Chainable Functional Commitments for Unbounded-Depth Circuits

David Balbás[1,2], Dario Catalano[3], Dario Fiore[1], and Russell W. F. Lai[4]

[1] IMDEA Software Institute, Madrid, Spain.
david.balbas@imdea.org
dario.fiore@imdea.org
[2] Universidad Politécnica de Madrid, Madrid, Spain.
[3] University of Catania, Catania, Italy.
catalano@dmi.unict.it
[4] Aalto University, Espoo, Finland.
russell.lai@aalto.fi

Abstract. A functional commitment (FC) scheme allows one to commit to a vector $\boldsymbol{x}$ and later produce a short opening proof of $(f, f(\boldsymbol{x}))$ for any admissible function f. Since their inception, FC schemes supporting ever more expressive classes of functions have been proposed.

In this work, we introduce a novel primitive that we call *chainable functional commitment* (CFC), which extends the functionality of FCs by allowing one to 1) open to functions of multiple inputs $f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$ that are committed independently, 2) while preserving the output also in committed form. We show that CFCs for quadratic polynomial maps generically imply FCs for circuits. Then, we efficiently realize CFCs for quadratic polynomials over pairing groups and lattices, resulting in the *first FC schemes for circuits of unbounded depth* based on either pairing-based or lattice-based falsifiable assumptions. Our FCs require fixing a-priori only the maximal width of the circuit to be evaluated, and have opening proofs whose size only depends on the depth of the circuit. Additionally, our FCs feature other nice properties such as being additively homomorphic and supporting sublinear-time verification after offline preprocessing.

Using a recent transformation that constructs homomorphic signatures (HS) from FCs, we obtain the first pairing- and lattice-based realisations of HS for bounded-width, but unbounded-depth, circuits. Prior to this work, the only HS for general circuits is lattice-based and requires bounding the circuit depth at setup time.

1 Introduction

Commitment schemes allow a sender to commit to a message x in such a way that the message remains secret until the moment she decides to open the commitment and reveal it (*hiding*), and they allow the receiver to get convinced that the opened message is the same x originally used at commitment time (*binding*).

Today, commitments are a ubiquitous building block in cryptographic protocols, including digital signatures, zero-knowledge proofs and multiparty computation, to name a few. As applications become more and more sophisticated, the basic commitment functionality may fall short. One particular limitation is that the opening mechanism is *all-or-nothing*: either the sender opens in full the commitment and the receiver learns the whole message, or the receiver gets nothing. A more flexible and useful functionality would be to open the commitment to a *function of the committed message*, that is to reveal $f(x)$ for some function f.

This advanced commitment notion has been formalized by Libert, Ramanna and Yung who called this primitive *Functional Commitments* (FC) [LRY16]. The property that makes functional commitments unique (and nontrivial to realize) is *succinctness*: assuming that the message is a large vector $\boldsymbol{x}$, then both the commitment and the openings should be short, e.g., polylogarithmic or constant in the size of $\boldsymbol{x}$. The main security requirement of functional commitments is *evaluation binding*: no polynomially bounded adversary should be able to, validly, open the commitment to two different values $y \neq y'$ for the same f. Additionally, FCs can also be hiding and zero-knowledge (a commitment and possibly several openings should not reveal additional information about $\boldsymbol{x}$).

Functional commitments are essentially a class of (commit-and-prove) succinct non-interactive arguments (SNARGs) with a weaker security property, that is evaluation binding instead of soundness. The notion of evaluation binding is not necessarily a weakness but can also be a feature: it is a falsifiable security notion that makes FCs potentially realizable from falsifiable assumptions in the standard model (i.e., without random oracles), without contradicting the celebrated result of Gentry and Wichs about impossibility of SNARGs from falsifiable assumptions [GW11]. For this reason, functional commitments can be an attractive alternative to SNARGs for implementing succinct arguments in cryptographic protocols where evaluation binding is sufficient (notably, without carrying the need of non-falsifiable assumptions). Examples of this case include homomorphic signatures and verifiable databases as shown in [CFT22], as well as the numerous applications that employ vector commitments [CFM08, LY10, CF13] or polynomial commitments [KZG10] (two primitives that are a special case of the FC notion). An additional motivation for studying evaluation binding FCs is that they can provide a different approach to construct SNARKs since any evaluation binding FC can be compiled into a SNARK by adding a simpler SNARK proof of "I know $\boldsymbol{x}$ that opens the commitment".

The state-of-the-art on realizations of FCs encompasses a limited set of functionalities that (besides the special cases of vector and polynomial commitments) include linear maps [LRY16, LM19], semi-sparse polynomials [LP20] and constant-degree polynomials [ACL+22, CFT22] (see Section 1.2 for a discussion on related and concurrent work).

1.1 Our Contribution

In this paper, we propose the first constructions of Functional Commitments that support the evaluation of arbitrary arithmetic circuits of unbounded depth[5] and are based on falsifiable assumptions. Our FC schemes are also *chainable*, meaning that it is possible to open to functions of multiple committed inputs while preserving the output to be in committed form. To capture such functionality, we introduce a novel primitive called *Chainable Functional Commitment* (CFC).

In our FC schemes only the maximal *width* of the circuits has to be fixed at setup time. The size of the commitments is fully succinct in the input size; the size of opening proofs grows with the multiplicative depth $d_{\mathcal{C}}$ of the evaluated circuit $\mathcal{C}$, but is otherwise independent of the circuit's size or the input length. Notably, our FC schemes provide an exponential improvement compared to previous FCs that could only support polynomials of degree $\delta = O(1)$ with an efficiency (prover time and parameter size) degrading exponentially in δ (as $O(n^{\delta})$)[6] [ACL+22, CFT22].

We design our FCs for circuits in two steps: (1) a generic construction of an FC for unbounded-depth circuits based on CFCs for quadratic functions, and (2) two realizations of CFCs, one based on bilinear pairings and one based on lattices. The pairing-based CFC relies on a new falsifiable assumption that we justify in the bilinear generic group model, while the lattice-based CFC relies on a slight extension of the k-R-ISIS assumption recently introduced in [ACL+22]. Using either one

[5] Looking ahead, our pairing-based instantiation supports arithmetic circuits over $\mathbb{Z}_q$, while our lattice-based instantiation supports arithmetic circuits over cyclotomic rings $\mathbb{Z}[\zeta]$ where wires carry values of bounded norm.

[6] Note, when used for a circuit of depth d these solutions may have efficiency doubly exponential in d since in general $\delta \approx 2^d$.

FC scheme	Functions	$\|\mathsf{pp}\|$	$\|\mathsf{com}\|$	$\|\pi\|$	AH
[LRY16] (pair.)	linear maps	λn	λ	$\lambda\ell$	✓
[LM19] (pair.)	linear maps	$\lambda\ell n$	λ	λ	✓
[LP20] (pair.)	semi-sparse poly	$\lambda\mu$	$\lambda\ell$	λ	–
[ACL+22] (latt.)	const. deg. poly	$p(\lambda)(n^{2\delta}+\ell)$	$p(\lambda)\log n$	$p(\lambda)\log^2 \ell n$	✓
[CFT22] (pair.)	const. deg. poly	$\lambda\ell n^{2\delta}$	$\lambda\delta_f$	$\lambda\delta_f$	✓
This work:					
Corol. 1.1 (pair.)	AC of width $\leq w$	λw^5	λ	$\lambda d_{\mathcal{C}}^2$	✓
Corol. 1.3 (pair.)	AC of size $\leq S$	λS^5	λ	$\lambda d_{\mathcal{C}}$	✓
Corol. 2.2 (latt.)	AC of width $\leq w$	$p(\lambda)w^5$	$p(\lambda)\log w$	$p(\lambda)d_{\mathcal{C}}\log^2 w$	✓

Table 1: Comparison of FC schemes for functions with n inputs and ℓ outputs. Constants are omitted, e.g., λn means $O(\lambda n)$ and $p(\cdot)$ represents some arbitrary polynomial function. For semi-sparse polynomials $\mu \geq n$ is a sparsity-dependent parameter (cf. [LP20]). For constant-degree polynomials δ_f is the degree of the polynomial f used in opening while δ is the maximum degree fixed at setup. AC means arithmetic circuits, $d_{\mathcal{C}}$ the depth of the circuit $\mathcal{C}$ used in opening, and note that $w \geq n, \ell$. AH means 'additively homomorphic'; schemes meeting this property can be turned into homomorphic signatures.

of these two CFC constructions (and considering a few tradeoffs of our generic construction), we obtain a variety of FC schemes; we summarize in Table 1 the most representative ones.

Our FC schemes enjoy useful additional properties.

1. They are *additively homomorphic*, which as shown in [CFT22] makes the FC updatable and allows for building homomorphic signatures (HS). Notably, our new FC for circuits yields new HS realizations that advance the state of the art (see slightly below for details).
2. They enjoy *amortized efficient verification*, which means that the verifier can precompute a verification key $\mathsf{vk}_{\mathcal{C}}$ associated to a circuit $\mathcal{C}$ and use this key (an unbounded number of times) to verify openings for $\mathcal{C}$ in time (asymptotically) faster than evaluating $\mathcal{C}$.
3. Our FC schemes can be trivially modified to have *perfectly hiding commitments* and efficiently compiled into FCs with *zero-knowledge openings*.

Both efficient verification and zero-knowledge openings are relevant in the construction of HS from FCs since, as showed in [CFT22], they imply the analogous properties of efficient verification [CFW14, GVW15] and *context hiding* [BF11] in the resulting HS schemes.

Application to Homomorphic Signatures. Homomorphic signatures (HS) [JMSW02, BF11] allow a signer to sign a large dataset $\boldsymbol{x}$ in such a way that anyone, holding a signature on $\boldsymbol{x}$, can perform a computation f on this data and derive a signature $\sigma_{f,y}$ on the output $y = f(\boldsymbol{x})$. This signature vouches for the correctness of y as output of f on some legitimately signed data and is publicly verifiable given a verification key, a description of f, and the result y. The most expressive HS in the state of the art is the scheme of Gorbunov, Vaikuntanathan and Wichs [GVW15] that is based on lattices and supports circuits with bounded number of inputs n and bounded (polynomial) depth d. In their scheme, the signature size grows polynomially with the depth of the evaluated circuit (precisely, as $d^3 \cdot \mathsf{poly}(\lambda)$).

By applying a recently proposed transformation [CFT22], our new FCs for circuits yield new HS that support the same class of functions and succinctness as supported by the FC, advancing the state of the art. Notably, we obtain:

– *The first HS for circuits based on pairings.* Previously existing HS based on pairings can capture at most circuits in NC^1 [KNYY19, CFT22] and need a bound on the circuit size. In contrast, our HS can evaluate circuits of any polynomial depth, achieving virtually the same capability of the lattice-based HS of [GVW15] and with better succinctness. We believe this result is interesting as it shows for the first time that we can build HS for circuits without the need of algebraic structures, such as lattices, that are notoriously powerful.
– *The first HS that do not require an a-priori bound on the depth.* The work of Gorbunov, Vaikuntanathan and Wichs [GVW15] left open the problem of constructing *fully-homomorphic* signatures, i.e., HS that can evaluate any computation in the class P without having to fix any bound at key generation time. In our new HS we do not need to fix a bound on the depth but we rather need a bound on the width of the circuits at key generation time. Although this result does not fully solve the open problem of realizing fully-homomorphic signatures, we believe that our schemes make one step ahead in this direction. Our observation is that dealing with a bound on the circuit's depth is more difficult than dealing with a bound on the width. As evidence for this, we show a variant of our FC scheme (see Section 5.1) for which one can fix a bound n and support circuits of larger width $O(n)$ with an $O(1)$ increase in proof size. Therefore, while our solution needs a bound on the width, this is not strict, as opposed to the depth bound in the HS of [GVW15].

Like the scheme of [GVW15], our HS constructions have efficient (offline/online) verification and are context-hiding. As a drawback, our HS allow only a limited form of multi-hop evaluation, that is the ability of computing on already evaluated signatures. In our case, we can compose computations sequentially (i.e., given a signature $\sigma_{f,y}$ for $\boldsymbol{y} = f(\boldsymbol{x})$ we can generate one for $\boldsymbol{z} = g(\boldsymbol{y}) = g(f(\boldsymbol{x}))$), while [GVW15] supports arbitrary compositions (e.g., given signatures for $\{\boldsymbol{y}_i = f_i(\boldsymbol{x})\}_i$, one can generate one for $z = g(f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})))$. On the other hand, for circuits with multiple outputs, the size of our signatures is independent of the output size, whereas in [GVW15] signatures grow linearly with the number of outputs.

Our Novel Tool: Chainable Functional Commitments. The key novelty that allows us to overcome the barrier in the state of the art and build the first FCs for circuits is the introduction and realization of chainable functional commitments (CFC) – a new primitive of potentially independent interest.

In brief, a CFC is a functional commitment where one can "open" to *committed outputs*. More concretely, while a (basic) FC allows proving statements of the form "$f(\boldsymbol{x}) = \boldsymbol{y}$" for committed $\boldsymbol{x}$ and publicly known $\boldsymbol{y}$, a CFC allows generating a proof π_f that com_y is a commitment to $\boldsymbol{y} = f(\boldsymbol{x}_1, \ldots \boldsymbol{x}_m)$ for vectors $\boldsymbol{x}_1, \ldots \boldsymbol{x}_m$, each independently committed in $\mathsf{com}_1, \ldots, \mathsf{com}_m$. In terms of security, CFCs must satisfy the analogue of evaluation binding, that is one cannot open the same input commitments $(\mathsf{com}_1, \ldots, \mathsf{com}_m)$ to two distinct output commitments $\mathsf{com}_y \neq \mathsf{com}'_y$ for the same f.

Keeping outputs committed is what makes CFCs "chainable", in the sense that committed outputs can serve as (committed) inputs for other openings. For instance, using the syntax above, one can compute an opening π_g proving that com_z is a commitment to $\boldsymbol{z} = g(\boldsymbol{y})$. This way, the concatenation of $\mathsf{com}_y, \pi_f, \pi_g$ yields a proof that $\boldsymbol{z} = g(f(\boldsymbol{x}_1, \ldots \boldsymbol{x}_m))$.

The introduction and realization of CFCs are in our opinion the main conceptual and technical contributions of this paper. From a conceptual point of view, the chaining functionality turns out to be a fundamental feature to tackle the challenge of supporting a computation as expressive as an arithmetic circuit. Indeed, we show that from a CFC for quadratic polynomial maps it is possible to construct a (C)FC for arithmetic circuits. From the technical point of view, we propose new techniques that depart from the ones of existing FCs for polynomials [ACL+22, CFT22] in that the latter only work when the output vector is known to the verifier and there is a single input commitment. We refer to Section 2 for an informal explanation of our techniques.

1.2 Related and Concurrent Work

As noticed in previous work, it is possible to construct an FC for arbitrary computations from a universal SNARK and a succinct commitment scheme by generating a succinct commitment to the input $\boldsymbol{x}$ and a SNARK proof for the statement "$f(\boldsymbol{x}) = y$ and $\boldsymbol{x}$ opens the commitment correctly". The drawback of this solution is that, by reducing to the knowledge-soundness of the SNARK, it would require non-falsifiable assumptions [GW11]. Alternatively, one could also reduce the security of this construction to the tautological (but falsifiable) assumption that the very same construction is secure. While such an argument is logically correct, it yields a non-standard assumption against the spirit of modern complexity-based cryptography. Hence, one of the goals in the FC literature is to construct schemes based on simple assumptions, which is the direction taken in this work.

The idea of a commitment scheme where one can open to functions of the committed data was implicitly suggested by Gorbunov, Vaikuntanathan and Wichs [GVW15], though their construction is not succinct as the commitment size is linear in the length of the vector. Libert, Ramanna and Yung [LRY16] were the first to formalize *succinct* functional commitments. They proposed a succinct FC for linear forms and showed applications of this primitive to polynomial commitments [KZG10] and accumulators. Recent works have extended FCs to support more expressive functions, including linear maps [LM19], semi-sparse polynomials [LP20], and constant-degree polynomials [ACL+22, CFT22]. Table 1 presents a comparison of these works with our results. Catalano, Fiore and Tucker [CFT22] also proposed an FC for monotone span programs, which only achieves a weaker notion of evaluation binding where the adversary must reveal the committed vector. A weaker security model is also considered in [PPS21], who introduced a lattice-based FC scheme where a trusted authority is assumed to generate, using a secret key, an opening key for each function for which the prover wants to release an opening.

Compared to these prior works, ours addresses the main question left open in the state of the art, which is to construct FCs for arbitrary computation from falsifiable assumptions.

Verifiable Computation. The functionality of functional commitments has similarities with verifiable computation (VC) schemes (also known as SNARGs for P). The main difference between VC and FC schemes is that in the latter, the input is committed as opposed to publicly known. Looking ahead, our generic construction of FC from CFCs presents a similar high-level approach as the SNARGs for P in [GR19] and [GZ21]. In particular, both constructions proceed level-by-level in the circuit (an idea that dates back to the GKR protocol [GKR08]). Then, the prover 1) computes a set of commitments to the wires at each level, and 2) proves that the committed vectors are consistent with respect to the circuit evaluation.

Beyond this similarity, our construction and [GR19, GZ21] differ in techniques and the level of security that we achieve. Notably, even though the verifier in [GR19, GZ21] may not need to see the opening of the commitment at each level, soundness only holds with respect to adversaries that reveal such opening. This translates into requiring the verifier to know the input (which is sufficient for VC but not for FC). Besides, [GR19, GZ21] have a function-specific setup, as opposed to FCs in which public parameters should be universal and functions are to be chosen at opening time.

Concurrent Work. Concurrently to our work, de Castro and Peikert [dCP23], and Wee and Wu [WW23], also propose lattice-based constructions of functional commitments for circuits (as well as polynomial and vector commitments). Their approaches differ significantly from ours, as they both rely on homomorphic evaluation techniques [GSW13].

The work of [dCP23] constructs a "dual" FC (where one commits to the function f and proves that $f(\boldsymbol{x}) = \boldsymbol{y}$ for a given $\boldsymbol{x}$)[7] for bounded-depth boolean circuits. Their construction is selectively secure under the standard SIS assumption and admits a transparent setup (i.e., the public parameters are a uniformly random string). Their FC does not have succinct openings though, as the opening size is linear in either the input size or the size of f (in our setting where one commits to f and opens to $\boldsymbol{x}$).

The FC in [WW23] supports circuits of bounded depth, needs a structured setup, and is secure under a new structured-BASIS assumption introduced in the same work. Their FC has succinct openings that are polylogarithmic in the input size and polynomial in the circuit depth.

In comparison to [dCP23, WW23], our FC schemes support circuits of bounded width but *unbounded depth*, with succinct openings that grow only with the depth of the circuit but are independent of the input size. As [WW23], we require a trusted setup and achieve adaptive security based on new falsifiable assumptions on either pairings or lattices. For the lattice-based FC we rely on the Twin-k-R-ISIS assumption which is weaker than the BASIS assumption of [WW23] (see Section 7 for a comparison). Our FC schemes are the only ones that (i) have openings succinct also in the output size,[8] and (ii) achieve fast verification with pre-processing (i.e., after an input-independent preprocessing verification time is sub-linear in the size of $|f|$ and $|\boldsymbol{x}|$).

2 A Technical Overview of Our Work

We construct our FCs for circuits in two main steps: (1) a generic construction of (C)FC for circuits from CFCs for quadratic polynomial maps (Section 5), and (2) the realization of these CFCs based on either pairings (Section 6) or lattices (Section 7). Below we give an informal overview of these constructions.

2.1 (C)FC for Circuits from CFCs for Quadratic Functions

Our first result is a transformation from CFCs for quadratic polynomials to FCs for circuits that is summarized in the following theorem.

Theorem 2 (informal) *Let* CFC *be a* chainable *functional commitment for quadratic polynomial maps* $f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) = \boldsymbol{y}$ *for any number of inputs* m*, such that each committed input vector* $\boldsymbol{x}_i$ *and the committed output* $\boldsymbol{y}$ *have length* n*. Then, there exist a functional commitment* FC *for arithmetic circuits of bounded width* n *and unbounded depth* d*, such that:*

- FC*'s commitment size is the same as that of* CFC*;*

[7] One can recover the standard notion of committing to $\boldsymbol{x}$ and opening to f via universal evaluators.

[8] This means that our FCs satisfy compactness as defined in [LM19] for subvector and linear map commitments.

- *if* CFC *has opening proofs of size* $s(n,m)$*, then* FC *has openings of size at most* $d \cdot s(n,d)$*. Moreover, if* CFC *is additively homomorphic and/or efficiently verifiable, so is* FC*.*

Our transform starts from the observation that the gates of an arithmetic circuit[9] can be partitioned into "levels" according to their multiplicative depth, i.e., level h contains all the gates of multiplicative depth h and level 0 contains the inputs. So, all the outputs of level h, denoted by $\boldsymbol{x}^{(h)}$, are computed by a quadratic polynomial map taking inputs from previous levels $< h$, and thus the evaluation of a circuit $\mathcal{C}$ of width $\leq n$ and depth d can be described as the sequential evaluation of quadratic polynomial maps $f^{(h)} : \mathcal{X}^{nh} \to \mathcal{X}^n$ for $h = 1$ to d.

The basic idea of our generic FC is that, starting with a commitment com_0 to the inputs $\boldsymbol{x}^{(0)}$, we can open it to $\boldsymbol{y} = \mathcal{C}(\boldsymbol{x}^{(0)})$ in two steps. First, we commit to the outputs of every level. Second, we use the CFC opening functionality to prove that these values are computed correctly from values committed in previous levels. Slightly more in detail, at level h we create a commitment com_h to the outputs $\boldsymbol{x}^{(h)} = f^{(h)}(\boldsymbol{x}^{(0)}, \ldots, \boldsymbol{x}^{(h-1)})$ and generate a CFC opening proof π_h to show consistency w.r.t. commitments $\mathsf{com}_0, \ldots, \mathsf{com}_h$. Eventually, this strategy reaches the commitment com_d of the last level that includes the outputs, which can be opened to $\boldsymbol{y}$ (or kept committed if one wants to build a CFC for circuits). The final proof π consists of all intermediate proofs and commitments, $\pi := (\pi_1, \ldots, \pi_d, \mathsf{com}_1, \ldots, \mathsf{com}_{d-1})$.

Security reduces to the security of the CFC for quadratic functions. To see this, consider an adversary that breaks FC evaluation binding by coming up with proofs π, π' that verify for $\boldsymbol{y} \neq \boldsymbol{y}'$ and for the same com_0. Then, there *must* exist some level h such that the intermediate commitments $\mathsf{com}_h \neq \mathsf{com}'_h$ differ (where possibly $h = d$). If we take h^* to be the smallest amongst such h, then we can break evaluation binding of the quadratic CFC at level h^*.

As one can see, this construction makes our opening proofs grow with the depth of the circuit. However, if the CFC commitments and opening proofs are short (e.g., $s(n,m)$ is constant/logarithmic in n, that is the circuit's width), then the FC openings keep only such dependence on the depth.[10] In addition, we describe different strategies to (a) reduce the opening size for not-so-densely connected circuits (for instance layered circuits), and (b) overcome the width bound without changing the parameters at the expense of increased opening size.

2.2 A Framework for CFCs for Quadratic Functions

We next overview our general strategy of construction CFCs for quadratic functions, which admits pairing- and lattice-based instantiations.

Theorems 4 and 5 (informal). *Assuming the n-HiKer assumption (resp. the Twin-k-R-*ISIS *assumption), our pairing-based (resp. lattice-based) CFC construction is a succinct CFC scheme for quadratic functions over any m vectors of length $\leq n$ that admits efficient verification, is additively homomorphic, and whose openings can be made zero-knowledge. For arbitrary quadratic functions, the opening proofs have size* $s(n,m) = \mathcal{O}(m^2)$ *(resp.* $s(n,m) = \mathcal{O}(m \cdot \mathsf{polylog}(m \cdot n))$*)*[11].

[9] In our model we assume wlog arithmetic circuits where every gate is a quadratic polynomial of unbounded fan-in.

[10] The CFC openings size $s(n,d)$ may be linear in d for "dense" quadratic functions, but this would contribute at most an additional factor of d to the succinctness of FC in the worst case.

[11] Following Theorem 2, this gives a proof size of $\mathcal{O}(d^3)$ for our pairing-based FC and $\mathcal{O}(d^2 \cdot \mathsf{polylog}(d \cdot w))$ for our lattice-based FC for circuits of depth d and width w. Nevertheless, the proof size can be reduced by a factor of d in both cases, as we show in Table 1. We refer to Sections 6 and 7 for details.

To build our CFCs we devise new commitment and opening techniques that capture a quadratic polynomial map $\boldsymbol{y} = f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$ where each input is committed in com_i, and the output is committed too in $\mathsf{com}_{\boldsymbol{y}}$. Our two constructions (pairing-based and lattice-based) of CFCs for quadratic functions have a similar high-level design that we introduce below.

For the pairing setting we adopt the implicit notation for bilinear groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order q by which $[\boldsymbol{x}]_s$ denotes the vector of group elements $(g_s^{x_1}, \ldots, g_s^{x_n}) \in \mathbb{G}_s^n$ for a fixed generator g_s. For the lattice setting, we let $\mathcal{R}$ be a cyclotomic ring and q be a large enough rational prime. In this overview, we adopt the bracket notation $[x]$ to express the representation of a given group or ring element without further distinction.

Abstract functionality. To start, we define three (vectors of) commitment keys $[\boldsymbol{\alpha}]$, $[\boldsymbol{\beta}]$, and $[\boldsymbol{\gamma}]$, that live either in $\mathbb{G}_1^n$ in the pairing setting, or in $\mathcal{R}_q^n$ in the lattice setting. A commitment of type α to a vector $\boldsymbol{x} \in \mathbb{Z}_q^n$ is computed *à la* Pedersen, i.e., via an inner product, as $X^{(\alpha)} = [\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle]$. Commitments of type β and γ are defined analogously.

In our CFCs the commitments generated by the commit algorithm Com and used by the opening algorithm Open are only those of type α, whereas commitments of type β and γ are used as auxiliary values in the opening proofs. In order to create a CFC opening to a quadratic polynomial, our main tool is a technique realizing the following functionality:

– $[(\alpha, \beta) \to \gamma]$-Quadratic opening: given m commitments for each of the keys $\{X_i^{(\alpha)} = [\langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle], X_i^{(\beta)} = [\langle \boldsymbol{x}_i, \boldsymbol{\beta} \rangle]\}_{i=1\ldots m}$ and a commitment $Y^{(\gamma)} = [\langle \boldsymbol{y}, \boldsymbol{\gamma} \rangle]$ generate a succinct opening proof $\pi_f^{(\gamma)}$ that $\boldsymbol{y} = f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$.

Before seeing how we generate this opening, we observe that $\pi_f^{(\gamma)}$ does not yet achieve our goal since it assumes the availability of both type-α and type-β commitments on the inputs, and it only allows us to "move" to a type-γ commitment of the output, preventing us from achieving chainability.

We solve both issues by designing two special cases of the functionality above:

– $[\alpha \to \beta]$-Identity opening: given a type-α commitment $X^{(\alpha)} = [\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle]$ show that a type-β commitment $X^{(\beta)}$ commits to the same $\boldsymbol{x}$, i.e., $X^{(\beta)} = [\langle \boldsymbol{x}, \boldsymbol{\beta} \rangle]$;
– $[\gamma \to \alpha]$-Identity opening: given a type-γ commitment $Y^{(\gamma)} = [\langle \boldsymbol{y}, \boldsymbol{\gamma} \rangle]$ show that a type-α commitment $Y^{(\alpha)}$ commits to the same $\boldsymbol{y}$, i.e., $Y^{(\alpha)} = [\langle \boldsymbol{y}, \boldsymbol{\alpha} \rangle]$.

We use the identity opening mechanisms to "close the circle" in such a way to obtain a quadratic opening mechanism where all inputs and outputs are only type-α commitments. To summarize, our CFC Open algorithm consists of the following steps:

(i) compute a type-β commitment $X_i^{(\beta)}$ to each input along with an $[\alpha \to \beta]$-Identity opening proof that $X_i^{(\beta)}$ commits to the same $\boldsymbol{x}_i$ in $X_i^{(\alpha)}$;
(ii) compute a type-γ commitment $Y^{(\gamma)}$ to the result $\boldsymbol{y} = f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$ and a $[(\alpha, \beta) \to \gamma]$-Quadratic opening proof attesting the validity of $\boldsymbol{y}$ w.r.t. the input commitment pairs $(X_i^{(\alpha)}, X_i^{(\beta)})$;
(iii) finally, use the $[\gamma \to \alpha]$-identity opening to ensure that $Y^{(\alpha)}$ is a commitment to the same $\boldsymbol{y}$ in the $Y^{(\gamma)}$ computed in (ii).

Our $[(\boldsymbol{\alpha}, \boldsymbol{\beta}) \to \boldsymbol{\gamma}]$-quadratic opening method. We use the fact that a quadratic polynomial map $f : \mathcal{X}^{nm} \to \mathcal{X}^n$ can be linearized via appropriately defined vector $\boldsymbol{e}$ and matrices $\mathbf{F}_i$ and $\mathbf{G}_{i,j}$ such that

$$\boldsymbol{y} = f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) = \boldsymbol{e} + \sum_i \mathbf{F}_i \cdot \boldsymbol{x}_i + \sum_{i,j \geq i} \mathbf{G}_{i,j} \cdot (\boldsymbol{x}_i \otimes \boldsymbol{x}_j)$$

where $\otimes$ denotes the tensor product.

In this overview, we only show how to produce an opening proof for a single quadratic term, i.e., to show that $\boldsymbol{y}_{i,j} = \mathbf{G}_{i,j} \cdot (\boldsymbol{x}_i \otimes \boldsymbol{x}_j)$ given input commitments $X_i^{(\alpha)}, X_i^{(\beta)}, X_j^{(\alpha)}, X_j^{(\beta)}$ and output $Y_{i,j}^{(\gamma)}$. This is the core of our technique since the full opening for f is obtained by doing an additive aggregation of openings for all the terms in the sum.

To open to $\mathbf{G}_{i,j}$, we first ensure that the verifier knows a commitment $Z_{i,j}$ to the tensor product $\boldsymbol{x}_i \otimes \boldsymbol{x}_j$, calculated as

$$Z_{i,j} := [\langle \boldsymbol{x}_i \otimes \boldsymbol{x}_j, \boldsymbol{\alpha} \otimes \boldsymbol{\beta} \rangle].$$

The way in which the verifier obtains $Z_{i,j}$ varies in the pairing and lattice constructions. Then, the prover generates a linear map opening that the vector $\boldsymbol{y}_{i,j}$ in the type-γ commitment $Y_{i,j}^{(\gamma)}$ is the result of applying $\mathbf{G}_{i,j}$ to the vector committed in $Z_{i,j}$. We compute this proof as follows. Denote with $\mathbf{G}_{i,j,k}$ the k-th row of $\mathbf{G}_{i,j}$ and with $\frac{1}{\boldsymbol{\alpha} \otimes \boldsymbol{\beta}}$ the component-wise inverse of $\boldsymbol{\alpha} \otimes \boldsymbol{\beta}$. Let

$$\mathit{\Gamma}_{i,j} = \sum_{k=1}^{n} \mathbf{G}_{i,j,k} \cdot \left[\frac{\gamma_k}{\boldsymbol{\alpha} \otimes \boldsymbol{\beta}} \right]$$

be an encoding of the matrix $\mathbf{G}_{i,j}$ that should be computable by the verifier (who can also precompute $\mathit{\Gamma}_{i,j}$). Then we rely on the fact that

$$Z_{i,j} \cdot \mathit{\Gamma}_{i,j} = [\langle \underbrace{\mathbf{G}_{i,j} \cdot \boldsymbol{x}_i \otimes \boldsymbol{x}_j}_{\boldsymbol{y}_{i,j}}, \boldsymbol{\gamma} \rangle] + \sum_{(h,l) \neq (h',l'),k} c_{h,l,h',l',k} \cdot \left[\frac{\alpha_{h'} \beta_{l'}}{\alpha_h \beta_l} \gamma_k \right]. \tag{1}$$

Namely, $Z_{i,j} \cdot \mathit{\Gamma}_{i,j}$ can be split into the sum between a non-rational term that actually encodes the (commitment to the) result $[\langle \boldsymbol{y}_{i,j}, \boldsymbol{\gamma} \rangle]$, and a linear combination of rational monomials, which is eventually encoded as part of the opening proof, and whose coefficients can be efficiently computed given $\mathbf{G}_{i,j}, \boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

For the prover to prove such splitting, and for the verifier to compute the encoding, we need to include additional elements in the public parameters. In particular, we add: $[\boldsymbol{\alpha} \otimes \boldsymbol{\beta}]$ for computing $Z_{i,j}$, $\left[\frac{\gamma_k}{\boldsymbol{\alpha} \otimes \boldsymbol{\beta}} \right]$ for computing $\mathit{\Gamma}_{i,j}$, and $\left[\frac{\alpha_{h'} \beta_{l'}}{\alpha_h \beta_l} \gamma_k \right]$ for computing the sum in (1). To obtain security, we instantiate the different commitments and verification checks over pairing groups and lattice rings, whose particularities we describe next.

2.3 Pairing-Based CFC

In our pairing-based CFC in Section 6, the elements in the public parameters belong to the groups $\mathbb{G}_1$ and $\mathbb{G}_2$, and the input commitment is computed in $\mathbb{G}_1$ as $X^{(\alpha)} = [\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle]_1$. For the verifier to obtain the commitment to the tensor product $Z_{i,j}$, the prover calculates and sends $Z_{i,j} := [\langle \boldsymbol{x}_i \otimes \boldsymbol{x}_j, \boldsymbol{\alpha} \otimes \boldsymbol{\beta} \rangle]_1$ and $X_i^{(2)} := [\langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle]_2$, and the verifier checks

$$e(X_i^{(\alpha)}, [1]_2) \stackrel{?}{=} e([1]_1, X_i^{(2)})$$

to test that $X_i^{(2)} \in \mathbb{G}_2$ encodes the same vector of $X_i^{(\alpha)} \in \mathbb{G}_1$, and

$$e(Z_{i,j}, [1]_2) \stackrel{?}{=} e(X_j^{(\beta)}, X_i^{(2)})$$

to test the well-formedness of $Z_{i,j}$. To let the prover compute this, we add elements $[\boldsymbol{\alpha}]_2$ in $\mathbb{G}_2$ to the public parameters.

Finally, the prover computes and sends $\pi_{i,j}^{(\gamma)} = \sum_{(h,l)\neq(h',l'),k} c_{h,l,h',l',k} \cdot [\frac{\alpha_{h'}\beta_{l'}}{\alpha_h \beta_l}\gamma_k \eta_\gamma]_1$. This way, the verifier can test equation (1) using pairings as

$$e\left(Z_{i,j}, [\mathit{\Gamma}_{i,j}]_2\right) \stackrel{?}{=} e\left(Y_{i,j}^{(\gamma)}, [\eta_\gamma]_2\right) e\left(\pi_{i,j}^{(\gamma)}, [1]_2\right). \tag{2}$$

Note that we are introducing an additional variable η_γ in the verification, which is central to the security of the scheme. More precisely, in our pairing-based CFC we provide in the public parameters the elements: $[\eta_\gamma]_2$ (to be used in the verification above), $\{[\frac{\gamma_k \eta_\gamma}{\boldsymbol{\alpha}\otimes\boldsymbol{\beta}}]_2\}_k$ (used to compute $[\mathit{\Gamma}_{i,j}]_2$), and $\{[\frac{\alpha_{h'}\beta_{l'}}{\alpha_h\beta_l}\gamma_k\eta_\gamma]_1\}_{(h,l)\neq(h',l'),k}$ (to compute the proof $\pi_{i,j}^{(\gamma)}$). The security of the scheme relies precisely on the fact that the public parameters do not include any term of the form $[\gamma_k\eta_\gamma]_1$ in the group $\mathbb{G}_1$.

To see how this relates to the scheme, suppose that one breaks evaluation binding by finding two proofs $\pi, \tilde{\pi}$ that open to different commitments $Y_{i,j}^{(\gamma)}$ and $\tilde{Y}_{i,j}^{(\gamma)}$ for the same function $\mathbf{G}_{i,j}$. Then, by (2), we can compute $U = \tilde{Y}_{i,j}^{(\gamma)}/Y_{i,j}^{(\gamma)}$ and $V = \pi/\tilde{\pi}$ such that (U, V) is in the linear span of $([1]_1, [\eta_\gamma]_1)$. However, elements of this form cannot be derived from linear combinations of group elements in the public parameters. This is captured formally by our HintedKernel (HiKer) assumption, which we justify in the generic (bilinear) group model (Appendix B). Our HiKer assumption can be seen as a "hinted" version of the KerMDH assumptions of [MRV16].[12]

In terms of succinctness, the opening proof size of our pairing-based CFC is linear in the density of the quadratic polynomial, that is the number of nonzero quadratic terms $\boldsymbol{x}_i\boldsymbol{x}_j$, which is in the worst case quadratic on the number m of input commitments. This is due to the fact that, even if we can compress all proofs $\pi_{i,j}^{(\gamma)}$ in one, the prover still needs to provide every $Z_{i,j}$ for $1 \leq i \leq j \leq m$. Fortunately, in our construction of FC for circuits, we can reduce the opening size of the CFC at each layer from quadratic to linear. We refer to Corollary 1 for further details.

2.4 Lattice-Based CFC

In our lattice-based CFC in Section 7, we sample commitment keys $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$ uniformly from $\mathcal{R}_q^n$. The public parameters also contain two trapdoored matrices $\mathbf{A}, \mathbf{B} \in \mathcal{R}_q^{\eta\times\ell}$ and a vector $\boldsymbol{t} \in \mathcal{R}_q^{\eta}$, where $\mathcal{R}$ is the ring of integers of a cyclotomic field, and η, ℓ are determined by the trapdoor sampling algorithm. Instead of providing the ring elements $\boldsymbol{\alpha}\otimes\boldsymbol{\beta}$ (and all elements that result from evaluating diverse monomials g on $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$) as in the pairing-based construction, we include a short preimage $\boldsymbol{u}_g$ of each ring element such that $\mathbf{A} \cdot \boldsymbol{u}_g \equiv \boldsymbol{t} \cdot g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \bmod q$, obtained with the help of the trapdoors[13].

Given commitments $X_i^{(\alpha)} = \langle \boldsymbol{\alpha}, \boldsymbol{x}_i \rangle$, the verifier can easily compute the commitment to the tensor product $Z_{i,j} = X_i^{(\alpha)} \cdot X_j^{(\beta)} = \langle \boldsymbol{\alpha}\otimes\boldsymbol{\beta}, \boldsymbol{x}_i \otimes \boldsymbol{x}_j \rangle$ thanks to the ring structure of $\mathcal{R}_q$. We note that in the scheme, we need to make an additional restriction that both the vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ and the coefficients of the polynomial map f are short. This implies that the coefficients $c_{h,l,h',l',k}$ in equation (1) are also short.

[12] For matrices $[\mathbf{A}]_2$ from certain (random) distributions, KerMDH asks the adversary to find a nonzero vector $[\boldsymbol{z}]_1$ such that $\mathbf{A}\boldsymbol{z} = \mathbf{0}$. In HiKer, the adversary is challenged to find a nonzero $[z]_1 = [u, v]_1$ such that $u\eta + v = 0$, when given $[\mathbf{A}]_2 = [1, \eta]_2$, but also other group elements, the "hints", that depend on η and other random variables.

[13] Preimages of some monomials with respect to $\mathbf{B}$ are also included in the public parameters, but we omit them in this overview.

With this restriction, we enable the proof of the split using the short preimages of each ring element $\frac{\alpha_{h'}\beta_{l'}}{\alpha_h\beta_l}\gamma_k$ available in the public parameters. This allows the prover to compute a short preimage $\boldsymbol{u}_{i,j}^{(\gamma)}$ for the element $Z_{i,j} \cdot \mathit{\Gamma}_{i,j} - Y_{i,j}^{(\gamma)}$, which the verifier can efficiently check by $\mathbf{A} \cdot \boldsymbol{u}_{i,j}^{(\gamma)} \equiv \boldsymbol{t} \cdot (Z_{i,j} \cdot \mathit{\Gamma}_{i,j} - Y_{i,j}^{(\gamma)}) \bmod q$ (note again that $\mathit{\Gamma}_{i,j}$ depends on the function and can be pre-computed), in addition to a norm check on $\boldsymbol{u}_{i,j}^{(\gamma)}$.

In comparison to our pairing-based CFC, here the prover no longer needs to provide the $Z_{i,j}$ elements, but only the $X_i^{(\beta)}$ for every $1 \leq i \leq m$ such that $\boldsymbol{x}_i\boldsymbol{x}_j$ is non-zero for some j. Thus, the opening proof size of our lattice-based CFC is (at most) linear in the number m of committed vectors. Naively, this results in opening proofs for our lattice-based FC for circuits that grow quadratically on the circuit depth. However, in Corollary 2 we show how to reduce this dependency from quadratic to linear for any circuit (even non-layered ones). We also note that the lattice parameters need to be set as a function of the size n of the committed vectors, therefore the proof also grows (logarithmically) in the circuit width.

The security of the scheme essentially relies on the fact that no short preimage for ring elements γ_k is available to the prover. We capture this fact via the Twin-k-R-ISIS assumption (Section 7), which extends the k-R-ISIS assumption from [ACL+22]. Essentially, k-R-ISIS states that even when given short preimages $\boldsymbol{u}_g$ satisfying $\mathbf{A} \cdot \boldsymbol{u}_g \equiv \boldsymbol{t} \cdot g(\boldsymbol{v}) \bmod q$ for all g in a given set of monomials, it is hard to find a SIS solution (i.e. a short non-zero preimage $\boldsymbol{u}^*$) such that $\mathbf{A} \cdot \boldsymbol{u}^* \equiv \mathbf{0} \bmod q$. Our Twin-$k$-$R$-$\mathsf{ISIS}$ states that finding a solution $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ for $\mathbf{A} \cdot \boldsymbol{u}^* + \mathbf{B} \cdot \boldsymbol{v}^* \equiv \mathbf{0} \bmod q$ is still hard even if we also provide preimages of a *strictly different* set of monomials with respect to a second (independent) matrix $\mathbf{B}$.

In Section 7.1, we analyse Twin-k-R-ISIS and compare it to k-R-ISIS and the BASIS assumption from [WW23]. Although both our Twin-k-R-ISIS and HiKer assumptions are new and non-standard, we remark that they are well-parametrized assumptions with a simple winning condition, which differs from that of the FC scheme. As typical in the lifetime of new cryptographic primitives, we expect that future work can fill this gap.

3 Preliminaries

Notation. We denote by $\mathbb{N}$ the set of natural numbers > 0. We denote the security parameter by $\lambda \in \mathbb{N}$. We call a function ϵ *negligible*, denoted $\epsilon(\lambda) = \mathsf{negl}(\lambda)$, if $\epsilon(\lambda) = O(\lambda^{-c})$ for every constant $c > 0$, and call a function $p(\lambda)$ *polynomial*, denoted $p(\lambda) = \mathsf{poly}$, if $p(\lambda) = O(\lambda^c)$ for some constant $c > 0$. We say that an algorithm is *probabilistic polynomial time* (PPT) if it consumes randomness and its running time is bounded by some $p(\lambda) = \mathsf{poly}(\lambda)$. For a finite set S, $x \leftarrow\$ S$ denotes sampling x uniformly at random in S. For an algorithm A, we write $y \leftarrow A(x)$ for the output of A on input x. For a positive $n \in \mathbb{N}$, $[n]$ is the set $\{1, \ldots, n\}$. We denote vectors $\boldsymbol{x}$ and matrices $\mathbf{M}$ using bold fonts. For a ring $\mathcal{R}$, given two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{R}^n$, $\boldsymbol{z} := (\boldsymbol{x} \otimes \boldsymbol{y}) \in \mathcal{R}^{n^2}$ denotes their Kronecker product (that is a vectorization of the outer product), i.e., $\forall i, j \in [n] : z_{i+(j-1)n} = x_i y_j$.

3.1 Functional Commitments

In this section we give the definition of functional commitments (FC) for generic classes of functions, by generalizing the one given in [LRY16] for linear functions. For notational simplicity and without loss of generality, we give our definitions for functions that have n inputs and n outputs.

Definition 1 (Functional Commitments). *Let $\mathcal{X}$ be some domain and let $\mathcal{F} \subseteq \{f : \mathcal{X}^n \to \mathcal{X}^n\}$ be a family of functions over $\mathcal{X}$, with n inputs and n outputs. A functional commitment scheme for $\mathcal{F}$ is a tuple of algorithms* $\mathsf{FC} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ *that work as follows and that satisfy correctness and succinctness defined below.*

$\mathsf{Setup}(1^\lambda, 1^n) \to \mathsf{ck}$ *on input the security parameter λ and the functions parameters n, outputs a commitment key* ck.

$\mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r) \to (\mathsf{com}, \mathsf{aux})$ *on input a vector $\boldsymbol{x} \in \mathcal{X}^n$ and (possibly) randomness r, outputs a commitment* com *and related auxiliary information* aux.[14]

$\mathsf{Open}(\mathsf{ck}, \mathsf{aux}, f) \to \pi$ *on input an auxiliary information* aux *and a function $f \in \mathcal{F}$, outputs an opening proof π.*

$\mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi) \to b \in \{0, 1\}$ *on input a commitment* com*, an opening proof π, a function $f \in \mathcal{F}$ and a value $\boldsymbol{y} \in \mathcal{X}^n$, accepts ($b = 1$) or rejects ($b = 0$).*

Correctness. FC *is correct if for any $n \in \mathbb{N}$, all* $\mathsf{ck} \leftarrow\$ \mathsf{Setup}(1^\lambda, 1^n)$*, any $f : \mathcal{X}^n \to \mathcal{X}^n$ in the class $\mathcal{F}$, and any $\boldsymbol{x} \in \mathcal{X}^n$, if* $(\mathsf{com}, \mathsf{aux}) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x})$*, then*

$$\Pr[\mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, f(\boldsymbol{x}), \mathsf{Open}(\mathsf{ck}, \mathsf{aux}, f)) = 1] = 1.$$

Succinctness. *Let us assume that the admissible functions can be partitioned as $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa \in \mathcal{K}}$ for some set $\mathcal{K}$, and let $s : \mathbb{N} \times \mathcal{K} \to \mathbb{N}$ be a function. A functional commitment* FC *for $\mathcal{F}$ is said to be $s(n, \kappa)$-*succinct *if there exists a polynomial $p(\lambda) = \mathsf{poly}(\lambda)$ such that for any $\kappa \in \mathcal{K}$, function $f : \mathcal{X}^n \to \mathcal{X}^n$ s.t. $f \in \mathcal{F}_\kappa$, honestly generated commitment key* $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$*, vector $\boldsymbol{x} \in \mathcal{X}^n$, commitment* $(\mathsf{com}, \mathsf{aux}) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x})$ *and opening* $\pi \leftarrow \mathsf{Open}(\mathsf{ck}, \mathsf{aux}, f)$*, it holds that* $|\mathsf{com}| \leq p(\lambda)$ *and $|\pi| \leq p(\lambda) \cdot s(n, \kappa)$.*

In order to model and compare different constructions, the notion of succinctness that we introduce is parametric with respect to a function $s(n, \kappa)$ that depends on the input-output length n and some parameter κ of the evaluated function. In some cases we will express the function s using asymptotic notation. To give some examples, κ could be an integer expressing the depth/size of a circuit (and thus $\mathcal{F}_\kappa$ are all circuits of depth/size κ), the degree of a polynomial, or the running time of a Turing machine. Accordingly, $\mathcal{K}$ is a set that partitions the class of admissible functions, e.g., $\mathcal{K} = [D]$ if the admissible functions are all circuits of depth $\leq D$, or $\mathcal{K} = \mathbb{N}$ if one wants to capture circuits of any depth.

The security definition of FCs proposed in [LRY16] is called evaluation binding and says that a PPT adversary cannot open a commitment to two distinct outputs for the same function.

Definition 2 (Evaluation Binding). *For any PPT adversary $\mathcal{A}$, the following probability is* $\mathsf{negl}(\lambda)$*:*

$$\mathbf{Adv}^{\mathsf{EvBind}}_{\mathcal{A},\mathsf{FC}}(\lambda) = \Pr\left[\begin{matrix}\mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi) = 1 \\ \wedge\ \boldsymbol{y} \neq \boldsymbol{y}' \wedge \\ \mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}', \pi') = 1\end{matrix} \ : \ \begin{matrix}\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ (\mathsf{com}, f, \boldsymbol{y}, \pi, \boldsymbol{y}', \pi') \leftarrow \mathcal{A}(\mathsf{ck})\end{matrix}\right]$$

[14] In our constructions, we often omit r from the inputs; in such a case we assume either that r is randomly sampled or that the commitment algorithm is deterministic.

For simplicity of presentation, in all our security definitions, we omit checking the domains of the elements returned by the adversary, e.g., that $f \in \mathcal{F}$ and $\boldsymbol{y} \in \mathcal{X}^n$ etc.

We show that evaluation binding implies the classical binding notion.

Proposition 1. *Let* FC *be an FC scheme satisfying evaluation binding. Then* $\mathsf{FC.Com}$ *is a computationally binding commitment scheme, namely any PPT adversary has probability* $\mathsf{negl}(\lambda)$ *of finding a tuple* $(\boldsymbol{x}, r, \boldsymbol{x}', r')$ *such that* $\boldsymbol{x} \neq \boldsymbol{x}'$ *and* $\mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r) = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}'; r')$.

Proof. The proof is rather simple and works as follows. Consider an adversary $\mathcal{A}$ that returns $(\boldsymbol{x}, r, \boldsymbol{x}', r')$ such that $\boldsymbol{x} \neq \boldsymbol{x}'$ and $\mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r) = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}'; r')$ with non-negligible probability. Then we can use it to build an adversary $\mathcal{B}$ that returns $(\mathsf{com}, f, \boldsymbol{y}, \pi, \boldsymbol{y}', \pi')$ such that $\mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi) = \mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}', \pi') = 1$ and $\boldsymbol{y} \neq \boldsymbol{y}'$. To do so, $\mathcal{B}$ runs $\mathcal{A}$ and then looks for a function f such that $\boldsymbol{y} = f(\boldsymbol{x}) \neq f(\boldsymbol{x}') = \boldsymbol{y}'$, and computes $(\mathsf{com}, \mathsf{aux}) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r)$, $(\mathsf{com}', \mathsf{aux}') \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}'; r')$, $\pi \leftarrow \mathsf{Open}(\mathsf{ck}, \mathsf{aux}, f)$, $\pi' \leftarrow \mathsf{Open}(\mathsf{ck}, \mathsf{aux}', f)$. By the correctness of FC, π and π' must verify for $\boldsymbol{y}$ and $\boldsymbol{y}'$ respectively, and for the same commitment $\mathsf{com} = \mathsf{com}'$ (due to the break of binding by $\mathcal{A}$). Therefore, $\mathcal{B}$'s output is a valid attack against evaluation binding.

In Appendix A, we also recall two security notions that are strictly stronger than evaluation binding. The first is strong evaluation binding, introduced in [LM19]. In this notion, the adversary outputs a commitment com and a collection of openings to one or several function-output pairs $\{f_i, \boldsymbol{y}_i\}$, and we say that it wins if these define an inconsistent system of equations (i.e., there is no valid $\boldsymbol{x}$ such that $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$ for all i). Then, we introduce the notion of knowledge extractability and prove that if an FC is knowledge extractable, then it also satisfies strong evaluation binding.

3.2 Additional Properties of FCs

Here we define three extra properties of functional commitments that can be useful in applications.

Additive-homomorphic FCs. These are functional commitments where, given two commitments com_1 and com_2 to vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ respectively, one can compute a commitment to $\boldsymbol{x}_1 + \boldsymbol{x}_2$.

Definition 3 (Additive-homomorphic FCs [CFT22]). *Let* FC *be a functional commitment scheme where* $\mathcal{X}$ *is a ring. Then* FC *is additive homomorphic if there exist deterministic algorithms* $\mathsf{FC.Add}(\mathsf{ck}, \mathsf{com}_1, \ldots, \mathsf{com}_n) \to \mathsf{com}$, $\mathsf{FC.Add}_{\mathsf{aux}}(\mathsf{ck}, \mathsf{aux}_1, \ldots, \mathsf{aux}_n) \to \mathsf{aux}$ *and* $\mathsf{FC.Add}_r(\mathsf{ck}, r_1, \ldots, r_n) \to r$ *such that for any* $\boldsymbol{x}_i \in \mathcal{X}$ *and* $(\mathsf{com}_i, \mathsf{aux}_i) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i; r_i)$, *if* $\mathsf{com} \leftarrow \mathsf{FC.Add}(\mathsf{ck}, \mathsf{com}_1, \ldots, \mathsf{com}_n)$, $\mathsf{aux} \leftarrow \mathsf{FC.Add}_{\mathsf{aux}}(\mathsf{ck}, \mathsf{aux}_1, \ldots, \mathsf{aux}_n)$, *and* $r \leftarrow \mathsf{FC.Add}_r(\mathsf{ck}, r_1, \ldots, r_n)$, *then* $(\mathsf{com}, \mathsf{aux}) = \mathsf{Com}(\mathsf{ck}, \sum_{i=1}^{n} \boldsymbol{x}_i; r)$.

As shown in [CFT22], an additive-homomorphic FC can be used to construct multi-input homomorphic signatures, and it is also updatable.

Efficient Amortized Verification. An FC with this property enables the verifier to precompute a verification key vk_f associated to the function f, with which they can check any opening for f in time asymptotically faster than running f.

Definition 4 (Amortized efficient verification). *A functional commitment scheme* FC *for* $\mathcal{F}$ *has* amortized efficient verification *if there exist two additional algorithms* $\mathsf{vk}_f \leftarrow \mathsf{VerPrep}(\mathsf{ck}, f)$ *and* $b \leftarrow \mathsf{EffVer}(\mathsf{vk}_f, \mathsf{com}, \boldsymbol{y}, \pi)$ *such that for any* $n = \mathsf{poly}(\lambda)$, *function* $f : \mathcal{X}^n \to \mathcal{X}^n$ *s.t.* $f \in \mathcal{F}$, *any honestly generated commitment key* $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, *vector* $\boldsymbol{x} \in \mathcal{X}^n$, *commitment* $(\mathsf{com}, \mathsf{aux}) \leftarrow$

$\mathsf{Com}(\mathsf{ck}, \boldsymbol{x})$ *and opening* $\pi \leftarrow \mathsf{Open}(\mathsf{ck}, \mathsf{aux}, f)$*, it holds: (a)* $\mathsf{EffVer}(\mathsf{VerPrep}(\mathsf{ck}, f), \mathsf{com}, \boldsymbol{y}, \pi) = \mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi)$*, and (b) the running time of* EffVer *is* $o(T)$ *where* $T = T(\lambda)$ *is the running time of* $\mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi)$*.*

Hiding and Zero Knowledge. Intuitively, an FC is hiding if the commitments produced through Com are hiding, in the classical sense. For zero-knowledge, the goal is that the openings produced by Open should not reveal more information about the committed vector beyond what can be deduced from the output, i.e., that $\boldsymbol{x}$ is such that $\boldsymbol{y} = f(\boldsymbol{x})$.

We use the formal definitions introduced in [CFT22].

Definition 5 (Com-Hiding [CFT22]). *A FC has perfectly (resp. statistically, computationally) hiding commitments if there are simulator algorithms* $\mathsf{Sim} = (\mathsf{Sim}_{\mathsf{Setup}}, \mathsf{Sim}_{\mathsf{Com}}, \mathsf{Sim}_{\mathsf{Equiv}})$ *such that*

- *(i)* $\mathsf{Sim}_{\mathsf{Setup}}$ *generates indistinguishable keys, along with a trapdoor, i.e., the distributions* $\{\mathsf{ck} : \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)\}$ *and* $\{\mathsf{ck} : (\mathsf{ck}, \mathsf{td}) \leftarrow \mathsf{Sim}_{\mathsf{Setup}}(1^\lambda, n)\}$ *are identical (resp. statistically, computationally indistinguishable).*
- *(ii) for any vector* $\boldsymbol{x} \in \mathcal{X}^n$*, keys* $(\mathsf{ck}, \mathsf{td}) \leftarrow \mathsf{Sim}_{\mathsf{Setup}}(1^\lambda, n)$*, the following distributions are identical (resp. statistically, computationally indistinguishable):*

$$\{\mathsf{Com}(\mathsf{ck}, \boldsymbol{x})\} \approx \{(\mathsf{com}, \mathsf{aux}) : (\mathsf{com}, \widetilde{\mathsf{aux}}) \leftarrow \mathsf{Sim}_{\mathsf{Com}}(\mathsf{td}), \mathsf{aux} \leftarrow \mathsf{Sim}_{\mathsf{Equiv}}(\mathsf{td}, \mathsf{com}, \widetilde{\mathsf{aux}}, \boldsymbol{x})\}$$

Definition 6 (Zero-knowledge openings). *An FC has perfect (resp. statistical, computational) zero-knowledge openings if there is a simulator* $\mathsf{Sim} = (\mathsf{Sim}_{\mathsf{Setup}}, \mathsf{Sim}_{\mathsf{Com}}, \mathsf{Sim}_{\mathsf{Equiv}}, \mathsf{Sim}_{\mathsf{Open}})$ *such that*

- *(i)* $\mathsf{Sim}_{\mathsf{Setup}}$ *generates indistinguishable keys, along with a trapdoor, i.e., the distributions* $\{\mathsf{ck} : \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)\}$ *and* $\{\mathsf{ck} : (\mathsf{ck}, \mathsf{td}) \leftarrow \mathsf{Sim}_{\mathsf{Setup}}(1^\lambda, n)\}$ *are identical (resp. statistically, computationally indistinguishable).*
- *(ii) for any vector* $\boldsymbol{x} \in \mathcal{X}^n$*, keys* $(\mathsf{ck}, \mathsf{td}) \leftarrow \mathsf{Sim}_{\mathsf{Setup}}(1^\lambda, n)$*, functions* $f_1, \ldots, f_Q \in \mathcal{F}$*, and commitments* $(\mathsf{com}, \mathsf{aux}) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x})$ *and* $(\widetilde{\mathsf{com}}, \widetilde{\mathsf{aux}}) \leftarrow \mathsf{Sim}_{\mathsf{Com}}(\mathsf{ck})$*, the following two distributions are identical (resp. statistically, computationally indistinguishable):*

$$(\widetilde{\mathsf{com}}, \{\mathsf{Sim}_{\mathsf{Open}}(\mathsf{td}, \widetilde{\mathsf{aux}}, \widetilde{\mathsf{com}}, f_j, f_j(\boldsymbol{x}))\}_{j=1}^{Q}) \approx (\mathsf{com}, \{\mathsf{Open}(\mathsf{ck}, \mathsf{aux}, f_j)\}_{j=1}^{Q})$$

We state a simple result showing that an FC with hiding commitments (but not necessarily zero-knowledge openings) can be converted, via the use of a NIZK scheme, into one that also achieves zero-knowledge openings. The proof is straightforward and we show it in Appendix A.1.

Theorem 1. *Let* FC *be an FC scheme that satisfies com-hiding (Definition 5), and let* Π *be a knowledge-sound NIZK for the NP relation* $R_{\mathsf{FC}} = \{((\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}); \pi) : \mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi) = 1\}$*. Then there exists an FC scheme* FC^* *for the same class of functions supported by* FC *that has com-hiding and zero-knowledge openings. Furthermore, if* FC *is additive-homomorphic, so is* FC^**; if* FC *has efficient verification and* Π *supports* $R'_{\mathsf{FC}} = \{(\mathsf{vk}_f, \mathsf{com}, \boldsymbol{y}; \pi) : \mathsf{EffVer}(\mathsf{vk}_f, \mathsf{com}, \boldsymbol{y}, \pi) = 1\}$*, then* FC^* *has also efficient verification.*

4 Chainable Functional Commitments

As described in the introduction, we introduce the notion of Chainable Functional Commitments (CFC), which is an extension of the FC primitive that allows one to "chain" multiple openings to different functions.

Definition 7 (Chainable Functional Commitments). *Let $\mathcal{X}$ be some domain, $n = \mathsf{poly}(\lambda)$ and let $\mathcal{F} \subseteq \{f : \mathcal{X}^{nm} \to \mathcal{X}^n\}$ be a family of functions over $\mathcal{X}$ for any integer $m = \mathsf{poly}(\lambda)$. A chainable functional commitment scheme for $\mathcal{F}$ is a tuple of algorithms $\mathsf{CFC} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ that works as follows and that satisfies correctness and succinctness.*

$\mathsf{Setup}(1^\lambda, 1^n) \to \mathsf{ck}$ *on input the security parameter λ and the vector length n, outputs a commitment key* ck.

$\mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r) \to (\mathsf{com}, \mathsf{aux})$ *on input a vector $\boldsymbol{x} \in \mathcal{X}^n$ and (possibly) randomness r, outputs a commitment* com *and related auxiliary information* aux.

$\mathsf{Open}(\mathsf{ck}, (\mathsf{aux}_i)_{i\in[m]}, f) \to \pi$ *given auxiliary informations $(\mathsf{aux}_i)_{i\in[m]}$, one for every committed input, and a function $f \in \mathcal{F}$, returns an opening proof π.*

$\mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi) \to b \in \{0,1\}$ *on input commitments $(\mathsf{com}_i)_{i\in[m]}$ to the m inputs and com_y to the output, an opening proof π, and a function $f \in \mathcal{F}$, accepts $(b = 1)$ or rejects $(b = 0)$.*

Correctness. CFC *is correct if for any $n, m \in \mathbb{N}$, all $\mathsf{ck} \leftarrow\$ \mathsf{Setup}(1^\lambda, 1^n)$, any $f : \mathcal{X}^{nm} \to \mathcal{X}^n$ in the class $\mathcal{F}$, and any set of vectors $\{\boldsymbol{x}_i\}_{i\in[m]}$ such that $\boldsymbol{x}_i \in \mathcal{X}^n$, if $(\mathsf{com}_i, \mathsf{aux}_i) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i)$ for every $i \in [m]$ and $(\mathsf{com}_y, \mathsf{aux}_y) \leftarrow \mathsf{Com}(\mathsf{ck}, f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m))$,*

$$\Pr\left[\mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \mathsf{Open}(\mathsf{ck}, (\mathsf{aux}_i)_{i\in[m]}, f)) = 1\right] = 1.$$

Succinctness. *Let $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathcal{K}}$ for some set $\mathcal{K}$ and let $s : \mathbb{N} \times \mathbb{N} \times \mathcal{K}$ be a function. A chainable functional commitment* CFC *is* $s(n, m, \kappa)$-succinct *if there exists a polynomial $p(\lambda) = \mathsf{poly}(\lambda)$ such that for any n, m and $\kappa \in \mathcal{K}$, function $f : \mathcal{X}^{mn} \to \mathcal{X}^n$, $f \in \mathcal{F}_\kappa$, honestly generated commitment key $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, vectors $\boldsymbol{x}_i \in \mathcal{X}^n$ and commitments $(\mathsf{com}_i, \mathsf{aux}_i) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i)$ for $i \in [m]$, $(\mathsf{com}_y, \mathsf{aux}_y) \leftarrow \mathsf{Com}(\mathsf{ck}, f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m))$, and opening $\pi \leftarrow \mathsf{Open}(\mathsf{ck}, (\mathsf{aux}_i)_{i\in[m]}, f)$, it holds that $|\mathsf{com}_i|, |\mathsf{com}_y| \le p(\lambda)$ for every $i \in [m]$ and $|\pi| \le p(\lambda) \cdot s(n, m, \kappa)$.*

As in the case of FCs (Definition 1) we define succinctness in a parametric way, and we are interested in CFC constructions supporting non-trivial functions $s(n, m, \kappa)$ that are sublinear or constant in n, m.

Additive homomorphism and efficient verification. As for functional commitments, a CFC can also be additively homomorphic and have amortized efficient verification. We omit the formal definitions of these properties as they are analogous to Definition 3 and Definition 4 respectively.

Definition 8 (Evaluation Binding). *For any PPT adversary $\mathcal{A}$, the following probability is* $\mathsf{negl}(\lambda)$*:*

$$\Pr\left[\begin{array}{c} \mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi) = 1 \\ \wedge\ \mathsf{com}_y \neq \mathsf{com}'_y \wedge \\ \mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}'_y, f, \pi') = 1 \end{array} : \begin{array}{r} \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ \left((\mathsf{com}_i)_{i\in[m]}, f, \begin{array}{c}\mathsf{com}_y, \pi, \\ \mathsf{com}'_y, \pi'\end{array}\right) \leftarrow \mathcal{A}(\mathsf{ck}) \end{array}\right]$$

As one can notice, the above notion of evaluation binding can only hold in the case when the output commitments com_y are generated deterministically. This is still enough for using CFCs to construct FCs with hiding commitments to inputs and zero-knowledge openings (thanks to Theorem 1). We leave the definition of CFCs with hiding output commitments for future work.

We introduce the definition of a knowledge extractable CFC in Appendix A.

5 FC for Circuits from CFC for Quadratic Polynomials

In this section we introduce a generic construction of a Functional Commitment scheme for arithmetic circuits of bounded width n, from any Chainable Functional Commitment for quadratic functions over inputs of length n.

Circuit model and notation. Let $\mathcal{R}$ be a commutative ring. We consider arithmetic circuits $\mathcal{C} : \mathcal{R}^n \to \mathcal{R}^n$ where every gate is a quadratic polynomial with bounded coefficients. It is not hard to see that such a model captures the more common model of arithmetic circuits consisting of fan-in-2 gates that compute either addition or multiplication.

More in detail, we model $\mathcal{C}$ as a directed acyclic graph (DAG) where every node is either an *input*, an *output* or a *gate*, and input (resp. output) nodes have in-degree (resp. out-degree) 0. We partition the nodes in the DAG defined by $\mathcal{C}$ in *levels* as follows. Level 0 contains all the input nodes. Let the *depth* of a gate g be the length of the longest path from any input to g, in the DAG defined by the circuit. Then, for $h \geq 1$, we define level h as the subset of gates of depth h. Note that any gate in level h has *at least* one input coming from a gate at level $h-1$ (while other inputs may come from gates at any other previous level $0, \ldots, h-2$). The *depth* of the circuit $\mathcal{C}$, denoted $d_{\mathcal{C}}$ (or simply d when clear from the context), is the number of levels of $\mathcal{C}$. Finally, we assume that the last level $d_{\mathcal{C}}$ also contains output nodes.[15]

In this model, we define the *width* of $\mathcal{C}$, denoted by n, as the maximum number of nodes in any level $h = 0$ to $d_{\mathcal{C}}$. Note that the width upper bounds the input length. For simplicity, we assume without loss of generality circuits with maximal n inputs and n gates in every level.

When we evaluate $\mathcal{C}$ on an input $\boldsymbol{x}$, we denote the input values by $\boldsymbol{x}^{(0)}$, and the outputs of the gates in level h by the vector $\boldsymbol{x}^{(h)}$. We note that, for every $k \in [n]$, the output of the k-th gate in level h can be defined as $x_k^{(h)} = f_k^{(h)}(\boldsymbol{x}^{(0)}, \ldots, \boldsymbol{x}^{(h-1)})$ where $f_k^{(h)} : \mathcal{R}^{nh} \to \mathcal{R}$ is a quadratic polynomial. We group all these n polynomials $f_1^{(h)}, \ldots, f_n^{(h)}$ into the quadratic polynomial map $f^{(h)} : \mathcal{R}^{nh} \to \mathcal{R}^n$ such that $\boldsymbol{x}^{(h)} = f^{(h)}(\boldsymbol{x}^{(0)}, \ldots, \boldsymbol{x}^{(h-1)})$. We denote the operation that extracts these functions $\{f^{(h)}\}$ from $\mathcal{C}$ by $(f^{(1)}, \ldots, f^{(d)}) \leftarrow \mathsf{Parse}(\mathcal{C})$.

Quadratic functions. As we mentioned above, a gate in our circuit model computes a quadratic polynomial. Thus all the gates at a given level form a vector of n quadratic polynomials that take up to $m = \mathsf{poly}(\lambda)$ vectors and output a single vector. We define this class of functions as

$$\mathcal{F}_{\mathsf{quad}} = \{f : \mathcal{R}^{nm} \to \mathcal{R}^n \ : f = (f_k)_{k\in[n]} \wedge \forall k \in [n]\, f_k \in \mathcal{R}[X_1^{(1)}, \ldots, X_n^{(m)}]^{\leq 2}\}.$$

A quadratic polynomial map $f \in \mathcal{F}_{\mathsf{quad}}$, $f : \mathcal{R}^{mn} \to \mathcal{R}^n$, such as those representing the computation done at a given level of a circuit, can be expressed in a compact form. For $f(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) = \boldsymbol{y}$, we can define d matrices $\mathbf{F}^{(h)} \in \mathcal{R}^{n\times n}$, $d(d+1)/2$ matrices $\mathbf{G}^{(h,h')} \in \mathcal{R}^{n\times n^2}$, and a vector $\boldsymbol{e} \in \mathbb{F}^n$ such that

$$f(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) = \boldsymbol{e} + \sum_{h\in\mathcal{S}_1(f)} \mathbf{F}^{(h)} \cdot \boldsymbol{x}^{(h)} + \sum_{(h,h')\in\mathcal{S}_2^{\otimes}(f)} \mathbf{G}^{(h,h')} \cdot (\boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}). \tag{3}$$

[15] This can be assumed without loss of generality. If we have an output $x_i^{(h)}$ at level $h < d$, we can introduce a linear gate at level d that takes $x_i^{(h)}$ and some arbitrary $x_j^{(d-1)}$ as input, and outputs $x_k^{(d)} = x_i^{(h)} + 0 \cdot x_j^{(d-1)}$.

The sets $\mathcal{S}_1(f)$ and $\mathcal{S}_2^{\otimes}(f)$ are the *linear support* and the *quadratic support* of f that we define below; for now $\mathcal{S}_1 = [m]$, $\mathcal{S}_2^{\otimes} = \{(h, h') \in [m] \times [m] : h \leq h'\}$.[16]

We note that, in an arbitrary circuit, the function $f^{(h)}$ at each level may depend on values from *any* previous level, but not necessarily from all of them. To capture such connectivity precisely, we define the *linear support* of $f \in \mathcal{F}_{\mathsf{quad}}$, denoted $\mathcal{S}_1(f) \subseteq [m]$, as the set of indices h where the linear part of f is nonzero with respect to any term $X_i^{(h)}$. Formally,

$$\mathcal{S}_1(f) := \{h \in [m] : \mathbf{F}^{(h)} \neq \mathbf{0}\}.$$

Analogously, we define the *quadratic support* of f, denoted $\mathcal{S}_2(f) \subseteq [m]$, as the indices h where f is nonzero with respect to any term $X_i^{(h)} \cdot X_j^{(h')}$ for one or more $h' \in [m]$. Formally,

$$\mathcal{S}_2(f) := \{h \in [m] : \exists h' \; \mathbf{G}^{(h,h')} \neq \mathbf{0}\}.$$

We will also express the quadratic support using pairs of indices,

$$\mathcal{S}_2^{\otimes}(f) := \{(h, h') \in [m] \times [m] : h \leq h' \wedge \mathbf{G}^{(h,h)} \neq \mathbf{0}\}.$$

We also say $h \in \mathcal{S}_2(f)$ whenever $(t, h) \in \mathcal{S}_2^{\otimes}(f)$ or $(h, t) \in \mathcal{S}_2^{\otimes}(f)$ for some $t \in [m]$. Finally, we define the *support* of f as the union of its linear and quadratic supports, namely $\mathcal{S}(f) = \mathcal{S}_1(f) \cup \mathcal{S}_2(f)$. By using the linear and quadratic supports, we can express polynomial functions in $\mathcal{F}_{\mathsf{quad}}$ as follows:

$$f(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) = \boldsymbol{e} + \sum_{h \in \mathcal{S}_1(f)} \mathbf{F}^{(h)} \cdot \boldsymbol{x}^{(h)} + \sum_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} \mathbf{G}^{(h,h')} \cdot (\boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}). \tag{4}$$

Consider a circuit $\mathcal{C}$ and let $(f^{(1)}, \ldots, f^{(d)}) \leftarrow \mathsf{Parse}(\mathcal{C})$. Then every function $f^{(h)}$ can be expressed and computed using only the inputs in $\mathcal{S}(f^{(h)})$, namely $f^{(h)}((\boldsymbol{x}^{(h')})_{h' \in \mathcal{S}(f^{(h)})}) = f^{(h)}(\boldsymbol{x}^{(0)}, \ldots, \boldsymbol{x}^{(h-1)})$.

We call the number of inputs in the support of $f^{(h)}$, namely $|\mathcal{S}(f^{(h)})|$, the *in-degree of level* h. We say that *a circuit* $\mathcal{C}$ *has in-degree* $t_{\mathcal{C}}$ if $t_{\mathcal{C}} = \max_{h \in [d_{\mathcal{C}}]} |\mathcal{S}(f^{(h)})|$. We call $\mathcal{C}$ a *layered circuit* if it has in-degree 1. Notice that for a layered circuit it holds that $\boldsymbol{x}^{(d)} = \mathcal{C}(\boldsymbol{x}^{(0)})$ where $\boldsymbol{x}^{(h)} = f^{(h)}(\boldsymbol{x}^{(h-1)})$ for all $h = 1$ to d.

Classes of circuits. To properly define the succinctness and the functions supported by our FC construction, we parametrize the circuits according to three parameters, the depth, the in-degree, and the width. Let $\mathcal{F}_{(d,t,w)} = \{\mathcal{C} : \mathcal{R}^n \to \mathcal{R}^n : d_{\mathcal{C}} = d, t_{\mathcal{C}} = t, w_{\mathcal{C}} = w\}$, where $d_{\mathcal{C}} \in \mathbb{N}$, $t_{\mathcal{C}} \leq d$, $w_{\mathcal{C}} \leq w$ are the depth, in-degree, and width of $\mathcal{C}$, respectively. Then our FC scheme supports any arithmetic circuit of width at most n, in the model described above. We denote this class by $\mathcal{F}_n := \{\mathcal{F}_{(d,t,w)}\}_{d \in \mathbb{N}, t \leq d, w \leq n}$.

Construction. In Figure 1 we present our FC construction for $\mathcal{F}_n$. We assume, without loss of generality, that the auxiliary input aux generated by $\mathsf{CFC.Com}$ contains the committed input $\boldsymbol{x}$. In the protocol, we retrieve $\boldsymbol{x}$ from aux via a Parse function. Note that the same construction becomes a CFC for $\mathcal{F}_n$ if the verifier takes com_d as input and skips line 4 of Figure 1.

Our goal in this section is to prove the following theorem.

[16] This representation is not unique as $\boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}$ contains repeated entries, but this can be solved by agreeing on appropriately placing zero coefficients.

$\mathsf{FC.Setup}(1^\lambda, 1^n)$

return $\mathsf{CFC.Setup}(1^\lambda, 1^n)$

$\mathsf{FC.Com}(\mathsf{ck}, \boldsymbol{x})$

return $\mathsf{CFC.Com}(\mathsf{ck}, \boldsymbol{x})$

$\mathsf{FC.Open}(\mathsf{ck}, \mathsf{aux}, \mathcal{C})$

$(f^{(1)}, \dots, f^{(d)}) \leftarrow \mathsf{Parse}(\mathcal{C})$

$\boldsymbol{x}^{(0)} \leftarrow \mathsf{Parse}(\mathsf{aux})$

for $h \in [d]$:

// Evaluate and commit to each level

$\boldsymbol{x}^{(h)} \leftarrow f^{(h)}(\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(h-1)})$

$(\mathsf{com}_h, \mathsf{aux}_h) \leftarrow \mathsf{CFC.Com}(\mathsf{ck}, \boldsymbol{x}^{(h)})$

// Compute the opening for the level

$\pi_h \leftarrow \mathsf{CFC.Open}(\mathsf{ck}, (\mathsf{aux}_{h'})_{h' \in \mathcal{S}(f^{(h)})}, f^{(h)})$

return $(\pi_1, \dots, \pi_d, \mathsf{com}_1, \dots, \mathsf{com}_{d-1})$

$\mathsf{FC.Ver}(\mathsf{ck}, \mathsf{com}, \mathcal{C}, \boldsymbol{y}, \pi)$

$(f^{(1)}, \dots, f^{(d)}) \leftarrow \mathsf{Parse}(\mathcal{C})$

$\mathsf{com}_0 \leftarrow \mathsf{com}$

$(\pi_1, \dots, \pi_d, \mathsf{com}_1, \dots, \mathsf{com}_{d-1}) \leftarrow \pi$

// Recompute commitment to output

$\mathsf{com}_d \leftarrow \mathsf{CFC.Com}(\mathsf{ck}, \boldsymbol{y})$

for $h \in [d]$:

// Verify all proofs

$b_h \leftarrow \mathsf{CFC.Ver}(\mathsf{ck}, (\mathsf{com}_{h'})_{h' \in \mathcal{S}(f^{(h)})}, \mathsf{com}_h, f^{(h)}, \pi_h)$

return $b_1 \wedge \dots \wedge b_d$

Fig. 1: Construction of our FC for circuits from a CFC for the class $\mathcal{F}_{\mathsf{quad}}$.

Theorem 2. *Let* $\mathsf{CFC} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ *be a chainable functional commitment scheme for the class of functions* $\mathcal{F}_{\mathsf{quad}}$*. Then, the scheme* FC *in Figure 1 is an FC for the class* $\mathcal{F}_n$ *of arithmetic circuits* $\mathcal{C} : \mathcal{R}^n \to \mathcal{R}^n$ *of width* $\leq n$*.*

Let $\mathcal{K}$ *be a partitioning of* $\mathcal{F}_{\mathsf{quad}}$ *such that* CFC *is* $s(n, m, \kappa)$*-succinct for* $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$*. Then* FC *is* $d \cdot (s_{\max}(n, t) + 1)$*-succinct for the class* $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}_{d \in \mathbb{N}, t \leq d, w \leq n}$*, where* $s_{\max}(n, t) := \max_{\kappa \in \mathcal{K}} s(n, t, \kappa)$*. Moreover, given an additively homomorphic and/or efficiently verifiable* CFC*, so is* FC*.*

Proof. Correctness and additive homomorphism of FC follow immediately from the respective properties of CFC.

Succinctness. If CFC is $s(n, m, \kappa)$-succinct for the class of quadratic polynomials in $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$, then FC is $s'(n, (d, t))$-succinct for $\mathcal{F}_n = \{\mathcal{F}_{(d,t,n)}\}$ where $s'(n, (d, t)) = d \cdot (s_{\max}(n, t) + 1)$. Indeed, $\mathsf{FC.Open}$ produces $d-1$ commitments com_h for $h \in [d-1]$, each of them having size bounded by a fixed polynomial $p(\lambda) = \mathsf{poly}(\lambda)$. Besides, it generates d CFC evaluation proofs π_h, each of them involving $|\mathcal{S}(f^{(h)})| \leq t$ input commitments, and thus having size $\leq p(\lambda) \cdot s(n, |\mathcal{S}(f^{(h)})|, \kappa) \leq p(\lambda) \cdot s_{\max}(n, t)$. Hence, we can bound the size of an $\mathsf{FC.Open}$ proof by $|\pi| \leq p(\lambda) \cdot d \cdot (s_{\max}(n, t) + 1)$. A particularly relevant case is that for layered circuits we obtain $|\pi| \leq p(\lambda) \cdot d \cdot (s_{\max}(n, 1) + 1)$.

We obtain a better succinctness by using a slightly different, yet general, circuit model. To keep the presentation of the main scheme more understandable, we present this optimization in Section 5.1. The proof size reduction is specific to our CFC construction from pairings (see Section 6.5 for the resulting efficiency).

Efficient verification. If CFC has amortized efficient verification (Definition 4), we can set $\mathsf{FC.VerPrep}(\mathsf{ck}, f)$ to obtain $\mathsf{vk}_h \leftarrow \mathsf{CFC.VerPrep}(\mathsf{ck}, f^{(h)})$ for $h \in [d]$ and output $\mathsf{vk}_f := (\mathsf{vk}_1, \dots, \mathsf{vk}_d)$. Then, $\mathsf{FC.EffVer}$ simply recomputes the commitment to the output com_d and runs $\mathsf{CFC.EffVer}$ for each circuit level. Let T_{CFC} be largest of the running times of $\mathsf{CFC.Ver}$ for all CFC instances in the

FC construction, and let T_{Com} be the running time of $\mathsf{CFC.Com}$. Then, the running time of $\mathsf{FC.Ver}$ is $T_{\mathsf{FC}} \leq d \cdot T_{\mathsf{CFC}} + T_{\mathsf{Com}}$. As the running time of $\mathsf{CFC.EffVer}$ is $o(T_{\mathsf{CFC}})$, the running time of $\mathsf{FC.EffVer}$ is $d \cdot o(T_{\mathsf{CFC}}) + T_{\mathsf{com}}$, which is $o(T_{\mathsf{FC}})$ whenever $T_{\mathsf{Com}} = o(d \cdot T_{\mathsf{CFC}})$. Usually, $T_{\mathsf{com}} = \mathcal{O}(|\boldsymbol{y}|)$ (and in fact $T_{\mathsf{com}} = \Omega(|\boldsymbol{y}|)$) where $|\boldsymbol{y}| \leq n$ is the length of the committed vector. Hence, in practice FC has amortized efficient verification unless $d = \mathcal{O}(|\mathcal{C}|)$, a case in which the proof size also becomes very large. We remark that for both our pairing-based and lattice-based CFC instances, the running time of $\mathsf{FC.EffVer}$ is actually bounded by $p(\lambda)(|\boldsymbol{y}| + |\pi|)$ where $p(\lambda) = \mathsf{poly}(\lambda)$, which is optimal since the verifier at least needs to parse the proof and the output.

Security. In Lemma 1, we prove that if CFC is evaluation binding, then FC is evaluation binding. In Appendix A, we show an analogous result for knowledge extractability (and therefore also for strong evaluation binding by Proposition 4).

Lemma 1. *If* CFC *is evaluation binding (Definition 8), then our FC construction for arbitrary circuits is also evaluation binding.*

Proof. Consider an adversary $\mathcal{A}$ who returns a tuple $(\mathsf{com}, \mathcal{C}, \boldsymbol{y}, \pi, \boldsymbol{y}', \pi')$ that breaks evaluation binding, and parse the proofs as follows

$$\pi := (\pi_1, \ldots, \pi_d, \mathsf{com}_1, \ldots, \mathsf{com}_{d-1})$$

$$\pi' := (\pi'_1, \ldots, \pi'_d, \mathsf{com}'_1, \ldots, \mathsf{com}'_{d-1})$$

We will show that, if both proofs π and π' verify for $\boldsymbol{y}$ and $\boldsymbol{y}'$ respectively, with $\boldsymbol{y} \neq \boldsymbol{y}'$, then we can construct an adversary $\mathcal{B}$ against the evaluation binding of the CFC. We construct $\mathcal{B}$ as follows.

First, $\mathcal{B}$ is given a commitment key ck and calls $\mathcal{A}(\mathsf{ck})$ to obtain the output $(\mathsf{com}, \mathcal{C}, \boldsymbol{y}, \pi, \boldsymbol{y}', \pi')$. Then, $\mathcal{B}$ obtains the commitments to the outputs $\mathsf{com}_y \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{y})$ and $\mathsf{com}_{y'} \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{y}')$.

If $\mathsf{com}_y = \mathsf{com}_{y'}$, then $\mathcal{B}$ can break the binding property of the commitment (and hence evaluation binding due to Proposition 1), since com_y opens to different $\boldsymbol{y} \neq \boldsymbol{y}'$.

Hence, let us assume $\mathsf{com}_y \neq \mathsf{com}'_y$, and denote $\mathsf{com}_0 = \mathsf{com}'_0 = \mathsf{com}$. Then, look at both proofs produced by $\mathcal{A}$ and set $1 \leq h^* \leq d$ to be the smallest index such that $\mathsf{com}_{h^*} \neq \mathsf{com}'_{h^*}$ and $\mathsf{com}_h = \mathsf{com}'_h$ for all $h = 0$ to $h^* - 1$. Notice that such index *must exist* since, at least, we have $\mathsf{com}_0 = \mathsf{com}'_0$ and $\mathsf{com}_d = \mathsf{com}_y \neq \mathsf{com}_{y'} = \mathsf{com}'_d$.

Then, $\mathcal{B}$ breaks evaluation binding of CFC by outputting $((\mathsf{com}_h)_{h \in \mathcal{S}(f^{(h^*)})}, f^{(h^*)}, \mathsf{com}_{h^*}, \pi_{h^*}, \mathsf{com}'_{h^*}, \pi'_{h^*})$.

□

5.1 Efficiency Tradeoffs

In this section we describe optimization strategies for our FC construction. Our main goals are to reduce the proof size in many cases, and to support circuits of larger width than initially specified at setup time.

A refined circuit model. We introduce a variant of our circuit model that results in a notable reduction of the proof size of our pairing-based CFC in Section 6. The new circuit model differs from the previous model in that here every quadratic monomial of each polynomial gate $f_k^{(h)}$ at level h is assumed to take at least one of its inputs from level $h-1$. In particular, the quadratic term of functions $f_k^{(h)}(\boldsymbol{x}^{(0)}, \ldots, \boldsymbol{x}^{(h-1)})$ is a linear combination of all products of variables $x_i^{(h-1)} \cdot x_j^{(h')}$, $\forall i, j \in [n]$, at levels $h-1$ and h' such that $0 \leq h' \leq h-1$.

This circuit model also generalizes the standard arithmetic circuit model with fan-in 2 additive or multiplicative gates. We denote the class of functions in the levels of the new model by $\mathcal{F}_{\mathsf{level}} \subset \mathcal{F}_{\mathsf{quad}}$, which we define as

$$\mathcal{F}_{\mathsf{level}} = \{f \in \mathcal{F}_{\mathsf{quad}} : \mathcal{S}_2^{\otimes}(f) \subseteq \{(h', m) \in [m] \times \{m\}\}\}.$$

Note that we can extend any parametrization $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$ to $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$ by setting $\mathcal{F}_{\mathsf{level},\kappa} := \mathcal{F}_{\mathsf{level}} \cap \mathcal{F}_{\mathsf{quad},\kappa}$. The main advantage of this model is that for any $f \in \mathcal{F}_{\mathsf{level}}$, $\left|\mathcal{S}_2^{\otimes}(f)\right| \leq m$, instead of being $\leq m^2$ in the more general case. When switching to this model, it is sufficient to instantiate our FC construction with a CFC scheme that only supports quadratic functions in $\mathcal{F}_{\mathsf{level}}$ and not all $\mathcal{F}_{\mathsf{quad}}$.

Reducing proof size. Assume that we want to evaluate a circuit $\mathcal{C}$ of width w and depth d that is densely interconnected (i.e. the in-degree $t = \mathcal{O}(d)$) when our commitment key ck supports circuits of width up to $n > w$. We present an optimization that reduces the proof size of our FC scheme.

Proposition 2. *Let* CFC *be a* $s(n, m, \kappa)$*-succinct CFC for* $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$ *(resp. for* $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$*), and let* $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}$ *be the class of circuits parametrized by depth* d*, in-degree* t*, and width* $w \leq n$*. Then, we can construct a* $s'(n, (d, t, w))$*-succinct FC scheme* FC *where* $s'(n, (d, t, w)) = d \cdot (s_{\max}(n, \lceil dw/n \rceil) + 1)$.

In particular, for circuits of bounded size $|\mathcal{C}| = d \cdot w \leq n$*, the proof size is the same as for layered circuits, namely* $s'(n, (d, t, w)) = d \cdot (s_{\max}(n, 1) + 1)$.

Proof. The construction of the optimized FC scheme consists in reshaping the original input circuit $\mathcal{C}$ into an equivalent semi-layered (i.e., $t \ll d$) circuit $\mathcal{C}'$ of depth d and width bounded by n. The FC scheme is then identical to the scheme in Figure 1. In fact, as FC needs to support circuits of any width $w \leq n$, $\mathsf{FC.Setup}(1^\lambda, n)$ outputs $\mathsf{ck} \leftarrow \mathsf{CFC.Setup}(1^\lambda, 1^n)$.

Let $r = \lfloor n/w \rfloor$. For each level h of $\mathcal{C}$ with values $\boldsymbol{x}^{(h)}$, we construct level h in circuit $\mathcal{C}'$ with values $\boldsymbol{z}^{(h)}$ as described below.

- Let $\boldsymbol{z}^{(0)} := \boldsymbol{x}$. For $h = 1, \ldots, r-1$, set $\boldsymbol{z}^{(h)} := \boldsymbol{x}^{(0)}||\boldsymbol{x}^{(1)}|| \cdots ||\boldsymbol{x}^{(h)}$ as the concatenation of variables from previous levels. Then, define the wiring in $\mathcal{C}'$ by introducing relay gates between levels, such that $\boldsymbol{x}^{(0)}$ is copied to levels $h = 1, \ldots, r-1$, $\boldsymbol{x}^{(1)}$ is copied to levels $h = 2, \ldots, r-1$, etc. Note that, up to level r, $\mathcal{C}'$ is the equivalent of $\mathcal{C}$ as a layered circuit.
- At level r, set $\boldsymbol{z}^{(r)} := \boldsymbol{x}^{(r)}$. Note that $\boldsymbol{z}^{(r)}$ only depends on inputs at level $r-1$ in $\mathcal{C}'$, since all $\boldsymbol{x}^{(0)}, \ldots, \boldsymbol{x}^{(r-1)}$ are duplicated at level $\boldsymbol{z}^{(r-1)}$.
- For levels $h = r+1, \ldots, 2r-1$, expand again as $\boldsymbol{z}^{(h)} := \boldsymbol{x}^{(r)}||\boldsymbol{x}^{(r+1)}|| \cdots ||\boldsymbol{x}^{(h)}$. Note that values at level h depend only on levels $r-1$ and $h-1$, as $\boldsymbol{z}^{(r-1)}$ contains all values from levels 0 to $r-1$ in $\mathcal{C}$.
- Repeat the steps above, bootstrapping the circuit at levels $2r, 3r, \ldots, d$.

The functions $f^{(1)}, \ldots, f^{(d)}$ that describe the levels of $\mathcal{C}'$ are such that level h has in-degree $|\mathcal{S}(f^{(h)})| = \lceil h/r \rceil$. Hence, if the CFC is $s(n, m, \kappa)$-succinct for $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$ (resp. for $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$) then the proof size of the FC scheme for $\mathcal{C}'$ becomes

$$|\pi| = \sum_{h=0}^{d-1} s(n, \lceil h/r \rceil, \kappa) + 1 \leq d \cdot (s_{\max}(n, \lceil d/r \rceil) + 1).$$

Note that the parameters can be tuned in a per-level basis, allowing for more succinct proofs in practice or when the initial in-degree is low.

Supporting circuits of arbitrary width. Suppose that the parameters of the FC scheme are set up for circuits of bounded width n, and that we want to evaluate a circuit $\mathcal{C}$ of width $w > n$. The following result shows that this is possible at the cost of increasing the proof size.

Proposition 3. *Let* CFC *be a* $s(n, m, \kappa)$*-succinct for* $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$ *(resp. for* $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$*). Let* FC *be our construction in Figure 1 for the class of circuits* $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}$ *of bounded width* $w \leq n$*. Then, we can construct an FC scheme* $\tilde{\mathsf{FC}}$ *for* $\mathcal{F} = \{\mathcal{F}_{(d,t,w)}\}$ *for any* $w \in \mathbb{N}$ *such that* $\tilde{\mathsf{FC}}.\mathsf{Setup}(1^\lambda) = \mathsf{FC}.\mathsf{Setup}(1^\lambda, n)$ *where the proof size is* $|\pi| \leq d \cdot \lceil w/n \rceil \cdot (s_{\max}(n, t \cdot \lceil w/n \rceil) + 1)$*.*

Proof. We describe $\tilde{\mathsf{FC}}$ in two steps. First, we introduce a circuit transformation from the original $\mathcal{C}$ to an equivalent $\mathcal{C}'$ of width n and larger depth. Then, we describe the $\tilde{\mathsf{FC}}.\mathsf{Com}$, $\tilde{\mathsf{FC}}.\mathsf{Open}$ and $\tilde{\mathsf{FC}}.\mathsf{Ver}$ algorithms. We can construct $\mathcal{C}'$ as follows:

- Let $r = \lceil w/n \rceil$. For each level $\boldsymbol{x}^{(h)}$, $h = 0, \ldots, d$ of $\mathcal{C}$, define sub-levels $\boldsymbol{z}^{(h,s)}$ with indices $(h, 1), \ldots, (h, r)$ in $\mathcal{C}'$ as the natural split of $\boldsymbol{x}^{(h)}$ in r blocks, i.e., $\boldsymbol{z}^{(h,s)} = (x^{(h)}_{(s-1)n}, x^{(h)}_{(s-1)n+1}, \ldots, x^{(h)}_{sn-1})$ for $s \in [r]$.
- For each level function $f^{(h)} : \mathcal{R}^{mw} \to \mathcal{R}^{w}$ corresponding to $\mathcal{C}$, let $m' = m \cdot r$ and define r functions $g^{(h,s)} : \mathcal{R}^{m'n} \to \mathcal{R}^{n}$ for $s \in [r]$ such that $g^{(h,s)}(\boldsymbol{z}^{(0,1)}, \ldots, \boldsymbol{z}^{(h-1,r)}) = \boldsymbol{z}^{(h,s)}$. Note that these functions can be built from a restriction of $f^{(h)}$ to a subset of its outputs.

The commit algorithm $\tilde{\mathsf{FC}}.\mathsf{Com}(\mathsf{ck}, \boldsymbol{x})$ partitions the input $\boldsymbol{x} \in \mathcal{R}^w$ in r blocks $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(r)}$ of size n as described above, obtains $(\mathsf{com}_{(s)}, \mathsf{aux}_{(s)}) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}^{(s)})$. It outputs $\tilde{\mathsf{com}} = (\mathsf{com}_{(1)}, \ldots, \mathsf{com}_{(r)})$ and $\tilde{\mathsf{aux}} = (\mathsf{aux}_{(1)}, \ldots, \mathsf{aux}_{(r)})$.

The opening algorithm $\tilde{\mathsf{FC}}.\mathsf{Open}(\mathsf{ck}, \tilde{\mathsf{aux}}, \mathcal{C})$ works as follows:

- Obtain $\mathcal{C}'$ from $\mathcal{C}$ as presented above, parse $(\boldsymbol{z}^{(0,1)}, \ldots, \boldsymbol{z}^{(0,r)}) \leftarrow \mathsf{Parse}(\tilde{\mathsf{aux}})$, and compute $\mathcal{C}'(\boldsymbol{z}^{(0,1)}, \ldots, \boldsymbol{z}^{(0,r)})$ and all the intermediate values $\boldsymbol{z}^{(h,s)}$ for $h \in [d]$ and $s \in [r]$.
- Commit to each $\boldsymbol{z}^{(h,s)}$ as $(\mathsf{com}_{(h,s)}, \mathsf{aux}_{(h,s)}) \leftarrow \mathsf{CFC}.\mathsf{Com}(\mathsf{ck}, \boldsymbol{z}^{(h,s)})$ for $h \in [d-1]$ and $s \in [r]$.
- Compute the opening proofs for all functions,

$$\forall\, h \in [d], s \in [r] :\ \pi_{(h,s)} \leftarrow \mathsf{CFC}.\mathsf{Open}(\mathsf{ck}, (\mathsf{aux}_{(h',s')})_{h' \in \mathcal{S}(f^{(h)}), s' \in [r]}, g^{(h,s)}).$$

- Return $\tilde{\pi} = (\pi_{(h,s)}, \mathsf{com}_{(h,s)})_{h \in [d], s \in [r]}$.

The verification algorithm $\tilde{\mathsf{FC}}.\mathsf{Ver}(\mathsf{ck}, \tilde{\mathsf{com}}, f, \boldsymbol{y}, \tilde{\pi})$ first computes r commitments to the output $\boldsymbol{z}^{(d,s)} \leftarrow \mathsf{Com}(\boldsymbol{y}^{(s)})$ for $s \in [r]$ and then verifies all opening proofs.

Overall, if the CFC is $s(n, m, \kappa)$-succinct for $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$ (resp. $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}$), and the original circuit $\mathcal{C} \in \mathcal{F}_{(d,t,w)}$ (i.e., the in-degree of $\mathcal{C}$ is bounded by t), then the proof size of the FC scheme for $\mathcal{C}'$ becomes

$$|\pi| = (d-1)r + r \cdot \sum_{h=0}^{d-1} s(n, hr, \kappa) \leq dr \cdot (s_{\max}(n, tr) + 1).$$

6 Paring-based CFC for Quadratic Functions

We present our construction of a chainable functional commitment for quadratic functions based on pairings. With our CFC, one can commit to a set of vectors $\boldsymbol{x}_1, \ldots \boldsymbol{x}_m$ of length n and then open the commitment to a quadratic function $f : \mathbb{F}^{mn} \to \mathbb{F}^n$, for any $m = \mathsf{poly}(\lambda)$. The opening proofs of our scheme are quadratic in the number m of input vectors, but constant in the (possibly padded) length n of each input vector and of the output. Security is proven in the standard model based on a new falsifiable assumption that we justify in the generic bilinear group model. In Section 6.5 we discuss the FCs for circuits that we obtain by applying the generic transform of Section 5 to this pairing-based CFC.

We present our CFC with deterministic commitments and openings. We detail how to make our commitments perfectly com-hiding in Section 6.8. We note that the FCs for circuits obtained from the com-hiding CFC are also com-hiding, and their openings can be made zero-knowledge by applying Theorem 1, which we can efficiently instantiate using, e.g., the Groth-Sahai [GS08] NIZK.

6.1 Preliminaries on Bilinear Groups and Assumption

A *bilinear group generator* $\mathcal{BG}(1^\lambda)$ is an algorithm that returns $\mathsf{bgp} := (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$, where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are groups of prime order q, $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ are fixed generators, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable, non-degenerate, bilinear map. In our work we use Type-3 groups in which it is assumed that there is no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. We use the bracket notation of [EHK+13] for group elements: for $s \in \{1, 2, T\}$ and $x \in \mathbb{Z}_q$, $[x]_s$ denotes $g_s^x \in \mathbb{G}_s$. We use additive notation for $\mathbb{G}_1$ and $\mathbb{G}_2$ and multiplicative notation for $\mathbb{G}_T$. We note that given an element $[x]_s \in \mathbb{G}_s$, for $s = 1, 2$, and a scalar a, one can efficiently compute $a \cdot [x] = [ax] = g_s^{ax} \in \mathbb{G}_s$; given group elements $[a]_1 \in \mathbb{G}_1$ and $[b]_2 \in \mathbb{G}_2$, one can efficiently compute $[ab]_T = e([a]_1, [b]_2)$.

We prove that our construction satisfies evaluation binding under a new falsifiable assumption, called HintedKernel (HiKer), that we justify in the generic group model (see Appendix B). The name of the assumption comes from its similarity with the KerMDH assumption of [MRV16] which for matrices $[\mathbf{A}]_2$ from certain (random) distributions asks the adversary to find a nonzero vector $[\boldsymbol{z}]_1$ such that $\mathbf{A}\boldsymbol{z} = \mathbf{0}$. In our case the adversary is challenged to find a nonzero $[u, v]_1$ such that $u\eta + v = 0$, when given $[1, \eta]_2$ but also other group elements, the "hints", that depend on η and other random variables.

Definition 9 (n-HiKer Assumption). *Let $\mathsf{bgp} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ be a bilinear group setting, let $n \in \mathbb{N}$ and let $\mathcal{G}_1, \mathcal{G}_2$ be the following two sets of Laurent monomials in $\mathbb{Z}_q[S_1, T_1, \ldots, S_n, T_n, H]$:*

$$\mathcal{G}_1(\boldsymbol{S}, \boldsymbol{T}, H) := \{S_i,\, T_i\}_{i\in[n]} \cup \{S_i \cdot T_j\}_{i,j\in[n]} \cup \left\{\frac{S_{i'}}{S_i} \cdot T_i \cdot H\right\}_{\substack{i,i'\in[n]\\ i\neq i'}} \cup \left\{\frac{S_{i'} \cdot T_{j'}}{S_i \cdot T_j} \cdot H\right\}_{\substack{i,j,i',j'\in[n]\\ (i,j)\neq(i',j')}}$$

$$\mathcal{G}_2(\boldsymbol{S}, \boldsymbol{T}, H) := \{H\} \cup \{S_i\}_{i\in[n]} \cup \left\{\frac{1}{S_i} \cdot T_i \cdot H,\, \frac{1}{S_i} \cdot H\right\}_{i\in[n]} \cup \left\{\frac{1}{S_i} \cdot \frac{1}{T_j} \cdot H\right\}_{i,j\in[n]}$$

The n-HintedKernel (n-HiKer) assumption holds if for every $n = \mathsf{poly}(\lambda)$ and any PPT $\mathcal{A}$, the following advantage is negligible

$$\mathbf{Adv}_{\mathcal{A}}^{n\text{-}HiKer}(\lambda) = \Pr\left[\begin{array}{c}(U, V) \neq (1, 1)_{\mathbb{G}_1} \,\wedge\\ e(U, [\eta]_2) = e(V, [1]_2)\end{array} \middle|\, (U, V) \leftarrow \mathcal{A}\left(\mathsf{bgp}, \begin{array}{l}[\mathcal{G}_1(\boldsymbol{\sigma}, \boldsymbol{\tau}, \eta)]_1,\\ [\mathcal{G}_2(\boldsymbol{\sigma}, \boldsymbol{\tau}, \eta)]_2\end{array}\right)\right]$$

where the probability is over the random choices of $\boldsymbol{\sigma}, \boldsymbol{\tau}, \eta$ and $\mathcal{A}$'s random coins.

6.2 Our CFC Construction

As defined in the previous section we express $f \in \mathcal{F}_{\mathsf{quad}}$ through a set of matrices $\mathbf{F}^{(h)} \in \mathbb{F}^{n \times n}$ and $\mathbf{G}^{(h,h')} \in \mathbb{F}^{n \times n^2}$, and a vector $\boldsymbol{e} \in \mathbb{F}^n$ such that

$$f(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) = \boldsymbol{e} + \sum_{h \in \mathcal{S}_1(f)} \mathbf{F}^{(h)} \cdot \boldsymbol{x}^{(h)} + \sum_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} \mathbf{G}^{(h,h')} \cdot (\boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}) \tag{5}$$

For the sake of defining the succinctness of our CFC we parametrize the class $\mathcal{F}_{\mathsf{quad}}$ by the size of the quadratic support of f. Formally, let $\mathcal{K} = \{0, 1, \ldots, m(m+1)/2\}$. Then we partition $\mathcal{F}_{\mathsf{quad}}$ as $\{\mathcal{F}_{\mathsf{quad},\kappa}\}_{\kappa \in \mathcal{K}}$ where each $\mathcal{F}_{\mathsf{quad},\kappa} = \{f \in \mathcal{F}_{\mathsf{quad}} : \mathcal{S}_2^{\otimes}(f) = \kappa\}$. Note that the parametrization extends naturally to the class $\mathcal{F}_{\mathsf{level}}$ as described in Section 5. Due to the definition of $\mathcal{F}_{\mathsf{level}}$, in that case we have at most m partitions, i.e,. $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}_{\kappa=0}^{m}$.

$\underline{\mathsf{Setup}(1^\lambda, 1^n)}$ Let $n \geq 1$ be an integer representing the width of each of the inputs of the functions to be computed at opening time. Generate a bilinear group description $\mathsf{bgp} := (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{BG}(1^\lambda)$, and let $\mathbb{F} := \mathbb{Z}_q$.
Next, sample random $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma} \leftarrow\$ \mathbb{F}^n$, $\eta_\alpha, \eta_\beta, \eta_\gamma \leftarrow\$ \mathbb{F}$, and output

$$\mathsf{ck} := \begin{pmatrix} [\boldsymbol{\alpha}]_1 , [\boldsymbol{\alpha}]_2 , [\boldsymbol{\beta}]_1 , [\boldsymbol{\gamma}]_1 , [\boldsymbol{\alpha} \otimes \boldsymbol{\beta}]_1 , [\eta_\alpha]_2 , [\eta_\beta]_2 , [\eta_\gamma]_2 \\ \left\{ \left[\alpha_i \frac{\gamma_{i'}}{\gamma_i} \eta_\alpha\right]_1 , \left[\frac{\alpha_{i'}}{\alpha_i} \beta_i \eta_\beta\right]_1 \right\}_{\substack{i,i' \in [n] \\ i \neq i'}} \left\{ \left[\frac{\alpha_{i'}\beta_{j'}}{\alpha_i \beta_j} \gamma_k \eta_\gamma\right]_1 \right\}_{\substack{i,j,i',j',k \in [n] \\ (i,j) \neq (i',j')}} \\ \left\{ \left[\frac{\alpha_i \eta_\alpha}{\gamma_i}\right]_2 , \left[\frac{\beta_i \eta_\beta}{\alpha_i}\right]_2 \right\}_{i \in [n]} , \left\{ \left[\frac{\gamma_k \eta_\gamma}{\alpha_i}\right]_2 \right\}_{i,k \in [n]} \left\{ , \left[\frac{\gamma_k \eta_\gamma}{\alpha_i \beta_j}\right]_2 \right\}_{i,j,k \in [n]} \end{pmatrix}.$$

$\underline{\mathsf{Com}(\mathsf{ck}, \boldsymbol{x})}$ output $\mathsf{com} := [\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle]_1$ and $\mathsf{aux} = \boldsymbol{x}$.

$\underline{\mathsf{Open}(\mathsf{ck}, (\mathsf{aux}_i)_{i \in [m]}, f) \to \pi}$ Let $\mathbf{F}^{(h)} \in \mathbb{F}^{n \times n}$ for $h \in \mathcal{S}_1(f)$, $\mathbf{G}^{(h,h')} \in \mathbb{F}^{n \times n^2}$ for $(h,h') \in \mathcal{S}_2^{\otimes}(f)$, and $\boldsymbol{e} \in \mathbb{F}^n$ be the matrices and vectors associated to $f : \mathbb{F}^{mn} \to \mathbb{F}^n$. The opening algorithm computes the output $\boldsymbol{y} = f(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)})$ and proceeds as follows.

- For every $h \in \mathcal{S}_2(f)$: compute $X_h^{(2)} := [\langle \boldsymbol{x}^{(h)}, \boldsymbol{\alpha} \rangle]_2$, $X_h^{(\beta)} := [\langle \boldsymbol{x}^{(h)}, \boldsymbol{\beta} \rangle]_1$, which are commitments to $\boldsymbol{x}^{(h)}$ under $\boldsymbol{\alpha}$ in $\mathbb{G}_2$ and under $\boldsymbol{\beta}$ in $\mathbb{G}_1$, resp.
- For every $h \in \mathcal{S}_2(f)$: compute a linear map opening proof for the identity function, to show that X_h and $X_h^{(\beta)}$ open to the same value:

$$\pi_h^{(\beta)} := \sum_{\substack{i,i' \in [n] \\ i \neq i'}} x_{i'}^{(h)} \cdot \left[\frac{\alpha_{i'}}{\alpha_i} \beta_i \eta_\beta\right]_1$$

- For every pair of inputs $\boldsymbol{x}^{(h)}, \boldsymbol{x}^{(h')}$ such that $(h,h') \in \mathcal{S}_2^{\otimes}(f)$, compute a commitment to their tensor products as follows:

$$Z_{h,h'} := \sum_{i,j \in [n]} x_i^{(h)} x_j^{(h')} \cdot [\alpha_i \beta_j]_1 = [\langle \boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}, \boldsymbol{\alpha} \otimes \boldsymbol{\beta} \rangle]_1.$$

– Compute a linear map opening proof to show that the vector $\boldsymbol{y}$ satisfies equation (5), with respect to all the inputs $\boldsymbol{x}^{(h)}$ committed in X_h and the inputs $\boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}$ committed in $Z_{h,h'}$:

$$\pi^{(\gamma)} := \sum_{h \in \mathcal{S}_1(f)} \sum_{\substack{i,i',k \in [n] \\ i \neq i'}} F_{k,i}^{(h)} \cdot x_{i'}^{(h)} \cdot \left[\frac{\alpha_{i'}}{\alpha_i} \gamma_k \eta_\gamma \right]_1 + \sum_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} \sum_{\substack{i,j,i',j',k \in [n] \\ (i,j) \neq (i',j')}} G_{k,(i,j)}^{(h,h')} \cdot x_{i'}^{(h)} x_{j'}^{(h')} \cdot \left[\frac{\alpha_{i'} \beta_{j'}}{\alpha_i \beta_j} \gamma_k \eta_\gamma \right]_1$$

Note that $\pi^{(\gamma)}$ is in fact a proof for the vector $\boldsymbol{e} - \boldsymbol{t}$; the linear shift will be addressed by the verifier in equation (11).

– Commit to the output $\boldsymbol{y}$ under $\boldsymbol{\gamma}$ by computing $Y^{(\gamma)} := [\langle \boldsymbol{y}, \boldsymbol{\gamma} \rangle]_1$. Then, compute a linear map opening proof for the identity function, to show that $Y^{(\gamma)}$ and the commitment to the output $\mathsf{com}_y \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{y})$ (which is under $\boldsymbol{\alpha}$) open to the same value:

$$\pi^{(\alpha)} := \sum_{\substack{i,i' \in [n] \\ i \neq i'}} y_{i'} \cdot \left[\alpha_i \frac{\gamma_{i'}}{\gamma_i} \eta_\alpha \right]_1$$

– Return $\pi := \left(\{X_h^{(2)}, X_h^{(\beta)}, \pi_h^{(\beta)}\}_{h \in \mathcal{S}_2(f)}, \{Z_{h,h'}\}_{(h,h') \in \mathcal{S}_2^{\otimes}(f)}, Y^{(\gamma)}, \pi^{(\alpha)}, \pi^{(\gamma)} \right)$.

$\underline{\mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i \in [m]}, \mathsf{com}_y, f, \pi) \to b \in \{0,1\}}$ Parse the proof π as above and set $X_h := \mathsf{com}_h$. Output 1 if all the following checks pass and 0 otherwise:

– Verify the consistency of all the commitments. Namely, verify that each X_h and $X_h^{(2)}$ are commitments to the same value in $\mathbb{G}_1$ and $\mathbb{G}_2$:

$$\forall h \in \mathcal{S}_2(f) : e\left(X_h, [1]_2\right) \stackrel{?}{=} e\left([1]_1, X_h^{(2)}\right) \tag{6}$$

– Verify the linear map commitment proofs $\pi_h^{(\beta)}$ that both $X_h^{(\beta)}, X_h$ commit to the same value in different sets of parameters:

$$\forall h \in \mathcal{S}_2(f) : e\left(X_h, \sum_{i \in [n]} \left[\frac{\beta_i \eta_\beta}{\alpha_i} \right]_2 \right) \stackrel{?}{=} e\left(\pi_h^{(\beta)}, [1]_2\right) e\left(X_h^{(\beta)}, [\eta_\beta]_2\right) \tag{7}$$

– Verify the consistency of the commitments to the tensor products, i.e., verify that $Z_{h,h'}$ is a commitment to $\boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}$:

$$\forall (h,h') \in \mathcal{S}_2^{\otimes}(f) : e\left(Z_{h,h'}, [1]_2\right) \stackrel{?}{=} e\left(X_{h'}^{(\beta)}, X_h^{(2)}\right) \tag{8}$$

– Verify the linear map commitment proof $\pi^{(\alpha)}$ that both $\mathsf{com}_y, Y^{(\gamma)}$ commit to the same value in different sets of parameters:

$$e\left(Y^{(\gamma)}, \sum_{i \in [n]} \left[\frac{\alpha_i \eta_\alpha}{\gamma_i} \right]_2 \right) \stackrel{?}{=} e\left(\pi^{(\alpha)}, [1]_2\right) e\left(\mathsf{com}_y, [\eta_\alpha]_2\right) \tag{9}$$

– Verify the linear map commitment proof to check that, intuitively, $Y^{(\gamma)}$ is a commitment under $\boldsymbol{\gamma}$ to the output of f, computed from the inputs committed in X_h and $Z_{h,h'}$. To this end, compute the encoding of the matrices $\mathbf{F}^{(h)}$ for $h \in \mathcal{S}_1(f)$, $\mathbf{G}^{(h,h')}$ for $(h,h') \in \mathcal{S}_2^{\otimes}(f)$ and the vector $\boldsymbol{e}$ as follows. Let $\Theta = [\langle \boldsymbol{e}, \boldsymbol{\gamma} \rangle]_1$ and

$$\Phi_h := \sum_{i,k \in [n]} F_{k,i}^{(h)} \cdot \left[\frac{\gamma_k \eta_\gamma}{\alpha_i}\right]_2, \quad \Gamma_{h,h'} := \sum_{i,j,k \in [n]} G_{k,(i,j)}^{(h,h')} \cdot \left[\frac{\gamma_k \eta_\gamma}{\alpha_i \beta_j}\right]_2 \tag{10}$$

and then verify that

$$\prod_{h \in \mathcal{S}_1(f)} e\left(X_h, \Phi_h\right) \cdot \prod_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} e\left(Z_{h,h'}, \Gamma_{h,h'}\right) \stackrel{?}{=} e\left(\pi^{(\gamma)}, [1]_2\right) e\left(Y^{(\gamma)} \cdot \Theta^{-1}, [\eta_\gamma]_2\right). \tag{11}$$

Theorem 3. *Assume that the n-HiKer assumption holds for a bilinear group setting generated by $\mathcal{BG}$. Then the construction* CFC *described above is an evaluation binding CFC scheme for the class $\mathcal{F}_{\mathsf{quad}}$ of quadratic functions over any $m = \mathsf{poly}(\lambda)$ vectors of length $\leq n$, that has efficient verification and is additively homomorphic. Considering the partitioning of $\mathcal{F}_{\mathsf{quad}} = \{\mathcal{F}_{\mathsf{quad},\kappa}\}_{\kappa=0}^{m(m+1)/2}$,* CFC *is $s(n,m,\kappa)$-succinct for $s(n,m,\kappa) = (\kappa + 3m + 3)$. Furthermore, when executed on the class of functions $\mathcal{F}_{\mathsf{level}} \subset \mathcal{F}_{\mathsf{quad}}$ introduced in Section 5.1 and partitioned as $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}_{\kappa=0}^{m}$, then* CFC *is $(4\kappa + 3)$-succinct.*

In the following sections we prove the theorem.

6.3 Correctness

To prove correctness, consider honestly generated input commitments $X_h = \left[\langle \boldsymbol{x}^{(h)}, \boldsymbol{\alpha} \rangle\right]_1$ for $h \in [m]$ and an honestly generated opening

$$\pi := \left(\{X_h^{(2)}, X_h^{(\beta)}, \pi_h^{(\beta)}\}_{h \in \mathcal{S}_2(f)}, \{Z_{h,h'}\}_{(h,h') \in \mathcal{S}_2^{\otimes}(\boldsymbol{f})}, Y^{(\gamma)}, \pi^{(\alpha)}, \pi^{(\gamma)}\right)$$

for a quadratic function f represented by the matrices $\boldsymbol{e}, \mathbf{F}^{(h)}, \mathbf{G}^{(h,h')}$ for $h \in \mathcal{S}_1(f)$ and $(h,h') \in \mathcal{S}_2^{\otimes}(f)$.

The correctness of equations (6) and (8) follows easily by construction since

$$e\left(X_h, [1]_2\right) = e\left(\left[\langle \boldsymbol{x}^{(h)}, \boldsymbol{\alpha} \rangle\right]_1, [1]_2\right) = e\left([1]_1, \left[\langle \boldsymbol{x}^{(h)}, \boldsymbol{\alpha} \rangle\right]_2\right) = e\left([1]_1, X_h^{(2)}\right),$$

$$\begin{aligned} e\left(Z_{h,h'}, [1]_2\right) &= e\left(\left[\langle \boldsymbol{x}^{(h)} \otimes \boldsymbol{x}^{(h')}, \boldsymbol{\alpha} \otimes \boldsymbol{\beta} \rangle\right]_1, [1]_2\right) = e\left(\left[\sum_{i,j \in [n]} x_i^{(h)} x_j^{(h')} \alpha_i \beta_j\right]_1, [1]_2\right) \\ &= e\left(\left[\langle \boldsymbol{x}^{(h')}, \boldsymbol{\beta} \rangle\right]_1, \left[\langle \boldsymbol{x}^{(h)}, \boldsymbol{\alpha} \rangle\right]_2\right) = e\left(X_{h'}^{(\beta)}, X_h^{(2)}\right). \end{aligned}$$

The correctness of equation (7) can be seen as follows. Given $h \in \mathcal{S}_2(f)$, we have that

$$\begin{aligned} e\left(X_h, \sum_{i \in [n]} \left[\frac{\beta_i \eta_\beta}{\alpha_i}\right]_2\right) &= \left[\left(\sum_{i \in [n]} x_i^{(h)} \alpha_i\right) \cdot \left(\sum_{i \in [n]} \frac{\beta_i \eta_\beta}{\alpha_i}\right)\right]_T = \left[\sum_{i,i' \in [n]} x_{i'}^{(h)} \frac{\alpha_{i'}}{\alpha_i} \beta_i \eta_\beta\right]_T \\ &= \left[\sum_{\substack{i,i' \in [n] \\ i \neq i'}} x_{i'}^{(h)} \frac{\alpha_{i'}}{\alpha_i} \beta_i \eta_\beta + \sum_{i \in [n]} x_i^{(h)} \beta_i \eta_\beta\right]_T = e\left(\pi_h^{(\beta)}, [1]_2\right) e\left(X_h^{(\beta)}, [\eta_\beta]_2\right). \end{aligned}$$

Similarly, for equation (9) we have that

$$e\left(Y^{(\gamma)}, \sum_{i\in[n]}\left[\frac{\alpha_i\eta_\alpha}{\gamma_i}\right]_2\right) = \left[\left(\sum_{i\in[n]} y_i\gamma_i\right)\cdot\left(\sum_{i\in[n]}\frac{\alpha_i\eta_\alpha}{\gamma_i}\right)\right]_T = \left[\sum_{i,i'\in[n]} y_{i'}\frac{\gamma_{i'}}{\gamma_i}\alpha_i\eta_\alpha\right]_T$$

$$= \left[\sum_{\substack{i,i'\in[n]\\ i\neq i'}} y_{i'}\alpha_i\frac{\gamma_{i'}}{\gamma_i}\eta_\alpha + \sum_{i\in[n]} y_i\alpha_i\eta_\alpha\right]_T = e\left(\pi^{(\alpha)}, [1]_2\right) e\left(\mathsf{com}_y, [\eta_\alpha]_2\right).$$

Finally, the correctness of equation (11) can be proven in an analogous way. First of all, we expand the pairing coefficients on the LHS in $\mathbb{G}_T$,

$$e\left(X_h, \Phi_h\right) = \left[\sum_{k\in[n]}\left(\sum_{i\in[n]} F_{k,i}^{(h)}\cdot x_i^{(h)}\right)\gamma_k\eta_\gamma + \sum_{\substack{i,i',k=1\\ i\neq i'}}^{n} F_{k,i}^{(h)}\cdot x_{i'}^{(h)}\cdot\frac{\alpha_{i'}}{\alpha_i}\gamma_k\eta_\gamma\right]_T$$

$$e\left(Z_{h,h'}, \Gamma_{h,h'}\right) = \left[\sum_{k\in[n]}\left(\sum_{i,j\in[n]} G_{k,(i,j)}^{(h,h')}\cdot x_i^{(h)}x_j^{(h')}\right)\gamma_k\eta_\gamma + \sum_{\substack{i,j,i',j',k\in[n]\\ (i,j)\neq(i',j')}} G_{k,(i,j)}^{(h,h')}\cdot x_{i'}^{(h)}x_{j'}^{(h')}\cdot\frac{\alpha_{i'}\beta_{j'}}{\alpha_i\beta_j}\gamma_k\eta_\gamma\right]_T.$$

By using the identities above and equation (5), we have

$$\prod_{h\in\mathcal{S}_1(f)} e\left(X_h, \Phi_h\right)\cdot\prod_{(h,h')\in\mathcal{S}_2^{\otimes}(f)} e\left(Z_{h,h'}, \Gamma_{h,h'}\right)$$

$$= \left[\sum_{\substack{h\in\mathcal{S}_1(f)\\ i,k\in[n]}} F_{k,i}^{(h)}\cdot x_i^{(h)}\cdot\gamma_k\eta_\gamma + \sum_{\substack{(h,h')\in\mathcal{S}_2^{\otimes}(f)\\ i,j,k\in[n]}} G_{k,(i,j)}^{(h,h')}\cdot x_i^{(h)}x_j^{(h')}\cdot\gamma_k\eta_\gamma\right]_T e\left(\pi^{(\gamma)}, [1]_2\right)$$

$$= \left[\sum_{k\in[n]}(y_k - e_k)\gamma_k\eta_\gamma\right]_T e\left(\pi^{(\gamma)}, [1]_2\right) = e\left(\mathsf{com}_y\cdot\Theta^{-1}, [\eta_\gamma]_2\right) e\left(\pi^{(\gamma)}, [1]_2\right).$$

Note that from the equations above it also follows that CFC is additively homomorphic.

6.4 Succinctness

An opening proof π to a given function $f\in\mathcal{F}_{\mathsf{quad},\kappa}$ includes $\left|\mathcal{S}_2^{\otimes}(f)\right| = \kappa$ commitments to tensored inputs $\tilde{X}_{h,h'}$, and the triples of elements $\{X_h^{(2)}, X_h^{(\beta)}, \pi_h^{(\beta)}\}_{h\in\mathcal{S}_2(f)}$, which are $3|\mathcal{S}_2(f)|$ group elements. Finally, π includes three additional group elements $Y^{(\gamma)}, \pi^{(\alpha)}, \pi^{(\gamma)}$. Hence, the proof consists of $\kappa + 3|\mathcal{S}_2(f)| + 3$ group elements, and essentially ranges from $\mathcal{O}(1)$ (in fact π has only 3 elements if f is a linear function) to $\mathcal{O}\left(m^2\right)$ depending on the quadratic support of f. Precisely, considering a fixed polynomial $p(\lambda)$ that upper bounds the size of a group element from $\mathbb{G}_1$ or $\mathbb{G}_2$, our CFC is $\mathcal{O}(\kappa)$-succinct.

When the CFC is executed on functions from the class $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$ introduced in Section 5.1 we have that $|\mathcal{S}_2(f)| = \kappa \leq m$. In this case a CFC opening contains $4\kappa + 3$ group elements and our CFC is also $\mathcal{O}(m)$-succinct.

6.5 Resulting Instantiations of FC for Circuits

We summarize the FC schemes that result from instantiating our generic construction of Section 5 with our pairing-based CFC.

Corollary 1. *Assume that the n-HiKer assumption holds for $\mathcal{BG}$. Then the following statements hold:*

1. *There exists an FC scheme for the class $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}$ of arithmetic circuits of width $w \leq n$ that is $O(d \cdot t)$-succinct. In particular, the FC is $O(d^2)$-succinct for an* arbitrary *arithmetic circuit of multiplicative depth d, and is $O(d)$-succinct for a* layered *arithmetic circuit of multiplicative depth d.*
2. *There exists an FC scheme for the class $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}$ of arithmetic circuits of width $w \leq n$ that is $\mathcal{O}\big(d^2 \cdot w \cdot n^{-1}\big)$-succinct.*
3. *There exists an FC scheme for the class of arithmetic circuits of size $\leq S$, that is $\mathcal{O}(d)$-succinct where d is the multiplicative depth of the circuit.*
4. *For any $w_0 \geq 2$, there exists an FC scheme for the class $\mathcal{F} = \{\mathcal{F}_{(d,t,w)}\}$ of circuits of arbitrary width $w > w_0$ that is $\mathcal{O}\big(d \cdot t \cdot (w/w_0)^2\big)$-succinct.*

Proof. Consider the FC construction in Section 5 instantiated with our pairing-based CFC for quadratic functions. More precisely, we consider arithmetic circuits following the model described in Section 5.1 which allows us to use CFC only with quadratic functions in $\mathcal{F}_{\mathsf{level}}$. The statements of the corollary can be obtained by combining the following observations.

1. For arbitrary circuits, note that the in-degree t of the circuit upper bounds the number m of inputs used in the CFC, and thus an FC proof consists of d CFC proofs, which makes a total of $4dt + 3d$ group elements. $O(d^2)$-succinctness for arbitrary arithmetic circuits follows from the fact that an arbitrary arithmetic circuit of depth d may have in-degree up to d, while $O(d)$-succinctness for layered circuits follows from the in-degree being 1 in such circuits.
2. The statement follows from the transformation that we present in Proposition 2.
3. To see this statement, let us consider the folklore transformation from arbitrary to layered arithmetic circuits (which is a special case of the transformation in Proposition 2). If one starts from a circuit $\mathcal{C}$ of width n and depth d, the circuit $\mathcal{C}'$ resulting from this transformation has the same depth, but width $\leq n \cdot d$, which is upper bounded by the circuit size S.
4. The statement follows directly from Proposition 3, where w_0 is the maximum width supported by the parameters of the given FC.

6.6 Proof of Security

In this section, we prove that our CFC satisfies evaluation binding. In Appendix C, we also show that our CFC satisfies knowledge extractability by relying on a non-falsifiable assumption.

Consider an adversary $\mathcal{A}$ who returns a tuple $((\mathsf{com}_h)_{h\in[m]}, \mathsf{com}_y, f, \pi, \tilde{\mathsf{com}}_y, \tilde{\pi})$ that breaks evaluation binding, set $X_h := \mathsf{com}_h$, and parse the proofs as follows

$$\pi := \left(\{X_h^{(2)}, X_h^{(\beta)}, \pi_h^{(\beta)}\}_{h\in\mathcal{S}_2(f)}, \{Z_{h,h'}\}_{(h,h')\in\mathcal{S}_2^{\otimes}(f)}, Y^{(\gamma)}, \pi^{(\alpha)}, \pi^{(\gamma)}\right)$$
$$\tilde{\pi} := \left(\{\tilde{X}_h^{(2)}, \tilde{X}_h^{(2)}, \tilde{\pi}_h^{(\beta)}\}_{h\in\mathcal{S}_2(f)}, \{\tilde{Z}_{h,h'}\}_{(h,h')\in\mathcal{S}_2^{\otimes}(f)}, \tilde{Y}^{(\gamma)}, \tilde{\pi}^{(\alpha)}, \tilde{\pi}^{(\gamma)}\right)$$

Recall that by definition of evaluation binding, if $\mathcal{A}$'s attack is successful, both proofs must verify for the same function f, the same input commitments X_h for $h \in [m]$, and for different output commitments $\mathsf{com}_y \neq \tilde{\mathsf{com}}_y$.

The intuition of the proof is that $\mathcal{A}$ can cheat in three possible ways, for which we define three events E_1, E_2, E_3 as follows:

- E_1 is the event that $Y^{(\gamma)} = \tilde{Y}^{(\gamma)}$. As $\mathsf{com}_y \neq \tilde{\mathsf{com}}_y$, this implies an evaluation binding break in the linear map commitment proof in equation (9).
- E_2 is the event that E_1 does not happen (i.e., $Y^{(\gamma)} \neq \tilde{Y}^{(\gamma)}$) and that $X_{h^*}^{(\beta)} \neq \tilde{X}_{h^*}^{(\beta)}$ for some $h^* \in \mathcal{S}_2(f)$. This means that the proofs $\pi_{h^*}^{(\beta)}, \tilde{\pi}_{h^*}^{(\beta)}$ open the commitment com_{h^*} to two different output commitments for the identity function, which breaks evaluation binding in equation (7).
- E_3 is the event that neither E_1 nor E_2 occur. In this case, we will show that evaluation binding breaks in equation (11).

For any of these events, we will use $\mathcal{A}$'s output to break the n-HiKer assumption if this is embedded into ck. For this embedding, $\mathcal{B}$ makes a guess $\hat{s} \in \{0,1\}$ such that $\hat{s} = 0$ corresponds to a guess that event E_1 occurs while $\hat{s} = 1$ corresponds to a guess that either E_2 or E_3 will occur. This $\hat{s}$ is perfectly hidden to $\mathcal{A}$.

Next we describe how to build $\mathcal{B}$ out of $\mathcal{A}$.

Commitment key generation. Let $\mathcal{B}$ be an adversary against the n-HiKer assumption. $\mathcal{B}$ uniformly samples a value $\hat{s} \leftarrow\$ \{0,1\}$ and simulates ck as follows.

Case $\hat{s} = 0$. $\mathcal{B}$ samples $\boldsymbol{\alpha}, \boldsymbol{\beta} \leftarrow\$ \mathbb{F}^n, \eta_\beta, \eta_\gamma \leftarrow\$ \mathbb{F}$ and implicitly sets $\boldsymbol{\gamma} := \boldsymbol{\sigma}$ and $\eta_\alpha := \eta$ from the input of the assumption. It is easy to see that this implicit setting allows $\mathcal{B}$ to compute all the elements in the first row of ck, namely:

$$[\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\alpha} \otimes \boldsymbol{\beta}]_1, [\boldsymbol{\alpha}, \eta_\alpha, \eta_\beta, \eta_\gamma]_2$$

We show how $\mathcal{B}$ can simulate the remaining elements in the second and third rows of ck starting from the inputs from the n-HiKer assumption as follows:

$$
\begin{aligned}
\forall i, i' \in [n], i \neq i' : \quad \alpha_i \left[\tfrac{\eta\sigma_{i'}}{\sigma_i}\right]_1 &= \left[\alpha_i \frac{\gamma_{i'}}{\gamma_i}\eta_\alpha\right]_1 \\
\tfrac{\alpha_{i'}}{\alpha_i}\beta_i\eta_\beta \left[1\right]_1 &= \left[\frac{\alpha_{i'}}{\alpha_i}\beta_i\eta_\beta\right]_1 \\
\forall i, j, i', j', k \in [n] : (i,j) \neq (i',j') : \tfrac{\alpha_{i'}\beta_{j'}}{\alpha_i\beta_j}\eta_\gamma \left[\sigma_k\right]_1 &= \left[\frac{\alpha_{i'}\beta_{j'}}{\alpha_i\beta_j}\gamma_k\eta_\gamma\right]_1 \\
\forall i \in [n] : \quad \alpha_i \left[\tfrac{\eta}{\sigma_i}\right]_2 &= \left[\frac{\alpha_i\eta_\alpha}{\gamma_i}\right]_2 \\
\tfrac{\beta_i\eta_\beta}{\alpha_i}\left[1\right]_2 &= \left[\frac{\beta_i\eta_\beta}{\alpha_i}\right]_2 \\
\forall i, k \in [n] : \quad \tfrac{\eta_\gamma}{\alpha_i}\left[\sigma_k\right]_2 &= \left[\frac{\gamma_k\eta_\gamma}{\alpha_i}\right]_2 \\
\forall i, j, k \in [n] : \quad \tfrac{\eta_\gamma}{\alpha_i\beta_j}\left[\sigma_k\right]_2 &= \left[\frac{\gamma_k\eta_\gamma}{\alpha_i\beta_j}\right]_2
\end{aligned}
$$

As one can notice, in this case of $\hat{s} = 0$ we embed in the commitment key only a subset of the elements of the assumption. This means that the reduction for adversaries causing event E_1 can actually be done based on a weaker version of the assumption which includes only the subset of the elements that we need for this case.

Case $\hat{s} = 1$. $\mathcal{B}$ samples $\eta_\alpha, r_\beta, r_\gamma \leftarrow\$ \mathbb{F}$ and $\boldsymbol{\gamma} \leftarrow\$ \mathbb{F}^n$ and implicitly sets $\boldsymbol{\alpha} := \boldsymbol{\sigma}, \boldsymbol{\beta} := \boldsymbol{\tau}, \eta_\beta := r_\beta \cdot \eta, \eta_\gamma := r_\gamma \cdot \eta$. As for the case of $\hat{s} = 0$, it is easy to see that this implicit setting allows $\mathcal{B}$ to compute all the elements in the first row of ck, namely $\left[\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\alpha} \otimes \boldsymbol{\beta}\right]_1, \left[\boldsymbol{\alpha}, \eta_\alpha, \eta_\beta, \eta_\gamma\right]_2$.

Next, we show how $\mathcal{B}$ can simulate the remaining elements in the second and third rows of ck starting from the inputs from the n-HiKer assumption as follows:

$$
\begin{aligned}
\forall i, i' \in [n], i \neq i' : \quad \tfrac{\gamma_{i'}}{\gamma_i}\eta_\alpha \left[\sigma_i\right]_1 &= \left[\alpha_i \frac{\gamma_{i'}}{\gamma_i}\eta_\alpha\right]_1 \\
r_\beta \left[\eta \tfrac{\sigma_{i'}}{\sigma_i}\tau_i\right]_1 &= \left[\frac{\alpha_{i'}}{\alpha_i}\beta_i\eta_\beta\right]_1 \\
\forall i, j, i', j', k \in [n] : (i,j) \neq (i',j') : r_\gamma\gamma_k \left[\eta \tfrac{\sigma_{i'}\tau_{j'}}{\sigma_i\tau_j}\right]_1 &= \left[\frac{\alpha_{i'}\beta_{j'}}{\alpha_i\beta_j}\gamma_k\eta_\gamma\right]_1 \\
\forall i \in [n] : \quad \tfrac{\eta_\alpha}{\gamma_i}\left[\sigma_i\right]_2 &= \left[\frac{\alpha_i\eta_\alpha}{\gamma_i}\right]_2 \\
r_\beta \left[\tfrac{\eta\tau_i}{\sigma_i}\right]_2 &= \left[\frac{\beta_i\eta_\beta}{\alpha_i}\right]_2 \\
\forall i, k \in [n] : \quad r_\gamma\gamma_k \left[\tfrac{\eta}{\sigma_i}\right]_2 &= \left[\frac{\gamma_k\eta_\gamma}{\alpha_i}\right]_2 \\
\forall i, j, k \in [n] : \quad r_\gamma\gamma_k \left[\tfrac{\eta}{\sigma_i\tau_j}\right]_2 &= \left[\frac{\gamma_k\eta_\gamma}{\alpha_i\beta_j}\right]_2
\end{aligned}
$$

Execution of $\mathcal{A}$. Once having generated ck as described above, $\mathcal{B}$ runs $\mathcal{A}(\mathsf{ck})$, receives the output $((\mathsf{com}_h)_{h\in[m]}, \mathsf{com}_y, f, \pi, \tilde{\mathsf{com}}_y, \tilde{\pi})$ and parses the proofs as before. Notice that ck is perfectly distributed as the one generated by Setup and thus the value $\hat{s}$ is perfectly hidden to $\mathcal{A}$.

The reduction proceeds differently according to the output produced by $\mathcal{A}$, that we split in the events E_1, E_2, E_3 as defined above.

E_1 **occurs:** If $\hat{s} \neq 0$, then $\mathcal{B}$ aborts. Otherwise it proceeds as follows. Recall that in this case we have that as $Y^{(\gamma)} = \tilde{Y}^{(\gamma)}$, then $\pi^{(\alpha)}, \tilde{\pi}^{(\alpha)}$ open to different $\mathsf{com}_y, \tilde{\mathsf{com}}_y$. Therefore, by equation (9) we have that

$$e\left(\pi^{(\alpha)}, [1]_2\right) e\left(\mathsf{com}_y, [\eta_\alpha]_2\right) = e\left(Y^{(\gamma)}, \sum_{i\in[n]} \left[\frac{\alpha_i \eta_\alpha}{\gamma_i}\right]_2\right) = e\left(\tilde{\pi}^{(\alpha)}, [1]_2\right) e\left(\tilde{\mathsf{com}}_y, [\eta_\alpha]_2\right)$$

Then, $\mathcal{B}$ returns (U, V) such that

$$U := \tilde{\mathsf{com}}_y / \mathsf{com}_y,\ V := \pi^{(\alpha)}, / \tilde{\pi}^{(\alpha)}.$$

If $\mathcal{B}$ did not abort, then $\hat{s} = 0$. Thus, $\eta_\alpha = \eta$ and $e\left(U, [\eta]_2\right) = e\left(V, [1]_2\right)$.

E_2 **occurs:** If $\hat{s} \neq 1$, then $\mathcal{B}$ aborts. Otherwise, let h^* be some index such that $X_{h^*}^{(\beta)} \neq \tilde{X}_{h^*}^{(\beta)}$; note that h^* must exist by definition of event E_2. Similarly as before, from equation (7) we have that

$$e\left(\pi_{h^*}^{(\beta)}, [1]_2\right) e\left(X_{h^*}^{(\beta)}, [\eta_\beta]_2\right) = e\left(X_{h^*}, \sum_{i\in[n]} \left[\frac{\beta_i \eta_\beta}{\alpha_i}\right]_2\right) = e\left(\tilde{\pi}_{h^*}^{(\beta)}, [1]_2\right) e\left(\tilde{X}_{h^*}^{(\beta)}, [\eta_\beta]_2\right).$$

Then, $\mathcal{B}$ returns (U, V) such that

$$U := (\tilde{X}_{h^*}^{(\beta)} / X_{h^*}^{(\beta)})^{r_\beta},\ V := \pi_{h^*}^{(\beta)} / \tilde{\pi}_{h^*}^{(\beta)}.$$

If $\mathcal{B}$ did not abort, then $\hat{s} = 1$. Thus, $\eta_\beta = r_\beta \cdot \eta$ and $e\left(U, [\eta]_2\right) = e\left(V, [1]_2\right)$.

E_3 **occurs:** If $\hat{s} \neq 1$, then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ proceeds as follows. First, note that since E_1 and E_2 did not occur, then $Y^{(\gamma)} = \tilde{Y}^{(\gamma)}$ and $X_h^{(\beta)} = \tilde{X}_h^{(2)}$ for every $h \in \mathcal{S}_2(f)$. Also, by equation (6) and by the non-degeneracy of the pairing, we have

$$e\left(X_h, [1]_2\right) = e\left([1]_1, X_h^{(2)}\right) = e\left([1]_1, \tilde{X}_h^{(2)}\right) \quad \text{which implies that } X_h^{(2)} = \tilde{X}_h^{(2)}.$$

From the equality above we can use equation (8) to also conclude that $Z_{h,h'} = \tilde{Z}_{h,h'}$ for all $(h, h') \in \mathcal{S}_2^{\otimes}(f)$. Then, since both proofs satisfy equation (11), we have

$$\begin{aligned} e\left(\pi^{(\gamma)}, [1]_2\right) e\left(Y^{(\gamma)} \cdot \Theta^{-1}, [\eta_\gamma]_2\right) &= \prod_{h\in\mathcal{S}_1(f)} e\left(X_h, \Phi_h\right) \cdot \prod_{(h,h')\in\mathcal{S}_2^{\otimes}(f)} e\left(Z_{h,h'}, \Gamma_{h,h'}\right) \\ &= e\left(\tilde{\pi}^{(\gamma)}, [1]_2\right) e\left(\tilde{Y}^{(\gamma)} \cdot \Theta^{-1}, [\eta_\gamma]_2\right). \end{aligned}$$

The reduction returns (U, V) computed as follows:

$$U := (\tilde{Y}^{(\gamma)} / Y^{(\gamma)})^{r_\gamma},\ V := \pi^{(\gamma)} / \tilde{\pi}^{(\gamma)}.$$

If $\mathcal{B}$ did not abort, then $\hat{s} = 1$ and $\eta_\gamma = r_\gamma \cdot \eta$. Thus, $e\left(U, [\eta]_2\right) = e\left(V, [1]_2\right)$. Since $\hat{s}$ is perfectly hidden $\mathcal{B}$ aborts with probability $1/2$. Hence, if $\mathcal{A}$ is successful with probability ϵ, then $\mathcal{B}$ breaks the assumption with probability $\epsilon/2$.

6.7 Efficient Verification

Our chainable functional commitment scheme CFC supports amortized efficient verification. We define the algorithms $\mathsf{VerPrep}$ and EffVer below, following Definition 4.

$\underline{\mathsf{VerPrep}(\mathsf{ck}, f)}$ Parse ck and compute the encodings $\Theta, \Phi_h, \Gamma_{h,h'}$ of f as done in the $\mathsf{CFC.Ver}$ algorithm following equation (10). Also, compute the encodings in equations (7) and (9), $\Psi^{(\beta)} = \sum_{i\in[n]} \left[\frac{\beta_i \eta_\beta}{\alpha_i}\right]_2$ and $\Psi^{(\alpha)} = \sum_{i\in[n]} \left[\frac{\alpha_i \eta_\alpha}{\gamma_i}\right]_2$.
Output $\mathsf{vk}_f := (\{\Theta, \Phi_h, \Gamma_{h,h'}\}_{(h,h')\in\mathcal{S}_2^{\otimes}(f)}, \Psi^{(\alpha)}, \Psi^{(\beta)})$.

$\underline{\mathsf{EffVer}(\mathsf{vk}_f, (\mathsf{com}_h)_{h\in[m]}, \mathsf{com}_y, \pi)}$ Parse vk_f, π and carry out all the pairing checks in the Ver algorithm, i.e., verify equations (6), (7), (8), (9), (11).

Following the description of succinctness in Section 6.4, given any $f \in \mathcal{F}_{\mathsf{quad},\kappa}$ then EffVer needs to parse a proof that has $\mathcal{O}(\kappa)$ group elements. Then, it verifies $\omega \leq \kappa$ pairing checks in equations (6) and (7), κ checks in equation (8), a single check in equation (7), and a single check involving $\kappa + \omega$ products in equation (11). Assuming that the running time of each pairing computation is bounded by some polynomial $p(\lambda) = \mathsf{poly}(\lambda)$, the running time of EffVer is therefore $\mathcal{O}(p(\lambda) \cdot |\kappa|) = \mathcal{O}(p(\lambda) \cdot |\pi|)$, which is essentially optimal.

6.8 Commitment Hiding

Our CFC construction can be made perfectly com-hiding (Definition 5) by adding randomness to the commitment. We describe the transformation $\widetilde{\mathsf{CFC}} = (\widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Com}}, \widetilde{\mathsf{Open}}, \widetilde{\mathsf{Ver}})$ below.

$\underline{\widetilde{\mathsf{Setup}}(1^\lambda, 1^n)}$ Output $\tilde{\mathsf{ck}} \leftarrow \mathsf{Setup}(1^\lambda, 1^{n+1})$.

$\underline{\widetilde{\mathsf{Com}}(\tilde{\mathsf{ck}}, \boldsymbol{x})}$ Let $r \leftarrow\!\!\$\, \mathbb{F}$. Output $(\mathsf{com}, \mathsf{aux})$ where $\mathsf{com} \leftarrow \mathsf{Com}(\tilde{\mathsf{ck}}, \boldsymbol{x}) + r \cdot [\alpha_{i+1}]_1$ and $\mathsf{aux} = (\boldsymbol{x}, r)$.

$\underline{\widetilde{\mathsf{Open}}(\tilde{\mathsf{ck}}, (\mathsf{aux}_i)_{i\in[m]}, f)}$ Let $\mathsf{aux}_i = (\boldsymbol{x}^{(i)}, r^{(i)})$. Output $\mathsf{Open}(\tilde{\mathsf{ck}}, (\mathsf{aux})_{i\in[m]}, f')$ where $f' = (f, 0)$.

$\underline{\widetilde{\mathsf{Ver}}(\tilde{\mathsf{ck}}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi)}$ Output $\mathsf{Ver}(\tilde{\mathsf{ck}}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi)$.

For the above scheme, it is easy to construct a simulator Sim as follows.

$\underline{\mathsf{Sim}_{\mathsf{Setup}}(1^\lambda, n)}$ Sample $\boldsymbol{\alpha} \leftarrow\!\!\$\, \mathbb{F}^{n+1}$ and generate $\tilde{\mathsf{ck}}$ as in $\mathsf{Setup}(1^\lambda, n+1)$, sampling additional field elements when necessary. Output $(\tilde{\mathsf{ck}}, \mathsf{td})$ where $\mathsf{td} = \boldsymbol{\alpha}$.

$\underline{\mathsf{Sim}_{\mathsf{Com}}(\mathsf{td})}$ Sample $r \leftarrow\!\!\$\, \mathbb{F}$ and output $(\mathsf{com}, \mathsf{aux})$ where $\mathsf{com} = r \cdot [\alpha_{n+1}]_1$ and $\mathsf{aux} = (\mathbf{0}, r)$.

$\underline{\mathsf{Sim}_{\mathsf{Equiv}}(\mathsf{td}, \mathsf{com}, \mathsf{aux}, \boldsymbol{x})}$ The algorithm uses the field elements in $\boldsymbol{\alpha}$ to find a value $r' \in \mathbb{F}$ such that $\mathsf{com} = \widetilde{\mathsf{Com}}(\boldsymbol{x}, r')$. It simply obtains the solution r' of the linear equation $\langle \boldsymbol{x}, \boldsymbol{\alpha}\rangle + \alpha_{n+1} r' = \alpha_{n+1} r$ and outputs $\mathsf{aux} = (\boldsymbol{x}, r')$.

7 Lattice-based CFC for Quadratic Functions

In this section, we present a lattice-based construction of a CFC for quadratic functions. Our construction can be seen as a lattice-analogue of the pairing-based scheme presented in Section 6 obtained via a slight generalisation of the translation technique in [ACL+22].

7.1 Lattice Preliminaries

Let $\mathcal{R} = \mathbb{Z}[\zeta]$, where ζ is a fixed primitive m-th root of unity, be the ring of integers of the m-th cyclotomic field of degree $d = \varphi(m)$, where elements are represented by their coefficient embedding $x = \sum_{i=0}^{d-1} x_i \cdot \zeta^i$. If m is a prime-power (resp. power of 2), we call $\mathcal{R}$ a prime-power (resp. power-of-two) cyclotomic ring. For the rest of this section we will assume that $m = \mathsf{poly}(\lambda)$.

For $x \in \mathcal{R}$, write $\|x\| \coloneqq \max_{i=0}^{d-1} |x_i|$ for the infinity norm induced on $\mathcal{R}$ by $\mathbb{Z}$. The norm generalises naturally to vectors $\boldsymbol{u} = (u_1, \ldots, u_n) \in \mathcal{R}^n$, with $\|\boldsymbol{u}\| \coloneqq \max_{i=1}^{n} \|u_i\|$. For $q \in \mathbb{N}$, write $\mathcal{R}_q \coloneqq \mathcal{R}/q\mathcal{R}$. We always assume that q is a (rational) prime. By a slight abuse of notation, we identity $\mathcal{R}_q$ with its balanced representation, i.e. if $x = \sum_{i=0}^{d-1} x_i \cdot \zeta^i \in \mathcal{R}_q$ then $|x_i| \leq q/2$ for all i. The set of units, i.e., invertible elements, in $\mathcal{R}_q$ is denoted by $\mathcal{R}_q^\times$.

The ring expansion factor $\gamma_\mathcal{R}$ of $\mathcal{R}$ is defined as $\gamma_\mathcal{R} \coloneqq \max_{a,b \in \mathcal{R}} \frac{\|a \cdot b\|}{\|a\| \cdot \|b\|}$. It is known [AL21] that if $\mathcal{R}$ is a prime-power cyclotomic ring then $\gamma_\mathcal{R} \leq 2 \cdot d$, and if $\mathcal{R}$ is a power-of-two cyclotomic ring then $\gamma_\mathcal{R} \leq d$.

Lattice Trapdoors. We will make use of the following standard algorithms (e.g. [GPV08, MP12, GM18]) associated to lattice trapdoors and their properties for sufficiently large "leftover hash lemma parameter" $\mathsf{lhl}(\mathcal{R}, \eta, q, \beta) = O(\eta \log_\beta q)$:

- $(\mathbf{A}, \mathsf{td}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(\mathcal{R}, 1^\eta, 1^\ell, q, \beta)$: The trapdoor generation algorithm generates a matrix $\mathbf{A} \in \mathcal{R}_q^{\eta \times \ell}$ along with a trapdoor $\mathsf{td}_\mathbf{A}$. It is assumed that (η, ℓ, q, β) are implicitly specified by $\mathsf{td}_\mathbf{A}$. When $\ell \geq \mathsf{lhl}(\mathcal{R}, \eta, q, \beta)$, the distribution of $\mathbf{A}$ is within $\mathsf{negl}(\lambda)$ statistical distance of $U(\mathcal{R}_q^{\eta \times \ell})$.
- $\boldsymbol{u} \leftarrow \mathsf{SampD}(\mathcal{R}, 1^\eta, 1^\ell, q, \beta')$: The domain sampling algorithm samples a vector $\boldsymbol{u} \in \mathcal{R}^\ell$ with norm $\|\boldsymbol{u}\| \leq \beta'$. When $\beta' \geq \beta$ and $\ell \geq \mathsf{lhl}(\mathcal{R}, \eta, q, \beta)$, then the distribution of $(\mathbf{A}, \mathbf{A} \cdot \boldsymbol{u} \bmod q)$ for a uniformly random $\mathbf{A} \leftarrow\$ \mathcal{R}_q^{\eta \times \ell}$ is within $\mathsf{negl}(\lambda)$ statistical distance of $U(\mathcal{R}_q^{\eta \times \ell} \times \mathcal{R}_q^\eta)$.
- $\boldsymbol{u} \leftarrow \mathsf{SampPre}(\mathsf{td}_\mathbf{A}, \boldsymbol{v}, \beta')$: The preimage sampling algorithm inputs a vector $\boldsymbol{v} \in \mathcal{R}_q^n$ and outputs a vector $\boldsymbol{u} \in \mathcal{R}^\ell$. If the parameters (η, ℓ, q, β) of $\mathsf{td}_\mathbf{A}$ satisfy $\beta' \geq \beta$ and $\ell \geq \mathsf{lhl}(\mathcal{R}, \eta, q, \beta)$, then $\boldsymbol{u}$ and $\boldsymbol{v}$ satisfy $\mathbf{A} \cdot \boldsymbol{u} = \boldsymbol{v} \bmod q$ and $\|\boldsymbol{u}\| \leq \beta'$. Furthermore, $\boldsymbol{u}$ is within $\mathsf{negl}(\lambda)$ statistical distance to $\boldsymbol{u} \leftarrow \mathsf{SampD}(\mathcal{R}, 1^n, 1^\ell, q, \beta')$ conditioned on $\mathbf{A} \cdot \boldsymbol{u} = \boldsymbol{v} \bmod q$.

7.2 Hardness Assumptions

The k-R-ISIS assumption family[17] was recently introduced in [ACL+22] as a natural extention of the standard short integer solution (SIS) assumption and a natural lattice-analogue of a certain class of pairing-based assumptions. The k-R-ISIS assumption family was accompanied by a translation technique outlined in [ACL+22] for translating pairing-based schemes and assumptions to their lattice-analogues.

For instance, a certain k-R-ISIS assumption could be parametrised by a set $\mathcal{G}$ of monomials. It states that even when given short preimages $\boldsymbol{u}_g$ satisfying $\mathbf{A} \cdot \boldsymbol{u}_g = \boldsymbol{t} \cdot g(\boldsymbol{v}) \bmod q$ for all $g \in \mathcal{G}$, it is hard to find a short non-zero preimage $\boldsymbol{u}^*$ satisfying $\mathbf{A} \cdot \boldsymbol{u}^* = \mathbf{0} \bmod q$.

Applying the translation technique in [ACL+22] to the pairing-based assumption (Definition 9) which underlies the security of the pairing-based CFC construction, we encounter an obstacle that there is no translation for the term $[\eta]_2$ in the challenge relation $e(U, [\eta]_2) = e(V, [1]_2)$.

[17] We use k-R-ISIS to refer to both the ring and module version. In [ACL+22], the module version is given the name k-M-ISIS.

To overcome the above obstacle, in the following, we introduce (a special case of) a generalisation of the k-R-ISIS assumption which we call the Twin-k-R-ISIS assumption. In a nutshell, instead of a single set $\mathcal{G}$ of monomials, we now have two (or in general more) sets $\mathcal{G}_A$ and $\mathcal{G}_B$ of non-overlapping monomials. The Twin-k-R-ISIS assumption states that even when given short preimages $\boldsymbol{u}_g$ satisfying $\mathbf{A} \cdot \boldsymbol{u}_g = \boldsymbol{t} \cdot g(\boldsymbol{v}) \bmod q$ for all $g \in \mathcal{G}_A$ and short preimages $\boldsymbol{w}_g$ satisfying $\mathbf{B} \cdot \boldsymbol{u}_g = \boldsymbol{t} \cdot g(\boldsymbol{v}) \bmod q$ for all $g \in \mathcal{G}_B$, it is hard to find a short non-zero preimage $(\boldsymbol{u}^*, \boldsymbol{w}^*)$ satisfying $\mathbf{A} \cdot \boldsymbol{u}^* + \mathbf{B} \cdot \boldsymbol{w}^* = \mathbf{0} \bmod q$. We stress that the non-overlapping requirement of $\mathcal{G}_A$ and $\mathcal{G}_B$ is crucial, for otherwise $(\boldsymbol{u}_g, -\boldsymbol{w}_g)$ would be a trivial solution for any $g \in \mathcal{G}_A \cap \mathcal{G}_B$. Other than this trivial attack (which is ruled out), it could be verified that the (failed) attack strategies discussed in [ACL+22] against the k-R-ISIS assumption also fail against the Twin-k-R-ISIS assumption.[18]

Definition 10 (Twin-k-R-ISIS Assumption). *Let $\ell, \eta \in \mathbb{N}$, q be a rational prime, $\beta, \beta^* \in \mathbb{R}^+$,*

$$\mathcal{G}_A := \left\{ \frac{X_{i'}}{X_i} \cdot \bar{X}_k,\ \frac{X_{i'}}{X_i} \cdot \check{X}_k,\ \frac{\bar{X}_{i'}}{\bar{X}_i} \cdot X_k \right\}_{i,i',k \in [n], i \neq i'} \cup \left\{ \frac{X_{i'} \cdot \check{X}_{j'}}{X_i \cdot \check{X}_j} \cdot \bar{X}_k \right\}_{\substack{i,i',j,j',k \in [n] \\ i \neq i', j \neq j'}},$$

$\mathcal{G}_B := \left\{X_k, \bar{X}_k, \check{X}_k\right\}_{k \in [n]}$, and $\mathcal{G} := \mathcal{G}_A \cup \mathcal{G}_B$. Let $\mathcal{D}$ be a distribution over $\mathcal{R}^\ell$. Write $\mathsf{pp} := (\mathcal{R}_q, \eta, \ell, n, \beta, \beta^*, \mathcal{G}_A, \mathcal{G}_B, \mathcal{D})$. The k-R-$\mathsf{ISIS}_{\mathsf{pp}}$ assumption states that for any PPT adversary $\mathcal{A}$ we have $\mathsf{Adv}^{\text{k-m-isis}}_{\mathsf{pp},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$, where

$$\mathsf{Adv}^{\text{k-m-isis}}_{\mathsf{pp},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l|l} & \mathbf{A} \leftarrow\!\!\$\ \mathcal{R}_q^{\eta \times \ell} \bmod q;\ \mathbf{B} \leftarrow\!\!\$\ \mathcal{R}_q^{\eta \times \ell} \bmod q \\ & \boldsymbol{t} \leftarrow\!\!\$\ (\mathcal{R}_q^\times)^\eta;\ \boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}} \leftarrow\!\!\$\ (\mathcal{R}^\times)^n \\ \mathbf{A} \cdot \boldsymbol{u}^* + \mathbf{B} \cdot \boldsymbol{w}^* \equiv \mathbf{0} \bmod q & \boldsymbol{u}_g \leftarrow\!\!\$\ \mathcal{D} : \mathbf{A} \cdot \boldsymbol{u}_g \equiv \boldsymbol{t} \cdot g(\boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}) \bmod q,\ \forall g \in \mathcal{G}_A \\ \wedge\ 0 < \|(\boldsymbol{u}^*, \boldsymbol{w}^*)\| \leq \beta^* & \boldsymbol{w}_g \leftarrow\!\!\$\ \mathcal{D} : \mathbf{B} \cdot \boldsymbol{w}_g \equiv \boldsymbol{t} \cdot g(\boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}) \bmod q,\ \forall g \in \mathcal{G}_B \\ & (\boldsymbol{u}^*, \boldsymbol{v}^*) \leftarrow \mathcal{A}\left(\mathbf{A}, \mathbf{B}, \boldsymbol{t}, \boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}, \{\boldsymbol{u}_{\mathcal{G}_A}, \boldsymbol{w}_{\mathcal{G}_B}\}\right) \end{array} \right].$$

We discuss briefly the relations between the Twin-k-R-ISIS assumption, the original k-R-ISIS assumption [ACL+22], and the recently introduced BASIS assumption [WW23]. Below, we adopt the notation $\mathbf{A}^{-1}(\boldsymbol{v})$ which refers to a short preimage $\boldsymbol{u}$ satisfying $\mathbf{A} \cdot \boldsymbol{u} = \boldsymbol{v} \bmod q$ sampled from some distribution.

The BASIS assumption [WW23] is a strong assumption which states the the SIS problem with respect to $\mathbf{C}$ even if a trapdoor for a matrix $\mathbf{C}$ related to $\mathbf{D}$ is given. Wee and Wu [WW23] discussed the relation between BASIS and k-R-ISIS and showed evidence that the former implies the latter (with appropriate parameters) but did not show a reduction. Below, we sketch a reduction from (a ring version of) BASIS to Twin-k-R-ISIS. Since the Twin-k-R-ISIS assumption clearly implies the original k-R-ISIS assumption for appropriate parameters, we also obtain a reduction from BASIS to k-R-ISIS.

[18] We refer to the attack strategies discussed in [ACL+22, Section 4.1]. There, the authors discussed two (they gave three, but the third generalises the second) attacks: 1) Direct SIS attack: Finding a short vector in the kernel of $(\mathbf{A}| - \boldsymbol{t} \cdot g^*(\boldsymbol{v}))$. 2) Find a (not necessarily short) linear combination $(z_1, \ldots, z_k)$ so that $s^* \cdot g^*(\boldsymbol{v}) = \sum_i z_i \cdot g_i(\boldsymbol{v})$ and $\boldsymbol{u}_{g^*} = \sum_i z_i \cdot \boldsymbol{u}_{g_i}$ is short. There seems to be no obvious way that either attack can take advantage of the two-slotted structure in the twin-kMISIS assumption.

We consider the following instantiation of the BASIS problem: For a matrix $\mathbf{C}$ and a vector $\boldsymbol{v}$, define the related matrix[19]

$$\mathbf{D} \coloneqq \left(\begin{array}{c|c|c} \mathsf{diag}\left((g(\boldsymbol{v})^{-1} \cdot \mathbf{A})_{g \in \mathcal{G}_A}\right) & & -\mathbf{G} \\ & & \vdots \\ & \mathsf{diag}\left((g(\boldsymbol{v})^{-1} \cdot \mathbf{B})_{g \in \mathcal{G}_B}\right) & -\mathbf{G} \end{array} \right)$$

where $\mathsf{diag}(\cdot)$ denotes block-diagonalisation, $\mathbf{A} = \mathbf{C} \cdot \mathbf{R}_A$ and $\mathbf{B} = \mathbf{C} \cdot \mathbf{R}_B$ for some short $\mathbf{R}_A$ and $\mathbf{R}_B$, and $\mathbf{G}$ is the gadget matrix. The BASIS problem is: Given $(\mathbf{C}, \mathbf{D})$ and a trapdoor of $\mathbf{D}$, find $\mathbf{C}^{-1}(\mathbf{0})$.

To generate a Twin-k-R-ISIS instance, use the trapdoor of $\mathbf{D}$ to sample $\mathbf{D}^{-1}(\mathbf{0}) = (\ldots, \boldsymbol{u}_{A,g}, \ldots, \boldsymbol{u}_{B,g}, \ldots, \tilde{\boldsymbol{t}})$. Let $\boldsymbol{t} \coloneqq \mathbf{G} \cdot \tilde{\boldsymbol{t}} \bmod q$. By construction, we have $\mathbf{A} \cdot \boldsymbol{u}_{A,g} = \boldsymbol{t} \cdot g(\boldsymbol{v}) \bmod q$ and $\mathbf{B} \cdot \boldsymbol{u}_{B,g} = \boldsymbol{t} \cdot g(\boldsymbol{v}) \bmod q$. We can therefore run the Twin-k-R-ISIS solver to obtain short $(\boldsymbol{u}^*, \boldsymbol{w}^*)$ such that

$$\begin{aligned} \mathbf{A} \cdot \boldsymbol{u}^* + \mathbf{B} \cdot \boldsymbol{w}^* &= \mathbf{0} \bmod q \\ \mathbf{C} \cdot (\mathbf{R}_A \cdot \boldsymbol{u}^* + \mathbf{R}_B \cdot \boldsymbol{w}^*) &= \mathbf{0} \bmod q \end{aligned}$$

It is not difficult to argue that the simulated Twin-k-R-ISIS instance is well-distributed, and the extracted BASIS solution $\mathbf{R}_A \cdot \boldsymbol{u}^* + \mathbf{R}_B \cdot \boldsymbol{w}^*$ is non-zero with non-negligible probability.

7.3 Construction

In the following, we construct a lattice-based chainable functional commitment scheme. Our construction is parametrised by a ring $\mathcal{R}$, dimensions η, ℓ, modulus q, norm bound β, an input length n, and the number of inputs m. Before describing the construction, we first introduce the following shorthands and notation.

For a quadratic polynomial map $f : \mathcal{R}^{mn} \to \mathcal{R}^n$, we express $f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$ similarly to previous sections,

$$f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) = \boldsymbol{e} + \sum_{h \in \mathcal{S}_1(f)} \mathbf{F}_h \cdot \boldsymbol{x}_h + \sum_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} \mathbf{G}_{h,h'} \cdot (\boldsymbol{x}_h \otimes \boldsymbol{x}_{h'})$$

for some $\mathbf{G}_{h,h'} \in \mathcal{R}^{n \times n^2}$, $\mathbf{F}_h \in \mathcal{R}^{n \times n}$, and $\boldsymbol{e} \in \mathcal{R}^n$.

Different from the pairing-based construction, our lattice-based construction is additionally parametrised by a norm bound $\alpha \in \mathbb{R}^+$. We assume that messages $\boldsymbol{x}$ and each coefficient of any quadratic polynomial map f to be opened have norm at most α, and f is such that for any $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ of norm at most α, it holds that $\|f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)\| \leq \alpha$.

For a vector $\boldsymbol{v} \in (\mathcal{R}_q^\times)^n$, denote its component-wise inverse by $\boldsymbol{v}^\dagger \coloneqq (v_i^{-1})_{i=1}^n$. Define $\mathbf{Z}_{\boldsymbol{v}} \coloneqq \boldsymbol{v}^\dagger \cdot \boldsymbol{v}^{\mathtt{T}} - \mathbf{I} = (z_{i,j})_{i,j}$ where

$$z_{i,j} = \begin{cases} 0 & i = j \\ v_i^{-1} \cdot v_j & i \neq j \end{cases}.$$

We are now ready to describe the construction as follows.

[19] The matrix $\mathbf{D}$ consists of a block-diagonal part on the left and a block-column on the right. The block-diagonal part can be split into two. On the top left, each block is given by $g(\boldsymbol{v})^{-1} \cdot \mathbf{A}$ for some $g \in \mathcal{G}_A$. On the bottom right, each block is given by $g(\boldsymbol{v})^{-1} \cdot \mathbf{B}$ for some $g \in \mathcal{G}_B$. The block-column consists of $-\mathbf{G}$ blocks.

$\underline{\mathsf{Setup}(1^\lambda, 1^n)}$

– Sample trapdoored matrices $(\mathbf{A}, \mathsf{td}_\mathbf{A}), (\mathbf{B}, \mathsf{td}_\mathbf{B}) \leftarrow \mathsf{TrapGen}(\mathcal{R}, 1^\eta, 1^\ell, q, \beta)$.
– Sample submodule generator $\boldsymbol{t} \leftarrow\!\$\; (\mathcal{R}_q^\times)^\eta$.
– Sample commitment key vectors $\boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}} \leftarrow\!\$\; \mathcal{R}_q^n$.
– Sample a short preimage $\boldsymbol{u}_g \leftarrow \mathsf{SampPre}(\mathsf{td}_\mathbf{A}, \boldsymbol{t} \cdot g(\boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}) \bmod q)$ for each $g \in \mathcal{G}_A$, where

$$\mathcal{G}_A := \left\{ \frac{X_{i'}}{X_i} \cdot \bar{X}_k,\ \frac{X_{i'}}{X_i} \cdot \check{X}_k,\ \frac{\bar{X}_{i'}}{\bar{X}_i} \cdot X_k \right\}_{i,i',k \in [n], i \neq i'} \cup \left\{ \frac{X_{i'} \cdot \check{X}_{j'}}{X_i \cdot \check{X}_j} \cdot \bar{X}_k \right\}_{\substack{i,i',j,j',k \in [n] \\ i \neq i', j \neq j'}}$$

– Sample a short preimage $\boldsymbol{w}_g \leftarrow \mathsf{SampPre}(\mathsf{td}_\mathbf{B}, \boldsymbol{t} \cdot g(\boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}) \bmod q)$ for each $g \in \mathcal{G}_B$, where

$$\mathcal{G}_B := \left\{X_k, \bar{X}_k, \check{X}_k\right\}_{k \in [n]}.$$

– Output $\mathsf{ck} := \left(\mathbf{A}, \mathbf{B}, \boldsymbol{t}, \boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}, (\boldsymbol{u}_g)_{g \in \mathcal{G}_A}, (\boldsymbol{w}_g)_{g \in \mathcal{G}_B}\right)$.

$\underline{\mathsf{Com}(\mathsf{ck}, \boldsymbol{x})}$

– Compute $c := \langle \boldsymbol{v}, \boldsymbol{x} \rangle \bmod q$.
– Output $\mathsf{com} = c$ and $\mathsf{aux} = \boldsymbol{x}$.

$\underline{\mathsf{Open}(\mathsf{ck}, (\mathsf{aux}_h)_{h \in [m]}, f)}$

– Parse aux_h as $\boldsymbol{x}_h$ for all $h \in [m]$ and let $\boldsymbol{y} := f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$.
– Compute $\boldsymbol{v}_1 := \mathsf{vec}(\mathbf{Z}_{\boldsymbol{v}}) \otimes \bar{\boldsymbol{v}}$ and $\boldsymbol{v}_2 := \mathsf{vec}((\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \otimes (\mathbf{I} + \mathbf{Z}_{\check{\boldsymbol{v}}}) - \mathbf{I}) \otimes \bar{\boldsymbol{v}}$.
– Pack the preimages vectors given in the public parameters as columns of the following matrices:
  - $\mathbf{U}_i$ such that $\mathbf{A} \cdot \mathbf{U}_i = \boldsymbol{t} \cdot \boldsymbol{v}_i^\mathtt{T} \bmod q$ for $i \in [2]$.
    For example, for $i = 1$, the first few columns of the R.H.S. of the equation are of the form

$$\boldsymbol{t} \cdot \boldsymbol{v}_1^\mathtt{T} = \boldsymbol{t} \cdot \begin{pmatrix} 0 & \frac{v_1}{v_2} \cdot \bar{v}_1 & \frac{v_1}{v_3} \cdot \bar{v}_1 & \ldots \end{pmatrix}.$$

    Notice that each column is either $\mathbf{0} \in \mathcal{R}_q^\eta$, for which $\mathbf{0} \in \mathcal{R}^\ell$ is a trivial preimage, or of the form $\boldsymbol{t} \cdot \frac{v_{i'}}{v_i} \cdot \bar{v}_k$ for some $i, i', k \in [n]$ with $i \neq i'$, for which a preimage is given in ck.
  - $\bar{\mathbf{U}}$ such that $\mathbf{A} \cdot \bar{\mathbf{U}} = \boldsymbol{t} \cdot \boldsymbol{v}^\mathtt{T} \cdot \mathbf{Z}_{\bar{\boldsymbol{v}}} \bmod q$.
  - $\check{\mathbf{U}}$ such that $\mathbf{A} \cdot \check{\mathbf{U}} = \boldsymbol{t} \cdot \check{\boldsymbol{v}}^\mathtt{T} \cdot \mathbf{Z}_{\boldsymbol{v}} \bmod q$.
  - $\mathbf{W}$ such that $\mathbf{B} \cdot \mathbf{W} = \boldsymbol{t} \cdot \boldsymbol{v}^\mathtt{T} \bmod q$.
  - $\bar{\mathbf{W}}$ such that $\mathbf{B} \cdot \bar{\mathbf{W}} = \boldsymbol{t} \cdot \bar{\boldsymbol{v}}^\mathtt{T} \bmod q$.
  - $\check{\mathbf{W}}$ such that $\mathbf{B} \cdot \check{\mathbf{W}} = \boldsymbol{t} \cdot \check{\boldsymbol{v}}^\mathtt{T} \bmod q$.
– Compute $\boldsymbol{u} := \sum_{h \in \mathcal{S}_1(f)} \mathbf{U}_1 \cdot \mathsf{vec}(\boldsymbol{x}_h^\mathtt{T} \otimes \mathbf{F}_h) + \sum_{(h,h') \in \mathcal{S}_2^\otimes(f)} \mathbf{U}_2 \cdot \mathsf{vec}((\boldsymbol{x}_h^\mathtt{T} \otimes \boldsymbol{x}_{h'}^\mathtt{T}) \otimes \mathbf{G}_{h,h'})$.
– Compute $\boldsymbol{w}_0 := \mathbf{W} \cdot \boldsymbol{y}$.
– Compute $\bar{\boldsymbol{u}}_0 := \bar{\mathbf{U}} \cdot \boldsymbol{y}$ and $\bar{\boldsymbol{w}}_0 := \bar{\mathbf{W}} \cdot \boldsymbol{y}$.
– Compute $\check{\boldsymbol{u}}_h := \check{\mathbf{U}} \cdot \boldsymbol{x}_h$ and $\check{\boldsymbol{w}}_h := \check{\mathbf{W}} \cdot \boldsymbol{x}_h$ for $h \in \mathcal{S}_2(f)$.
– Output $(\boldsymbol{u}, \boldsymbol{w}_0, \bar{\boldsymbol{u}}_0, \bar{\boldsymbol{w}}_0, (\check{\boldsymbol{u}}_h, \check{\boldsymbol{w}}_h)_{h \in \mathcal{S}_2(f)})$.

$\underline{\mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_h)_{h \in [m]}, \mathsf{com}_0, f, \pi)}$

– Define $\hat{f}(C_1, \ldots, C_m, \check{C}_1, \ldots, \check{C}_m)$

$$:= \bar{\boldsymbol{v}}^\mathtt{T} \cdot \left(\sum_{(h,h') \in \mathcal{S}_2(f)} \mathbf{G}_{h,h'} \cdot (\boldsymbol{v}^\dagger \otimes \check{\boldsymbol{v}}^\dagger) \cdot C_h \cdot \check{C}_{h'} + \sum_{h \in \mathcal{S}_1(f)} \mathbf{F}_h \cdot \boldsymbol{v}^\dagger \cdot C_h + \boldsymbol{e}^\mathtt{T} \right).$$

- Check if $\|\boldsymbol{w}_0\| \leq \beta^*$ and $\|\bar{\boldsymbol{w}}_0\| \leq \beta^*$.
- For $h \in [m] \setminus \mathcal{S}_2(f)$, set $\check{c}_h = 0$ and check if $\|\check{\boldsymbol{w}}_h\| \leq \beta^*$.
- Check if $\mathbf{B} \cdot \boldsymbol{w}_0 = \boldsymbol{t} \cdot c_0 \bmod q$.
- Check if there exists (unique) $\bar{c}_0$ such that $\mathbf{B} \cdot \bar{\boldsymbol{w}}_0 = \boldsymbol{t} \cdot \bar{c}_0 \bmod q$.
- Check if there exists (unique) $\check{c}_h$ such that $\mathbf{B} \cdot \check{\boldsymbol{w}}_h = \boldsymbol{t} \cdot \check{c}_h \bmod q$ for $h \in \mathcal{S}_2(f)$.
- Check if $\mathbf{A} \cdot \boldsymbol{u} = \boldsymbol{t} \cdot (\hat{f}(c_1, \ldots, c_m, \check{c}_1, \ldots, \check{c}_m) - \bar{c}_0) \bmod q$ and $\|\boldsymbol{u}\| \leq \beta^*$.
- Check if $\mathbf{A} \cdot \bar{\boldsymbol{u}}_0 = \boldsymbol{t} \cdot (\boldsymbol{v}^{\mathtt{T}} \cdot \bar{\boldsymbol{v}}^{\dagger} \cdot \bar{c}_0 - c_0) \bmod q$ and $\|\bar{\boldsymbol{u}}_0\| \leq \beta^*$.
- Check if $\mathbf{A} \cdot \check{\boldsymbol{u}}_h = \boldsymbol{t} \cdot (\check{\boldsymbol{v}}^{\mathtt{T}} \cdot \boldsymbol{v}^{\dagger} \cdot c_h - \check{c}_h) \bmod q$ and $\|\check{\boldsymbol{u}}_h\| \leq \beta^*$ for $h \in \mathcal{S}_2(f)$.
- Accept, i.e. output 1, if all checks pass. Otherwise, output 0.

Theorem 4. *Let $\ell \geq \mathsf{lhl}(\mathcal{R}, \eta, q, \beta)$, $\beta^* \geq 2 \cdot n^4 \cdot \hat{m}^2 \cdot \alpha^3 \cdot \beta \cdot \gamma_{\mathcal{R}}^3$, and $\mathcal{D} = \mathsf{SampD}(\mathcal{R}, 1^\eta, 1^\ell, q, \beta)$, and assume that the twin-k-R-$\mathsf{ISIS}_{\mathcal{R}_q,\eta,\ell,n,\beta,\beta^*,\mathcal{G}_A,\mathcal{G}_B,\mathcal{D}}$ assumption holds. Then, the construction* CFC *described above is an evaluation binding CFC for the class $\mathcal{F}_{\mathsf{quad}}$ of quadratic functions over any $m \leq \hat{m}$ vectors of length $\leq n$, has efficient verification, and is (almost) additively homomorphic. For a function $f \in \mathcal{F}_{\mathsf{quad}}$, the proof size of* CFC *is $|\pi| = |\mathcal{S}_2(f)| \cdot \log^2(m \cdot n) \cdot \mathsf{poly}(\lambda)$, and for the class $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$, our* CFC *is $s(n, m, \kappa)$-succinct where $s(n, m, \kappa) = \kappa \cdot \log^2(m \cdot n)$. Furthermore, by setting $\hat{m} = \lambda^{\omega(1)}$ the CFC supports quadratic functions over any $m = \mathsf{poly}(\lambda)$ vectors and is $\kappa \cdot \log^2(n)$-succinct.*

In the following sections we prove the theorem.

7.4 Correctness

To prove correctness, we first state a claim which abstracts away most of the tedious calculations. The claim is proven in Appendix D.

Claim. Let $f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) = \boldsymbol{y}$. For $h \in \mathcal{S}(f)$, let $c_h = \langle \boldsymbol{v}, \boldsymbol{x}_h \rangle \bmod q$. For $h \in \mathcal{S}_2(f)$, let $\check{c}_h = \langle \check{\boldsymbol{v}}, \boldsymbol{x}_h \rangle \bmod q$. For $h \in [m] \setminus \mathcal{S}_2(f)$, let $\check{c}_h = 0$. Let $c_0 = \langle \boldsymbol{v}, \boldsymbol{y} \rangle \bmod q$ and $\bar{c}_0 = \langle \bar{\boldsymbol{v}}, \boldsymbol{y} \rangle \bmod q$. Let $\boldsymbol{v}_2 = \mathsf{vec}((\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \otimes (\mathbf{I} + \mathbf{Z}_{\check{\boldsymbol{v}}}) - \mathbf{I}) \otimes \bar{\boldsymbol{v}}$ and $\boldsymbol{v}_1 = \mathsf{vec}(\mathbf{Z}_{\boldsymbol{v}}) \otimes \bar{\boldsymbol{v}}$. It holds that

$$\hat{f}(c_1, \ldots, c_m, \check{c}_1, \ldots, \check{c}_m) - \bar{c}_0 = \sum_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} \boldsymbol{v}_2^{\mathtt{T}} \cdot \mathsf{vec}(\boldsymbol{x}_h^{\mathtt{T}} \otimes \boldsymbol{x}_{h'}^{\mathtt{T}} \otimes \mathbf{G}_{h,h'}) + \sum_{h \in \mathcal{S}_1(f)} \boldsymbol{v}_1^{\mathtt{T}} \cdot \mathsf{vec}(\boldsymbol{x}_h^{\mathtt{T}} \otimes \mathbf{F}_h),$$

$$\boldsymbol{v}^{\mathtt{T}} \cdot \bar{\boldsymbol{v}}^{\dagger} \cdot \bar{c}_0 - c_0 = \boldsymbol{v}^{\mathtt{T}} \cdot \mathbf{Z}_{\bar{\boldsymbol{v}}} \cdot \boldsymbol{y}, \text{ and}$$

$$\check{\boldsymbol{v}}^{\mathtt{T}} \cdot \boldsymbol{v}^{\dagger} \cdot c_h - \check{c}_h = \check{\boldsymbol{v}}^{\mathtt{T}} \cdot \mathbf{Z}_{\boldsymbol{v}} \cdot \boldsymbol{y} \text{ for all } h \in \mathcal{S}_2(f).$$

Recall that

$$\boldsymbol{u} = \sum_{(h,h') \in \mathcal{S}_2^{\otimes}(f)} \mathbf{U}_2 \cdot \mathsf{vec}(\boldsymbol{x}_h^{\mathtt{T}} \otimes \boldsymbol{x}_{h'}^{\mathtt{T}} \otimes \mathbf{G}_{h,h'}) + \sum_{h \in \mathcal{S}_1(f)} \mathbf{U}_1 \cdot \mathsf{vec}(\boldsymbol{x}_h^{\mathtt{T}} \otimes \mathbf{F}_h),$$

$$\bar{\boldsymbol{u}}_0 = \bar{\mathbf{U}} \cdot \boldsymbol{y}, \text{ and}$$

$$\check{\boldsymbol{u}}_h = \check{\mathbf{U}} \cdot \boldsymbol{x}_h \text{ for } h \in \mathcal{S}_2(f)$$

are computed using $(\mathbf{U}_2, \mathbf{U}_1, \bar{\mathbf{U}}, \check{\mathbf{U}})$ satisfying

$$\mathbf{A} \cdot \mathbf{U}_h = \boldsymbol{t} \cdot \boldsymbol{v}_h^{\mathtt{T}} \bmod q,$$

$$\mathbf{A} \cdot \bar{\mathbf{U}} = \boldsymbol{t} \cdot \bar{\boldsymbol{v}}^{\mathtt{T}} \cdot \mathbf{Z}_{\boldsymbol{v}} \bmod q, \text{ and}$$

$$\mathbf{A} \cdot \check{\mathbf{U}} = \boldsymbol{t} \cdot \bar{\boldsymbol{v}}^{\mathtt{T}} \cdot \mathbf{Z}_{\boldsymbol{v}} \bmod q.$$

It follows that

$$
\begin{aligned}
\mathbf{A} \cdot \boldsymbol{u} &= \boldsymbol{t} \cdot (\hat{f}(c_1, \ldots, c_m, \check{c}_1, \ldots, \check{c}_m) - \bar{c}_0) \bmod q, \\
\mathbf{A} \cdot \bar{\boldsymbol{u}}_0 &= \boldsymbol{t} \cdot (\boldsymbol{v}^{\mathsf{T}} \cdot \bar{\boldsymbol{v}}^{\dagger} \cdot \bar{c}_0 - c_0) \bmod q, \text{ and} \\
\mathbf{A} \cdot \check{\boldsymbol{u}}_h &= \boldsymbol{t} \cdot (\check{\boldsymbol{v}}^{\mathsf{T}} \cdot \boldsymbol{v}^{\dagger} \cdot c_h - \check{c}_h) \bmod q \text{ for all } h \in \mathcal{S}_2(f).
\end{aligned}
$$

It remains to analyse the norms of the preimages. The norms of $\boldsymbol{w}_0$, $\bar{\boldsymbol{w}}_0$, $(\check{\boldsymbol{w}}_h)_{h \in \mathcal{S}_2(f)}$ are easy to verify. By the properties discussed in Section 7.1, each column in the matrices $\mathbf{U}_2$, $\mathbf{U}_1$, $\bar{\mathbf{U}}$, and $\check{\mathbf{U}}$ has norm as most β. By our choice of parameters, each entry in $\mathbf{G}_{h,h'}$, $\mathbf{F}_h$, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ and $\boldsymbol{y}$ has norm at most α. It follows that

$$
\begin{aligned}
\|\boldsymbol{u}\| &\le n^4 \cdot \mathcal{S}_2^{\otimes}(f) \cdot \alpha^3 \cdot \beta \cdot \gamma_{\mathcal{R}}^3 + n^3 \cdot \mathcal{S}_1(f) \cdot \alpha^2 \cdot \beta \cdot \gamma_{\mathcal{R}}^2 < \beta^*, \\
\|\bar{\boldsymbol{u}}\|_0 &\le n \cdot \alpha \cdot \beta \cdot \gamma_{\mathcal{R}} < \beta^*, \text{ and} \\
\|\check{\boldsymbol{u}}\|_h &\le n \cdot \alpha \cdot \beta \cdot \gamma_{\mathcal{R}} < \beta^* \ \forall\ h \in \mathcal{S}_2(f).
\end{aligned}
$$

Additive homomorphism. As is common in the lattice setting, our construction is almost additively homomorphic in the following sense: Although the commitment function $\boldsymbol{x} \mapsto \langle \boldsymbol{v}, \boldsymbol{x} \rangle \bmod q$ is a linear function, the bounded-norm restriction on messages could be violated since $\|\boldsymbol{x}\| \le \alpha$ and $\|\boldsymbol{x}'\| \le \alpha$ in general do not imply $\|\boldsymbol{x} + \boldsymbol{x}'\| \le \alpha$. As such, correctness is only guaranteed after homomorphic evaluation if $\|\boldsymbol{x} + \boldsymbol{x}'\| \le \alpha$.

7.5 Succinctness

We measure the succinctness of our construction. A commitment consists of a single $\mathcal{R}_q$ element. An opening proof consists of $2\,\mathcal{S}_2(f) + 3$ vectors in $\mathcal{R}^\ell$ each of norm at most β^*. Setting $\ell = \mathsf{lhl}(\mathcal{R}, \eta, q, \beta) = \log q \cdot \mathsf{poly}(\lambda)$ for the guarantees of lattice trapdoor algorithms, $\beta^* = 2 \cdot n^4 \cdot m^2 \cdot \alpha^3 \cdot \beta \cdot \gamma_{\mathcal{R}}^3 = n^4 \cdot m^2 \cdot \mathsf{poly}(\lambda)$ so that correctness holds, and $q = \beta^* \cdot \mathsf{poly}(\lambda)$ to be large enough so that the Twin-k-R-ISIS assumption plausibly holds, a commitment can be described with $\log q \cdot \mathsf{poly}(\lambda) = (\log n + \log m) \cdot \mathsf{poly}(\lambda)$ bits, while an opening proof for a function $f \in \mathcal{F}_{\mathsf{quad}}$ can be described with $(2\,|\mathcal{S}_2(f)| + 3) \cdot \ell \cdot \log \beta^* \cdot \mathsf{poly}(\lambda) = |\mathcal{S}_2(f)| \cdot \log^2(m \cdot n) \cdot \mathsf{poly}(\lambda)$ bits. Note that for $f \in \mathcal{F}_{\mathsf{level}}$, then $|\mathcal{S}_2(f)| = |\mathcal{S}_2^{\otimes}(f)| = \kappa$. Hence, our CFC is $s(n, m, \kappa)$-succinct for the class $\mathcal{F}_{\mathsf{level}} = \{\mathcal{F}_{\mathsf{level},\kappa}\}$, where $s(n, m, \kappa) = \kappa \cdot \log^2(m \cdot n)$.

Remark 1 (Removing the dependence on m). According to the choice of parameters above, commitments and openings have a logarithmic dependence on the number of inputs m (in addition to the input length n). More importantly, for correctness to hold, one should fix q depending on the largest m to be supported. This is a limitation, especially when plugging this CFC in the FC transformation as there m is in the worst case the depth of the circuit. However, since the dependence is only logarithmic we can actually set $\beta^* = 2 \cdot n^4 \cdot \hat{m}^2 \cdot \alpha^3 \cdot \beta \cdot \gamma_{\mathcal{R}}^3$ where $\hat{m} = \lambda^{\omega(1)}$ is superpolynomial in the security parameter, in such a way that correctness holds for any $m = \mathsf{poly}(\lambda)$. This change makes $q = \lambda^{\omega(1)}$ (a choice that does not affect the plausibility of the assumption according to the analysis of [ACL+22]) and makes the CFC scheme $\kappa \cdot \log^2(n)$-succinct.

7.6 Resulting Instantiations of FC for Circuits

As in the previous section, we summarize the concrete FC schemes that result from instantiating our generic construction of Section 5 with our lattice-based CFC.

Corollary 2. *Assume that all the conditions of Theorem 4 are satisfied. Then the following statements hold:*

1. *There exists an FC scheme for the class $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}$ of arithmetic circuits of width w bounded by $\leq n$ and in-degree bounded by $\leq t_{\max}$ that is $O(d \cdot \log^2(t_{\max} \cdot n))$-succinct.*
2. *Using the choice of parameters of Remark 1, there exists an FC scheme for $\mathcal{F}_n = \{\mathcal{F}_{(d,t,w)}\}$ of width $w \leq n$ that is $\mathcal{O}(d)$-succinct.*
3. *For any $w_0 \geq 2$, there exists an FC scheme for the class $\mathcal{F} = \{\mathcal{F}_{(d,t,w)}\}$ of circuits of arbitrary width $w > w_0$ that is $\mathcal{O}\big(d \cdot (w/w_0)^2\big)$-succinct.*

Case (1) follows by observing that in the FC construction from CFCs the number of CFC inputs is bounded by the in-degree of the admissible circuits. In case (1) we fix a concrete $m = t_{\max}$ in the choice of $q = \beta^* \cdot \mathsf{poly}(\lambda)$ while in points (2)–(3) we consider the parameters choice of Remark 1 that let us support any in-degree $t = \mathsf{poly}(\lambda)$.

As opposed to our pairing-based construction, the linear dependency on the depth does not follow from a black-box application of our FC from CFC construction. In fact, Theorem 2 gives a proof size of $\mathcal{O}\big(d \cdot t \cdot \log^2(t_{\max} \cdot n)\big)$. We can supress the t factor by noticing that, for each circuit layer h, the *same* vectors $(\check{\boldsymbol{u}}_h, \check{\boldsymbol{w}}_h)$ are included in the openings at every layer h' such that $h \in \mathcal{S}_2(f^{(h')})$. The result follows by including them only once in the FC opening proof.

We observe that the resulting lattice-based FC schemes yield shorter proofs (with respect to circuit depth) than their pairing-based counterparts. This feature can be seen as a natural consequence of the additional capability to perform computations over encrypted (in this case, committed) data that lattices provide. Indeed, in our pairing-based construction, the prover needs to provide $\mathcal{O}(d \cdot t)$ commitments $X_{h,h'}$ to the tensor product of every pair of layers in the circuit. This is avoided in our lattice-based scheme, as the verifier can multiply commitments $C_h \cdot \check{C}_{h'}$ by herself.

7.7 Proof of Security

Suppose there exists a PPT adversary $\mathcal{A}$ against evaluation binding of the CFC construction, we construct a PPT algorithm $\mathcal{B}$ for the Twin-k-R-ISIS problem as follows. Given a Twin-k-R-ISIS instance ck, $\mathcal{B}$ passes ck to $\mathcal{A}$. The adversary $\mathcal{A}$ returns input commitments $(c_h)_{h\in[m]}$, a quadratic function f, two output commitments c_0 and c_0', and two opening proofs π and π', where $\pi = (\boldsymbol{u}, \boldsymbol{w}_0, \bar{\boldsymbol{u}}_0, \bar{\boldsymbol{w}}_0, (\check{\boldsymbol{u}}_h, \check{\boldsymbol{w}}_h)_{h\in\mathcal{S}(f)})$ and $\pi' = (\boldsymbol{u}', \boldsymbol{w}_0', \bar{\boldsymbol{u}}_0', \bar{\boldsymbol{w}}_0', (\check{\boldsymbol{u}}_h', \check{\boldsymbol{w}}_h')_{h\in\mathcal{S}_2(f)})$. By our assumption on $\mathcal{A}$, with non-negligible probability, π (and analogously π') satisfies

$$\begin{aligned}
\mathbf{A} \cdot \boldsymbol{u} &= \boldsymbol{t} \cdot (\hat{f}(c_1, \ldots, c_m, \check{c}_1, \ldots, \check{c}_m) - \bar{c}_0) \bmod q, \\
\mathbf{A} \cdot \bar{\boldsymbol{u}}_0 &= \boldsymbol{t} \cdot (\boldsymbol{v}^{\mathtt{T}} \cdot \bar{\boldsymbol{v}}^{\dagger} \cdot \bar{c}_0 - c_0) \bmod q, \text{ and} \\
\mathbf{A} \cdot \check{\boldsymbol{u}}_h &= \boldsymbol{t} \cdot (\check{\boldsymbol{v}}^{\mathtt{T}} \cdot \boldsymbol{v}^{\dagger} \cdot c_h - \check{c}_h) \bmod q \text{ for all } h \in \mathcal{S}_2(f),
\end{aligned}$$

where $\mathbf{B} \cdot \bar{\boldsymbol{w}}_0 = \boldsymbol{t} \cdot \bar{c}_0 \bmod q$ and $\mathbf{B} \cdot \check{\boldsymbol{w}}_h = \boldsymbol{t} \cdot \check{c}_h \bmod q$.

For any $h \in \mathcal{S}_2(f)$, suppose $\check{\boldsymbol{w}}_h \neq \check{\boldsymbol{w}}_h'$, then from the third equation $(\check{\boldsymbol{u}}_h - \check{\boldsymbol{u}}_h', \check{\boldsymbol{w}}_h - \check{\boldsymbol{w}}_h')$ would be a non-zero vector of norm at most $2\beta^*$ satisfying $\mathbf{A} \cdot (\check{\boldsymbol{u}}_h - \check{\boldsymbol{u}}_h') + \mathbf{B} \cdot (\check{\boldsymbol{w}}_h - \check{\boldsymbol{w}}_h') = \mathbf{0} \bmod q$, contradicting the twin-k-R-ISIS assumption. We therefore have $\check{\boldsymbol{w}}_h = \check{\boldsymbol{w}}_h'$ and hence $\check{c}_h = \check{c}_h'$ for all $h \in \mathcal{S}_2(f)$.

Next, suppose $\bar{\boldsymbol{w}}_0 \neq \bar{\boldsymbol{w}}_0'$, then from the first equation $(\boldsymbol{u} - \boldsymbol{u}', \bar{\boldsymbol{w}}_0 - \bar{\boldsymbol{w}}_0')$ would be a non-zero vector of norm at most $2\beta^*$ satisfying $\mathbf{A} \cdot (\boldsymbol{u} - \boldsymbol{u}') + \mathbf{B} \cdot (\bar{\boldsymbol{w}}_0 - \bar{\boldsymbol{w}}_0') = \mathbf{0} \bmod q$, contradicting the twin-k-R-ISIS assumption. We therefore have $\bar{\boldsymbol{w}}_0 = \bar{\boldsymbol{w}}_0'$ and hence $\bar{c}_0 = \bar{c}_0'$.

Finally, suppose $\boldsymbol{w}_0 \neq \boldsymbol{w}_0'$, then from the second equation $(\bar{\boldsymbol{u}}_0 - \bar{\boldsymbol{u}}_0', \boldsymbol{w}_0 - \boldsymbol{w}_0')$ would be a non-zero vector of norm at most $2\beta^*$ satisfying $\mathbf{A} \cdot (\bar{\boldsymbol{u}}_0 - \bar{\boldsymbol{u}}_0') + \mathbf{B} \cdot (\boldsymbol{w}_0 - \boldsymbol{w}_0') = \mathbf{0} \bmod q$, contradicting the twin-k-R-ISIS assumption. We therefore have $\boldsymbol{w}_0 = \boldsymbol{w}_0'$ and hence $c_0 = c_0'$, meaning that $\mathcal{A}$ cannot be a successful adversary against evaluation binding.

7.8 Efficient Verification

Our CFC construction also supports amortized efficient verification. We observe that in our construction the Vf algorithm can be split into an offline preprocessing step and an online verification step:

- $\mathsf{VerPrep}(\mathsf{ck}, f)$: Compute the polynomials $\hat{f}$, $\bar{\mathsf{id}}$, and $\check{\mathsf{id}}$, and output $\mathsf{vk}_f \coloneqq (\mathbf{A}, \mathbf{B}, \boldsymbol{t}, \hat{f}, \bar{\mathsf{id}}, \check{\mathsf{id}})$.
- $\mathsf{EffVer}(\mathsf{vk}_f, (\mathsf{com}_h)_{h \in [m]}, \mathsf{com}_0, \pi)$: Perform all the checks described in Vf.

Clearly, the runtime of EffVer is $(\mathcal{S}_2^{\otimes}(f) + \mathcal{S}_1(f)) \cdot \log q \cdot \mathsf{poly} \leq m^2 \cdot \log(m \cdot n) \cdot \mathsf{poly}(\lambda)$, which is logarithmic in n.

7.9 Commitment Hiding

Commitment hiding can be achieved by extending the dimension of the input vector and dedicating some entries for commitment randomness. We outline such a transformation in the following.

First, we modify the setup so that the vectors $\boldsymbol{v}, \bar{\boldsymbol{v}}, \check{\boldsymbol{v}}$ are now sampled from $\mathcal{R}_q^{n+\ell}$. The sets $\mathcal{G}_A$ and $\mathcal{G}_B$ of monomials are adjusted accordingly. To commit to $\boldsymbol{x} \in \mathcal{R}^n$, sample a uniformly random vector $\boldsymbol{r} \leftarrow\$ \mathcal{R}^\ell$ with $\|\boldsymbol{r}\| \leq \alpha$, and compute $c \coloneqq \left\langle \boldsymbol{v}, \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{r} \end{pmatrix} \right\rangle \bmod q$. Opening and verifying are almost identical as in the base scheme, except that f is treated as a polynomial on $(\boldsymbol{x}_1, \boldsymbol{r}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{r}_m)$ but with zero coefficients for all terms involving any entry of $(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m)$. It can be verified that the modified scheme retains correctness and evaluation binding. For $\ell \geq \mathsf{lhl}(\mathcal{R}, \eta, q, \beta)$, which we anyway need for correctness, commitment hiding is immediate from the leftover hash lemma.

To make the verification more friendly to zero-knowledge arguments, we need to make one more minor change to the scheme: The opening algorithm additionally includes the commitments $(\bar{c}_0, (\check{c}_h)_{h \in \mathcal{S}_2(f)})$ in an opening proof. This makes the verification NIZK-friendly, since it boils down to proving the following SIS relations in zero-knowledge: There exists $(\boldsymbol{u}, \boldsymbol{w}_0, \bar{\boldsymbol{u}}_0, \bar{\boldsymbol{w}}_0, (\check{\boldsymbol{u}}_h, \check{\boldsymbol{w}}_h)_{h \in \mathcal{S}_2(f)}) \in (\mathcal{R}^\ell)^{2\mathcal{S}_2(f)+3}$ such that

$$\begin{cases} \mathbf{A} \cdot \boldsymbol{u} = \boldsymbol{t} \cdot (\hat{f}(c_1, \ldots, c_m, \check{c}_1, \ldots, \check{c}_m) - \bar{c}_0) \bmod q & \wedge \ \|\boldsymbol{u}\| \leq \beta^* \\ \mathbf{A} \cdot \bar{\boldsymbol{u}}_0 = \boldsymbol{t} \cdot (\boldsymbol{v}^{\mathtt{T}} \cdot \bar{\boldsymbol{v}}^{\dagger} \cdot \bar{c}_0 - c_0) \bmod q & \wedge \ \|\bar{\boldsymbol{u}}_0\| \leq \beta^* \\ \mathbf{A} \cdot \check{\boldsymbol{u}}_h = \boldsymbol{t} \cdot (\check{\boldsymbol{v}}^{\mathtt{T}} \cdot \boldsymbol{v}^{\dagger} \cdot c_h - \check{c}_h) \bmod q & \wedge \ \|\check{\boldsymbol{u}}_h\| \leq \beta^* \quad \forall\, h \in \mathcal{S}_2(f) \\ \mathbf{B} \cdot \boldsymbol{w}_0 = \boldsymbol{t} \cdot c_0 \bmod q & \wedge \ \|\boldsymbol{w}_0\| \leq \beta^* \\ \mathbf{B} \cdot \bar{\boldsymbol{w}}_0 = \boldsymbol{t} \cdot \bar{c}_0 \bmod q & \wedge \ \|\bar{\boldsymbol{w}}_0\| \leq \beta^* \\ \mathbf{B} \cdot \check{\boldsymbol{w}}_h = \boldsymbol{t} \cdot \check{c}_h \bmod q & \wedge \ \|\check{\boldsymbol{w}}_h\| \leq \beta^* \quad \forall\, h \in \mathcal{S}_2(f). \end{cases}$$

By slightly adjusting the parameters of the k-R-ISIS assumption, the scheme remains evaluation binding even if the NIZK argument can only guarantee that the norm of the witness is bounded by some $\beta^{**} > \beta^*$ (although the prover has a witness of norm bounded by β^*). This allows to use efficient NIZK (e.g. [Lyu09]) for proving SIS relations with relaxed soundness.

8 Conclusions

In this work, we present the first constructions of functional commitments for circuits based on falsifiable assumptions. Our results leave some open questions for future work. The first one concerns the current need of fixing a bound on the maximal width of the circuits at setup time. Constructing an FC whose setup procedure only depends on the input size, or ideally on no bound, would be a remarkable result that would also imply fully-homomorphic signatures. Another interesting direction is to construct functional commitments with more succinct opening proofs, e.g., sublinear in the circuit depth. Finally, we believe that there is room for improvement towards the design of FC schemes that rely on simpler, or more standard, cryptographic assumptions.

Acknowledgements

This work is supported by the PICOCRYPT project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101001283), partially supported by PRODIGY Project (TED2021-132464B-I00) funded by MCIN/AEI/10.13039/501100011033/ and the European Union NextGenerationEU / PRTR, and partially funded by Ministerio de Universidades (FPU21/00600). This research has been supported in part by the Programma ricerca di ateneo UNICT 35 2020-22 linea 2 and by research gifts from Protocol Labs.

References

ACL+22. M. R. Albrecht, V. Cini, R. W. F. Lai, G. Malavolta, and S. A. K. Thyagarajan. Lattice-Based SNARKs: Publicly Verifiable, Preprocessing, and Recursively Composable - (Extended Abstract). In *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 102–132. Springer, Heidelberg, August 2022.

AL21. M. R. Albrecht and R. W. F. Lai. Subtractive Sets over Cyclotomic Rings - Limits of Schnorr-Like Arguments over Lattices. In *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 519–548, Virtual Event, August 2021. Springer, Heidelberg.

BF11. D. Boneh and D. M. Freeman. Homomorphic Signatures for Polynomial Functions. In *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 149–168. Springer, Heidelberg, May 2011.

CF13. D. Catalano and D. Fiore. Vector Commitments and Their Applications. In *PKC 2013*, volume 7778 of *LNCS*, pages 55–72. Springer, Heidelberg, February / March 2013.

CFM08. D. Catalano, D. Fiore, and M. Messina. Zero-Knowledge Sets with Short Proofs. In *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 433–450. Springer, Heidelberg, April 2008.

CFT22. D. Catalano, D. Fiore, and I. Tucker. Additive-Homomorphic Functional Commitments and Applications to Homomorphic Signatures. In *ASIACRYPT 2022*, 2022. To appear.

CFW14. D. Catalano, D. Fiore, and B. Warinschi. Homomorphic Signatures with Efficient Verification for Polynomial Functions. In *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 371–389. Springer, Heidelberg, August 2014.

dCP23. L. de Castro and C. Peikert. Functional Commitments for All Functions, with Transparent Setup. In *EUROCRYPT 2023*, 2023. To appear.

EHK+13. A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An Algebraic Framework for Diffie-Hellman Assumptions. In *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013.

GKR08. S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: interactive proofs for muggles. In *40th ACM STOC*, pages 113–122. ACM Press, May 2008.

GM18. N. Genise and D. Micciancio. Faster Gaussian Sampling for Trapdoor Lattices with Arbitrary Modulus. In *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 174–203. Springer, Heidelberg, April / May 2018.

GPV08. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

GR19. A. González and C. Ràfols. Shorter Pairing-Based Arguments Under Standard Assumptions. In *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 728–757. Springer, Heidelberg, December 2019.

GS08. J. Groth and A. Sahai. Efficient Non-interactive Proof Systems for Bilinear Groups. In *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg, April 2008.

GSW13. C. Gentry, A. Sahai, and B. Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

GVW15. S. Gorbunov, V. Vaikuntanathan, and D. Wichs. Leveled Fully Homomorphic Signatures from Standard Lattices. In *47th ACM STOC*, pages 469–477. ACM Press, June 2015.

GW11. C. Gentry and D. Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *43rd ACM STOC*, pages 99–108. ACM Press, June 2011.

GZ21. A. González and A. Zacharakis. Fully-Succinct Publicly Verifiable Delegation from Constant-Size Assumptions. In *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 529–557. Springer, Heidelberg, November 2021.

JMSW02. R. Johnson, D. Molnar, D. X. Song, and D. Wagner. Homomorphic Signature Schemes. In *CT-RSA 2002*, volume 2271 of *LNCS*, pages 244–262. Springer, Heidelberg, February 2002.

KNYY19. S. Katsumata, R. Nishimaki, S. Yamada, and T. Yamakawa. Designated Verifier/Prover and Preprocessing NIZKs from Diffie-Hellman Assumptions. In *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 622–651. Springer, Heidelberg, May 2019.

KZG10. A. Kate, G. M. Zaverucha, and I. Goldberg. Constant-Size Commitments to Polynomials and Their Applications. In *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194. Springer, Heidelberg, December 2010.

LM19. R. W. F. Lai and G. Malavolta. Subvector Commitments with Application to Succinct Arguments. In *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 530–560. Springer, Heidelberg, August 2019.

LP20. H. Lipmaa and K. Pavlyk. Succinct Functional Commitment for a Large Class of Arithmetic Circuits. In *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 686–716. Springer, Heidelberg, December 2020.

LRY16. B. Libert, S. C. Ramanna, and M. Yung. Functional Commitment Schemes: From Polynomial Commitments to Pairing-Based Accumulators from Simple Assumptions. In *ICALP 2016*, volume 55 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl, July 2016.

LY10. B. Libert and M. Yung. Concise Mercurial Vector Commitments and Independent Zero-Knowledge Sets with Short Proofs. In *TCC 2010*, volume 5978 of *LNCS*, pages 499–517. Springer, Heidelberg, February 2010.

Lyu09. V. Lyubashevsky. Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures. In *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

MP12. D. Micciancio and C. Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012.

MRV16. P. Morillo, C. Ràfols, and J. L. Villar. The Kernel Matrix Diffie-Hellman Assumption. In *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 729–758. Springer, Heidelberg, December 2016.

PPS21. C. Peikert, Z. Pepin, and C. Sharp. Vector and Functional Commitments from Lattices. In *TCC 2021, Part III*, volume 13044 of *LNCS*, pages 480–511. Springer, Heidelberg, November 2021.

WW23. H. Wee and D. J. Wu. Succinct Vector, Polynomial, and Functional Commitments from Lattices. In *EUROCRYPT 2023*, 2023. To appear.

A More on FCs and CFCs security notions

A.1 Proof of Theorem 1 (zero-knowledge FC)

Let $\mathit{\Pi} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Sim})$ be a NIZK for the relation $R_{\mathsf{FC}} = \{((\mathsf{ck}, C, f, \boldsymbol{y}); \pi) : \mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi) = 1\}$. Then, we can construct a FC scheme FC^* that satisfies com-hiding and zero knowledge openings as follows:

– $\mathsf{FC}^*.\mathsf{Setup}(1^\lambda)$ runs $\mathsf{ck} \leftarrow \mathsf{FC}.\mathsf{Setup}(1^\lambda)$ and $\mathsf{crs} \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$, and outputs $(\mathsf{ck}, \mathsf{crs})$.

– $\mathsf{FC}^*.\mathsf{Com}((\mathsf{ck},\mathsf{crs}),\boldsymbol{x};\mathsf{r})$ directly outputs $(\mathsf{com},\mathsf{aux}) \leftarrow \mathsf{FC}.\mathsf{Com}(\mathsf{ck},\boldsymbol{x};r)$ (note that $\mathsf{FC}.\mathsf{Com}$ is com-hiding).
– $\mathsf{FC}^*.\mathsf{Open}((\mathsf{ck},\mathsf{crs}),\mathsf{aux},\mathsf{f})$ runs $\pi \leftarrow \mathsf{FC}.\mathsf{Open}(\mathsf{ck},\mathsf{aux},f)$ and then outputs $\pi^* \leftarrow \mathit{\Pi}.\mathsf{Prove}(\mathsf{crs},(\mathsf{ck},\mathsf{com},\mathsf{f},\boldsymbol{y}),\pi)$.
– $\mathsf{FC}^*.\mathsf{Ver}((\mathsf{ck},\mathsf{crs}),\mathsf{com},\mathsf{f},\boldsymbol{y},\pi^*)$ outputs $b \leftarrow \mathit{\Pi}.\mathsf{Ver}(\mathsf{crs},(\mathsf{ck},\mathsf{com},\mathsf{f},\boldsymbol{y}),\pi^*)$.

Additive homomorphism and efficient verification of FC^* follow from the respective properties of FC.

Zero knowledge follows from the zero knowledge property of the NIZK, since we can construct a simulator $\mathsf{FC}^*.\mathsf{Sim}$ given the NIZK simulator $\mathit{\Pi}.\mathsf{Sim}$ as follows. $\mathsf{FC}^*.\mathsf{Sim}_{\mathsf{Com}}, \mathsf{FC}^*.\mathsf{Sim}_{\mathsf{Equiv}}$ are the same as their respective FC com-hiding simulators. $\mathsf{FC}^*.\mathsf{Sim}_{\mathsf{Setup}}$ runs both $\mathsf{FC}.\mathsf{Sim}_{\mathsf{Setup}}$ and the NIZK simulator $\mathit{\Pi}.\mathsf{Sim}_{\mathsf{Prove}}$ and outputs simulated $\widetilde{crs}, \widetilde{ck}$ and respective trapdoors. Finally, $\mathsf{FC}^*.\mathsf{Sim}_{\mathsf{Open}}$ runs the NIZK simulator $\mathit{\Pi}.\mathsf{Sim}_{\mathsf{Prove}}$ on the simulated $\widetilde{\mathsf{crs}}$ and its trapdoor.

Evaluation binding for FC^* requires the knowledge soundness of the NIZK. Namely, given two proofs $\pi^*, \pi^{*\prime}$ for different outputs $\boldsymbol{y}, \boldsymbol{y}'$ and the same commitment com, one can run the NIZK extractor to obtain π, π' from $\pi^*, \pi^{*\prime}$. Then, it is possible to make a reduction to the security (evaluation binding) of FC.

A.2 Strong Evaluation Binding and Extractability

Definition 11 (Strong Evaluation Binding). *For any PPT adversary $\mathcal{A}$, the following advantage is* $\mathsf{negl}(\lambda)$*:*

$$\mathbf{Adv}^{\mathsf{sEvBind}}_{\mathcal{A},\mathsf{FC}}(\lambda) = \Pr\left[\begin{matrix} \quad\forall i \in [Q], \\ \mathsf{Ver}(\mathsf{ck},\mathsf{com},f_i,\boldsymbol{y}_i,\pi_i) = 1 \\ \wedge\ \nexists \boldsymbol{x} \in \mathcal{X} : f_i(\boldsymbol{x}) = \boldsymbol{y}_i \end{matrix} \ :\ \begin{matrix} \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda,1^n) \\ (\mathsf{com},\{f_i,\boldsymbol{y}_i,\pi_i\}_{i=1}^{Q}) \leftarrow \mathcal{A}(\mathsf{ck}) \end{matrix}\right]$$

Definition 12 (FC Extractability). FC *is knowledge extractable for an auxiliary input distribution $\mathcal{Z}$ if for any polynomial time adversary $\mathcal{A}$ there exists a PPT extractor $\mathcal{E}$ such that the following advantage is* $\mathsf{negl}(\lambda)$*:*

$$\mathbf{Adv}^{\mathsf{extr}}_{\mathcal{A},\mathsf{FC}}(\lambda) = \Pr\left[\begin{matrix} \mathsf{Ver}(\mathsf{ck},\mathsf{com},f,\boldsymbol{y},\pi) = 1 \\ \wedge\ (\mathsf{com} \neq \mathsf{com}' \\ \vee\ f(\boldsymbol{x}) \neq \boldsymbol{y}) \end{matrix} \ :\ \begin{matrix} \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda,1^n) \\ \mathsf{aux}_Z \leftarrow \mathcal{Z}(1^\lambda) \\ (\mathsf{com},f,\boldsymbol{y},\pi) \leftarrow (\mathcal{A})(\mathsf{ck},\mathsf{aux}_Z) \\ (\boldsymbol{x};r) \leftarrow \mathcal{E}(\mathsf{ck},\mathsf{aux}_Z) \\ (\mathsf{com}',\mathsf{aux}') \leftarrow \mathsf{Com}(\mathsf{ck},\boldsymbol{x};r) \end{matrix}\right]$$

Proposition 4. *Let* FC *be a knowledge extractable FC. Then,* FC *satisfies strong evaluation binding.*

Proof. Let $\mathcal{A}$ be an adversary against strong evaluation binding that on input $(\mathsf{ck},\mathsf{aux}_Z)$ returns $(\mathsf{com},\{f_i,\boldsymbol{y}_i,\pi_i\}_{i=1}^{Q})$ such that $\forall i \in [Q], \mathsf{Ver}(\mathsf{ck},\mathsf{com},f_i,\boldsymbol{y}_i,\pi_i) = 1 \wedge \nexists \boldsymbol{x} \in \mathcal{X} : f_i(\boldsymbol{x}) = \boldsymbol{y}_i$. We will show that if $\mathcal{A}$ is a succesful adversary then FC is not knowledge extractable.

We proceed by contradiction; assume that FC is knowledge extractable. We define Q adversaries $\mathcal{B}_1(\mathsf{ck},\mathsf{aux}_Z),\ldots,\mathcal{B}_Q(\mathsf{ck},\mathsf{aux}_Z)$ against FC extractability as follows: $\mathcal{B}_i$ runs $\mathcal{A}(\mathsf{ck},\mathsf{aux}_Z)$ and returns the i-th tuple $(\mathsf{com},f_i,\boldsymbol{y}_i,\pi_i)$ from $\mathcal{A}$'s output. Notice that $\mathcal{A}$ is a deterministic machine; nevertheless $\mathcal{A}$ can take input randomness in aux_Z. As FC is knowledge extractable, for every $\mathcal{B}_i$ there exists an extractor $\mathcal{E}_i(\mathsf{ck}, \mathsf{aux}_Z)$ that returns $\boldsymbol{x}_i; r_i$ such that (abusing notation) $\mathsf{com} \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i; r_i)$ and $f_i(\boldsymbol{x}_i) = \boldsymbol{y}_i$.

We now distinguish two cases. First, suppose that $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ for some $i, j \in [Q]$. Then, we have that $\mathsf{com} = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i; r_i) = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_j; r_j)$, which is a contradiction as this breaks commitment binding. Otherwise, let $\boldsymbol{x}$ be the vector such that $\boldsymbol{x} = \boldsymbol{x}_i$ for all $i \in [Q]$. Then, by correctness of the extractors $\mathcal{E}_i$ we have that $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$ for every $i \in [Q]$, which is a contradiction with respect to $\mathcal{A}$ breaking evaluation binding.

Definition 13 (CFC Extractability). CFC *is knowledge extractable for an auxiliary input distribution $\mathcal{Z}$ if for any polynomial time adversary $\mathcal{A}$ there exists an extractor $\mathcal{E}$ such that the following probability is* $\mathsf{negl}(\lambda)$*:*

$$\Pr\left[\begin{array}{l} \mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi) = 1 \\ \wedge\ (\exists i \in [m] : \mathsf{com}_i \neq \mathsf{com}'_i \\ \quad \vee\ \mathsf{com}_y \neq \mathsf{com}'_y \\ \quad \vee\ f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \neq \boldsymbol{y}) \end{array} : \begin{array}{r} \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ \mathsf{aux}_Z \leftarrow \mathcal{Z}(1^\lambda) \\ ((\mathsf{com}_i)_{i\in[m]}, f, \mathsf{com}_y, \pi) \leftarrow \mathcal{A}(\mathsf{ck}, \mathsf{aux}_Z) \\ ((\boldsymbol{x}_i; r_i)_{i\in[m]}, (\boldsymbol{y}; r_y)) \leftarrow \mathcal{E}(\mathsf{ck}, \mathsf{aux}_Z) \\ (\mathsf{com}'_i, \mathsf{aux}'_i) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i; r_i) \\ (\mathsf{com}'_y, \mathsf{aux}'_y) \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{y}; r_y) \end{array}\right]$$

A.3 Extractability of FC from CFC

Theorem 5. *If* CFC *is a knowledge extractable CFC, then our* FC *in Figure 1 is knowledge extractable.*

Proof. Let $\mathcal{A}$ be an adversary against FC extractability (Definition 12) with respect to an auxiliary input distribution $\mathcal{Z}$. On input ck, $\mathcal{A}$ returns $(\mathsf{com}, f, \boldsymbol{y}, \pi) \leftarrow \mathcal{A}(\mathsf{ck})$ such that $\mathsf{FC.Ver}(\mathsf{ck}, \mathsf{com}, f, \boldsymbol{y}, \pi) = 1$. Our goal is to construct an extractor $\mathcal{E}_\mathcal{A}$ for FC and argue that it is successful with overwhelming probability. The intuition of the proof is that we can use $\mathcal{A}$ to create an adversary $\mathcal{B}$ against CFC extractability (Definition 13) with respect to the same input distribution $\mathcal{Z}$. Then, we use the extractor $\mathcal{E}_\mathcal{B}$ for CFC to build $\mathcal{E}_\mathcal{A}$. We describe $\mathcal{B}$ and $\mathcal{E}_\mathcal{A}$ in Figure 2.

Let $(\mathsf{com}, \mathsf{com}_1, \pi_1, f^{(1)})$ be the output of $\mathcal{B}$. It follows that $\mathcal{B}$ is a valid adversary against CFC extractability and that $\mathsf{Ver}(\mathsf{ck}, \mathsf{com}, f^{(1)}, \mathsf{com}_1, \pi_1) = 1$. As CFC is knowledge extractable, there exists an extractor $\mathcal{E}_\mathcal{B}$ that returns $\boldsymbol{x}, \boldsymbol{x}^{(1)}$ such that

$$\Pr[\mathsf{com} = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}) \wedge \mathsf{com}_1 = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}^{(1)}) \wedge f^{(1)}(\boldsymbol{x}) = \boldsymbol{x}^{(1)}] = 1 - \mathsf{negl}(\lambda).$$

Next, we show that the extractor $\mathcal{E}_\mathcal{A}$ for FC succeeds with overwhelming probability, i.e., that

$$\Pr[\mathsf{com} \neq \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}) \vee f(\boldsymbol{x}) \neq \boldsymbol{y}] = \mathsf{negl}(\lambda).$$

For the first clause, we have that $\Pr[\mathsf{com} \neq \mathsf{Com}(\mathsf{ck}, \boldsymbol{x})] = \mathsf{negl}(\lambda)$ as otherwise the extractor $\mathcal{E}_\mathcal{B}$ is not successful (and $\mathcal{B}$ wins with non-negligible probability). For the second clause, we can recompute $\boldsymbol{y}' := f(\boldsymbol{x})$. If $f(\boldsymbol{x}) \neq \boldsymbol{y}$, then we can break evaluation binding of FC by creating an honest proof $\pi' \leftarrow \mathsf{FC.Open}(\mathsf{ck}, \boldsymbol{x}, f)$ and outputting $(\mathsf{com}, f, \pi, \boldsymbol{y}, \pi', \boldsymbol{y}')$. Hence, $\Pr[f(\boldsymbol{x}) \neq \boldsymbol{y}] = \mathsf{negl}(\lambda)$ and the result follows by the union bound.

$\mathcal{B}(\mathsf{ck}, \mathsf{aux}_Z)$	$\mathcal{E}_{\mathcal{A}}(\mathsf{ck}, \mathsf{aux}_Z)$
$(\mathsf{com}, f, \boldsymbol{y}, \pi) \leftarrow \mathcal{A}(\mathsf{ck}, \mathsf{aux}_Z)$	$(\boldsymbol{x}, \boldsymbol{x}^{(1)}) \leftarrow \mathcal{E}_{\mathcal{B}}(\mathsf{ck}, \mathsf{aux}_Z)$
$(f^{(1)}, \ldots, f^{(d)}) \leftarrow \mathsf{Parse}(f)$	**return** $\boldsymbol{x}$
$(\pi_1, \ldots, \pi_d, \mathsf{com}_1, \ldots, \mathsf{com}_{d-1}) \leftarrow \mathsf{Parse}(\pi)$	
return $(\mathsf{com}, \mathsf{com}_1, \pi_1, f^{(1)})$	

Fig. 2: Adversary $\mathcal{B}$ and extractor $\mathcal{E}_{\mathcal{A}}$ for the proof of Theorem 5.

B Analysis of the HiKer Assumption in the Generic Bilinear Group Model

Lemma 2. *The n-HiKer assumption holds in the generic bilinear group model.*

Proof. First of all, note that the assumption is equivalent to an assumption without rational terms. Indeed, for a uniformly sampled η', consider the assumption above where $\eta = \eta' \prod_{i,j \in [n]} \sigma_i \tau_j$.

The intuition is that since the solution (U, V) satisfies the equation $e(U, [\eta]_2) = e(V, [1]_2)$ then it must be of the form $(U, V) = [u, \eta u]_1$ for some u. However, if we look at the input of the adversary in $\mathbb{G}_1$, there is no pair of elements in the linear span of $[1, \eta]_1$. Note also that elements in $\mathbb{G}_2$ cannot be used by a GGM extractor as bgp is a Type-III bilinear group setting. A detailed proof follows.

More formally, let $\mathcal{A}$ be an adversary that on input (bgp, Ω) outputs two elements $U, V \in \mathbb{G}_1$ such that $e(U, [\eta]_2) = e(V, [1]_2)$. Then, the GGM extractor must output two polynomials $p_u(\boldsymbol{S}, \boldsymbol{T}, H), p_v(\boldsymbol{S}, \boldsymbol{T}, H)$ with coefficients $u_0, u_{\sigma,i}, u_{\tau,i}, u_{i,j}, u_{i,i'}, u_{i,j,i',j'}$ and $v_0, v_{\sigma,i}, v_{\tau,i}, v_{i,j}, v_{i,i'}, v_{i,j,i',j'}$ such that:

$$
\begin{aligned}
0 = {} & p_u(\boldsymbol{S}, \boldsymbol{T}, H)H + p_v(\boldsymbol{S}, \boldsymbol{T}, H) = \\
& u_0 H + v_0 + \sum_i \left[(u_{\sigma,i} S_i + u_{\tau,i} T_i)H + v_{\sigma,i} S_i + v_{\tau,i} T_i\right] + \sum_{i,j} \left[u_{i,j} S_i T_j H + v_{i,j} S_i T_j\right] \\
& + \sum_{\substack{i,i' \in [n] \\ i \neq i'}} \left[u_{i,i'} \frac{S_{i'}}{S_i} T_i H^2 + v_{i,i'} \frac{S_{i'}}{S_i} T_i H\right] + \sum_{\substack{i,j,i',j' \in [n] \\ (i,j) \neq (i',j')}} \left[u_{i,j,i',j'} \frac{S_{i'} T_{j'}}{S_i T_j} H^2 + v_{i,j,i',j'} \frac{S_{i'} T_{j'}}{S_i T_j} H\right].
\end{aligned}
$$

Due to the equivalence mentioned above, we can effectively do a change of variable $H \mapsto HA$ where $A = \prod_{i,j \in [n]} S_i T_j$ and reorganize the expression as a polynomial $c_0 + c_1 H + c_2 H^2$ in H, where

$$
c_0 = \left[v_0 + \sum_{i \in [n]} (v_{\sigma,i} S_i + v_{\tau,i} T_i) + \sum_{i,j \in [n]} v_{i,j} S_i S_j\right],
$$

$$
c_1 = \left[u_0 + \sum_{i \in [n]} (u_{\sigma,i} S_i + u_{\tau,i} T_i) + \sum_{i,j \in [n]} u_{i,j} S_i T_j + \sum_{\substack{i,i' \in [n] \\ i \neq i'}} v_{i,i'} \frac{S_{i'}}{S_i} T_i + \sum_{\substack{i,j,i',j' \in [n] \\ (i,j) \neq (i',j')}} v_{i,j,i',j'} \frac{S_{i'} T_{j'}}{S_i T_j}\right] A,
$$

$$
c_2 = \left[\sum_{\substack{i,j,i',j' \in [n] \\ (i,j) \neq (i',j')}} u_{i,j,i',j'} \frac{S_{i'} T_{j'}}{S_i T_j} + \sum_{\substack{i,i' \in [n] \\ i \neq i'}} u_{i,i'} \frac{S_{i'}}{S_i} T_i\right] A^2.
$$

For the above to equal the zero polynomial in H, all terms must cancel. We analyze the constant, linear, and quadratic terms separately. Note that as all fractions are multiplied by $A = \prod_{i,j\in[n]} S_i T_j$, all denominators vanish.

- The constant term does not include cross-terms, so all monomials are linearly independent and the expression cancels only if $v_0 = v_{\sigma,i} = v_{\tau,i} = v_{i,j} = 0$.
- The linear term is formed by pairwise distinct monomials which are all independent; no allowed choice of indices i, j, i', j' or i, i' produces a monomial in the linear span of any others. In particular, note that variables in $(S_{i'}/S_i)T_i$ only cancel for $i = i'$ which is not in the sum. Also, the denominator of $(S_{i'}T_{j'})/(S_iT_j)$ only cancels if $(i, j) = (i', j')$ which is also not in the sum.
- For the quadratic term, we reason analogously to conclude that all terms are independent.

It follows that all coefficients of $p_u(\boldsymbol{S}, \boldsymbol{T}, H)$ and $p_v(\boldsymbol{S}, \boldsymbol{T}, H)$ must be zero, so $p_u = p_v = 0$ and the assumption holds.

C Knowledge Extractability of the Pairing-based CFC for Quadratic Functions

In this section, we prove that the pairing-based CFC for quadratic functions of Section 6 is knowledge extractable (hence strong evaluation binding by Proposition 4) under an extractability (non-falsifiable) assumption that we define below. This result implies that CFC can be seen as a SNARK for quadratic polynomial maps. In particular, its use with a single input can be used to prove arithmetic circuit satisfiability, and thus it is a (commit-and-prove) SNARK for NP with constant-size proofs.

Extractability assumption. First of all, we introduce our extractability assumption which is a slight extension of classical knowledge-of-exponent assumptions in bilinear groups. The intuition is that if the adversary produces two group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ that share the same discrete logarithm, then it must know coefficients that explain both elements as a linear combination of (a subset of) its inputs that have the same representation in $\mathbb{G}_1$ and $\mathbb{G}_2$. Note that our assumption can only hold in type-III bilinear groups, as for type-I and type-II groups there exists an efficient map $\mathbb{G}_2 \to \mathbb{G}_1$.

Definition 14 (Assumption 1). *Let* $\mathsf{bgp} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ *be a bilinear group setting and let* $\mathcal{Z}$ *be a PPT auxiliary input generator. The assumption holds for* bgp *and* $\mathcal{Z}$ *if, for every* $n = \mathsf{poly}(\lambda)$ *and any polynomial-time* $\mathcal{A}$*, given* $\Omega(\boldsymbol{\sigma}) := \left([\boldsymbol{\sigma}]_1, [\boldsymbol{\sigma}]_2, \left\{\left[\frac{\sigma_{i'}}{\sigma_i}\right]_1\right\}_{i,i'\in[n}, \left\{\left[\frac{1}{\sigma_i}\right]_2\right\}_{i\in[n]}\right)$, *then there exists a polynomial-time extractor* $\mathcal{E}$ *such that*

$$\Pr\left[\begin{array}{c} e(U, [1]_2) = e([1]_1, V) \\ \wedge\, U \neq [x_0 + \langle \boldsymbol{x}, \boldsymbol{\sigma}\rangle]_1 \end{array} \middle| \begin{array}{r} \mathsf{aux}_Z \leftarrow \mathcal{Z}(\Omega) \\ (U, V) \leftarrow \mathcal{A}(\mathsf{bgp}, \Omega; \mathsf{aux}_Z) \\ (x_0, \{x_i\}_{i=1}^n) \leftarrow \mathcal{E}(\mathsf{bgp}, \Omega; \mathsf{aux}_Z) \end{array}\right] = \mathsf{negl}(\lambda)$$

where the probability is taken over the choice of $\boldsymbol{\sigma} \leftarrow\$\, \mathbb{F}^n$ *and the random coins of* $\mathcal{Z}$.

We now describe the auxiliary generators $\mathcal{Z}$ for which we can argue (in the generic group model) that Assumption 1 holds. We say that a PPT input generator $\mathcal{Z}$ is admissible if, on input $\Omega(\boldsymbol{\sigma})$, outputs a set aux_Z of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ such that:

– There exists no pair of elements $A \in \mathbb{G}_1, B \in \mathbb{G}_2$ in the linear span of $\Omega \cup \mathsf{aux}_Z$, except for linear combinations of $[\sigma_i]_1, [\sigma_i]_2$ and the group generators $[1]_1, [1]_2$, such that $e(A, [1]_2) = e([1]_1, B)$.
– All elements provided in aux_Z can be independently generated from the input of the assumption Ω and from the random coins of $\mathcal{Z}$.

Our CFC Auxiliary input generator. In order to show extractability of CFC, we define an input generator $\mathcal{Z}_{\mathsf{CFC}}$ that, on input Ω, outputs a set aux_Z which has an identical distribution to a commitment key ck of the CFC. $\mathcal{Z}_{\mathsf{CFC}}$ proceeds as follows. First, it samples $\boldsymbol{\beta}, \boldsymbol{\gamma} \leftarrow\$ \mathbb{F}^n$ and $\eta_\alpha, \eta_\beta, \eta_\gamma \leftarrow\$ \mathbb{F}$. Then, it generates all parameters as follows, implicitly setting $\boldsymbol{\alpha} := \boldsymbol{\sigma}$:

$$\mathsf{aux}_Z := \begin{pmatrix} [\boldsymbol{\sigma}]_1\,, [\boldsymbol{\sigma}]_2\,, \boldsymbol{\beta}\,[1]_1\,, \boldsymbol{\gamma}\,[1]_1\,, [\boldsymbol{\sigma}]_1 \otimes \boldsymbol{\beta}, \eta_\alpha\,[1]_2\,, \eta_\beta\,[1]_2\,, \eta_\gamma\,[1]_2 \\ \left\{\frac{\gamma_{i'}}{\gamma_i}\eta_\alpha\,[\sigma_i]_1\,, \beta_i\eta_\beta\left[\frac{\sigma_{i'}}{\sigma_i}\right]_1\right\}_{\substack{i,i'\in[n]\\ i\neq i'}} \left\{\frac{\beta_{j'}}{\beta_j}\gamma_k\eta_\gamma\left[\frac{\sigma_{i'}}{\sigma_i}\right]_1\right\}_{\substack{i,j,i',j',k\in[n]\\ (i,j)\neq(i',j')}} \\ \left\{\frac{\eta_\alpha}{\gamma_i}\,[\sigma_i]_2\,, \beta_i\eta_\beta\left[\frac{1}{\sigma_i}\right]_2\right\}_{i\in[n]}, \left\{\gamma_k\eta_\gamma\left[\frac{1}{\sigma_i}\right]_2\right\}_{i,k\in[n]}, \left\{\frac{\gamma_k\eta_\gamma}{\beta_j}\left[\frac{1}{\sigma_i}\right]_2\right\}_{i,j,k\in[n]} \end{pmatrix}$$

Clearly, the distribution of aux_Z, given a random generation of Ω, is identical to $\mathsf{ck} \leftarrow \mathsf{CFC.Setup}(1^\lambda, 1^n)$ and the conditions specified above on $\mathcal{Z}$ hold.

Extending our HiKer assumption. Our extractability proof requires a second assumption which is an extension of the Hinted Kernel assumption introduced in 9. We define it below.

Definition 15 (Assumption 2). *Let $\mathsf{bgp} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ be a bilinear group setting, let $n \in \mathbb{N}$ and let $\mathcal{G}_1, \mathcal{G}_2$ be the following two sets of Laurent monomials in $\mathbb{Z}_q[S_1, T_1, \ldots, S_n, T_n, H]$:*

$$\mathcal{G}_1(\boldsymbol{S}, \boldsymbol{T}, H) := \{S_i,\, T_i\}_{i\in[n]} \cup \{S_i \cdot T_j\}_{i,j\in[n]} \cup \left\{\frac{S_{i'}}{S_i} \cdot T_i \cdot H\right\}_{\substack{i,i'\in[n]\\ i\neq i'}} \cup \left\{\frac{S_{i'} \cdot T_{j'}}{S_i \cdot T_j} \cdot H\right\}_{\substack{i,j,i',j'\in[n]\\ (i,j)\neq(i',j')}}$$

$$\mathcal{G}_2(\boldsymbol{S}, \boldsymbol{T}, H) := \{H\} \cup \{S_i\}_{i\in[n]} \cup \left\{\frac{1}{S_i} \cdot T_i \cdot H,\, \frac{1}{S_i} \cdot H\right\}_{i\in[n]} \cup \left\{\frac{1}{S_i} \cdot \frac{1}{T_j} \cdot H\right\}_{i,j\in[n]}$$

The assumption holds if for every $n = \mathsf{poly}(\lambda)$ and any PPT $\mathcal{A}$, the following advantage is negligible

$$\Pr\left[\begin{array}{c} (U, V) \neq (1, 1)_{\mathbb{G}_1} \wedge \\ e(U, [\eta]_2) \cdot e(V, [1]_2) = \prod_i e\left(W_i, \left[\frac{\eta\tau_i}{\sigma_i}\right]_2\right) \end{array} \middle|\, (U, V, \{W_i\}_{i\in[n]}) \leftarrow \mathcal{A}\left(\mathsf{bgp}, \begin{array}{c}[\mathcal{G}_1(\boldsymbol{\sigma}, \boldsymbol{\tau}, \eta)]_1, \\ [\mathcal{G}_2(\boldsymbol{\sigma}, \boldsymbol{\tau}, \eta)]_2\end{array}\right)\right]$$

where the probability is over the random choices of $\boldsymbol{\sigma}, \boldsymbol{\tau}, \eta$ and $\mathcal{A}$'s random coins.

The HiKer assumption is a special case of Assumption 2 in which every $W_i = [0]_1$. The assumption can be justified in the GGM in an analogous way as the HiKer assumption; note that there are no terms of the form T_i/S_i or $H \cdot T_i/S_i$ in $\mathcal{G}_1$.

Extractability proof. We are now ready to state and prove the extractability of our pairing-based CFC for quadratic functions. The broad idea of the proof is that, given a valid CFC proof π for f, we can use the extractor of Assumption 1 to obtain coefficients $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ and values $x_{0,1}, \ldots, x_{0,m}$ such that the commitments to the inputs are of the form $\mathsf{com}_i = [\langle \boldsymbol{x}_i, \boldsymbol{\alpha}\rangle]_1 + [x_{0,i}]_1$

for every $i \in [m]$. Besides, using Assumption 2, we can assert that all $x_{0,i} = 0$ with overwhelming probability. This implies that the commitments com_i are correctly distributed as $\mathsf{com}_i = [\langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle]_1$ and that we can extract the committed inputs $\boldsymbol{x}_i$. Finally, we show that com_y must be a commitment to $\boldsymbol{y} = f(\boldsymbol{x}_1, \dots, x_m)$ as otherwise we break evaluation binding.

Theorem 6. *Assuming Assumption 1 for the input generator $\mathcal{Z}_{\mathsf{CFC}}$ described above, and Assumption 2, the pairing-based CFC scheme of Section 6 is knowledge extractable.*

Proof. Let $\mathcal{A}$ be a deterministic, polynomial-time adversary against CFC extractability. On input $(\mathsf{ck}, \mathsf{aux}_Z)$ for some auxiliary input $\mathsf{aux}_Z \leftarrow \mathcal{Z}$, $\mathcal{A}$ outputs $((\mathsf{com}_i)_{i\in[m]}, f, \mathsf{com}_y, \pi)$ such that $\mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi) = 1$. Our goal is to show that we can construct an extractor $\mathcal{E}_{\mathcal{A}}$ that on input $(\mathsf{ck}, \mathsf{aux}_Z)$ returns vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m, \boldsymbol{y}$ such that the advantage of $\mathcal{A}$ in the extractability game is

$$\Pr\left[\begin{array}{l} \quad \mathsf{Ver}(\mathsf{ck}, (\mathsf{com}_i)_{i\in[m]}, \mathsf{com}_y, f, \pi) = 1 \\ \wedge\ (\ \exists i \in [m] : \mathsf{com}_i \neq \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i) \\ \vee\ \mathsf{com}_y \neq \mathsf{Com}(\mathsf{ck}, \boldsymbol{y}) \\ \vee\ f(\boldsymbol{x}_1, \dots, \boldsymbol{x}_m) \neq \boldsymbol{y}\ ) \end{array}\right] = \mathsf{negl}(\lambda).$$

$\mathcal{B}_0^{(j)}(\Omega, \mathsf{aux}_Z)$	$\mathcal{E}_{\mathcal{A}}(\mathsf{ck}, \mathsf{aux}_Z)$
$1:\ \mathsf{ck} \leftarrow \mathsf{aux}_Z$	$1:\ \textbf{for } j \in [m]:$
$2:\ (\mathsf{com}_i)_{i\in[m]}, f, \mathsf{com}_y, \pi) \leftarrow \mathcal{A}(\mathsf{ck}, \mathsf{aux}_Z)$	$2:\ \quad (U, V) \leftarrow \mathcal{B}_0^{(j)}(\Omega, \mathsf{aux}_Z)$
$3:\ (X_i, X_i^{(2)})_{i\in[m]} \leftarrow \mathsf{Parse}(\pi)$	$3:\ \quad (x_j, x_{0,j}) \leftarrow \mathcal{E}_{\mathcal{B}}^{(j)}(\Omega, \mathsf{aux}_Z)$
$4:\ \textbf{return } (X_j, X_j^{(2)})$	$4:\ \boldsymbol{y} \leftarrow f(\boldsymbol{x}_1, \dots, \boldsymbol{x}_m)$
	$5:\ \textbf{return } (\boldsymbol{x}_1, \dots, \boldsymbol{x}_m, \boldsymbol{y})$

Fig. 3: Adversaries $\mathcal{B}_0^{(j)}$ and extractor $\mathcal{E}_{\mathcal{A}}$ for the proof of Theorem 6. Note that, for $\mathsf{aux}_Z \leftarrow \mathcal{Z}_{\mathsf{CFC}}(\Omega)$, we have that $\Omega \subset \mathsf{aux}_Z$.

First of all, we show how to construct $\mathcal{E}_{\mathcal{A}}$. We define m adversaries $\mathcal{B}_0^{(j)}$ for $j \in [m]$ against Assumption 1 in Figure 3. $\mathcal{B}_0^{(j)}$ takes Ω, aux_Z as input, where aux_Z is generated by $\mathcal{Z}_{\mathsf{CFC}}(\Omega)$ as explained before, and includes all the elements of a valid ck for the CFC scheme that are consistent with Ω. We define $\mathcal{E}_{\mathcal{A}}$ in Figure 3.

In the rest of the proof, we show that the extractor succeeds except with negligible probability. Consider the pair $(\mathcal{A}, \mathcal{E}_{\mathcal{A}})$ and their outputs, and let Win be the event in which $(\mathcal{A}, \mathcal{E}_{\mathcal{A}})$ win the CFC extractability game. We will reduce $(\mathcal{A}, \mathcal{E}_{\mathcal{A}})$ to an adversary $\mathcal{B}$ against Assumption 2 to show that $\Pr[\mathsf{Win}] \leq \mathsf{negl}(\lambda)$. The adversary $\mathcal{B}$ will embed the input of the assumption into a simulated commitment key ck (we detail this procedure below), then run $(\mathcal{A} \| \mathcal{E}_{\mathcal{A}})(\mathsf{ck})$ and parse its output $((\mathsf{com}_i)_{i\in[m]}, f, \mathsf{com}_y, \pi)$ and $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_m, \boldsymbol{y})$. Depending on such output, we distinguish between the following (nested) events:

- Event BadExt as the event in which $\mathsf{com}_j \neq [\langle \boldsymbol{x}_j, \boldsymbol{\alpha} \rangle]_1 + [x_{0,j}]_1$ for some $j \in [m]$ and $x_{0,j}$.
- Event BadCom as the event in which BadExt does not occur and $x_{0,j} \neq 0$ for some $j \in [m]$.

– Event BadY as the event in which $\mathsf{com}_y \neq \mathsf{Com}(\mathsf{ck}, f(\boldsymbol{x}_1, \ldots, x_m)$ and BadCom (and hence also BadExt) does not occur.

First, we bound the adversary's winning probability given a bad extraction, $\Pr[\mathsf{Win} \wedge \mathsf{BadExt}]$.

Event BadExt. The output of each $\mathcal{B}_0^{(j)}$, which corresponds to the vectors extracted by $\mathcal{E}_{\mathcal{A}}$, is $(\mathsf{com}_j, X_j^{(2)})$ in the proof π. Also, by the pairing checks in the $\mathsf{CFC.Ver}$ algorithm, these satisfy $e(\mathsf{com}_j, [1]_2) = e([1]_1, X_j^{(2)})$. As ck is perfectly distributed, the output of $\mathcal{E}_{\mathcal{A}}$ satisfies that $\mathsf{com}_i \neq [\langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle]_1 + [x_{0,i}]_1$ for some $x_{0,i}$, and for every i, unless any of the extractors $\mathcal{E}_{\mathcal{B}}^{(j)}$ fails. By Assumption 1, this occurs with negligible probability, so by the union bound we have

$$\Pr[\mathsf{Win} \wedge \mathsf{BadExt}] \leq \sum_{i=1}^{m} \Pr[\mathsf{com}_i \neq [\langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle]_1 + [x_{0,i}]_1] \leq m \cdot \epsilon_{\mathsf{Ass1}} = \mathsf{negl}(\lambda).$$

Note that

$$\Pr[\mathsf{Win}] \leq \Pr[\mathsf{Win} \wedge \mathsf{BadExt}] + \Pr[\mathsf{Win} | \neg \mathsf{BadExt}]$$

Next, we will bound $\Pr[\mathsf{Win} | \neg \mathsf{BadExt}]$ by showing a reduction to Assumption 2.

Commitment key generation. Based on the events above, $\mathcal{B}$ makes a secret guess $\hat{b} \leftarrow\$ \{0, 1\}$. Intuitively, $\hat{b} = 1$ corresponds to event BadCom, and a subcase of event BadY, whereas $\hat{b} = 0$ corresponds to a different subcase of BadY. Then, $\mathcal{B}$ simulates ck depending on $\hat{b}$:

– If $\hat{b} = 0$, then $\mathcal{B}$ receives the input of the assumption and generates ck exactly as in the case $\hat{s} = 0$ of the proof of evaluation binding for CFC. Namely, it samples $\boldsymbol{\alpha}, \boldsymbol{\beta} \leftarrow\$ \mathbb{F}^n, \eta_\beta, \eta_\gamma \leftarrow\$ \mathbb{F}$ and implicitly sets $\boldsymbol{\gamma} := \boldsymbol{\sigma}$ and $\eta_\alpha := \eta$ from the input of the assumption. Then, it simulates the remaining terms in ck accordingly.
– If $\hat{b} = 1$, then $\mathcal{B}$ proceeds as in the case $\hat{s} = 1$ of the proof of evaluation binding for CFC. Namely, $\mathcal{B}$ samples $\eta_\alpha, r_\beta, r_\gamma \leftarrow\$ \mathbb{F}$, $\boldsymbol{\gamma} \leftarrow\$ \mathbb{F}^n$ and implicitly sets $\boldsymbol{\alpha} := \boldsymbol{\sigma}, \boldsymbol{\beta} := \boldsymbol{\tau}, \eta_\beta := r_\beta \cdot \eta, \eta_\gamma := r_\gamma \cdot \eta$. Later, it simulates ck accordingly.

Next, $\mathcal{B}$ runs $(\mathcal{A} || \mathcal{E}_{\mathcal{A}})(\mathsf{ck})$ and parses the output as detailed before. The reduction proceeds differently depending on the events above.

Event BadCom**.** If $\hat{b} \neq 1$, then $\mathcal{B}$ aborts. We analyze the probability that $\mathsf{Win} \wedge \mathsf{BadCom}$ occurs given that BadExt does not occur. By $\neg\mathsf{BadExt}$, there exist values $x_{0,i}$ such that $X_i = \mathsf{com}_i = [\langle \boldsymbol{x}_i, \boldsymbol{\alpha} \rangle]_1 + [x_{0,i}]_1$ for all $i \in [m]$. By the occurrence of BadCom there must be an index h such that $x_{0,h} \neq 0$. Using the fact that the proof produced by the adversary correctly verifies, we have that for such h the following pairing identity (from the $\alpha \to \beta$ conversion) holds:

$$e\left([\langle \boldsymbol{x}_h, \boldsymbol{\alpha} \rangle]_1 + [x_{0,h}]_1, \sum_{i \in [n]} \left[\frac{\beta_i \eta_\beta}{\alpha_i}\right]_2\right) = e\left(\pi_h^{(\beta)}, [1]_2\right) e\left(X_h^{(\beta)}, [\eta_\beta]_2\right)$$

Let $\mathcal{B}$ compute $\tilde{X}_h = [\langle \boldsymbol{x}_h, \boldsymbol{\alpha} \rangle]_1$ and $\tilde{X}_h^{(\beta)} = [\langle \boldsymbol{x}_h, \boldsymbol{\beta} \rangle]_1$. Furthermore, $\mathcal{B}$ computes an honest identity proof $\tilde{\pi}_h^{(\beta)}$ to show that $\tilde{X}_h$ and $\tilde{X}_h^{(\beta)}$ commit to the same value (i.e., for an $\alpha \to \beta$ conversion). Then, we can write the pairing identity as follows

$$e\left(\tilde{X}_h, S\right) \cdot e\left([x_{0,h}]_1, S\right) = e\left(\pi_h^{(\beta)}, [1]_2\right) e\left(X_h^{(\beta)}, [\eta_\beta]_2\right)$$
$$e\left(\tilde{\pi}_h^{(\beta)}, [1]_1\right) \cdot e\left(\tilde{X}_h^{(\beta)}, [\eta_\beta]_2\right) \cdot e\left([x_{0,h}]_1, S\right) = e\left(\pi_h^{(\beta)}, [1]_2\right) e\left(X_h^{(\beta)}, [\eta_\beta]_2\right)$$

where the second equality follows by the correctness of $\tilde{\pi}_h^{(\beta)}$. Also, for brevity, we let $S = \sum_{i\in[n]} \left[\frac{\beta_i\eta_\beta}{\alpha_i}\right]_2$. Moving terms to the right-hand side, we have:

$$e\left([x_{0,h}]_1, S\right) = x_{0,h} \left[\sum_{i\in[n]} \frac{\beta_i\eta_\beta}{\alpha_i}\right]_T = e\left(\pi_h^{(\beta)}/\tilde{\pi}_h^{(\beta)}, [1]_2\right) e\left(X_h^{(\beta)}/\tilde{X}_h^{(\beta)}, [\eta_\beta]_2\right)$$

Then, $\mathcal{B}$ outputs $(U, V, \{W_i\}_{i\in[n]})$ such that $U := (X_h^{(\beta)}/\tilde{X}_h^{(\beta)})^{r_\beta}$, $V := \pi_h^{(\beta)}/\tilde{\pi}_h^{(\beta)}$, and $W_i = r_\beta \cdot [x_{0,h}]_1$ for every $i \in [n]$. By the ck simulation procedure $\eta_\beta = r_\beta \cdot \eta$, therefore we have that $e(U, [\eta]_2) \cdot e(V, [1]_2) = \prod_i e(W_i, S)$ breaking Assumption 2.

Event BadY: We will turn $\mathcal{A}$ into an adversary against Assumption 2. As an intermediate step, we define a subroutine $\mathcal{B}^*$ that, on input ck and access to $(\mathcal{A}||\mathcal{E}_\mathcal{A})$ outputs a tuple $((\mathsf{com}_i)_{i\in[m}, f, \mathsf{com}_y, \pi, \mathsf{com}'_y, \pi')$ against evaluation binding.

We build $\mathcal{B}^*$ from $\mathcal{A}$ as follows. First, $\mathcal{B}^*$ calls $(\mathcal{A}||\mathcal{E}_\mathcal{A})(\mathsf{ck})$, parses their output as before, and computes $\boldsymbol{y}' = f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$. Then, $\mathcal{B}^*$ calculates $\mathsf{com}'_y \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{y})$ and outputs

$$((\mathsf{com}_i)_{i\in[m}, f, \mathsf{com}_y, \pi, \mathsf{com}'_y, \pi')$$

where π' is an honestly generated proof $\pi' \leftarrow \mathsf{CFC.Open}(\mathsf{ck}, (\mathsf{aux}_i)_{i\in[m]}, y, f)$.

Now, we show that if BadY occurs, then $\mathcal{B}^*$ breaks evaluation binding. First, note that since BadCom does not occur, we know that $\mathsf{com}_i = \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}_i)$ for every $i \in [m]$. If $\boldsymbol{y} = \boldsymbol{y}'$, then as $\mathcal{A}$ wins it must be the case that $\mathsf{com}_y \neq \mathsf{Com}(\mathsf{ck}, \boldsymbol{y}) = \mathsf{com}'_y$. Otherwise, if $\boldsymbol{y} \neq \boldsymbol{y}'$, then we have that $\mathsf{com}_y \neq \mathsf{com}'_y$ (as the commitment is binding). In both cases, we break evaluation binding by opening to two different commitments via a honest opening proof π' to com'_y.

Finally, we define $\mathcal{B}$ as follows. After embedding the assumption on ck based on the choice of $\hat{b}$, $\mathcal{B}$ uses the output of $(\mathcal{A}||\mathcal{E}_\mathcal{A})$ to build $\mathcal{B}^*(\mathsf{ck})$ as an adversary against evaluation binding. Then, it proceeds exactly as in the evaluation binding proof, aborting or not depending whether the output of $\mathcal{B}^*(\mathsf{ck})$ is consistent with the guess $\hat{b}$. Hence, if $\mathcal{B}^*$ succeeds with probability ϵ, then $\mathcal{B}$ breaks the HiKer assumption with probability $\epsilon/2$.

We conclude the proof by noting that, since Assumption 2 implies the HiKer assumption, then

$$\Pr[\mathsf{Win}|\neg\mathsf{BadExt}] \leq 2 \cdot \epsilon_{\mathsf{Ass2}} = \mathsf{negl}(\lambda).$$

D Proof of Claim in Correctness of Lattice-based CFC

The proof of the claim relies on the following fact about Kronecker products and vectorization.

Lemma 3. *Let $\mathbf{L}, \mathbf{Z}$ be matrices and $\boldsymbol{v}, \boldsymbol{x}$ be vectors of compatible dimensions so that the product $\boldsymbol{v}^\mathsf{T} \cdot \mathbf{L} \cdot \mathbf{Z} \cdot \boldsymbol{x}$ is well-defined. It holds that*

$$\boldsymbol{v}^\mathsf{T} \cdot \mathbf{L} \cdot \mathbf{Z} \cdot \boldsymbol{x} = (\mathsf{vec}(\mathbf{Z})^\mathsf{T} \otimes \boldsymbol{v}^\mathsf{T}) \cdot \mathsf{vec}(\boldsymbol{x}^\mathsf{T} \otimes \mathbf{L}).$$

Proof. The proof involves repeated applications of the identities $\mathsf{vec}(\mathbf{ABC}) = (\mathbf{C}^{\mathsf{T}} \otimes \mathbf{A}) \cdot \mathsf{vec}(\mathbf{B})$ and $\mathsf{vec}(\boldsymbol{x}) = \boldsymbol{x}$. We observe the following:

$$\begin{aligned}\boldsymbol{v}^{\mathsf{T}} \cdot \mathbf{L} \cdot \mathbf{Z} \cdot \boldsymbol{x} &= \boldsymbol{v}^{\mathsf{T}} \cdot \mathsf{vec}(\mathbf{L} \cdot \mathbf{Z} \cdot \boldsymbol{x}) = \boldsymbol{v}^{\mathsf{T}} \cdot (\boldsymbol{x}^{\mathsf{T}} \otimes \mathbf{L}) \cdot \mathsf{vec}(\mathbf{Z}) \\ &= \mathsf{vec}(\boldsymbol{v}^{\mathsf{T}} \cdot (\boldsymbol{x}^{\mathsf{T}} \otimes \mathbf{L}) \cdot \mathsf{vec}(\mathbf{Z})) = (\mathsf{vec}(\mathbf{Z})^{\mathsf{T}} \otimes \boldsymbol{v}^{\mathsf{T}}) \cdot \mathsf{vec}(\boldsymbol{x}^{\mathsf{T}} \otimes \mathbf{L})\end{aligned}$$

□

We are now ready to prove the claim in the correctness proof. We prove it by directly calculating

$$\begin{aligned}
&\hat{f}(c_1, \ldots, c_m, \check{c}_1, \ldots, \check{c}_m) \\
=\ & \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \left(\sum_{h,h' \in [m]} \mathbf{G}_{h,h'} \cdot (\boldsymbol{v}^{\dagger} \otimes \check{\boldsymbol{v}}^{\dagger}) \cdot c_h \cdot \check{c}_{h'} + \sum_{i \in [m]} \mathbf{F}_h \cdot \boldsymbol{v}^{\dagger} \cdot c_h + \boldsymbol{e} \right) \\
=\ & \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \left(\sum_{h,h' \in [m]} \mathbf{G}_{h,h'} \cdot (\boldsymbol{v}^{\dagger} \otimes \check{\boldsymbol{v}}^{\dagger}) \cdot (\boldsymbol{v}^{\mathsf{T}} \otimes \check{\boldsymbol{v}}^{\mathsf{T}}) \cdot (\boldsymbol{x}_h \otimes \boldsymbol{x}_{h'}) + \sum_{i \in [m]} \mathbf{F}_h \cdot \boldsymbol{v}^{\dagger} \cdot \boldsymbol{v}^{\mathsf{T}} \cdot \boldsymbol{x}_h + \boldsymbol{e} \right) \\
=\ & \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \left(\sum_{h,h' \in [m]} \mathbf{G}_{h,h'} \cdot ((\boldsymbol{v}^{\dagger} \cdot \boldsymbol{v}^{\mathsf{T}}) \otimes (\check{\boldsymbol{v}}^{\dagger} \cdot \check{\boldsymbol{v}}^{\mathsf{T}})) \cdot (\boldsymbol{x}_h \otimes \boldsymbol{x}_{h'}) + \sum_{i \in [m]} \mathbf{F}_h \cdot \boldsymbol{v}^{\dagger} \cdot \boldsymbol{v}^{\mathsf{T}} \cdot \boldsymbol{x}_h + \boldsymbol{e} \right) \\
=\ & \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \left(\sum_{h,h' \in [m]} \mathbf{G}_{h,h'} \cdot ((\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \otimes (\mathbf{I} + \mathbf{Z}_{\check{\boldsymbol{v}}})) \cdot (\boldsymbol{x}_h \otimes \boldsymbol{x}_{h'}) + \sum_{i \in [m]} \mathbf{F}_h \cdot (\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \cdot \boldsymbol{x}_h + \boldsymbol{e} \right) \\
=\ & \bar{c}_0 + \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \sum_{h,h' \in [m]} \mathbf{G}_{h,h'} \cdot ((\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \otimes (\mathbf{I} + \mathbf{Z}_{\check{\boldsymbol{v}}}) - \mathbf{I}) \cdot (\boldsymbol{x}_h \otimes \boldsymbol{x}_{h'}) + \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \sum_{i \in [m]} \mathbf{F}_h \cdot \mathbf{Z}_{\boldsymbol{v}} \cdot \boldsymbol{x}_h \\
=\ & \bar{c}_0 + \sum_{h,h' \in [m]} (\mathsf{vec}((\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \otimes (\mathbf{I} + \mathbf{Z}_{\check{\boldsymbol{v}}}) - \mathbf{I})^{\mathsf{T}} \otimes \bar{\boldsymbol{v}}^{\mathsf{T}}) \cdot \mathsf{vec}(\boldsymbol{x}_h^{\mathsf{T}} \otimes \boldsymbol{x}_{h'}^{\mathsf{T}} \otimes \mathbf{G}_{h,h'}) \\
&\quad + \sum_{i \in [m]} (\mathsf{vec}(\mathbf{Z}_{\boldsymbol{v}})^{\mathsf{T}} \otimes \bar{\boldsymbol{v}}^{\mathsf{T}}) \cdot \mathsf{vec}(\boldsymbol{x}_h^{\mathsf{T}} \otimes \mathbf{F}_h),
\end{aligned}$$

where the last equality follows from Lemma 3,

$$\begin{aligned}\bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \boldsymbol{v}^{\dagger} \cdot \boldsymbol{v}^{\mathsf{T}} \cdot \boldsymbol{y} = \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot (\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \cdot \boldsymbol{y} = \bar{c}_0 + \bar{\boldsymbol{v}}^{\mathsf{T}} \cdot \mathbf{Z}_{\boldsymbol{v}} \cdot \boldsymbol{y}, \text{ and} \\ \check{\boldsymbol{v}}^{\mathsf{T}} \cdot \boldsymbol{v}^{\dagger} \cdot \boldsymbol{v}^{\mathsf{T}} \cdot \boldsymbol{y} = \check{\boldsymbol{v}}^{\mathsf{T}} \cdot (\mathbf{I} + \mathbf{Z}_{\boldsymbol{v}}) \cdot \boldsymbol{y} = \check{c}_0 + \check{\boldsymbol{v}}^{\mathsf{T}} \cdot \mathbf{Z}_{\boldsymbol{v}} \cdot \boldsymbol{y}.\end{aligned}$$

Table of Contents