconic and instance-independent assumption [33,44].) Nevertheless, the security proof of the degree-L FE scheme (the only component that relies on multilinear maps) can be adapted into a proof in the degree-5 ideal multilinear map model *without* homomorphic scalar multiplication. Security in the ideal model does not imply security against known cryptanalytic attacks [26,19,32,16,20,46,6,18]. It is unclear whether these instantiations are secure against them — we have no concrete attacks nor formal arguments that validate their security against known attacks, such as, a security proof in the weak multilinear map model [29]. See Section 2.6 for a more detailed discussion.

1.2 Concurrent and Independent Work

In a concurrent work, Ananth and Sahai [4] (AS) showed a similar result. Both works convey the same high-level message that "IO can be constructed from 5-linear maps and locality-5 PRG, assuming sub-exponential LWE". But, the concrete theorem statements differ. First, while our construction relies on the classical 5-linear maps, the AS construction uses degree-5 *set-based* graded encodings, which, as discussed above, is more powerful. Second, a main contribution of this paper is basing security of IO on the SXDH assumption, which is laconic and instance dependent. In comparison, the AS construction is proven secure based on two assumptions on graded encodings that are tailored to their construction and justified in the ideal model, and the security of their FE scheme follows immediately from the assumptions. In terms of techniques, both works follow the paradigm of IO construction in [44]. The two works propose different notions of FE for low-degree polynomials, and use completely different methods to construct them.

1.3 Subsequent Works

Given that locality 4 PRGs do not exist [47], the approach (in this and recent works [44,4]) of using local PRGs to reduce the degree of multilinear maps used in IO constructions hits a barrier at degree 5. In a subsequent work, Lin and Tessaro [43] overcame this barrier and further reduced the degree of multilinear maps needed to 3. More specifically, they showed that assuming sub-exponential LWE, IO can be based on the SXDH assumption on L-linear maps and PRGs with a new notion of *block-wise locality* L. Roughly speaking, a PRG has block-wise locality L if every output bit depends on at most L *input blocks*, each containing up to $\log \lambda$ bits. Their result crucially relies on our IO construction, with the modification of replacing locality L PRGs with block-wise locality L PRGs in the first bootstrapping step (the rest of the construction, such as, the low-degree FE scheme, is kept the same). They further initiated the study of block-wise local PRGs based on Goldreich's local functions and their (in)security. In particular, they showed that the security of candidates with block-wise locality $L \geq 3$ is backed by similar validation as candidates with (conventional) locality 5. Soon after their work, two exciting cryptanalytic works [45,9] showed that, unfortunately, (polynomial-stretch) PRGs with block-wise locality 2 do not exist.

Summarizing the new state-of-the-art: Assuming sub-exponential LWE, there is a construction of IO from trilinear maps and PRGs with block-wise locality 3 — we are one degree away from the dream statement of "building IO from bilinear maps".

Organization Next, we proceed to give an overview of our FE and IO constructions and their security proofs. Due to the lack of space, we leave the formal description of constructions and proofs to the full version [42]. In Section 2.6, we discuss in more detail issues related to instantiating our schemes with noisy multilinear maps.

2 Overview

In this work, scalars are written in normal font, such as a, b, and vectors are written in boldface, such as $\mathbf{v}, \mathbf{w}$.

2.1 Bootstrapping

Our bootstrapping theorem follows the same two step approach as [41,44]. To construct IO for $\mathsf{P/poly}$,

Step 1. First, construct sub-exponentially secure single-key FE schemes **CFE** for NC^1 that are *weakly compact*, meaning that encryption time scales polynomially in the security parameter λ and the input length N, but also scales *sublinearly* in the maximal size S of the circuits for which secret keys are generated. More precisely, a FE scheme is said to be $(1-\varepsilon)$-weakly-compact if its encryption time is $\mathrm{poly}(\lambda, N)S^{1-\varepsilon}$.

Step 2. If the FE schemes obtained from Step 1 are public-key schemes, invoke the result of [2,14] that any public-key (single-key) weakly-compact FE schemes (for any $\varepsilon > 0$) imply IO for P/poly.
Otherwise, if the FE schemes obtained are secret-key schemes, then invoke the recent result of [13] that any secret-key weakly-compact FE schemes also imply IO for P/poly, assuming additionally sub-exponential LWE.

The challenging task is constructing (public-key or secret-key) weakly-compact FE schemes for NC^1 from simpler primitives. In [44] (LV), they constructed such schemes from (public key or secret key respectively) *collusion resistant* FE schemes for NC^0 with *linear efficiency*, assuming the existence of a polynomial-stretch PRG in NC^0. We observe that in their construction, if the PRG has locality L, the NC^0-FE scheme is used to compute polynomials with low degree $3L + 2$. In this work, we show that the degree of the FE schemes (*i.e.*, the degree of the polynomials supported) can be reduced to L. Below, we start with reviewing the LV construction of weakly-compact FE for NC^1, and then modify their construction to reduce the degree of the underlying FE scheme. (In the exposition below, we do not differentiate public-key vs secret-key schemes, since they are handled in the same way.)

The LV Weakly-Compact FE for NC^1. To construct weakly-compact FE schemes for NC^1 from FE schemes for NC^0, LV uses Randomized Encodings (RE) [37,8] to represent every NC^1 function $f(\mathbf{x})$, as a simpler NC^0 randomized function $\hat{f}(\mathbf{x};\ \mathbf{r})$. Then, to enable computing $f(\mathbf{x})$, it suffices to publish a secret key for $\hat{f} \in \mathsf{NC}^0$, and a ciphertext of $(\mathbf{x}, \mathbf{r})$, which can be done using the NC^0-FE scheme. But, the resulting ciphertext is not compact, since the randomness $\mathbf{r}$ for computing the randomized encoding is at least of length $S(\lambda)\,\mathrm{poly}(\lambda)$, where $S(\lambda)$ is the size of the circuit computing f. The key idea of LV is using a polynomial-stretch PRG $\mathbf{PRG} : \{0,1\}^n \to \{0,1\}^{n^{1+\alpha}}$ in NC^0 to generate pseudorandomness for RE, that is, computing instead $g(\mathbf{x}, \mathbf{s}) = \hat{f}(\mathbf{x}; \mathbf{PRG}(\mathbf{s}))$. Now the input of the function becomes $(\mathbf{x}, \mathbf{s})$, whose length is sublinear in $S(\lambda)$ thanks to the fact that the PRG has polynomial stretch. Since the NC^0-FE scheme has linear efficiency, the ciphertext size is also sublinear in $S(\lambda)$. In addition, the function g can still be computed in NC^0.

Observe that if g can be computed by a degree-D polynomial in some ring $\mathcal{R}$, then one can instantiate the LV construction with degree-D FE schemes in $\mathcal{R}$. The question is how large is the degree D? Plug in the randomized encoding scheme by Applebaum, Ishai, and Kushilevitz [8], whose encodings $\hat{f}(\mathbf{x}; \mathbf{r})$ are computable in NC^0_4 and has degree 1 in $\mathbf{x}$ and degree 3 in $\mathbf{r}$. Then, the degree of g is determined by the degree D_{PRG} of the PRG (*i.e.*, the minimal degree of polynomials that computes PRG in $\mathcal{R}$), namely, $D = 3D_{\mathrm{PRG}} + 1$. As the degree of PRG is upper bounded by its locality $D_{\mathrm{PRG}} \le L$, the degree of g is bounded by $3L+1$. For the security proof to work out, the actual functions used in the LV construction are more complicated and has degree $3L + 2$. For simplicity of this overview, it is convenient to ignore this issue, as it does not affect understanding the main ideas.

A formal description of the LV weakly-compact FE scheme $\mathbf{CFE}^{N,D,S}$ for NC^1 circuits with input-length $N = N(\lambda)$, depth $D = D(\lambda)$, and size $S = S(\lambda)$ can be found in Figure 1; it relies on the following tools:

- A (collusion resistant) FE scheme for degree-$(3D+2)$ polynomials $\{\mathbf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})\}$ in some ring $\mathcal{R}$ with linear efficiency.
- A pseudorandom generator **PRG** with $n^{1+\alpha}$-stretch for any $\alpha > 0$ that is computable in degree D in ring $\mathcal{R}$.
- A weak PRF F in NC^1.
- A specific randomized encoding scheme, which is the composition of Yao's garbling scheme [51,52] and the AIK randomized encoding scheme in NC^0 [8]. Denote by $\hat{C}_x = \mathsf{Yao}(C, \mathbf{x};\ \mathbf{r})$ Yao's garbling algorithm that compiles a circuit C and an input $\mathbf{x}$ into a garbled circuit $\hat{C}_x$, and by $\Pi = \mathsf{AIK}(f, \mathbf{x}\ ;\ \mathbf{r})$ the AIK encoding algorithm.

We refer the reader to [44] for the correctness and security of the scheme, and to our full version [42] for the analysis of compactness and degree.

Relying on Degree-L FE To reduce the degree of polynomials computed using the low-degree FE, our key idea is *pre-processing* the input $(\mathbf{x}, \mathbf{s})$, so that, part of the computation of the function g is already done at *encryption time*. To illustrate the idea, recall that g is linear in $\mathbf{x}$. Thus, if one pre-computes $\mathbf{x} \otimes \mathbf{s}$ (where $\mathbf{x} \otimes \mathbf{s}$ is the tensor product of $\mathbf{x}$ and $\mathbf{s}$), then g can be computed with one degree less. More specifically, there exists another function g' that takes input $(\mathbf{x}, \mathbf{s}, \mathbf{x} \otimes \mathbf{s})$ and computes $g(\mathbf{x}, \mathbf{s})$ in degree $3L$, by replacing every monomial of form $x_i s_{i_1} s_{i_2} \cdots$ with $(x_i s_{i_1})\ s_{i_2} \cdots$, where $x_i s_{i_1}$ is taken directly from $\mathbf{x} \otimes \mathbf{s}$. Therefore, we can modify the LV construction to encrypt $(\mathbf{x}, \mathbf{s}, \mathbf{x} \otimes \mathbf{s})$, whose length is still sublinear in $S(\lambda)$, and generate keys for functions g' that have degree $3L$.

The more tricky part is how to further reduce the degree of g in $\mathbf{s}$. The naive method of pre-computing $\mathbf{s} \otimes \mathbf{s}$ at encryption time would not work, since it would make encryption time exceed $S(\lambda)$, losing compactness. To avoid this, consider a simple case where the NC^1 function f to be computed is *decomposable*, in the sense that it has $I = S(\lambda)/\operatorname{poly}(\lambda)$ output bits, and every output bit $i \in [I]$ can be computed by a function f_i of fixed polynomial size $\operatorname{poly}(\lambda)$. (In fact, it is w.l.o.g. to assume this, since every function f can be turned into one that is decomposable using Yao's garbled circuits.) Then, the AIK randomized encoding of f consists of $\{\hat{f}_i(\mathbf{x}, \mathbf{r}[i])\}_{i \in [I]}$, where the random tape $\mathbf{r}[i]$ for every encoding has a fixed polynomial length $Q = \operatorname{poly}(\lambda)$, since $|f_i| = \operatorname{poly}(\lambda)$.

In LV, all the random tapes $\{\mathbf{r}[i]\}$ are generated by evaluating a PRG on a single seed $\mathbf{r} = \mathbf{PRG}(\mathbf{s})$. We first modify how these random tapes are generated. Parse $\mathbf{s}$ as Q equally-long seeds, $\mathbf{s}_1, \cdots \mathbf{s}_Q$, and use $\mathbf{s}_q$ to generate the q^{th} bit in all the random tapes, that is,

$$\forall\ q \in [Q],\ i \in [I], \quad \mathbf{r}[i]_q = \mathbf{PRG}(\mathbf{s}_q)|_i = \mathbf{PRG}_i(\{s_{q,\gamma}\}_{\gamma \in \Gamma(i)})\ ,$$

where $\mathbf{PRG}_i$ is the function that computes the i^{th} output bit of the PRG, which depends on at most L seed bits with indexes $\gamma \in \Gamma(i)$. $\mathbf{PRG}(\mathbf{s}_q)$ is a length-I

Single-key Compact FE Scheme CFE by [44]

<u>Setup:</u> $\mathsf{CFE.Setup}(1^\lambda)$ samples $(\mathsf{mpk}, \mathsf{msk}) \overset{\$}{\leftarrow} \mathsf{FE.Setup}(1^\lambda)$.

<u>Key Generation:</u> $\mathsf{CFE.KeyGen}(\mathsf{msk}, f)$ does the following:

- Sample $\mathbf{CT} \overset{\$}{\leftarrow} \{0,1\}^\ell$, where $\ell = \ell(\lambda)$ is set below.
- Define function g as follows: On input $\mathbf{x}$ of length N, a weak PRF key $\mathbf{k}$ of length $\mathrm{poly}(\lambda)$, two PRG seeds $\mathbf{s}, \mathbf{s}'$ each of length $\ell^{1/(1+\alpha)}$ and a bit b,

  $g(\mathbf{x}, \mathbf{k}, \mathbf{s}, \mathbf{s}', b)$ does the following:
  - Let $h_i(\mathbf{x}, \mathbf{k})$ denote the function that computes the i^{th} bit in Yao's garbling of $(f, \mathbf{x})$ using pseudo-randomness generated by a weak PRF

    $$\forall i \in [I], \qquad h_i(\mathbf{x}, \mathbf{k}) := \mathsf{Yao}_i(f, \mathbf{x}\ ;\ \mathbf{r} = \{r_j = \mathsf{F}(\mathbf{k}, j)\})\ ,$$

    where I is the length of Yao's garbling of $(f, \mathbf{x})$. (Note that $h \in \mathsf{NC}^1$ since Yao's garbling algorithm and the weak PRF are both computable in NC^1.)
  - If $b = 0$, for every $i \in [I]$, compute the AIK encoding $\Pi[i]$ of computation $(h_i, (\mathbf{x}, \mathbf{k}))$, using pseudo-randomness generated by a PRG

    $$\forall\ i \in [I], \qquad \Pi[i] = \mathsf{AIK}(h_i,\ (\mathbf{x}, \mathbf{k})\ ;\ \ \mathbf{r}[i])\ ,\ \text{where } \mathbf{r}[i] = \mathbf{PRG}[i](\mathbf{s})$$

    where $\mathbf{PRG}[i](\mathbf{s})$ denotes the i^{th} *portion* in the output of $\mathbf{PRG}$, and each portion has equal length $\mathrm{poly}(\lambda)$.
    Output $\Pi = \{\Pi[i]\}_i$.
  - If $b = 1$, output $\Pi = \mathbf{CT} \oplus \mathbf{PRG}(\mathbf{s}')$.

  For every $l \in [\ell = |\Pi|]$, let P_l denote the degree-$(3D+2)$ polynomial in $\mathcal{R}$ that computes the l^{th} output bit of g. (See the full version [42] for a proof that every output bit of g can indeed be computed by a degree-$(3D+2)$ polynomial in $\mathcal{R}$.)
- For every $l \in [\ell]$, generate a secret key $\mathsf{SK}_l \overset{\$}{\leftarrow} \mathsf{FE.KeyGen}(\mathsf{msk}, P_l)$ for P_l.

Output $\mathsf{SK} = \{\mathsf{SK}_l\}_{l \in [\ell]}$.

<u>Encryption:</u> $\mathsf{CFE.Enc}(\mathsf{mpk}, \mathbf{x})$ samples $\mathbf{k} \overset{\$}{\leftarrow} \{0,1\}^{\mathrm{poly}(\lambda)}$, $\mathbf{s}, \mathbf{s}' \overset{\$}{\leftarrow} \{0,1\}^{\ell^{1/(1+\alpha)}}$, and generates

$$\mathsf{CT} \overset{\$}{\leftarrow} \mathsf{FE.Enc}(\mathsf{mpk}, (\mathbf{x}, \mathbf{k}, \mathbf{s}, \mathbf{s}', 0))$$

<u>Decryption:</u> $\mathsf{CFE.Dec}(\mathsf{SK}, \mathsf{CT})$ computes $\Pi = \{\mathsf{FE.Dec}(\mathsf{SK}_l, \mathsf{CT})\}_{l \in [\ell]}$, parses $\Pi = \{\Pi[i]\}_{i \in I}$, and decodes every $\Pi[i]$ using the AIK decoding algorithm to obtain a garbled circuit, which is further decoded to obtain the output $f(\mathbf{x})$.

Fig. 1. Single-key Compact FE **CFE** by [44]

string, and hence the length $|\mathbf{s}_q|$ of each seed $\mathbf{s}_q$ is sublinear in $S(\lambda)$. Since each encoding $\hat{f}_i$ has degree 3 in its random tape $\mathbf{r}[i]$, consider an arbitrary degree 3 monomial $\mathbf{r}[i]_{q_1}\mathbf{r}[i]_{q_2}\mathbf{r}[i]_{q_2}$.

$$\begin{aligned}\mathbf{r}[i]_{q_1}\mathbf{r}[i]_{q_2}\mathbf{r}[i]_{q_2} &= \mathbf{PRG}_i(\{s_{q_1,\gamma}\}_{\gamma\in\Gamma(i)})\ \mathbf{PRG}_i(\{s_{q_2,\gamma}\}_{\gamma\in\Gamma(i)})\ \mathbf{PRG}_i(\{s_{q_3,\gamma}\}_{\gamma\in\Gamma(i)}) \\ &= \sum_{\substack{\text{Monomials} \\ X,Y,Z \text{ in } \mathbf{PRG}_i}} \begin{pmatrix} X(s_{q_1,\gamma_1},\cdots,s_{q_1,\gamma_L}) \\ \times\ Y(s_{q_2,\gamma_1},\cdots,s_{q_2,\gamma_L}) \\ \times\ Z(s_{q_3,\gamma_1},\cdots,s_{q_3,\gamma_L}) \end{pmatrix} ,\end{aligned}$$

where $\Gamma(i) = \{\gamma_1, \cdots, \gamma_L\}$. Now, suppose that for every index $\gamma \in [|\mathbf{s}_q|]$ in all seeds, the encryptor pre-compute all the degree ≤ 3 monomials over the γ^{th} bits in all Q seeds; denote this set as

$$M^3(\mathbf{s},\gamma) = \left\{ \text{ degree} \leq 3 \text{ monomials over } \{\mathbf{s}_{q,\gamma}\}_{q\in[Q]} \right\} .$$

Note that given $M^3(\mathbf{s},\gamma)$ for every $\gamma \in \Gamma(i)$, the above monomial $\mathbf{r}[i]_{q_1}\mathbf{r}[i]_{q_2}\mathbf{r}[i]_{q_2}$ can be computed in just degree L. Therefore, given $M^3(\mathbf{s},\gamma)$ for every $\gamma \in [|\mathbf{s}_q|]$, the function g can be computed in degree L (with additionally the above-mentioned trick for reducing the degree in $\mathbf{x}$). More precisely, there exists a degree-L polynomial g'' that, on input $\mathbf{x}$, $\{M^3(\mathbf{s},\gamma)\}_\gamma$, and their tensor product, computes $g(\mathbf{x},\mathbf{s})$.

Finally, we need to make sure that the total number of such degree ≤ 3 monomials is sublinear in $S(\lambda)$, so that, encryption remains weakly-compact. Note that, for each $\gamma \in [|\mathbf{s}_q|]$, the number of degree ≤ 3 monomials over the γ^{th} bits in these Q seeds is bounded by $(Q+1)^3 = \text{poly}(\lambda)$. Moreover, the length of each seed $|s_q|$ is still sublinear in $S(\lambda)$. Thus, the total number of monomials to be pre-computed is sublinear in $S(\lambda)$.

A formal description of our weakly-compact FE scheme can be found in Figure 2. Important difference from the LV scheme is highlighted with red underline.

2.2 Quadratic Secret-Key FE

Before proceeding to constructing degree-D FE schemes from SXDH on degree-D MMaps, we describe a self-contained construction of FE for quadratic polynomials from SXDH on bilinear maps. The degree-D scheme is a generalization of the quadratic scheme.

We start with reviewing the tool, Inner Product Encryption (IPE), for constructing quadratic FE. A (public key or secret key) IPE scheme allows to encode vectors $\mathbf{y}$ and $\mathbf{x}$ in a ring $\mathcal{R}$, in a function key $\mathsf{iSK}(\mathbf{y})$ and ciphertext $\mathsf{iCT}(\mathbf{x})$ respectively, and decryption evaluates the inner product $\langle \mathbf{y}, \mathbf{x} \rangle$. In this work (like in [44]), we use specific IPEs that compute the inner product *in the exponent*, which, in particular, allows the decryptor to test whether the inner product is zero, or whether it falls into any polynomial-sized range. [5]

[5] Such IPEs should be contrasted with functional encryption for testing the orthogonality of two vectors (see, e.g., [38,40] and many others), which reveals *only* whether

Our Single-key Compact FE Scheme CFE

SETUP: $\mathsf{CFE.Setup}(1^\lambda)$ samples $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{FE.Setup}(1^\lambda)$.

KEY GENERATION: $\mathsf{CFE.KeyGen}(\mathsf{msk}, f)$ does the following:

- Sample $\mathbf{CT} \xleftarrow{\$} \{0,1\}^\ell$, where $\ell = \ell(\lambda)$ is set below.
- Define function g defined as follows: On input $\mathbf{x}$ of length N, a weak PRF key $\mathbf{k}$ of length $\mathrm{poly}(\lambda)$, PRG seeds $\mathbf{s}$ and $\mathbf{s}'$ of length $I^{1/(1+\alpha)} \times Q$ and $\ell^{1/1+\alpha}$ respectively, and a bit b,

  $g(\mathbf{x}, \mathbf{k}, \mathbf{s}, \mathbf{s}', b)$ does the following:
  - Let $h_i(\mathbf{x}, \mathbf{k})$ denote the function that computes the i^{th} bit in Yao's garbling of $(f, \mathbf{x})$,

    $$\forall i \in [I], \qquad h_i(\mathbf{x}, \mathbf{k}) := \mathsf{Yao}_i(f, \mathbf{x}\ ;\ \mathbf{r} = \{r_j = \mathsf{F}(\mathbf{k}, j)\})\ ,$$

    where I is the length of Yao's garbling of $(f, \mathbf{x})$.
  - If $b = 0$, parse $\mathbf{s}$ into Q strings, $\mathbf{s} = \mathbf{s}_1 || \cdots || \mathbf{s}_Q$, of equal length $I^{1/(1+\alpha)}$, and compute

    $$\forall\ i \in [I],\ \ \Pi[i] = \mathsf{AIK}(h_i,\ (\mathbf{x}, \mathbf{k})\ ;\ \mathbf{r}[i])\ ,$$
    $$\text{where } Q = |\mathbf{r}[i]| \text{ and } \forall\ q \in [Q]\ ,\ \mathbf{r}[i]_q = \mathbf{PRG}_i(\mathbf{s}_q)$$

    Output $\Pi = \{\Pi[i]\}_i$.
  - If $b = 1$, output $\Pi = \mathbf{CT} \oplus \mathbf{PRG}(\mathbf{s}')$.

  For every $l \in [\ell = |\Pi|]$, let P_l denote the degree-$(3D+2)$ polynomial in $\mathcal{R}_\lambda$ that computes the l^{th} output bit of g. Moreover, define

  $$P'_l((1||\mathbf{x}||\mathbf{k}||b) \otimes (1||\mathbf{S}), (b(\mathbf{x}||\mathbf{k})) \otimes \mathbf{S}, (1||b) \otimes (1||\mathbf{s}'))$$
  $$:= \text{ The degree } L \text{ polynomial that computes } P_l(\mathbf{x}, \mathbf{k}, \mathbf{s}, \mathbf{s}', b) \text{ in Figure 1}$$

  where L is the locality of $\mathbf{PRG}$ and
  $\mathbf{S} = \{(1||\mathbf{s}_{\star,\gamma}) \otimes (1||\mathbf{s}_{\star,\gamma}) \otimes (1||\mathbf{s}_{\star,\gamma})\}_{\gamma \in [I^{1/(1+\alpha)}]}$.
- For every $l \in [\ell]$, generate a secret key $\mathsf{SK}_l \xleftarrow{\$} \mathsf{FE.KeyGen}(\mathsf{msk}, P'_l)$ for P'_l.

Output $\mathsf{SK} = \{\mathsf{SK}_l\}_{l \in [\ell]}$.

ENCRYPTION: $\mathsf{CFE.Enc}(\mathsf{mpk}, \mathbf{x})$ samples $\mathbf{k} \xleftarrow{\$} \{0,1\}^{\mathrm{poly}(\lambda)}$, $\mathbf{s} \xleftarrow{\$} \{0,1\}^{I^{1/(1+\alpha)} \times Q}$, and $\mathbf{s}' \xleftarrow{\$} \{0,1\}^{\ell^{1/(1+\alpha)}}$, and generates

$$\mathsf{CT} \xleftarrow{\$} \mathsf{FE.Enc}(\mathsf{mpk}, (1||\mathbf{x}||\mathbf{k}||0) \otimes (1||\mathbf{S}),\ \ (0(\mathbf{x}||\mathbf{k})) \otimes \mathbf{S},\ \ (1||0) \otimes (1||\mathbf{s}'))$$

DECRYPTION: $\mathsf{CFE.Dec}(\mathsf{SK}, \mathsf{CT})$ computes $\Pi = \{\mathsf{FE.Dec}(\mathsf{SK}_l, \mathsf{CT})\}_{l \in [\ell]}$, parse $\Pi = \{\Pi[i]\}_{i \in I}$, and decodes every $\Pi[i]$ using the AIK decoding algorithm to obtain a garbled circuit, which is further decoded to obtain the output $f(\mathbf{x})$.

Fig. 2. Single-key Compact FE **CFE** from locality-L PRG and degree-L FE

Given IPE schemes, it is trivial to implement FE for quadratic polynomials, or quadratic FE schemes for short: Simply write a quadratic function f as a linear function over quadratic monomials $f(x) = \Sigma_{i,j} c_{i,j} x_i x_j = \langle \mathbf{c}, \mathbf{x} \otimes \mathbf{x} \rangle$. Then, generate an IPE secret key $\mathsf{iSK}(\mathbf{c})$, and an IPE ciphertext $\mathsf{iSK}(\mathbf{x} \otimes \mathbf{x})$, from which the function output can be computed. However, such a scheme has encryption time quadratic in the input length $N = |\mathbf{x}|$. The key challenge is improving the encryption time to be linear in the input length under standard assumptions (*e.g.* bilinear maps).

In this work, we do so based on SXDH on bilinear maps, where the exponent space $\mathcal{R}$ of the bilinear map is the ring in which quadratic polynomials are evaluated. At a high-level, our key idea is "compressing" the encryption time of the above trivial quadratic FE schemes from quadratic to linear, by publishing some "compressed information" of linear size at encryption time, which can be expanded to an IPE ciphertext of $\mathbf{x} \otimes \mathbf{x}$ at decryption time. To make this idea work, we will use, as our basis, the public key IPE scheme by Abdalla, Bourse, De Caro, Pointcheval (ABDP) [1] based on the DDH assumption; we briefly review their scheme.

The ABDP public key IPE scheme The ABDP scheme **IPE** resembles the El Gamal encryption and is quite simple. Let G be a cyclic group of order p with generator g, in which DDH holds. A master secret key of the ABDP scheme is a random vector $\mathbf{s} = s_1, \cdots, s_N \xleftarrow{\$} \mathbb{Z}_p^N$, and its corresponding public key is $\mathsf{iMPK} = g^{s_1}, \cdots g^{s_N}$. A ciphertext encrypting a vector $\mathbf{x} = x_1, \cdots, x_N$ looks like $\mathsf{iCT} = g^{-r},\ g^{rs_1 + x_1},\ \cdots,\ g^{rs_N + x_N}$, where r is the random scalar "shared" for encrypting every coordinate. It is easy to see that it follows from DDH that this encryption is semantically secure.

To turn the above scheme into an IPE, observe that given a vector $\mathbf{y} \in \mathbb{Z}_p^N$, and in addition the inner product $\langle \mathbf{y}, \mathbf{s} \rangle$ in the clear, one can homomorphically compute inner product in the exponent to obtain $g^{-r\langle \mathbf{y}, \mathbf{s} \rangle} g^{r\langle \mathbf{s}, \mathbf{y} \rangle + \langle \mathbf{x}, \mathbf{y} \rangle} = g^{\langle \mathbf{x}, \mathbf{y} \rangle}$, which reveals whether the inner product $\langle \mathbf{x}, \mathbf{y} \rangle$ is zero or not. Therefore, the ABDP scheme sets the secret key to be $\mathsf{iSK} = \langle \mathbf{s}, \mathbf{y} \rangle \,||\mathbf{y}$.

In this work, we will use the bracket notation $[x]_l = g_l^x$ to represent elements in group G_l, and omit l when there is no need to specify the group. Under this notation, the ABDP scheme can be written as,

$$\mathsf{iMSK} = \mathbf{s} \xleftarrow{\$} \mathbb{Z}_p, \quad \mathsf{iMPK} = [\mathbf{s}], \quad \mathsf{iCT} = [-r \mid\mid (r\ \mathbf{s} + \mathbf{x})] \quad \mathsf{iSK} = \langle \mathbf{s}, \mathbf{y} \rangle \,||\mathbf{y}$$

where $a\mathbf{u}$ denotes coordinate-wise multiplication with a scalar a and $\mathbf{u}+\mathbf{v}$ denotes coordinate-wise addition between two vectors. We will also refer to $[x]_l$ as an encoding of x in group G_l.

Compress an ABDP ciphertext $\mathsf{iCT}(\mathbf{x} \otimes \mathbf{x})$ The first difficulty with "compressing" a ciphertext $\mathsf{iCT} = \mathsf{iCT}(\mathbf{x} \otimes \mathbf{x}) = [-r \mid\mid (r\ \mathbf{s} + \mathbf{x} \otimes \mathbf{x})]$ is that it contains

the inner product is zero and nothing else. In particular, they do not compute the inner product in the exponent in a way that allows for further computation, which is needed for our quadratic FE construction.

information of the master secret key $\mathbf{s}$ of quadratic length, which is truely random and cannot be "compressed".

Our idea is replacing the truly random secret key $\mathbf{s}$ with the tensor product of two length-N vectors $\mathbf{s}^1 \otimes \mathbf{s}^2$, so that, the new ciphertext depends only on information, namely $(r, \mathbf{s}^1, \mathbf{s}^2, \mathbf{x})$, of linear size. The reason that we use the tensor product $\mathbf{s}^1 \otimes \mathbf{s}^2$ as the secret key is that under DDH, encodings $\left[\mathbf{s}^1 \otimes \mathbf{s}^2\right]$ is indistinguishable to encodings of N^2 truely random elements, and hence there is hope that $\mathbf{s}^1 \otimes \mathbf{s}^2$ is "as good as" a truly random master secret key. As we will see later, this hope is true, however through complicated security proof.

Now, it is information theoretically possible to compress $\mathsf{iCT}(\mathbf{x} \otimes \mathbf{x})$. However, simply publishing $(r, \mathbf{s}^1, \mathbf{s}^2, \mathbf{x})$ would blatantly violate security. We need a way to securely and succinctly encode them so that only the ciphertext iCT is revealed. Classical cryptographic tools for hiding computation like garbled circuits or randomized encodings do not help here, since the output length is quadratic, and garbled circuits or randomized encodings have at least quadratic size too. Instead, we leverage the special structure of iCT: Each of the last N^2 encodings of iCT encodes an element that is the inner product of two length-2 vectors,

$$\mathsf{iCT}[0] = [-r], \quad \Big(\mathsf{iCT}[i,j] = [\ \langle\ x_i || s^1_i\ ,\ x_j || r s^2_j\ \rangle\ ]\Big)_{i \in [N], j \in [N]}$$

Here, for convenience, we use 0 and $\{(i,j)\}$ to index different encodings in iCT.

Suppose that we have a (secret key) IPE scheme **cIPE** that is function hiding (defined shortly) from bilinear maps, and has certain *canonical form*: In particular, its ciphertexts and secret keys encodes the input and function vectors in different source groups G_1, G_2 of the bilinear map, and decryption simply uses pairing to produce an *encoding of the output inner product* in the target group G_3. (Unfortunately, off-the-shelf function hiding IPEs [12,44,24] do not have the canonical form and we discuss how to construct such a scheme later.)

Then, we can use a canonical function hiding IPE, to generate the last N^2 encodings $\{\mathsf{iCT}[i,j]\}$: Publish N ciphertext $\{\mathsf{cCT}_i\}$ where each cCT_i encrypts vector $(x_i || s^1_i)$, and N secret keys $\{\mathsf{cSK}_j\}$ where each cSK_j encrypts vector $(x_j || r s^2_j)$. To obtain the $(i,j)^{\text{th}}$ encoding, one can simply decrypt the i^{th} ciphertext using the j^{th} secret key, which produces

$$\mathsf{iCT}[i,j] = [\ \langle\ x_i || s^1_i\ ,\ x_j || r s^2_j\ \rangle\ ] = \mathsf{cIPE.Dec}(\mathsf{cSK}_j, \mathsf{cCT}_i)$$

In order to hide r, x_j's, and s^1_j, s^2_j's, it is necessary that the IPE scheme is function hiding, which guarantees that secret keys and ciphertexts for two sets of vectors $\{\mathbf{u}_i, \mathbf{v}_i\}$ and $\{\mathbf{u}'_i, \mathbf{v}'_i\}$ are indistinguishable if they produce identical inner products $\langle \mathbf{u}_i, \mathbf{v}_j \rangle = \langle \mathbf{u}'_i, \mathbf{v}'_j \rangle$. The hope is that function hiding is also sufficient, as, intuitively, it ensures that only the set of possible outputs $\{\mathsf{iCT}[i,j]\}$ is revealed, and all other information of $(r, \mathbf{x}, \mathbf{s}^1, \mathbf{s}^2)$ is hidden. (This intuition is not precise, as the IPE scheme is not simulation-secure, but is a good starting point.)

In summary, we now have the first version of our quadratic FE schemes.

Version 1 of Our Secret Key Quadratic FE scheme **qFE**

- Setup: A master secret key msk consists of two random vectors $\mathbf{s}^1, \mathbf{s}^2$ of length N.
- Key Generation: A secret key $\mathsf{SK}(\mathbf{c})$ of a function $f_{\mathbf{c}}(\mathbf{x}) = \langle \mathbf{c}, \mathbf{x} \otimes \mathbf{x} \rangle$ consists of

$$\mathsf{SK}(\mathbf{c}) = \left(\left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c} \right\rangle, \ \mathbf{c} \ \right) \ .$$

- Encryption: Sample a random scalar $r \xleftarrow{\$} \mathbb{Z}_p$. A ciphertext $\mathsf{CT}(\mathbf{x})$ of input vector $\mathbf{x}$ contains

$$\mathsf{CT}(\mathbf{x}) = \Big(\ [-r], \ \{\mathsf{cCT}_i(\boldsymbol{\chi}_i^1), \ \mathsf{cSK}_i(\boldsymbol{\chi}_i^2)\}_{i \in [N]} \ \Big)$$

$$\text{where } \boldsymbol{\chi}_i^d = \begin{cases} x_i || s_i^1 & \text{if } d = 1 \\ x_i || r s_i^2 & \text{if } d = 2 \end{cases} \tag{1}$$

  and $\{\mathsf{cSK}_j, \ \mathsf{cCT}_i\}$ are generated using a *freshly sampled* master secret key cMSK of a canonical function hiding IPE **cIPE**.
- Decryption: For every $(i, j) \in [N]^2$, decrypt cCT_i using cSK_j to obtain

$$\mathsf{cIPE.Dec}(\mathsf{cSK}_j, \mathsf{cCT}_i) = \left[\left\langle \boldsymbol{\chi}_i^1, \boldsymbol{\chi}_j^2 \right\rangle\right] = \left[r s_i^1 s_j^2 + x_i x_j\right] = \mathsf{iCT}[i, j] \ . \tag{2}$$

  Homomorphically compute $\varLambda_1 = \left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c} \right\rangle [-r] = \left[-r \left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c} \right\rangle\right]$, and $\varLambda_2 = \langle \{\mathsf{iCT}[i, j]\} \ , \ \mathbf{c} \rangle$. Homomorphically add $\varLambda_1 + \varLambda_2$ to produce an encoding of the output $[f_{\mathbf{c}}(\mathbf{x})]$.

Next, we move to describing ideas for the security proof. As we develop the proof ideas, we will need to make several modifications to the above scheme.

Selective IND-Security of Our Quadratic FE Scheme. We want to show that ciphertexts of **qFE** of one set of inputs $\{\mathbf{u}_i\}$ is indistinguishable from that of another $\{\mathbf{v}_i\}$, as long as all the secret keys published are associated with functions $\{f_{\mathbf{c}_j}\}$ that do not separate these inputs, that is, $f_{\mathbf{c}_j}(\mathbf{u}_i) = f_{\mathbf{c}_j}(\mathbf{v}_i)$ for all i, j. For simplicity of this overview, we restrict our attention to the simpler case where only a single ciphertext and many secret keys are published. The security proof for the general case with many ciphertexts follows from a hybrid argument where the encrypted vectors are switched one by one from $\mathbf{u}_i$ to $\mathbf{v}_i$, and the indistinguishability of each step is proven using the same ideas to the single-ciphertext case.

Naturally, we want to reduce the security of **qFE** the security of the ABDP IPE scheme **IPE** and the function hiding of **cIPE**. Our intuition is that given a ciphertext $\mathsf{CT}(\mathbf{x})$ for $\mathbf{x} = \mathbf{u}$ or $\mathbf{v}$, the security of **cIPE** ensures that the N ciphertexts and secret keys $\{\mathsf{cCT}_i\}, \{\mathsf{cSK}_j\}$ contained in ciphertext $\mathsf{CT}(\mathbf{x})$ reveals only the output encodings $\{\mathsf{iCT}[i, j]\}$ and nothing else. Then, the security of the ABDP scheme ensures that the derived ciphertext iCT encrypting either $\mathbf{u} \otimes \mathbf{u}$ or $\mathbf{v} \otimes \mathbf{v}$ is indistinguishable, at the presence of secret keys for vectors $\{\mathbf{c}_j\}$ that do not separate them. This intuition would go through if the two building blocks **cIPE** and **IPE** provide very strong security guarantees: Naturally, **cIPE** has simulation security, so that, its ciphertexts and secret keys $\{\mathsf{cCT}_i\}, \{\mathsf{cSK}_j\}$ can be simulated from the set of output encodings $\{\mathsf{iCT}[i,j]\}$, and second, the ABDP scheme is secure even when the master secret keys are generated as a tensor product $\mathbf{s}^1 \otimes \mathbf{s}^2$ as opposed to be truely random. Unfortunately, our building blocks do not provide such strong security guarantees, which leads to the following challenges.

- *Challenge 1 — Relying only on indistinguishability-based function hiding of* **cIPE**. The simulation security of **cIPE** essentially allows one to easily reduce the security of **qFE** to that of **IPE**. With only indistinguishability-based security of **cIPE**, the reduction to security of **IPE** becomes significantly harder. Typically, one build a black-box security reduction that receives from its challenger **IPE** secret keys and a ciphertext, in this case $\{\mathsf{SK}_j\}, \mathsf{iCT}$, and embeds them in the view of the adversary attacking the **qFE** scheme. However, the ciphertext CT of **qFE** has only linear size, but iCT has quadratic size — there is not enough space for embedding. [6]

  To resolve this problem, our idea is to *embed* iCT *in "piecemeal"*. Observe that the ABDP scheme encrypts its input vector *element by element* using different master secret key elements, and a shared random scalar. Thus, we can flexibly view its ciphertext iCT either as a single ciphertext, or as a list of many ciphertexts encrypting a list of vectors of shorter length. In particular, we will "cut" the ciphertext into N pieces, each of length N and indexed by $i \in [N]$.

  $$\mathsf{iCT} = [r], \quad \left\{ \mathsf{iCT}[i,\star] = \{ \left[r s_i^1 s_j^2 + x_i x_j \right] \}_{j \in [N]} \right\}_{i \in [N]} .$$

  Since the i^{th} ciphertext-piece can be viewed as an **IPE** ciphertext of vector $x_i\mathbf{x}$, generated with master secret key $s_i^1\mathbf{s}^2$ and shared random scalar r. Our idea is gradually switching the values of $x_i\mathbf{x}$ from $u_i\mathbf{u}$ to $v_i\mathbf{v}$ piece by piece in N steps. In each step, we first apply the function hiding of **cIPE** to move to a hybrid distribution where the challenge-piece $\mathsf{iCT}[i,\star]$ is directly hardwired in the **qFE** ciphertext; since $|\mathsf{iCT}[i,\star]| = N$, there is enough space for it. Then, we rely on the indistinguishability-security of **IPE** to argue that switching the plaintext-piece underlying $\mathsf{iCT}[i,\star]$ from $u_i\mathbf{u}$ to $v_i\mathbf{v}$ is indistinguishable.
- *Challenge 2 — Relying on the security of the ABDP scheme under correlated randomness.* Arguing the indistinguishability of switching the vectors underlying each ciphertext-piece $\mathsf{iCT}[i,\star]$ from $u_i\mathbf{u}$ to $v_i\mathbf{v}$ turns out to be tricky. First, An acute reader might have already noticed the problem that changing pieces in the tensor product would affect the function output, which is noticeable. For example, after switching the first plaintext piece to $v_i\mathbf{v}$, the function output changes to $\langle \mathbf{c}_j, \mathbf{u} \otimes \mathbf{u} \rangle \neq \langle \mathbf{c}_j, v_1\mathbf{v} || \mathbf{u}_{\geq 1} \otimes \mathbf{u} \rangle$. To resolve this problem, we modify the scheme to build in an *offset* value Δ_j in every secret key SK_j to ensure that the function output remains the same throughout all steps.

[6] Non-black-box security reduction may get around this difficulty, but is unclear how one can design a non-black-box reduction here.

Second, the challenge ciphertext-piece is generated with master secret key $s_i^1\mathbf{s}^2$, which is not truly random, since the vector $\mathbf{s}^2$ is used for generating the master secret keys $s_k^1\mathbf{s}^2$ of other ciphertext-pieces for $k \neq i$. We overcome this by relying on the SXDH assumption to argue that encodings of $s_i^1\mathbf{s}^2$, given encodings of s_i^1 and $\mathbf{s}^2$, are indistinguishable to encodings of random elements, and hence as good as a truly random master secret key. Similar idea was used in [44].

Next, we discuss in more detail how to overcome these two challenges.

Overcoming Challenge 1 — Embed ABDP IPE ciphertext in piecemeal. Our goal is switching *piece by piece* the tensor product underlying the derived **IPE** ciphertext from $\mathbf{u} \otimes \mathbf{u}$ to $\mathbf{v} \otimes \mathbf{v}$, which corresponds to changing the encrypted input from $\mathbf{u}$ to $\mathbf{v}$. To do so, we build a sequence of $2N$ hybrids $\{H_\rho^b\}_{\rho\in[N],b\in\{0,1\}}$ satisfying the following desiderata:

1. In H_ρ^b, the ρ^{th} ciphertext-piece $\mathsf{iCT}[\rho,\star]$ is embedded in the **qFE** ciphertext CT,
2. The derived **IPE** ciphertext iCT encrypts the following "hybrid" vectors.

$$\begin{array}{ll} \text{In } H_\rho^0, & v_1\mathbf{v}||\cdots||v_{\rho-1}\mathbf{v}||\underline{\,u_\rho\mathbf{u}\,}||u_{\rho+1}\mathbf{u}||\cdots||u_N\mathbf{u} \\ \text{In } H_\rho^1, & v_1\mathbf{v}||\cdots||v_{\rho-1}\mathbf{v}||\underline{\,v_\rho\mathbf{v}\,}||u_{\rho+1}\mathbf{u}||\cdots||u_N\mathbf{u} \end{array}$$

To build such hybrids, we need to modify our **qFE** scheme to build in more "redundant space" in its ciphertext.

Version 2 of Our Secret Key Quadratic FE scheme **qFE**

– Encryption: A ciphertext $\mathsf{CT}(\mathbf{x})$ consists of

$$\mathsf{CT}(\mathbf{x}) = \Big(\ [-r],\ \{\mathsf{cCT}_i(\underline{\,\mathbf{X}_i^1\,})\}_{i\in[N]},\ \{\mathsf{cSK}_j(\underline{\,\mathbf{X}_j^2\,})\}_{j\in[N]}\ \Big)\ ,$$
$$\text{where } \underline{\mathbf{X}_i^d = (\boldsymbol{\chi}_i^d||\mathbf{0},0)} \qquad (3)$$

where $\{\mathsf{cCT}_i\}$ and $\{\mathsf{cSK}_j\}$ encode vectors $\boldsymbol{\chi}_i^d$ like before, but now padded with 3 zeros.

We refer to the first 4 elements in $\mathbf{X}$'s as the first slot, which holds two vectors of length 2, and the last element as the second slot. In the honest executions, these vectors $\{\mathbf{X}_i^d\}$ are set to either $(\boldsymbol{\mu}^d||\mathbf{0},0)$ if $\mathbf{u}$ is encrypted, or $(\boldsymbol{\nu}^d||\mathbf{0},0)$ if $\mathbf{v}$ is encrypted, with $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ defined as $\boldsymbol{\chi}$ in Equation 1 but replacing x_i with u_i or v_i respectively.

Set the vector $\mathbf{X}$*'s in hybrid* H_ρ^b. Hybrid H_ρ^b uses the following set of vectors $\mathbf{X}$'s, which leverages the "space" of the additional zeros to satisfy the above desiderata.

$$\mathbf{X}_i^1 = \left(\begin{cases} \mathbf{0} \mid\mid \boldsymbol{\nu}_i^1 & \text{if } i < \rho \\ \boldsymbol{\mu}_i^1 \mid\mid \mathbf{0} & \text{if } i > \rho \\ \mathbf{0} \mid\mid \mathbf{0} & \text{if } i = \rho \end{cases} , \begin{cases} 0 & \text{if } i < \rho \\ 0 & \text{if } i > \rho \\ 1 & \text{if } i = \rho \end{cases} \right)$$

$$\mathbf{X}_j^2 = \left(\boldsymbol{\mu}_j^2 || \boldsymbol{\nu}_j^2 \ , \ \begin{cases} \langle \boldsymbol{\mu}_\rho^1, \boldsymbol{\mu}_j^2 \rangle & \text{in } H_\rho^0 \\ \langle \boldsymbol{\nu}_\rho^1, \boldsymbol{\nu}_j^2 \rangle & \text{in } H_\rho^1 \end{cases} \right)$$

Let us first see how the challenge ciphertext-piece $\mathsf{iCT}[\rho, \star]$ is hardwired. Observe that the last slots of $\mathbf{X}_j^2$'s contain exactly the values encoded in $\mathsf{iCT}[\rho, \star]$: In H_ρ^0, they are set to $\{\langle \boldsymbol{\mu}_\rho^1, \boldsymbol{\mu}_j^2 \rangle = rs_\rho^1 s_j^2 + u_\rho u_j\}_{j \in [N]}$ (see Equation 2), corresponding to encrypting $u_\rho \mathbf{u}$, while in H_ρ^1, they are set to $\{\langle \boldsymbol{\nu}_\rho^1, \boldsymbol{\nu}_j^2 \rangle = rs_\rho^1 s_j^2 + v_\rho v_j\}_j$, encrypting $v_\rho \mathbf{v}$. By the fact that **cIPE** encodes its function vectors, $\mathbf{X}_j^2$'s here, in a bilinear source group, $[\mathbf{X}_j^2]$ is effectively embedded in cSK_j's and hence so is $\mathsf{iCT}[\rho, \star]$. Next, we check that the **IPE** ciphertext derived by decrypting every pair $(\mathsf{cCT}_i, \mathsf{cSK}_j)$ indeed encrypts the right hybrid vector.

$$\begin{aligned} \mathsf{cIPE.Dec}(\mathsf{cSK}_j, \mathsf{cCT}_i) &= \left[\langle \mathbf{X}_i^1, \mathbf{X}_j^2 \rangle \right] \\ &= \left[\begin{cases} \langle \mathbf{0} \mid\mid \boldsymbol{\nu}_i^1 \mid\mid 0 \ , \ \boldsymbol{\mu}_j^2 \mid\mid \boldsymbol{\nu}_j^2 \mid\mid \star \rangle = \langle \boldsymbol{\nu}_i^1 \ , \ \boldsymbol{\nu}_j^2 \rangle & \text{if } i < \rho \\ \langle \boldsymbol{\mu}_i^1 \mid\mid \mathbf{0} \mid\mid 0 \ , \ \boldsymbol{\mu}_j^2 \mid\mid \boldsymbol{\nu}_j^2 \mid\mid \star \rangle = \langle \boldsymbol{\mu}_i^1 \ , \ \boldsymbol{\mu}_j^2 \rangle & \text{if } i > \rho \\ \langle \ \mathbf{0} \ \mid\mid \mathbf{0} \mid\mid 1 \ , \ \boldsymbol{\mu}_j^2 \mid\mid \boldsymbol{\nu}_j^2 \mid\mid \star \rangle = \qquad \star & \text{if } i = \rho \end{cases} \right] \end{aligned}$$

In the case $i = \rho$, $\mathsf{iCT}[\rho, \star]$ encodes exactly the values hardwired in the last slot, which as argued above encrypts $u_\rho \mathbf{u}$ in H_ρ^0 and $v_\rho \mathbf{v}$ in H_ρ^1 as desired. In the case $i < \rho$, the derived ciphertext-piece $\mathsf{iCT}[i, \star]$ encodes values $\{\langle \boldsymbol{\nu}_i^1, \boldsymbol{\nu}_j^2 \rangle\}_j$, corresponding to encrypting $v_i \mathbf{v}$; and similarly, when $i > \rho$, the ciphertext-piece $\mathsf{iCT}[i, \star]$ encrypts $u_i \mathbf{u}$ as desired. Therefore, all desiderata above are satisfied.

Now, to show the security of **qFE**, it suffices to argue that every pair of neighboring hybrids is indistinguishable. Note that the only difference between different hybrids lies in the values of the $\mathbf{X}$ vectors encoded in the ciphertexts and secret keys of **cIPE**. Observe first that in hybrids H_ρ^1 and $H_{\rho+1}^0$, every pair of vectors $(\mathbf{X}_i^1, \mathbf{X}_j^2)$ produce the *same* inner products, and hence the indistinguishability of H_ρ^1 and $H_{\rho+1}^0$ follows immediately from the function hiding property of **cIPE**. This is, however, not the case in hybrids H_ρ^0 and H_ρ^1, where for the special index ρ, the challenge ciphertext-piece change from encrypting $u_\rho \mathbf{u}$ to $v_\rho \mathbf{v}$. Next, we show how to reduce the indistinguishability of H_ρ^0 and H_ρ^1 to the security of the ABDP IPE scheme, which turns out to be quite tricky.

Overcoming Challenge 2: Indistinguishability of H_ρ^0 and H_ρ^1 from IPE security The goal is relying on the security of **IPE** to argue that the embedded challenge ciphertext-pieces in H_ρ^0 and H_ρ^1 are indistinguishable, and hence so are the hybrids. But, we immediately encounter a problem: The function outputs obtained when decrypting the derived ciphertext iCT using secret keys SK_j's are different in H^0_ρ and H^1_ρ, namely

$$\begin{aligned}&\left\langle v_1\mathbf{v}||\cdots||v_{\rho-1}\mathbf{v}||\,\underline{u_\rho\mathbf{u}}\,||u_{\rho+1}\mathbf{u}||\cdots||u_N\mathbf{u},\ ,\ \mathbf{c}_j\right\rangle\\ &\qquad\qquad \neq \left\langle v_1\mathbf{v}||\cdots||v_{\rho-1}\mathbf{v}||\,\underline{v_\rho\mathbf{v}}\,||u_{\rho+1}\mathbf{u}||\cdots||u_N\mathbf{u}\ ,\ \mathbf{c}_j\right\rangle\ .\end{aligned}$$

This means the hybrids are clearly distinguishable. To fix this, we modify our **qFE** scheme to build in an offset value Δ in its secret keys, which will be added to the decryption output. In the honest execution, the offsets are set to zero, whereas in hybrid H^b_ρ, they are set to $\Delta^b_j(\rho)$ in each secret key SK_j, so that, the above inner products when added with $\Delta^0_j(\rho)$ in the left hand side and $\Delta^1_j(\rho)$ in the right hand side become equal. Clearly, whether the offset values Δ are used (set to non-zero) at all and their values must be hidden, we do so by encoding it using **cIPE**, as described below.

Version 3 of Our Secret Key Quadratic FE schemes **qFE**

- Setup: A master secret key $\mathsf{msk} = (\mathbf{s}^1, \mathbf{s}^2, \mathsf{cMSK}')$ contains additionally a master secret key cMSK' of **cIPE**.
- Key Generation: In the secret key $\mathsf{SK}(\mathbf{c})$, the inner product $\left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c}\right\rangle$ is now encoded, together with an offset value Δ, using cMSK' of **cIPE**:

$$\mathsf{SK}(\mathbf{c}) = \left(\ \underline{\mathsf{cSK}'\left(\ \left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c}\right\rangle ||\Delta = 0\ \right)},\ \mathbf{c}\ \right)\ .$$

- Encryption: In the ciphertext $\mathsf{CT}(\mathbf{x})$, the random scalar r is now encrypted, with an additional 0, using cMSK' of **cIPE**:

$$\mathsf{CT}(\mathbf{x}) = \left(\ \underline{\mathsf{cCT}'(-r||0)},\ \left\{\mathsf{cCT}_i(\mathbf{X}^2_j)\right\}_{i\in[N]},\ \left\{\mathsf{cSK}_j(\mathbf{X}^2_j)\right\}_{j\in[N]}\ \right)\ .$$

- Decryption: Decryption proceeds as before, except that now encoding Λ_1 is obtained by decrypting cCT' using cSK', which yields $\left[-r\left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c}\right\rangle + \Delta\right]$ as desired.

With the new offset value in secret key, we can now fix our hybrids so that the function outputs always stay the same.

Set the offsets in hybrid H^b_ρ. In hybrid H^b_ρ, not only that the vectors $\mathbf{X}$'s are set differently as above, the **cIPE** ciphertext cCT' in ciphertext CT encrypts $(0||1)$ instead of $(-r||0)$ and the corresponding **cIPE** secret key cSK'_j in SK_j encodes vector $(\left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c}\right\rangle\ ||\ r\left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c}\right\rangle + \Delta^b_j(\rho))$, instead of $(\left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c}\right\rangle ||0)$. At decryption time, the offset $\Delta^b_j(\rho)$ is added to the inner product between $\mathbf{c}_j$ and hybrid vector underlying iCT. Setting $\Delta^b_j(\rho)$ appropriately ensures that

$$\begin{aligned}&\left\langle v_1\mathbf{v}||\cdots||v_{\rho-1}\mathbf{v}||\,\underline{u_\rho\mathbf{u}}\,||u_{\rho+1}\mathbf{u}||\cdots||u_N\mathbf{u},\ ,\ \mathbf{c}_j\right\rangle + \Delta^0_j(\rho)\\ &\qquad = \left\langle v_1\mathbf{v}||\cdots||v_{\rho-1}\mathbf{v}||\,\underline{v_\rho\mathbf{v}}\,||u_{\rho+1}\mathbf{u}||\cdots||u_N\mathbf{u}\ ,\ \mathbf{c}_j\right\rangle + \Delta^1_j(\rho) = f_{\mathbf{c}}(\mathbf{u})\ .\end{aligned}$$

Now H^0_ρ and H^1_ρ have the same function outputs. But, to formally reduce their indistinguishability to the security of **IPE**, we need a way to incorporate the offsets Δ's into the challenge **IPE** ciphertexts. We do so by viewing

Δ_j's as extension of the plaintext. More specifically, we implicitly switch from encrypting $\mathbf{U} = u_\rho \mathbf{u}||\Delta_1^0(\rho)||\cdots||\Delta_L^0(\rho)$ to $\mathbf{V} = v_\rho \mathbf{v}||\Delta_1^1(\rho)||\cdots||\Delta_L^1(\rho)$ using master secret key $\mathbf{S} = s_\rho^1 \mathbf{s}^2||t_1||\cdots||t_L$, at the presence of secret keys for vectors $\mathbf{Y}_j = \{\mathbf{c}_j[\rho, \star]||e_j\}_j$, where L is the total number of keys, t_j's are implicitly sampled secret key elements, and e_j is the unit vector of length L with a single one at index j. Observe that from such ciphertexts and secret keys, one can extract the challenge ciphertext-piece $\mathsf{iCT}[\rho, \star]$ encrypting $u_\rho \mathbf{u}$ or $v_\rho \mathbf{v}$, and obtain an encoding of $-r \left\langle \mathbf{s}^1 \otimes \mathbf{s}^2, \mathbf{c} \right\rangle + \Delta_j^b(\rho)$ embedded in each secret key cSK_j' — these are the only parts that hybrids H_ρ^0 and H_ρ^1 differ at. Given that $\langle \mathbf{U}, \mathbf{Y}_j \rangle = \langle \mathbf{V}, \mathbf{Y}_j \rangle$ for every j, we are almost done: Apply the security of **IPE** to argue that H_ρ^0 and H_ρ^1 are indistinguishable, except that we must overcome one last hurdle — the master secret key for encrypting $u_i \mathbf{u}$ or $v_i \mathbf{v}$ is not truely random.

Pseudorandomness from SXDH The master secret key of the challenge ciphertext-piece is $s_\rho^1 \mathbf{s}^2$. It is not truely random since $\mathbf{s}^2$ is also used for generating the master secret keys of other ciphertext-pieces. But, observe that both the challenge ciphertext-piece and $\mathbf{s}^2$ are embedded in secret keys $\{\mathsf{cSK}_j\}$, and hence encoded in the same bilinear map source group. Furthermore, thanks to the fact that in H_ρ^b, the ρ^{th} ciphertext cCT_ρ encrypts the vector $(\mathbf{0}||\mathbf{0}, 1)$, the key element s_ρ^1 does not appear in the other source group. Therefore, we can apply the SXDH assumption to argue that encodings of $s_\rho^1 \mathbf{s}^2$ is indistinguishable to that of a truly random vector $\mathbf{w}$ — in other words, the master secret key $s_\rho^1 \mathbf{s}^2$ is pseudorandom, *inside encodings*. Therefore, the security of **IPE** applies, and we conclude that hybrid H_ρ^0 and H_ρ^1 are indistinguishable.

2.3 Degree-D Secret-Key FE

Generalizing from quadratic FE to degree-D secret key FE, the natural idea is again starting from the trivial **IPE**-based construction that encrypts all degree-D monomials, denoted as $\otimes \mathbf{x}^{\leq D} = \otimes_{d \in [D]} \mathbf{x}$, and compressing the N^D-size ciphertext into linear size. Naturally, instead of compressing a ciphertext generated using a truly random master secret key, we will use a structured master secret key $\otimes \mathbf{s}^{\leq D} = \otimes_{d \in [D]} \mathbf{s}^i$. Thus the **IPE** ciphertext to be compressed looks like:

$$\mathsf{iCT}[0] = [-r], \qquad \mathsf{iCT}[I_1, \cdots, I_d] = \left[r s_{I_1}^1 \cdots s_{I_D}^D + x_{I_1} \cdots x_{I_D} \right]$$

The challenge is how to generate the N^D encodings $\mathsf{iCT}[I]$ from just linear-sized information?

Key Tool: High-Degree IPE We generalize IPE to the notion of high-degree IPE, or HIPE for short. More precisely, a degree-D HIPE is a *multi-input* functional encryption scheme for degree-D inner product defined as follows,

$$\left\langle \mathbf{x}^1, \cdots, \mathbf{x}^D \right\rangle = \Sigma_{i \in [N]} x_i^1 x_i^2 \cdots x_i^D$$

Introduced by [35], a multi-input functional encryption allows one to encrypt inputs at different coordinates, and generate secret keys associated with multi-input functions, so that, decryption computes the output of the function evaluated on inputs encrypted at different coordinates. In the context of HIPE, a degree-D HIPE encryption scheme **hIPE** allows one to generate a ciphertext $\mathsf{hCT}^d(\mathbf{x}^d)$ encrypting an input vector $\mathbf{x}^d$ at a coordinate $d \in [D-1]$, and a secret key $\mathsf{hSK}(\mathbf{x}^D)$ at coordinate D, so that, decryption reveals whether the degree-D inner product $\left\langle \mathbf{x}^1 \cdots \mathbf{x}^D \right\rangle$ is zero or not. Under this generalization, standard IPE is a special case of HIPE for degree $D = 2$.

In terms of security, the notion of function hiding also generalizes naturally, HIPE is function hiding, if ciphertexts and keys $\{\mathsf{hCT}^1_i, \cdots, \mathsf{hCT}^{D-1}_i, \mathsf{hSK}_i\}_{i\in[L]}$ encoding two sets of vectors $\{\mathbf{u}^1_i, \cdots, \mathbf{u}^{D-1}_i, \mathbf{u}^D_i\}_{i\in[L]}$ and $\{\mathbf{v}^1_i, \cdots, \mathbf{v}^{D-1}_i, \mathbf{v}^D_i\}_{i\in[L]}$ are indistinguishable, whenever all degree-D inner products that can be computed from them are identical, that is,

$$\forall I \in [L]^D, \left\langle \mathbf{u}^1_{I_1}, \cdots \mathbf{u}^D_{I_D} \right\rangle = \left\langle \mathbf{v}^1_{I_1}, \cdots \mathbf{v}^D_{I_D} \right\rangle$$

In this work, we give a construction of function hiding degree-D HIPE scheme from the SXDH assumption on degree-D multilinear maps. Our construction starts from a canonical function hiding IPE scheme (for $D = 2$), and inductively build degree-$(D+1)$ HIPE scheme, by composing a degree-D HIPE scheme and a special-purpose function hiding IPE scheme. Our HIPE schemes have *canonical form* (similar to the canonical form for standard IPE): Ciphertexts (or secret keys) at coordinate d (or D) consist of encodings in the d^{th} (or D^{th} respectively) MMap source group, and decryption uses degree-D pairing to produce an encoding of the degree-D inner product. That is,

$$\mathsf{hIPE.Dec}(\mathsf{hSK}(\mathbf{x}^D), \mathsf{hCT}^1(\mathbf{x}^1), \cdots, \mathsf{hCT}^D(\mathbf{x}^{D-1})) = \left[\left\langle \mathbf{x}^1, \cdots, \mathbf{x}^D \right\rangle\right]$$

From Degree-D HIPE to Degree-D FE HIPE works perfectly for our goal of compressing the ciphertext iCT. Generalizing **qFE**, our degree-D FE scheme **dFE** generates ciphertexts as follows:

$$\mathsf{CT}(\mathbf{x}) = \left(\mathsf{cCT}'(-r||0),\ \left\{\mathsf{cCT}^1_i(\mathbf{X}^1_i),\ \cdots,\ \mathsf{cCT}^{D-1}_i(\mathbf{X}^{D-1}_i),\ \mathsf{cSK}_i(\mathbf{X}^D_i)\right\}_{i\in[N]} \right)$$

$$\text{where } \mathbf{X}^d_i = \boldsymbol{\chi}^d_i||\mathbf{0} \text{ and } \boldsymbol{\chi}^d_i = \begin{cases} x_i||s^d_i & \text{if } d < D \\ x_i||rs^D_i & \text{if } d = D \end{cases}.$$

From such a ciphertext, a decryptor can "expand" out a size-N^D **IPE** ciphertext iCT by decrypting every combination of HIPE ciphertexts and secret keys. Namely, for every $I \in [N]^D$,

$$\begin{aligned} \mathsf{hIPE.Dec}(\mathsf{cCT}^1_{I_1},\ \cdots,\ \mathsf{cCT}^{D-1}_{I_{D-1}},\ \mathsf{cSK}_{I_D}) &= \left[\left\langle \mathbf{X}^1_{I_1}, \cdots, \mathbf{X}^D_{I_D} \right\rangle\right] \\ &= \left[r \prod_{d\in[D]} s^d_{I_d} + \prod_{d\in[D]} x_{I_d} \right] = \mathsf{iCT}[I] \end{aligned}$$

where $\mathsf{iCT}[I]$ encrypts the I^{th} degree-D monomial $\prod_{d\in[D]} x_{I_d}$, using the I^{th} key element $\prod_{d\in[D]} s^d_{I_d}$.

To show security of **dFE**, we, again, switch the degree-D monomials encrypted in the **IPE** ciphertext iCT in piecemeal. In each step, we can still only embed a size-N ciphertext-piece; naturally we embed $\mathsf{iCT}[\rho, \star]$ for a prefix $\rho \in [N]^{D-1}$ of length $D-1$. Thus, the N^D encrypted monomials are changed piece by piece in N^{D-1} steps, where in the ρ^{th} step, all monomials with index I smaller than ρ (*i.e.*, $I_{\leq D-1} < \rho$) have already been switched to $\prod_{d\in[D]} v_{I_d}$, monomials with index I larger than ρ (*i.e.*, $I_{\leq D-1} > \rho$) remain to be $\prod_{d\in[D]} u_{I_d}$, and monomials with index I that agrees with ρ (*i.e.*, $I_{\leq D-1} = \rho$) are being switched from $\prod_{d\in[D]} u_{I_d}$ in H^0_ρ to $\prod_{d\in[D]} v_{I_d}$ in H^1_ρ.

Creating a sequence of hybrids that carry out these steps is more complex than the case for degree 2. First, we need more space in the ciphertext to make sure that the right monomials are encrypted for every index I; thus, the vectors $\mathbf{X}$'s are padded to length $2D-1$. Second, it becomes significantly harder to argue that the key elements $(\prod_{d\in[D-1]} s^d_{\rho_d})\mathbf{s}^{\leq D}$ are pseudorandom, as the shares s^d_i's are encoded in different MMap source groups, and unlike the degree 2 case, we cannot eliminate the appearance of all shares $\{s^d_{\rho_d}\}$ since they are also used for generating the master secret keys of other ciphertext-pieces (whereas in the degree 2 case, s^1_ρ is only used for generating $s^1_\rho \mathbf{s}^2$). To resolve this, we apply the SXDH assumption iteratively to gradually replace every partial product $\prod_{d\in[d^\star]} s^d_{\rho_d}$ with an independent and random element w^d_ρ, so that, the master secret keys for other ciphertext-pieces are generated using independent w elements.

2.4 Construction of HIPE

We construct function hiding HIPE schemes by induction in the degree D.

- **For the base case of** $D = 2$, function hiding degree-2 HIPE is identical to function hiding IPE, which we give a new construction discussed shortly in the next subsection.
- **For the induction step**, we show that for any $D \geq 2$, if there exist a function hiding degree-D HIPE scheme, denoted as **dIPE**, from SXDH on degree-D MMap, then there exist a function-hiding degree-$(D+1)$ HIPE scheme, denoted as **hIPE**, from SXDH on degree-$(D+1)$ MMap. Our induction keeps the invariant that both **dIPE** and **hIPE** have canonical form.

In the induction step, we construct the degree-$D+1$ scheme **hIPE**, by combining the degree-D scheme **dIPE**, with a special purpose IPE scheme **sIPE**. Denote by $(\mathsf{hCT}^1, \cdots, \mathsf{hCT}^D)$ and hSK the ciphertexts and secret key of **hIPE**, $(\mathsf{dCT}^1, \cdots, \mathsf{dCT}^{D-1})$ and dSK that of **dIPE**, and sCT and sSK that of **sIPE**.

To achieve functionality, we need to specify how to generate ciphertexts and secret key for input vectors $\mathbf{x}^1, \cdots, \mathbf{x}^D$ and $\mathbf{x}^{D+1}$, so that,

$$\mathsf{hIPE.Dec}(\mathsf{hSK}, \mathsf{hCT}^1, \cdots, \mathsf{hCT}^D) = \left[\left\langle \mathbf{x}^1, \cdots, \mathbf{x}^D, \mathbf{x}^{D+1} \right\rangle\right] .$$

Observe that a degree-$(D+1)$ inner product of $\mathbf{x}^1, \cdots \mathbf{x}^{D+1}$, can be computed as the inner product between $\mathbf{x}^{D+1}$ and the coordinate-wise product of the first

D vectors $\prod_{d\in[D]} \mathbf{x}^d$, denoted as $\mathbf{x}^{\leq D}$, that is,

$$y = \left\langle \mathbf{x}^1, \cdots \mathbf{x}^{D+1} \right\rangle = \left\langle \prod_{d\in[D]} \mathbf{x}^d, \mathbf{x}^{D+1} \right\rangle = \left\langle \mathbf{x}^{\leq D}, \mathbf{x}^{D+1} \right\rangle$$

Therefore, if the decryptor obtains a pair of **sIPE** ciphertext and secret key $(\mathsf{sCT}, \mathsf{sSK})$ for $(\mathbf{x}^{\leq D}, \mathbf{x}^{D+1})$, he/she can decrypt to obtain $[y]$. To do so, our new scheme **hIPE** simply publishes sSK as its secret key,

Secret key of hIPE: $\quad \mathsf{hSK} = \mathsf{sSK} \leftarrow \mathsf{sIPE.KeyGen}(\mathsf{sMSK}, \mathbf{x}^{D+1})$.

However, it cannot directly publish a ciphertext of $\mathbf{x}^{\leq D}$, as $\mathbf{x}^{\leq D}$ is the product of D input vectors, but each encryption algorithm $\mathsf{hIPE.Enc}^d$ receives only a single vector $\mathbf{x}^d$ as input and cannot compute $\mathbf{x}^{\leq D}$. The idea is to include in the D ciphertexts $\mathsf{hCT}^1, \cdots, \mathsf{hCT}^D$ of **hIPE**, ciphertexts and secret keys of the degree-D scheme, so that the decryptor can combine them to generate a ciphertext sCT of $\mathbf{x}^{\leq D}$.

Towards this end, we rely on the first property of **sIPE** that its ciphertext sCT consists of many encodings $\{\mathsf{sCT}_l\}_{l\in[L]}$. Suppose that the element encoded sct_l in every encoding sCT_l can be expressed as the inner product of D vectors

Condition C: $\quad \mathrm{sct}_l = \left\langle \boldsymbol{\chi}_l^1, \cdots \boldsymbol{\chi}_l^D \right\rangle$, and each $\boldsymbol{\chi}_l^d$ depends only on $\mathbf{x}^d$,

Then, it suffices to encode these vectors in a tuple $(\mathsf{dCT}_l^1, \cdots \mathsf{dCT}_l^{D-1}, \mathsf{dSK}_l)$ of ciphertexts and secret key of **dIPE** using an independently sampled master secret key dMSK_l, from which the decryptor can obtain exactly sCT_l. Thus, the D ciphertexts $\mathsf{hCT}^1, \cdots, \mathsf{hCT}^D$ of our new scheme **hIPE** consists of exactly one such tuple $(\mathsf{dCT}_l^1, \cdots \mathsf{dCT}_l^{D-1}, \mathsf{dSK}_l)$ for every l, namely,

Ciphertext of hIPE:

$$\mathsf{hCT}^d = \begin{cases} \left\{\mathsf{dCT}_l^d \leftarrow \mathsf{dIPE.Enc}(\mathsf{dMSK}_l, \boldsymbol{\chi}_l^d)\right\}_{l\in[L]} & \text{if } d \leq D \\ \left\{\mathsf{dSK}_l \leftarrow \mathsf{dIPE.KeyGen}(\mathsf{dMSK}_l, \boldsymbol{\chi}_l^D)\right\}_{l\in[L]} & \text{if } d = D \end{cases} .$$

Given $(\mathsf{hCT}^1, \cdots, \mathsf{hCT}^D)$ and hSK as specified above, the decryptor proceeds in two steps:

1. First, decrypt for every l, the tuple $(\mathsf{dCT}_l^1, \cdots \mathsf{dCT}_l^{D-1}, \mathsf{dSK}_l)$ using the decryption algorithm of **dIPE** to obtain sCT_l; put them together to get a ciphertext sCT of $\mathbf{x}^{\leq D}$.
2. Then, decrypt the obtained ciphertext sCT using the decryption algorithm of **sIPE** and secret key $\mathsf{hSK} = \mathsf{sSK}$ of $\mathbf{x}^{D+1}$ to obtain an encoding of the final inner product y, as illustrated below.

$$\underbrace{\underbrace{\left\{\mathsf{hCT}^1 = \{\mathsf{dCT}_l^1\}_l \ , \ \cdots \ , \ \mathsf{hCT}^{D-1} = \{\mathsf{dCT}_l^{D-1}\}_l \ , \ \mathsf{hCT}^D = \{\mathsf{dSK}_l\}_l\right\}_l}_{\text{Decrypt to } \mathsf{sCT}} \quad \mathsf{hSK} = \mathsf{sSK}}_{\text{Decrypt to } [y]}$$

Setting Condition C – A First Attempt We now argue that **Condition C** above indeed holds. This relies on a second property of the special-purpose IPE scheme **sIPE** that the elements $\{\mathrm{sct}_l\}$ encoded in its ciphertext sCT, depends *linearly* in the encrypted vector $\mathbf{x}^{\leq D}$ and randomness $\mathbf{r}$ of encryption. More specifically, when the master secret key sMSK is fixed, each element sct_l is the output of a linear function $h_l^{(\mathsf{sMSK})}$ on input $(\mathbf{x}^{\leq D}, \mathbf{r})$,

$$\begin{aligned}&\mathsf{sCT} = \mathsf{sIPE.Enc}(\mathsf{sMSK},\ \mathbf{x}^{\leq D}\ ;\ \mathbf{r}) = \{[\mathrm{sct}_l]\}_l\ ,\\ &\text{with } \mathrm{sct}_l = h_l^{(\mathsf{sMSK})}(\mathbf{x}^{\leq D},\ \mathbf{r}) = \left\langle \mathbf{c}_l^{(\mathsf{sMSK})}, (\mathbf{x}^{\leq D}||\mathbf{r})\ \right\rangle\ ,\end{aligned}$$

where $\mathbf{c}_l^{(\mathsf{sMSK})}$ is the coefficient vector of $h_l^{(\mathsf{sMSK})}$. Then, since $\mathbf{x}^{\leq D} = \mathbf{x}^1 \cdots \mathbf{x}^D$, we can represent sct_l as the inner product of D vectors $\boldsymbol{\chi}_l^1, \cdots, \boldsymbol{\chi}_l^D$, each depending on only one input vector $\mathbf{x}^D$, as follows:

$$\mathrm{sct}_l = \langle \boldsymbol{\chi}_l^1, \boldsymbol{\chi}_l^2, \cdots \boldsymbol{\chi}_l^D \rangle \qquad \boldsymbol{\chi}_l^d = \begin{cases} \mathbf{x}^1||\underline{\mathbf{r}} & \text{if } d = 1 \\ \mathbf{x}^d||\mathbf{1} & \text{if } 1 < d < D \\ (\mathbf{x}^D||\mathbf{1})\underline{(\mathbf{c}_l^{(\mathsf{sMSK})})} & \text{if } d = D \end{cases} .$$

Therefore, as discussed above, encrypting the vectors $\{\boldsymbol{\chi}_l^d\}$ in the ciphertexts of **hIPE** guarantees that the decryptor can obtain sCT from the ciphertexts, and decrypting the ciphertext sCT further produces an encoding of the correct output y.

A Security Issue The above way of setting the vectors $\{\boldsymbol{\chi}_l^d\}_{d,l}$ achieves functionality, but, does not guarantee security. A security issue stems from the fact that the randomness $\mathbf{r}$ used for generating the ciphertext sCT is hardcoded entirely in the input vectors $\{\boldsymbol{\chi}_l^1\}_l$ encrypted at the first coordinate. Consider a simple scenario where a single ciphertext of **hIPE** at the first coordinate, two ciphertexts at each other coordinate, and a single secret key, are published:

$$\begin{array}{cccccc} \mathsf{hCT}^1, & \mathsf{hCT}_0^2, & \cdots, & \mathsf{hCT}_0^D\ , & \mathsf{hSK} \\ & \mathsf{hCT}_1^2, & \cdots, & \mathsf{hCT}_1^D & \end{array}$$

Since the randomness $\mathbf{r}$ is embedded in hCT^1, different combinations of ciphertexts, say hCT^1 and $\mathsf{hCT}_{b_2}^2 \cdots\ \mathsf{hCT}_{b_D}^D$, produce **sIPE** ciphertexts encrypting different vectors, $\mathbf{x}^1\mathbf{x}_{b_2}^2 \cdots \mathbf{x}_{b_D}^D$, but using the same random coins $\mathbf{r}$. The security of **sIPE** does not hold when attackers can observe ciphertexts with shared randomness, and in particular, information of the encrypted vector $\mathbf{x}^1\mathbf{x}_{b_2}^2 \cdots \mathbf{x}_{b_D}^D$ may be revealed. On the other hand, the function hiding property requires that only the final degree-$(D+1)$ inner products $\mathbf{x}^1\mathbf{x}_{b_2}^2 \cdots \mathbf{x}_{b_D}^D\mathbf{x}^{D+1}$ are revealed, and nothing else.

Setting Condition C, Right To address this security issue, we need to ensure that ciphertexts sCT produced by different combinations of ciphertexts of **hIPE** correspond to (at the very least) distinct randomness. To do so, we embed fresh randomness $\mathbf{r}^d$ in ciphertexts at every coordinate by modifying the encrypted vectors $\boldsymbol{\chi}_l^d$ to the following:

$$\boldsymbol{\chi}_l^d = \begin{cases} \mathbf{x}^d || \underline{\mathbf{r}^d} & \text{if } d < D \\ (\mathbf{x}^D || \underline{\mathbf{r}^D})(\mathbf{c}_l^{(\mathsf{sMSK})}) & \text{if } d = D \end{cases}$$

Note that the inner products of these vectors correspond to a ciphertext sCT generated using random coins $\mathbf{r}^{\leq D} = \prod_{d \in [D]} \mathbf{r}^d$. That is,

$$\langle \boldsymbol{\chi}_1, \cdots, \boldsymbol{\chi}_D \rangle = \left\langle \mathbf{c}_l^{(\mathsf{sMSK})}, (\mathbf{x}^{\leq D} || \mathbf{r}^{\leq D}) \right\rangle = h_l^{(\mathsf{sMSK})}(\mathbf{x}^{\leq D},\ \mathbf{r}^{\leq D}) = \mathrm{sct}_l \ ,$$
$$\mathsf{sCT} = \{[\mathrm{sct}_l]\}_l = \mathsf{sIPE.Enc}(\mathsf{sMSK},\ \mathbf{x}^{\leq D}\ ;\ \mathbf{r}^{\leq D}) \ .$$

In the simple scenario above, combining $\mathsf{hCT}^1, \mathsf{hCT}^2_{b_2} \cdots, \mathsf{hCT}^D_{b_D}$ now produces sCT with randomness $\mathbf{r}^1 \mathbf{r}^2_{b_2} \cdots \mathbf{r}^D_{b_D}$, which is distinct for each combination.

Having distinct randomness is still not enough for applying the security of **sIPE**, which requires independently and uniformly sampled randomness. We will rely on the SXDH assumption to argue that they are indeed pseudorandom. The security analysis of the above scheme turns out to be quite complicated, and in fact for security to hold, the scheme needs to further pad the vectors $\boldsymbol{\chi}_l^d$ with zeros, serving as redundant space for hardwiring information in different hybrids in the security proof.

2.5 Simple Function Hiding IPE

As described above, our construction of degree-D FE crucially relies on a *canonical* function hiding IPE. However, known secret-key IPE schemes [12,24,44] do not have the canonical form, in particular, their decryption does not produce an encoding of the output inner product $[\langle \mathbf{x}, \mathbf{y} \rangle]$, but produce the inner product masked by a scalar $[\langle \mathbf{x}, \mathbf{y} \rangle\, \theta]$ together with $[\theta]$, where the scalar θ is determined by the randomness used in key generation and encryption. In this work, we give a construction of a *canonical* function hiding IPE. Our construction is extremely simple and may be of independent interests. We now summarize the idea of the construction in one paragraph.

Lin and Vaikuntanathan [44] give a simple transformation from IPE with weak function hiding to IPE with full function hiding. Our construction starts from the ABDP public key IPE scheme, whose secret key for a vector $\mathbf{y}$ reveals $\mathbf{y}$ and its inner product with the master secret key $\langle \mathbf{s}, \mathbf{y} \rangle$ in the clear. To achieve weak function hiding, we need to hide $\mathbf{y}$. Our idea is to simply encrypt the secret key as an input vector using the ABDP scheme itself, with an independently sampled master secret key $\mathbf{s}'$ of length $N+1$, which yields the new secret key $\mathsf{iSK}' = [r'\mathbf{s}' + (\langle \mathbf{s}, \mathbf{y} \rangle || \mathbf{y})]$. Recall that decryption of the ABDP scheme simply computes (homomorphically) the inner product between its secret key and ciphertext. Now that the original secret key is encrypted, we correspondingly encode the original ciphertext in a secret key using $\mathbf{s}'$, which gives the new ciphertext $\mathsf{iCT}' = [\langle \mathbf{s}' \ , \ (r\mathbf{s} + \mathbf{x}) \rangle \ || \ (r\mathbf{s} + \mathbf{x})]$. Computing the "inner product" of iCT' and iSK' using paring simultaneously decrypts both "layers" of ABDP encryption, and produce exactly an encoding of the output inner product.

We have described ideas underlying our FE and IO constructions; due to the lack of space, we refer the reader to the full version [42] for their formal description and proofs. With a better view of the constructions and security proofs, next, we revisit the topic of instantiating our schemes with known noisy multilinear map candidates in more detail.

2.6 On Instantiation with Noisy Multilinear Maps

As mentioned in the introduction when replacing algebraic multilinear maps with noisy ones [26,21,39,22,31], the constructions work as-is, but not the security proofs. Nevertheless, the security proof can be modified into an ideal model proof, or a proof based on a family of more complex assumptions.

The FE Security Proof Fails. The only component in our IO construction that relies on MMaps is the low-degree FE scheme. When using known noisy MMap candidates, its security proofs fail for two reasons:

1. *The SXDH assumption does not hold on known noisy MMap candidates.* Roughly speaking, a noisy multilinear map scheme can encode a ring element a and a label l with some noise. Let L be a set of labels that correspond to the set of source groups in algebraic MMaps. Translating the SXDH assumption to the noisy setting would require for every label $l \in L$, the distribution of randomly sampled encodings of a, b, ab with label l to be indistinguishable to that of a, b, r, for random ring elements a, b, r, *even when low-level encodings of 1 with each label $l \in L$ is published.* Unfortunately, given these encodings of 1, known noisy MMap candidates can be completely broken.
2. The security reduction uses the *homomorphic scalar multiplication* functionality of algebraic MMaps, which is not support by current candidates.

The reason that encodings of 1 is needed in the assumption and homomorphic scalar multiplication is needed for the reduction is as follows. The security of the FE scheme is based on the SXDH assumption, via a security reduction that turns FE attackers to SXDH distinguishers. To do so, given a challenge sampled according to (one of the two distributions specified in) the SXDH assumption, our reduction internally simulates the view of the attacker in the FE security game, and appropriately *embeds* the challenge into the view. Since the challenge is "laconic" — containing only a constant number of encodings. To concoct the attacker's view, the reduction needs to i) generate new encodings and ii) randomize some encodings in the challenge for embedding. It does so using encodings of 1 in the challenge and homomorphic scalar multiplication. It seems (to us) that any reduction to a *laconic* and/or *instance-independent* assumption (*i.e.*, one that is independent of the scheme and the attacker) necessarily needs the capabilities of generating and randomizing encodings. This is indeed the case for previous such reductions [33,44] and they also require homomorphic scalar multiplication. Designing a reduction that does not rely on homomorphic scalar multiplication, or rely only on homomorphic scalar multiplication with *small* scalars is an interesting open question.

Security Proofs to Non-Laconic Assumptions, and in Ideal MMap Model Above problems can be eliminated if we give up on having a security reduction to a laconic and instance-independent assumption. In particular, our security proof presents a sequence of hybrids that gradually "morph" from one honest execution of the FE scheme to another (where the attacker receives secret keys and ciphertexts of different functions and inputs as specified in the security definition of FE). Each pair of neighboring hybrids defines an indistinguishability assumption that simply states that the attacker's views in these two hybrids are indistinguishable, and the security of FE can be based on such a family of non-laconic and instance-dependent assumptions, without using encodings of 1 and homomorphic scalar multiplication. Such a security proof is non-trivial since the assumptions only require indistinguishability of distributions that are almost identical modulo the difference induced by switching a single DDH tuple to a random tuple. Moreover, since these assumptions hold in the ideal model, such a proof also gives a proof in degree-5 ideal multilinear map model.

Instantiating the Construction with Noisy Multilinear Maps. We can instantiate our FE scheme with noisy MMaps and correctness holds. The above-discussed issues w.r.t. the security proof do not appear when instantiating the construction. This is because the secret keys and ciphertexts of our FE scheme do not contain any low-level encodings of 0 or 1, in fact, they contain only encodings of large randomized elements, and its algorithms do not rely on homomorphic scalar multiplication. We note, however, decryption may generate *top-level* encodings of 0 or 1 for correctness. It is unclear (to us) whether these instantiations are secure against known cryptanalytic attacks. We do not know whether known attacks can be adapted to break their security, nor have formal arguments that validate their security against known attacks. Obtaining a concrete attack or give some formal proof, such as, a security proof in the weak MMap model [29], are interesting open problems.

Acknowledgements. The author thanks Benny Applebaum, Nir Bitansky, Stefano Tessaro, and Vinod Vaikuntanathan for many helpful and insightful discussions.

References

1. M. Abdalla, F. Bourse, A. D. Caro, and D. Pointcheval, "Simple functional encryption schemes for inner products," in *Public-Key Cryptography - PKC 2015 - 18th*

IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings, vol. 9020 of *Lecture Notes in Computer Science*, pp. 733–751, 2015.

2. P. Ananth and A. Jain, "Indistinguishability obfuscation from compact functional encryption," in *CRYPTO 2015, Part I*, vol. 9215 of *LNCS*, (Santa Barbara, CA, USA), pp. 308–326, Aug. 16–20, 2015.
3. P. Ananth, A. Jain, and A. Sahai, "Achieving compactness generically: Indistinguishability obfuscation from non-compact functional encryption," *IACR Cryptology ePrint Archive*, vol. 2015, p. 730, 2015.
4. P. Ananth and A. Sahai, "Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps," in *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, pp. 152–181, 2017.
5. P. V. Ananth, D. Gupta, Y. Ishai, and A. Sahai, "Optimizing obfuscation: Avoiding Barrington's theorem," in *ACM CCS 14*, (Scottsdale, AZ, USA), pp. 646–658, Nov. 3–7, 2014.
6. D. Apon, N. Döttling, S. Garg, and P. Mukherjee, "Cryptanalysis of indistinguishability obfuscations of circuits over GGH13," in *ICALP 2017*, LNCS, 2017.
7. B. Applebaum and Z. Brakerski, "Obfuscating circuits via composite-order graded encoding," in *TCC 2015, Part II*, vol. 9015 of *LNCS*, (Warsaw, Poland), pp. 528–556, Mar. 23–25, 2015.
8. B. Applebaum, Y. Ishai, and E. Kushilevitz, "Cryptography in nc^0," in *FOCS*, pp. 166–175, 2004.
9. B. Barak, Z. Brakerski, I. Komargodski, and P. K. Kothari, "Limits on low-degree pseudorandom generators (or: Sum-of-squares meets program obfuscation)." Cryptology ePrint Archive, Report 2017/312, 2017. http://eprint.iacr.org/2017/312.
10. B. Barak, S. Garg, Y. T. Kalai, O. Paneth, and A. Sahai, "Protecting obfuscation against algebraic attacks," in *EUROCRYPT 2014*, vol. 8441 of *LNCS*, (Copenhagen, Denmark), pp. 221–238, May 11–15, 2014.
11. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang, "On the (im)possibility of obfuscating programs," in *Advances in Cryptology CRYPTO 2001*, pp. 1–18, Springer, 2001.
12. A. Bishop, A. Jain, and L. Kowalczyk, "Function-hiding inner product encryption," in *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, vol. 9452 of *Lecture Notes in Computer Science*, pp. 470–491, 2015.
13. N. Bitansky, R. Nishimaki, A. Passelègue, and D. Wichs, "From cryptomania to obfustopia through secret-key functional encryption," in *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, vol. 9986 of *Lecture Notes in Computer Science*, pp. 391–418, 2016.
14. N. Bitansky and V. Vaikuntanathan, "Indistinguishability obfuscation from functional encryption," in *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pp. 171–190, 2015.
15. D. Boneh and A. Silverberg, "Applications of multilinear forms to cryptography," *Contemporary Mathematics*, vol. 324, pp. 71–90, 2002.

16. D. Boneh, D. J. Wu, and J. Zimmerman, “Immunizing multilinear maps against zeroizing attacks.” Cryptology ePrint Archive, Report 2014/930, 2014. http://eprint.iacr.org/2014/930.
17. Z. Brakerski and G. N. Rothblum, “Virtual black-box obfuscation for all circuits via generic graded encoding,” in *TCC 2014*, vol. 8349 of *LNCS*, (San Diego, CA, USA), pp. 1–25, Feb. 24–26, 2014.
18. Y. Chen, C. Gentry, and S. Halevi, “Cryptanalyses of candidate branching program obfuscators,” in *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, pp. 278–307, 2017.
19. J. H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé, “Cryptanalysis of the multilinear map over the integers,” in *EUROCRYPT 2015, Part I*, vol. 9056 of *LNCS*, (Sofia, Bulgaria), pp. 3–12, Apr. 26–30, 2015.
20. J.-S. Coron, C. Gentry, S. Halevi, T. Lepoint, H. K. Maji, E. Miles, M. Raykova, A. Sahai, and M. Tibouchi, “Zeroizing without low-level zeroes: New MMAP attacks and their limitations,” in *CRYPTO 2015, Part I*, vol. 9215 of *LNCS*, (Santa Barbara, CA, USA), pp. 247–266, Aug. 16–20, 2015.
21. J.-S. Coron, T. Lepoint, and M. Tibouchi, “Practical multilinear maps over the integers,” in *CRYPTO 2013, Part I*, vol. 8042 of *LNCS*, (Santa Barbara, CA, USA), pp. 476–493, Aug. 18–22, 2013.
22. J.-S. Coron, T. Lepoint, and M. Tibouchi, “New multilinear maps over the integers,” in *CRYPTO 2015, Part I*, vol. 9215 of *LNCS*, (Santa Barbara, CA, USA), pp. 267–286, Aug. 16–20, 2015.
23. M. Cryan and P. B. Miltersen, “On pseudorandom generators in nc0.” In Proc. 26th MFCS, 2001.
24. P. Datta, R. Dutta, and S. Mukhopadhyay, “Functional encryption for inner product with full function privacy,” in *Public-Key Cryptography - PKC 2016 - 19th IACR International Conference on Practice and Theory in Public-Key Cryptography, Taipei, Taiwan, March 6-9, 2016, Proceedings, Part I*, vol. 9614 of *Lecture Notes in Computer Science*, pp. 164–195, 2016.
25. N. Döttling, S. Garg, D. Gupta, P. Miao, and P. Mukherjee, “Obfuscation from low noise multilinear maps.” Cryptology ePrint Archive, Report 2016/599, 2016. http://eprint.iacr.org/2016/599.
26. S. Garg, C. Gentry, and S. Halevi, “Candidate multilinear maps from ideal lattices,” in *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, vol. 7881 of *Lecture Notes in Computer Science*, pp. 1–17, 2013.
27. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, “Candidate indistinguishability obfuscation and functional encryption for all circuits,” in *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pp. 40–49, 2013.
28. S. Garg, C. Gentry, S. Halevi, and M. Zhandry, “Functional encryption without obfuscation,” in *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, vol. 9563 of *Lecture Notes in Computer Science*, pp. 480–511, 2016.
29. S. Garg, E. Miles, P. Mukherjee, A. Sahai, A. Srinivasan, and M. Zhandry, “Secure obfuscation in a weak multilinear map model,” in *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, pp. 241–268, 2016.

30. S. Garg, P. Mukherjee, and A. Srinivasan, "Obfuscation without the vulnerabilities of multilinear maps," *IACR Cryptology ePrint Archive*, vol. 2016, p. 390, 2016.
31. C. Gentry, S. Gorbunov, and S. Halevi, "Graph-induced multilinear maps from lattices," in *TCC 2015, Part II*, vol. 9015 of *LNCS*, (Warsaw, Poland), pp. 498–527, Mar. 23–25, 2015.
32. C. Gentry, S. Halevi, H. K. Maji, and A. Sahai, "Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero." Cryptology ePrint Archive, Report 2014/929, 2014. http://eprint.iacr.org/2014/929.
33. C. Gentry, A. B. Lewko, A. Sahai, and B. Waters, "Indistinguishability obfuscation from the multilinear subgroup elimination assumption," in Guruswami [36], pp. 151–170.
34. O. Goldreich, "Candidate one-way functions based on expander graphs," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 7, no. 90, 2000.
35. S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F. Liu, A. Sahai, E. Shi, and H. Zhou, "Multi-input functional encryption," in *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, vol. 8441 of *Lecture Notes in Computer Science*, pp. 578–602, 2014.
36. V. Guruswami, ed., *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, IEEE Computer Society, 2015.
37. Y. Ishai and E. Kushilevitz, "Perfect constant-round secure computation via perfect randomizing polynomials," in *ICALP*, pp. 244–256, 2002.
38. J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *EUROCRYPT 2008*, pp. 146–162, 2008.
39. A. Langlois, D. Stehlé, and R. Steinfeld, "Gghlite: More efficient multilinear maps from ideal lattices," in *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, vol. 8441 of *Lecture Notes in Computer Science*, pp. 239–256, 2014.
40. A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *EUROCRYPT 2010*, vol. 6110 of *LNCS*, (French Riviera), pp. 62–91, May 30 – June 3, 2010.
41. H. Lin, "Indistinguishability obfuscation from constant-degree graded encoding schemes," 2016. To Appear in Eurocrypt'16.
42. H. Lin, "Indistinguishability obfuscation from ddh on 5-linear maps and locality-5 prgs." Cryptology ePrint Archive, Report 2016/1096, 2016. http://eprint.iacr.org/2016/1096.
43. H. Lin and S. Tessaro, "Indistinguishability obfuscation from trilinear maps and block-wise local prgs," in *To appear, CRYPTO 2017*, 2017.
44. H. Lin and V. Vaikuntanathan, "Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings," in *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, New Brunswick, NJ, USA, 9-11 October, 2016*, 2016.
45. A. Lombardi and V. Vaikuntanathan, "On the non-existence of blockwise 2-local prgs with applications to indistinguishability obfuscation." Cryptology ePrint Archive, Report 2017/301, 2017. http://eprint.iacr.org/2017/301.

46. E. Miles, A. Sahai, and M. Zhandry, "Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13," *IACR Cryptology ePrint Archive*, vol. 2016, p. 147, 2016.
47. E. Mossel, A. Shpilka, and L. Trevisan, "On e-biased generators in NC0," in *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pp. 136–145, 2003.
48. R. O'Donnell and D. Witmer, "Goldreich's PRG: evidence for near-optimal polynomial stretch," in *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pp. 1–12, 2014.
49. R. Pass, K. Seth, and S. Telang, "Indistinguishability obfuscation from semantically-secure multilinear encodings," in *CRYPTO 2014, Part I*, vol. 8616 of *LNCS*, (Santa Barbara, CA, USA), pp. 500–517, Aug. 17–21, 2014.
50. R. Rothblum, "On the circular security of bit-encryption," in *TCC*, pp. 579–598, 2013.
51. A. C. Yao, "Protocols for secure computations (extended abstract)," in *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pp. 160–164, 1982.
52. A. C.-C. Yao, "How to generate and exchange secrets (extended abstract)," in *FOCS*, pp. 162–167, 1986.
53. J. Zimmerman, "How to obfuscate programs directly," in *EUROCRYPT 2015, Part II*, vol. 9057 of *LNCS*, (Sofia, Bulgaria), pp. 439–467, Apr. 26–30, 2015.