# Succinct Functional Commitment for a Large Class of Arithmetic Circuits*

**July 8, 2021**

Helger Lipmaa and Kateryna Pavlyk

Simula UiB, Bergen, Norway

**Abstract.** A succinct functional commitment (SFC) scheme for a circuit class $\mathbf{CC}$ enables, for any circuit $\mathcal{C} \in \mathbf{CC}$, the committer to first succinctly commit to a vector $\boldsymbol{\alpha}$, and later succinctly open the commitment to $\mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta})$, where the verifier chooses $\boldsymbol{\beta}$ at the time of opening. Unfortunately, SFC commitment schemes are known only for severely limited function classes like the class of inner products. By making non-black-box use of SNARK-construction techniques, we propose an SFC scheme for the large class of semi-sparse polynomials. The new SFC scheme can be used to, say, efficiently (1) implement sparse polynomials, and (2) aggregate various interesting SFC (e.g., vector commitment and polynomial commitment) schemes. The new scheme is evaluation-binding under a new instantiation of the computational uber-assumption. We provide a thorough analysis of the new assumption.

**Keywords:** aggregated functional commitment, Déjà Q, functional commitment, SNARK, uber-assumption, vector commitment

## 1 Introduction

A succinct functional commitment (SFC) scheme [LRY16] for a circuit class $\mathbf{CC}$ enables the committer, for any $\mathcal{C} \in \mathbf{CC}$, to first commit to a vector $\boldsymbol{\alpha}$, and later open the commitment to $\mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta})$, where the verifier chooses $\boldsymbol{\beta}$ at the time of opening. An SFC scheme must be evaluation-binding (given a commitment, it is intractable to open it to $\boldsymbol{\xi} = \mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ and $\boldsymbol{\xi}' = \mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ for $\boldsymbol{\xi} \neq \boldsymbol{\xi}'$) and hiding (a commitment and possibly many openings should not reveal any additional information about $\boldsymbol{\alpha}$). Succinctness means that both the commitment and the opening have length $\mathsf{polylog}(|\boldsymbol{\alpha}|, |\boldsymbol{\beta}|)$.

In particular, an SFC scheme for inner products (SIPFC) assumes that $\mathcal{C}$ computes the inner product $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \to \langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle$ [LY10,ILV11,LRY16]. As explained in [LRY16], one can use an SIPFC scheme to construct succinct vector commitment schemes [CF13], polynomial commitment schemes [KZG10], and accumulators [Bd94]. Each of these primitives has a large number of independent applications. Succinct polynomial commitment schemes have recently become very popular since they can be used to construct (updatable) SNARKs [ZGK+17,WTS+18,MBKM19,CHM+20] (a direction somewhat opposite to the one we will pursue in the current paper). Since in several applications (e.g., in cryptocurrencies, [TAB+20]), one has to run many instances of SFC in parallel, there is a recent surge of interest in aggregatable SFC schemes, [BBF19,LM19,BDFG20,GRWZ20,TAB+20]. All mentioned papers propose *succinct* FC schemes for limited functionalities.

Since there are no prior SFC schemes for broader classes of functions, there is a large gap between function classes for which an SFC scheme is known and the class of all efficiently verifiable functions (e.g., poly-size arithmetic circuits). Filling a similar gap is notoriously hard in the case of related primitives like functional encryption, homomorphic encryption, and NIZK. A natural question to ask is whether something similar holds in the case of functional commitment.

It is easy to construct an SFC for all poly-size circuits under non-falsifiable assumptions: given a commitment to $\boldsymbol{\alpha}$, the opening consists of a SNARK argument [Gro10,Lip12,GGPR13,PHGR13,Lip13,Gro16] that

* Full version of [LP20]. It differs from [LP20] by having appendices, more standard AGM security proofs of certain theorems, and general readability improvements.

$\mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \boldsymbol{\xi}$. However, while non-falsifiable assumptions are required to construct SNARKs [GW11], they are not needed in the case of SFC schemes. Thus, just using SNARK as a black box is not a satisfactory solution.

Moreover, since one can construct non-succinct NIZK for $\mathsf{NP}$ from falsifiable assumptions, one can construct a non-succinct FC (nSFC) for $\mathsf{NP}$ from falsifiable assumptions. Bitansky [Bit17] pursued this approach, proposing an nSFC for all circuits that uses NIZK as a black-box. Using non-interactive witness-indistinguishable proofs in a non-black-box manner, Bitansky proposed another, non-trivial, nSFC scheme that does not achieve zero-knowledge (the strongest form of hiding, defined in the current paper) but does not require the CRS model. Alternatively, consider the FC scheme where the commitment consists of fully-homomorphic encryptions $C_i$ of individual coefficients $\alpha_i$, and the opening is the randomizer $R$ of the evaluation of the circuit $\mathcal{C}$ on them. The verifier can re-evaluate the circuit on $C_i$ and her input and then check that the result is equal to $\mathsf{Enc}(\boldsymbol{\xi}; R)$. However, the resulting FC is not succinct since one has to encrypt all $\alpha_i$ individually.

Thus, the main question is to construct *succinct* FC schemes, under falsifiable assumptions, for a wide variety of functionalities.

**Our Contributions.** We propose a falsifiable SFC scheme $\mathsf{FC}_{\mathsf{sn}}$ for the class of *semi-sparse polynomials* $\mathbf{CC} = \mathbf{CC}_{\Sigma\Pi\forall}$ whose correct computation can be verified by using an arbitrary polynomial-size arithmetic circuit that is "compilable" according to the definition, given in a few paragraphs. Notably, $\mathsf{FC}_{\mathsf{sn}}$ allows efficient aggregation of various SFC schemes, e.g., vector commitments with inner-product commitments and polynomial commitments. We analyze the power of $\mathbf{CC}_{\Sigma\Pi\forall}$ by using techniques from algebraic complexity theory; the name of the class will be explained in Section 4.

We prove that $\mathsf{FC}_{\mathsf{sn}}$ is secure under a new falsifiable assumption (computational span-uber-assumption in a group $\mathbb{G}_1$) that is reminiscent of the well-known computational uber-assumption in $\mathbb{G}_1$. We then thoroughly analyze the security of the new assumption.

**Our Techniques.** Next, we provide a high-level overview of our technical contributions. The construction of $\mathsf{FC}_{\mathsf{sn}}$ consists of the following steps.

1. Compilation of the original circuit $\mathcal{C}$ computing the fixed function $\boldsymbol{\mathcal{F}} \in \mathbf{CC}$ to a circuit $\mathcal{C}^*$ consisting of four public subcircuits.
2. Representation of $\mathcal{C}^*$ in the QAP language which SNARKs usually use.
3. Construction of SFC for the QAP representation, by using SNARK techniques in a non-black-box way.

Next we describe these steps in detail.

*Circuit Compilation.* Let $\mathcal{C} : \mathbb{Z}_p^{\mu_\alpha} \times \mathbb{Z}_p^{\mu_\beta} \to \mathbb{Z}_p^{\kappa}$ be a polynomial-size arithmetic circuit that, on input $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, outputs $\boldsymbol{\xi} = \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = (\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta}))_{i=1}^{\kappa}$. Here, the committer's input $\boldsymbol{\alpha}$ is secret, and the verifier's input is public. We modify the circuit $\mathcal{C}$ to a compiled circuit $\mathcal{C}^*$, see Fig. 1, that consists of the subcircuits $\mathcal{C}_\phi$, $\mathcal{C}_\psi$, $\mathcal{C}_\chi$, and $\mathcal{C}_\xi$. In the commitment phase, the committer uses the circuit $\mathcal{C}_\phi$ to compute several polynomials $\phi_i(\boldsymbol{\alpha})$ depending on only 1 (this allows the output polynomials to have a non-zero constant term) and $\boldsymbol{\alpha}$. In the opening phase, the verifier sends $\boldsymbol{\beta}$ to the committer, who uses the circuit $\mathcal{C}_\psi$ to compute several polynomials $\psi_i(\boldsymbol{\beta})$ depending on 1 and $\boldsymbol{\beta}$. The verifier can redo this part of the computation. After that, the committer uses the circuit $\mathcal{C}_\chi$ to compute several polynomials $\chi_i(\boldsymbol{\alpha}, \boldsymbol{\beta})$ from the inputs and outputs of $\mathcal{C}_\phi$ and $\mathcal{C}_\psi$. Finally, the committer uses $\mathcal{C}_\xi$ to compute the outputs $\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of $\mathcal{C}^*$. We will explain more thoroughly this compilation in Section 3.

Intuitively, the compilation restricts the class of circuits in two ways. First, we add a small circuit $\mathcal{C}_\xi$ at the top of the compiled circuit to guarantee that the R1CS representation of $\mathcal{C}^*$ has several all-zero columns and rows, which helps us in the security reduction. This does, however, not restrict the circuit class for which the SFC is defined, and it only increases the number of gates by $\kappa$. Second, $\mathcal{C}_\chi$ is restricted to have multiplicative depth 1, that is, it sums up products of polynomials in $\boldsymbol{\alpha}$ with polynomials in $\boldsymbol{\beta}$. This guarantees that in a collision, the two accepted openings have a linear relation that does not depend on the secret data $\boldsymbol{\alpha}$. The latter makes it possible for the reduction to break the underlying falsifiable assumption.

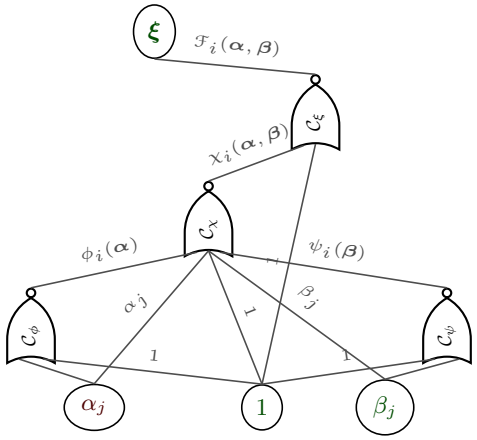


**Fig. 1.** The compiled circuit $\mathcal{C}^*$.

Thus, we are restricted to the class $\mathbf{CC}_{\Sigma\Pi\forall}$ of circuits where each output can be written as $\sum_{i,j}\phi_i(\boldsymbol{\alpha})\psi_j(\boldsymbol{\beta})$, for efficiently computable polynomials $\phi_i$ and $\psi_j$, and the sum is taken over $\mathsf{poly}(\lambda)$ products.

By employing tools from the algebraic complexity theory, in Section 4, we study the class $\mathbf{CC}_{\Sigma\Pi\forall}$ of "compilable" (according to the given definition) arithmetic circuits. We say that a polynomial $f \in \mathbf{CC}_{\Sigma\Pi\forall}$ if $f$ has a circuit that belongs to $\mathbf{CC}_{\Sigma\Pi\forall}$. The new SFC scheme can implement $f$ iff $f \in \mathbf{CC}_{\Sigma\Pi\forall}$. First, we show that any sparse polynomial (over indeterminates, chosen by both the committer and the verifier) belongs to $\mathbf{CC}_{\Sigma\Pi\forall}$. Second, we construct a non-sparse polynomial $f \in \mathbf{CC}_{\Sigma\Pi\forall}$. This relies on a result of Ben-Or who constructed an $O(n^2)$-size arithmetic circuit that simultaneously computes the $d$th symmetric polynomial $\sigma_d(X_1,\ldots,X_n)$, for $d \in [1,n]$. Third, we construct a polynomial $f \in \mathbf{VP}$ such that $f \notin \mathbf{CC}_{\Sigma\Pi\forall}$, where $\mathbf{VP}$ is the class of poly-degree polynomials that have poly-size circuits, [Val79].

*R1CS/QAP Representation.* Let $\mathcal{C}$ be an arithmetic circuit, and $\mathcal{C}^*$ be its compilation. A circuit evaluation can be verified by verifying a matrix equation, where matrices define the circuit uniquely and reflect all the circuit constraints. SNARKs usually use QAP (Quadratic Arithmetic Program, [GGPR13]), a polynomial version of R1CS, which allows for better efficiency.

*Constructing the Underlying SNARK.* Intuitively, we start constructing a SNARK for $\mathcal{C}^*$ by following the approach of Groth [Gro16] who proposed the most efficient known zk-SNARK, or more precisely, its recent modification by Lipmaa [Lip19]. However, we modify this approach whenever it suits our goals. The new SFC inherits the efficiency of Groth's SNARK; this is the main reason we chose Groth's SNARK; it may be the case that SFCs constructed from less efficient SNARKs have other desirable properties, but this is out of the scope of the current paper. We chose the modified version of [Lip19] due to its versatility: [Lip19] explains sufficiently well how to construct a SNARK for QAP so that it is feasible to modify its approach to suit the current paper.

*The New SFC Scheme.* In the SNARKs of [Gro16,Lip19], the argument consists of three group elements, $\pi = ([\mathsf{A}]_1, [\mathsf{B}]_2, [\mathsf{C}]_1)$. (We use the bracket additive notation, see Section 2.) Due to our restrictions on $\mathcal{C}^*$, both $[\mathsf{A}]_1$ and $[\mathsf{B}]_2$ can be written as sums of a non-functional commitment that depends only on the secret data and a non-functional commitment that depends only on the public data. By the public data we mean $(\boldsymbol{\beta}, \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}))$; any other function of $\boldsymbol{\alpha}$ is a part of the secret data. E.g., $[\mathsf{A}]_1 = [\mathsf{A}_s]_1 + [\mathsf{A}_p]_1$, where $[\mathsf{A}_s]_1$ is computed by the committer before $\boldsymbol{\beta}$ becomes available, and $[\mathsf{A}_p]_1$ can be recomputed by the verifier since it only depends on the public data. However, $[\mathsf{C}]_1 = [\mathsf{C}_{sp}]_1 + [\mathsf{C}_p]_1$, where $[\mathsf{C}_p]_1$ depends only on the public data but $[\mathsf{C}_{sp}]_1$ depends both on the public and private data.

In the new SFC commitment scheme, the functional commitment is $C = ([\mathsf{A}_s]_1, [\mathsf{B}_s]_2)$ and the opening is $[\mathsf{C}_{sp}]_1$. After receiving the opening, the verifier recomputes $[\mathsf{A}_p]_1$, $[\mathsf{B}_p]_2$, and $[\mathsf{C}_p]_1$, and then runs the SNARK verifier on the argument $\pi = ([\mathsf{A}_s]_1 + [\mathsf{A}_p]_1, [\mathsf{B}_s]_2 + [\mathsf{B}_p]_2, [\mathsf{C}_{sp}]_1 + [\mathsf{C}_p]_1)$. However, as we will see later, the commitment also includes auxiliary elements $[\mathsf{B}_i^{\mathsf{aux}}]_1$ needed to obtain an efficient security reduction.

We will denote the new SFC commitment scheme by $\mathsf{FC}_{\mathsf{sn}}$. We denote by $\mathsf{FC}_{\mathsf{sn}}^{\mathcal{C}}$ its specialization to the circuit $\mathcal{C}$.

**Applications.** To demonstrate the usefulness of $\mathsf{FC}_{\mathsf{sn}}$, we will give several applications; some of them are well-known, and some are new. In all cases, the function of interest can be rewritten as a semi-sparse polynomial in $(\boldsymbol{\alpha}, \boldsymbol{\beta})$. Some of these examples are closely related to but still sufficiently different from IPFC. In particular, [LRY16] showed how to use an efficient IPFC to construct SFC for polynomial commitments [KZG10], accumulators [Bd94], and vector commitments [CF13] (See Appendix A.1.). We use $\mathsf{FC}_{\mathsf{sn}}$ to construct subvector commitments [LM19], aggregated polynomial commitment [CHM+20,BDFG20] (one can commit to multiple polynomials at once, each of which can be opened at a different point), and multivariate polynomial commitments [BGH19]. Also, we outline a few seemingly new applications like the aggregated inner product (that, in particular, can be used to implement subvector commitment) and evaluation-point commitment schemes. (See Appendices A.2 and A.3.) All described commitment schemes are succinct.

$\mathsf{FC}_{\mathsf{sn}}$ achieves aggregation easily in a more general sense. Let $\mathcal{C}_i$ be some circuits for which efficient SFC schemes exist. We can then construct an efficient SFC for the circuit that consists of the sequential composition of $\mathcal{C}_i$-s. In particular, we can aggregate multiple polynomial commitment schemes, some vector commitment schemes, and say an evaluation-point commitment scheme. Some of the referred papers [BBF19,LM19,BDFG20,TAB+20,GRWZ20] construct aggregated commitment schemes for a concrete circuit (e.g., an aggregated polynomial commitment scheme). Importantly, $\mathsf{FC}_{\mathsf{sn}}$ allows one to aggregate different SFC schemes.

**Security.** The correctness and perfect hiding proofs are straightforward. The main thing worthy of note here is that we have three definitions of hiding (com-hiding, open-hiding, and zero-knowledge, see Section 2). For the sake of completeness, we also give three different hiding proofs. The SFC schemes must work in the CRS model to obtain zero knowledge. However, since zero-knowledge is stronger than the other two definitions, the proof of zero-knowledge, which follows roughly from the zero-knowledge of the related SNARK, suffices. Note that say [LRY16] only considered the weakest hiding notion (com-hiding).

The evaluation-binding proof differs significantly from the knowledge-soundness proofs of SNARKs. The knowledge-soundness of SNARKs can only be proven under non-falsifiable assumptions [GW11]. In particular, Groth proved the knowledge-soundness of the SNARK from [Gro16] in the generic group model while Lipmaa [Lip19] proved it under a tautological knowledge assumption and a known computational assumption (namely, $q$-PDL [Lip12]). Such assumptions have very little in common with assumptions we use. As expected, a knowledge-soundness proof that uses non-falsifiable assumptions has a very different flavor compared to an evaluation-binding proof that only uses falsifiable assumptions. We emphasize it is not clear a priori that an SFC constructed from SNARKs could rely on falsifiable assumptions.

We prove the evaluation-binding of $\mathsf{FC}_{\mathsf{sn}}$ under the new $(\mathcal{R}, \mathcal{S}, \{f_i\})$-*computational span-uber-assumption* in source group $\mathbb{G}_1$, where $\mathcal{R}, \mathcal{S} \subset \mathbb{Z}_p[X, Y]$ and $f_i \in \mathbb{Z}_p[X, Y]$ with $f_i \notin \mathrm{span}(\mathcal{R})$ are fixed by the circuit $\mathcal{C}$. This assumption states that given a commitment key $\mathsf{ck} = ([\varrho(\chi, y) : \varrho \in \mathcal{R}]_1, [\sigma(\chi, y) : \sigma \in \mathcal{S}]_2)$, where $\chi, y$ are random trapdoors, it is difficult to compute $(\boldsymbol{\Delta} \neq \mathbf{0}, \sum_{i=1}^{\kappa} \Delta_i [f_i(\chi, y)]_1)$, where $\boldsymbol{\Delta}$ is adversarially chosen. (See Definition 6 for a formal definition.) Importantly, if $\kappa = 1$ then we just have an uber-assumption in $\mathbb{G}_1$. We show that (see Theorem 2), for all $\mathcal{R}$ and $f_i$ needed by our new SFC, $f_i(X, Y) \notin \mathrm{span}(\mathcal{R})$.

The full evaluation-binding proof is quite tricky and relies significantly on the structure of matrices $U$, $V$, $W$ and the commitment key. Given a collision, we "almost" compute $(\boldsymbol{\Delta}, \sum \Delta_i [f_i(\chi, y)]_1)$, where $\boldsymbol{\Delta}$ is the component-wise difference between two claimed values of $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}, \boldsymbol{\beta})$. To eliminate "almost" in the previous sentence, the committer outputs $\kappa$ additional "helper" elements $[\mathsf{B}_i^{\mathsf{aux}}]_1$, where extra care has to be used to guarantee that the helper elements can be computed given the commitment key. In both cases, to succeed, we

need to assume that the matrices $(U, V, W)$ satisfy some natural restrictions stated in individual theorems. These restrictions, that hold for the compiled circuit, are collected together in Theorem 1.

**Analysis of the Span-Uber-Assumption.** The span-uber-assumption is falsifiable and, thus, significantly more realistic than non-falsifiable (knowledge) assumptions needed to prove the adaptive soundness of SNARKs. Still, it is a new assumption and thus we have written down *three* different proofs that it follows from already known assumptions. (See Lemma 2 and Theorems 4 and 5.)

In Lemma 2, we prove that the span-uber-assumption in $\mathbb{G}_1$ holds under the known $(\mathcal{R}, \mathcal{S}, f'_i)$-computational uber-assumption in the target group $\mathbb{G}_T$ [BBG05]. Here, $f'_i$ are different from but related to $f_i$. We also prove that $f'_i \notin \mathrm{span}(\mathcal{R}\mathcal{S})$. Since $f_i(X, Y) \notin \mathrm{span}(\mathcal{R})$ and $f'_i(X, Y) \notin \mathrm{span}(\mathcal{R}\mathcal{S})$ (in the case of the uber-assumption in $\mathbb{G}_T$), we have an instantiation of the computational uber-assumption, known to be secure [BBG05] in the generic group model.

Since the generic group model is very restrictive and has known weaknesses [Fis00,Den02] not shared by well-chosen knowledge assumptions, we will use the newer methodology of [FKL18]. In Appendix B, Theorem 5, we prove that if $f_i \notin \mathrm{span}(\mathcal{R})$ then the $(\mathcal{R}, \mathcal{S}, \{f_i\})$-computational span-uber-assumption in $\mathbb{G}_1$ holds in the algebraic group model (AGM, [FKL18]) under a PDL assumption. Since uber-assumption in $\mathbb{G}_T$ is not secure in the original AGM of [FKL18] (the latter only handles the case the adversary outputs elements in source groups since the target group is non-generic), this result is orthogonal to the previous result. As a corollary of independent interest, we get that if $f_i(X, Y) \notin \mathrm{span}(\mathcal{R})$ then uber-assumption in $\mathbb{G}_1$ holds in the AGM under a PDL assumption.

In composite-order bilinear groups, the computational uber-assumption in $\mathbb{G}_T$ holds under a subgroup hiding assumption [CMM16]. Thus, due to Lemma 2, a composite-order group span-uber-assumption (and also the new SFC) is secure under a subgroup hiding assumption. In Theorem 4, we use the Déjà Q approach of [CM14] to prove that the span-uber-assumption in $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$, is secure under a subgroup hiding assumption. This proof is more direct than the reduction through an uber-assumption in $\mathbb{G}_T$. Moreover, the Déjà Q approach is more applicable if one is working in the source group. Whether a similar reduction holds in the case of prime-order groups is an interesting open question.

**Efficiency.** It is tedious to provide a detailed efficiency comparison of our newly constructed scheme to all the abundant existing work in all applications. $\mathsf{FC}_{\mathsf{sn}}$ is *generic*, works for a large class of circuits, and can tackle scenarios, not possible with previous work, but at the same time, it can also be used to solve the much simpler case of, e.g., inner product. We stress that $\mathsf{FC}_{\mathsf{sn}}$, when straightforwardly specialized to the IPFC case, is nearly as efficient as the most efficient known prior IPFC, losing ground only in the CRS length. On the other hand, we are not aware of *any* previous aggregated IPFC schemes (See Appendix A.2).

This paper uses heavily a yet unpublished paper [Lip19] of the first author.

## 2 Preliminaries

If $\mathcal{R} = (\varrho_1(\boldsymbol{X}), \ldots, \varrho_n(\boldsymbol{X}))$ is a tuple of polynomials over $\mathbb{Z}_p[\boldsymbol{X}]$ and $\boldsymbol{x}$ is a vector of integers then $\mathcal{R}(\boldsymbol{x}) := (\varrho_1(\boldsymbol{x}), \ldots, \varrho_n(\boldsymbol{x}))$. Let $\mathbb{Z}_p^{(\le d)}[X]$ be the set of degree-$\le d$ polynomials over $\mathbb{Z}_p$. For a matrix $U$, let $\boldsymbol{U}_i$ be its $i$th row, $\boldsymbol{U}^{(j)}$ be its $j$th column. Let $\boldsymbol{a} \circ \boldsymbol{b}$ denote the component-wise product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, $(\boldsymbol{a} \circ \boldsymbol{b})_i = a_i b_i$. Let $\boldsymbol{a}_1 // \ldots // \boldsymbol{a}_n = \begin{pmatrix} \boldsymbol{a}_1 \\ \cdots \\ \boldsymbol{a}_n \end{pmatrix}$ denote the vertical concatenation of vectors $\boldsymbol{a}_i$. $\lambda$ is the security parameter, and $1^\lambda$ denotes its unary representation. PPT denotes probabilistic polynomial-time. For an algorithm $\mathcal{A}$, $\mathrm{range}(\mathcal{A})$ is the range of $\mathcal{A}$, that is, the set of valid outputs of $\mathcal{A}$, $\mathsf{RND}_\lambda(\mathcal{A})$ denotes the random tape of $\mathcal{A}$ (assuming the given value of $\lambda$). Finally, $r \leftarrow\!\!\$\, \mathcal{S}$ denotes the uniformly random choice of a randomizer $r$ from the set/distribution $\mathcal{S}$.

*Interpolation.* Assume $\nu$ is a power of two, and let $\omega$ be the $\nu$th primitive root of unity modulo $p$. Such $\omega$ exists, given that $\nu \mid (p-1)$. Then,

- $\ell(X) := \prod_{i=1}^{\nu}(X-\omega^{i-1}) = X^\nu - 1$ is the unique degree $\nu$ monic polynomial such that $\ell(\omega^{i-1}) = 0$ for all $i \in [1,\nu]$.
- For $i \in [1,\nu]$, $\ell_i(X)$ is the *ith Lagrange basis polynomial*, that is, the unique degree $\nu-1$ polynomial, such that $\ell_i(\omega^{i-1}) = 1$ and $\ell_i(\omega^{j-1}) = 0$ for $i \neq j$. Clearly, $\ell_i(X) := \ell(X)/(\ell'(\omega^{i-1})(X-\omega^{i-1})) = (X^\nu-1)\omega^{i-1}/(\nu(X-\omega^{i-1}))$.

Moreover, $(\ell_j(\omega^{i-1}))_{i=1}^{\nu} = \boldsymbol{e}_j$ (the $j$th unit vector) and $(\ell(\omega^{i-1}))_{i=1}^{\nu} = \mathbf{0}_\nu$.

*Bilinear Pairings.* Let $\nu$ be an integer parameter (the circuit size in our application). A bilinear group generator $\mathsf{Pgen}(1^\lambda,\nu)$ returns $(p,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T,\hat{e},\mathsf{P}_1,\mathsf{P}_2)$, where $\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T$ are three additive cyclic groups of prime order $p$, $\hat{e}:\mathbb{G}_1\times\mathbb{G}_2\to\mathbb{G}_T$ is a non-degenerate efficiently computable bilinear pairing, and $\mathsf{P}_\iota$ is a fixed generator of $\mathbb{G}_\iota$. We assume $\mathsf{P}_T = \hat{e}(\mathsf{P}_1,\mathsf{P}_2)$. We require the bilinear pairing to be Type-3, that is, there is no efficient isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. For efficient interpolation, we assume that $p$ is such that $\nu \mid (p-1)$. When emphasizing efficiency is not important, we drop the parameter $\nu$ and just write $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda)$. We use additive notation together with the standard elliptic-curve "bracket" notation. Namely, we write $[a]_\iota$ to denote $a\mathsf{P}_\iota$, and $[a]_1 \bullet [b]_2$ to denote $\hat{e}([a]_1,[b]_2)$. We use freely the bracket notation together with matrix notation, e.g., if $AB = C$ as matrices then $[A]_1 \bullet [B]_2 = [C]_T$.

*Uber-Assumption.* The following assumption is a special case of the more general uber-assumption of [BBG05,Boy08].

**Definition 1 ([BBG05,Boy08]).** *Let $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda)$. Let $\mathcal{R}$, $\mathcal{S}$, and $\mathcal{T}$ be three tuples of bivariate polynomials from $\mathbb{Z}_p[X,Y]$. Let $f$ be a bivariate polynomial from $\mathbb{Z}_p[X,Y]$. The $(\mathcal{R},\mathcal{S},\mathcal{T},f)$-*computational uber-assumption *for $\mathsf{Pgen}$ in group $\mathbb{G}_\iota$, where $\iota \in \{1,2,T\}$, states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{uber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},f,\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, where $\mathsf{Adv}^{\mathrm{uber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},f,\mathcal{A}}(\lambda) :=$*

$$\Pr\begin{bmatrix}\mathsf{p}\leftarrow\mathsf{Pgen}(1^\lambda);\chi,y\leftarrow\$\,\mathbb{Z}_p^*;\mathsf{ck}\leftarrow([\mathcal{R}(\chi,y)]_1,[\mathcal{S}(\chi,y)]_2,[\mathcal{T}(\chi,y)]_T):\\ \mathcal{A}(\mathsf{ck}) = [f(\chi,y)]_\iota\end{bmatrix}.$$

[BBG05,Boy08] considered the general case of $c$-variate polynomials for any $c$. In our case, $\mathcal{T} = \emptyset$; then, we have an $(\mathcal{R},\mathcal{S},f)$-computational uber-assumption in $\mathbb{G}_\iota$.

Importantly [BBG05,Boy08], (i) if $f(X,Y)$ is not in the span of $\{\varrho(X,Y)\}$ then the $(\mathcal{R},\mathcal{S},\mathcal{T},f)$-computational uber-assumption for $\mathbb{G}_1$ holds in the generic group model, and (ii) if $f(X,Y)$ is not in the span of $\{\varrho(X,Y)\sigma(X,Y)+\tau(X,Y)\}$ then the $(\mathcal{R},\mathcal{S},\mathcal{T},f)$-computational uber-assumption for $\mathbb{G}_T$ holds in the generic group model. We will only invoke the uber-assumption when it is secure in the generic group model.

*QAP.* Let $\mathbf{R} = \{(\mathbb{x},\mathbb{w})\}$ be a relation between statements and witnesses. Quadratic Arithmetic Program (QAP) was introduced in [GGPR13] as a language where for an input $\mathbb{x}$ and witness $\mathbb{w}$, $(\mathbb{x},\mathbb{w}) \in \mathbf{R}$ can be verified by using a parallel quadratic check. QAP has an efficient reduction from the (either Boolean or Arithmetic) Circuit-SAT. Thus, an efficient zk-SNARK for QAP results in an efficient zk-SNARK for Circuit-SAT.

We consider arithmetic circuits that consist only of fan-in-2 multiplication gates, but either input of each multiplication gate can be any weighted sum of wire values, [GGPR13]. Let $\mu_0 < \mu$ be a non-negative integer. In the case of arithmetic circuits, $\nu$ is the number of multiplication gates, $\mu$ is the number of wires, and $\mu_0$ is the number of public inputs.

Let $\mathbb{F} = \mathbb{Z}_p$, such that $\omega$ is the $\nu$-th primitive root of unity modulo $p$. This requirement is needed for the sake of efficiency, and we will make it implicitly throughout the paper. However, it is not needed for the new SFC to work. Let $U$, $V$, and $W$ be instance-dependent matrices and let $\mathbf{a}$ be a witness. A QAP is characterized by the constraint $U\mathbf{a} \circ V\mathbf{a} = W\mathbf{a}$. Let $L_{\boldsymbol{a}}(X) := \sum_{i=1}^{\nu} a_i\ell_i(X)$ be the interpolating polynomial of $\boldsymbol{a} = (a_1,\ldots,a_\nu)^\top$ at points $\omega^{i-1}$, with $L_{\boldsymbol{a}}(\omega^{i-1}) = a_i$ for $i \in [1,\nu]$. For $j \in [1,\mu]$, define $u_j(X) := L_{\boldsymbol{U}^{(j)}}(X)$, $v_j(X) := L_{\boldsymbol{V}^{(j)}}(X)$, and $w_j(X) := L_{\boldsymbol{W}^{(j)}}(X)$ to be interpolating polynomials of the $j$th column of the corresponding matrix. Thus, $u_j,v_j,w_j \in \mathbb{Z}_p^{(\leq\nu-1)}[X]$. Let $u(X) = \sum_{j=1}^{\mu}\mathsf{a}_j u_j(X)$, $v(X) = \sum_{j=1}^{\mu}\mathsf{a}_j v_j(X)$,

and $w(X) = \sum_{j=1}^{\mu} \mathsf{a}_j w_j(X)$. Then $U\mathbf{a} \circ V\mathbf{a} = W\mathbf{a}$ iff $\ell(X) \mid u(X)v(X) - w(X)$ iff $u(X)v(X) \equiv w(X) \pmod{\ell(X)}$ iff there exists a polynomial $\mathcal{H}(X)$ such that $u(X)v(X) - w(X) = \mathcal{H}(X)\ell(X)$.

A QAP instance $\mathcal{I}_{\mathsf{qap}}$ is equal to $(\mathbb{Z}_p, \mu_0, \{u_j, v_j, w_j\}_{j=1}^{\mu})$. $\mathcal{I}_{\mathsf{qap}}$ defines the following relation:

$$\mathbf{R}_{\mathcal{I}_{\mathsf{qap}}} = \left\{(\mathbb{x}, \mathbb{w})\colon \mathbb{x} = (\mathsf{a}_1, \ldots, \mathsf{a}_{\mu_0})^\top \wedge \mathbb{w} = (\mathsf{a}_{\mu_0+1}, \ldots, \mathsf{a}_{\mu})^\top \wedge u(X)v(X) \equiv w(X) \pmod{\ell(X)}\right\} \ , \quad (1)$$

where $u(X)$, $v(X)$, and $w(X)$ are as above. Alternatively, $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$ if there exists a (degree $\leq \nu - 2$) polynomial $\mathcal{H}(X)$, such that the following key equation holds:

$$\chi(X) := u(X)v(X) - w(X) - \mathcal{H}(X)\ell(X) = 0 \ . \quad (2)$$

On top of checking Eq. (2), the verifier also needs to check that $u(X)$, $v(X)$, and $w(X)$ are correctly computed: that is, (i) the first $\mu_0$ coefficients $\mathsf{a}_j$ in $u(X)$ are equal to the public inputs, and (ii) $u(X)$, $v(X)$, and $w(X)$ are all computed by using the same coefficients $\mathsf{a}_j$ for $j \leq \mu$.

Since in a functional commitment scheme, both the committer and the verifier have inputs, we will use a variation of QAP that handles public inputs differently (see Section 3). In particular, we will use different parameters instead of $\mu_0$.

*SNARKs.* Let $\mathcal{R}$ be a relation generator, such that $\mathcal{R}(1^\lambda)$ returns a polynomial-time decidable binary relation $\mathbf{R} = \{(\mathbb{x}, \mathbb{w})\}$. Here, $\mathbb{x}$ is a statement, and $\mathbb{w}$ is a witness. $\mathcal{R}$ also outputs the system parameters $\mathsf{p}$ that will be given to the honest parties and the adversary. A *non-interactive zero-knowledge (NIZK) argument system* $\Psi = (\mathsf{K}_{\mathsf{crs}}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$ for $\mathcal{R}$ consists of four PPT algorithms:

**CRS generator:** $\mathsf{K}_{\mathsf{crs}}$ is a probabilistic algorithm that, given $(\mathbf{R}, \mathsf{p}) \in \mathrm{range}(\mathcal{R}(1^\lambda))$, outputs $(\mathsf{crs}, \mathsf{td})$ where $\mathsf{crs}$ is a CRS and $\mathsf{td}$ is a simulation trapdoor. Otherwise, it outputs a special symbol $\perp$.

**Prover:** $\mathsf{P}$ is a probabilistic algorithm that, given $(\mathbf{R}, \mathsf{p}, \mathsf{crs}, \mathbb{x}, \mathbb{w})$ for $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$, outputs an argument $\pi$. Otherwise, it outputs $\perp$.

**Verifier:** $\mathsf{V}$ is a probabilistic algorithm that, given $(\mathbf{R}, \mathsf{p}, \mathsf{crs}, \mathbb{x}, \pi)$, returns either 0 (reject) or 1 (accept).

**Simulator:** $\mathsf{Sim}$ is a probabilistic algorithm that, given $(\mathbf{R}, \mathsf{p}, \mathsf{crs}, \mathsf{td}, \mathbb{x})$, outputs an argument $\pi$.

A NIZK argument system must satisfy completeness (an honest verifier accepts an honest prover), knowledge-soundness (if a prover makes an honest verifier accept, then one can extract from the prover the corresponding witness $\mathbb{w}$), and zero-knowledge (there exists a simulator that, knowing the CRS trapdoor but not the witness, can produce accepting statements with the verifier's view being indistinguishable from the view when interacting with an honest prover). See Appendix C for formal definitions. A *SNARK (succinct non-interactive argument of knowledge, [Gro10,Lip12,GGPR13,PHGR13,Lip13,Gro16])* is a NIZK argument system where the argument is sublinear in the input size.

*Functional Commitment Schemes.* Let $\mathcal{D}$ be some domain. In a functional commitment scheme for a circuit $\mathcal{C} : \mathcal{D}^{\mu_\alpha} \times \mathcal{D}^{\mu_\beta} \to \mathcal{D}^{\kappa}$, the committer first commits to a vector $\boldsymbol{\alpha} \in \mathcal{D}^{\mu_\alpha}$, obtaining a functional commitment $C$. The goal is to allow the committer to later open $C$ to $\boldsymbol{\xi} = \mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathcal{D}^{\kappa}$, where $\boldsymbol{\beta} \in \mathcal{D}^{\mu_\beta}$ is a public input that is chosen by the verifier before the opening. We generalize the notion of functional commitment, given in [LRY16], from inner products to arbitrary circuits. Compared to [LRY16], we also provide a stronger hiding definition.

Let $\mathbf{CC}$ be a class of circuits $\mathcal{C} : \mathcal{D}^{\mu_\alpha} \times \mathcal{D}^{\mu_\beta} \to \mathcal{D}^{\kappa}$. A *functional commitment scheme* $\mathsf{FC}$ *for* $\mathbf{CC}$ is a tuple of four (possibly probabilistic) polynomial-time algorithms $(\mathsf{KC}, \mathsf{com}, \mathsf{open}, \mathsf{V})$, where

**Commitment-key generator:** $\mathsf{KC}(1^\lambda, \mathcal{C})$ is a probabilistic algorithm that, given a security parameter $\lambda \in \mathbb{N}$ and a circuit $\mathcal{C} \in \mathbf{CC}$, outputs a commitment key $\mathsf{ck}$ and a trapdoor key $\mathsf{tk}$. We implicitly assume $1^\lambda$ and $\mathcal{C}$ are described by $\mathsf{ck}$.

**Commitment:** $\mathsf{com}(\mathsf{ck}, \boldsymbol{\alpha}; r)$ is a probabilistic algorithm that takes as input a commitment key $\mathsf{ck}$, a message vector $\boldsymbol{\alpha} \in \mathcal{D}^{\mu_\alpha}$ and a randomizer $r$. It outputs $(C, D)$, where $C$ is a commitment to $\boldsymbol{\alpha}$ and $D$ is a decommitment information. We denote the first output $C$ of $\mathsf{com}(\mathsf{ck}; \boldsymbol{\alpha}; r)$ by $\mathsf{com}_1(\mathsf{ck}; \boldsymbol{\alpha}; r)$.

**Opening:** $\mathsf{open}(\mathsf{ck}, C, D, \boldsymbol{\beta})$ is a deterministic algorithm that takes as input a commitment key $\mathsf{ck}$, a commitment $C$ (to $\boldsymbol{\alpha}$), a decommitment information $D$, and a vector $\boldsymbol{\beta} \in \mathcal{D}^{\mu_\beta}$. Assume that the $i$th output value of the circuit $\mathcal{C}$ is $\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta})$, where $\mathcal{F}_i$ is a public function. It computes an opening $\mathrm{op}_{\boldsymbol{\xi}}$ to $\boldsymbol{\xi} = \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}, \boldsymbol{\beta}) := (\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta}))_{i=1}^{\kappa}$.

**Verification:** $\mathsf{V}(\mathsf{ck}, C, \mathrm{op}_{\boldsymbol{\xi}}, \boldsymbol{\beta}, \boldsymbol{\xi})$ is a deterministic algorithm that takes as input a commitment key $\mathsf{ck}$, a commitment $C$, an opening $\mathrm{op}_{\boldsymbol{\xi}}$, a vector $\boldsymbol{\beta} \in \mathcal{D}^{\mu_\beta}$, and $\boldsymbol{\xi} \in \mathcal{D}^{\kappa}$. It outputs 1 if $\mathrm{op}_{\boldsymbol{\xi}}$ is a valid opening for $C$ being a commitment to some $\boldsymbol{\alpha} \in \mathcal{D}^{\mu_\alpha}$ such that $\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \boldsymbol{\xi}$ and outputs 0 otherwise.

*Security of FC.* Next, we give three definitions of the hiding property for FC schemes of increasing strength. The first definition corresponds to the definition of hiding given in [LRY16] and essentially states that commitments do not reveal any information about $\boldsymbol{\alpha}$. The other two definitions seem to be novel at least in the context of general FC. We provide all three definitions, since in some applications, a weaker definition might be sufficient. Moreover, the third definition (zero-knowledge) makes only sense in the CRS model; in a CRS-less model, one can rely on the open-hiding property.

**Definition 2 (Perfect com-hiding).** *A functional commitment scheme* $\mathsf{FC} = (\mathsf{KC}, \mathsf{com}, \mathsf{open}, \mathsf{V})$ *for circuit class* $\mathbf{CC}$ *is* perfectly hiding *if for any* $\lambda$, $\mathcal{C} \in \mathbf{CC}$, $(\mathsf{ck}, \mathsf{tk}) \leftarrow \mathsf{KC}(1^\lambda, \mathcal{C})$, *and* $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathcal{D}^{\mu_\alpha}$ *with* $\boldsymbol{\alpha}_1 \neq \boldsymbol{\alpha}_2$, *the two distributions* $\delta_1$ *and* $\delta_2$ *are identical, where*

$$\delta_b := \{(\mathsf{ck}, C_b) : r \leftarrow\!\!\$\, \mathsf{RND}_\lambda(\mathsf{com}); (C_b, D_b) \leftarrow \mathsf{com}(\mathsf{ck}, \boldsymbol{\alpha}_b; r)\} \ .$$

The open-hiding property is considerably stronger, stating that the commitment and the openings together do not reveal more information on $\boldsymbol{\alpha}$ than the values $\mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta}_i)$ on queried values $\boldsymbol{\beta}_i$. Trivial non-succinct FC schemes, where one uses a perfectly-hiding commitment scheme to commit to $\boldsymbol{\beta}$, and then in the opening phase, opens the whole database, are com-hiding but not open-hiding.

**Definition 3 (Perfect open-hiding).** *A functional commitment scheme* $\mathsf{FC} = (\mathsf{KC}, \mathsf{com}, \mathsf{open}, \mathsf{V})$ *for circuit class* $\mathbf{CC}$ *is* perfectly open-hiding *if for any* $\lambda$, $\mathcal{C} \in \mathbf{CC}$, $(\mathsf{ck}, \mathsf{tk}) \leftarrow \mathsf{KC}(1^\lambda, \mathcal{C})$, *for all* $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathcal{D}^{\mu_\alpha}$ *with* $\boldsymbol{\alpha}_1 \neq \boldsymbol{\alpha}_2$, *and* $Q = \mathsf{poly}(\lambda)$ *of* $\boldsymbol{\beta}_i$ *such that* $\mathcal{C}(\boldsymbol{\alpha}_1, \boldsymbol{\beta}_i) = \mathcal{C}(\boldsymbol{\alpha}_2, \boldsymbol{\beta}_i)$ *for all* $i \leq Q$, *the two distributions* $\delta_1$ *and* $\delta_2$ *are identical, where*

$$\delta_b := \{(\mathsf{ck}, C_b, \{\mathsf{open}(\mathsf{ck}, C_b, D_b, \boldsymbol{\beta}_i)\}) : r \leftarrow\!\!\$\, \mathsf{RND}_\lambda(\mathsf{com}); (C_b, D_b) \leftarrow \mathsf{com}(\mathsf{ck}, \boldsymbol{\alpha}_b; r)\} \ .$$

Finally, zero-knowledge FC schemes have simulation-based hiding. While simulation-based security is a gold standard in cryptography, it is usually more complicated to achieve than game-based security. In particular, one needs (at least, when not using random oracles) to have a trusted $\mathsf{ck}$ (and its trapdoor) to achieve zero-knowledge. We will leave it as an open problem whether one can use instead the much weaker bare public key (BPK) model (that is, subversion-secure), by using the techniques of [BFS16,ABLZ17,Fuc18,ALSZ20,ALSZ21]. Note that [Lip19] showed that their SNARKs are all secure in the BPK model.

**Definition 4 (Perfect zero-knowledge).** *An FC scheme* $\mathsf{FC} = (\mathsf{KC}, \mathsf{com}, \mathsf{open}, \mathsf{V})$ *for* $\mathbf{CC}$ *is* perfectly zero-knowledge *if there exists a PPT simulator* $\mathsf{Sim}$, *such that for all* $\lambda$, *all* $\mathcal{C} \in \mathbf{CC}$, $(\mathsf{ck}, \mathsf{tk}) \leftarrow \mathsf{KC}(1^\lambda, \mathcal{C})$, *for all* $\boldsymbol{\alpha} \in \mathcal{D}^{\mu_\alpha}$, *for any poly-size set of* $\boldsymbol{\beta}_i$, $\delta_0$ *and* $\delta_1$ *are identical, where*

$$\begin{aligned}\delta_0 &:= \{(\mathsf{ck}, C, \{\mathsf{open}(\mathsf{ck}, C, D, \boldsymbol{\beta}_i)\}) : r \leftarrow\!\!\$\, \mathsf{RND}_\lambda(\mathsf{com}); (C, D) \leftarrow \mathsf{com}(\mathsf{ck}, \boldsymbol{\alpha}; r)\} \ , \\ \delta_1 &:= \{(\mathsf{ck}, \mathsf{Sim}(\mathsf{ck}, \mathsf{td}, \{\boldsymbol{\beta}_i\}, \{\mathcal{C}(\boldsymbol{\alpha}, \boldsymbol{\beta}_i)\}))\} \ .\end{aligned}$$

Next, we will define evaluation-binding. Evaluation-binding can be weaker than binding, but sometimes the two notions are equivalent. (Consider the case of the inner product when the adversary asks the committer to open a commitment for $\boldsymbol{\beta} = \boldsymbol{e}_i$ for each $i$.) In the context of FC schemes, evaluation-binding is *the* distinguishing security notion.

**Definition 5 (Computational evaluation-binding).** *A functional commitment scheme* $\mathsf{FC} = (\mathsf{KC}, \mathsf{com}, \mathsf{open}, \mathsf{V})$ *for circuit class* $\mathbf{CC}$ *is* computationally evaluation-binding *if for any* $\lambda$, $\mathcal{C} \in \mathbf{CC}$, *and a non-uniform PPT adversary* $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{bind}}_{\mathsf{FC},\mathcal{C},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, *where*

$$\mathsf{Adv}^{\mathrm{bind}}_{\mathsf{FC},\mathcal{C},\mathcal{A}}(\lambda) := \Pr\begin{bmatrix}(\mathsf{ck}, \mathsf{tk}) \leftarrow \mathsf{KC}(1^\lambda, \mathcal{C}); (C, \boldsymbol{\beta}, \boldsymbol{\xi}, \mathrm{op}_{\boldsymbol{\xi}}, \widetilde{\boldsymbol{\xi}}, \widetilde{\mathrm{op}}_{\boldsymbol{\xi}}) \leftarrow \mathcal{A}(\mathsf{ck}) : \boldsymbol{\beta} \in \mathcal{D}^{\mu_\beta} \wedge \\ \boldsymbol{\xi} \neq \widetilde{\boldsymbol{\xi}} \in \mathcal{D}^{\kappa} \wedge \mathsf{V}(\mathsf{ck}, C, \mathrm{op}_{\boldsymbol{\xi}}, \boldsymbol{\beta}, \boldsymbol{\xi}) = \mathsf{V}(\mathsf{ck}, C, \widetilde{\mathrm{op}}_{\boldsymbol{\xi}}, \boldsymbol{\beta}, \widetilde{\boldsymbol{\xi}}) = 1\end{bmatrix} \ .$$

An FC scheme is *succinct* (SFC), if both the commitments and openings have length that is polylogarithmic in $|\boldsymbol{\alpha}|$ and $|\boldsymbol{\beta}|$.

## 3 The New SFC Scheme

In this section, we will construct a succinct functional commitment (SFC) scheme for a large class of polynomial-size arithmetic circuits by mixing techniques from SNARKs with original ideas, needed to construct an SFC scheme. Let $\boldsymbol{\mathcal{F}}$ be a fixed vector function that takes inputs from two parties, the committer and the verifier. Let $\mathcal{C}$ be an arithmetic circuit that inputs $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ and computes $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) = (\mathcal{F}_i(\boldsymbol{\alpha},\boldsymbol{\beta}))_{i=1}^{\kappa}$, where $\boldsymbol{\alpha}$ is the private input of the committer, used in the commitment, and $\boldsymbol{\beta}$ is the public input of the verifier, possibly chosen only when opening the commitment. We compile $\mathcal{C}$ to a circuit $\mathcal{C}^*$ that consists of four subcircuits $\mathcal{C}_\phi$, $\mathcal{C}_\psi$, $\mathcal{C}_\chi$ and $\mathcal{C}_\xi$. We need the division to four subcircuits to prove evaluation-binding; we will give more details later.

After that, we use the QAP-representation [GGPR13] (more precisely, the approach of [Lip19]) of arithmetic circuits, obtaining polynomials $\mathsf{A}(X,Y)$, $\mathsf{B}(X,Y)$ (the "commitment polynomials" to all left/right inputs of all gates of $\mathcal{C}^*$, correspondingly), and $\mathsf{C}(X,Y)$ (the "opening polynomial"), such that $\mathsf{C}(X,Y)$ is in the linear span of the "polynomial commitment key" $\mathsf{ck}_1 = (\varrho(X,Y) : \varrho \in \mathcal{R})$ if and only if the committer was honest. The circuit compilation allows us to additively divide the polynomials to "private" parts (transmitted during the commitment) and "public" parts (trasmitted during the opening), such that one can, given two different openings for the same commitment, break a computational assumption. We then use SNARK-based techniques to construct the SFC for $\mathcal{C}^*$ with succinct commitment and opening. We postpone security proofs to Section 5; here, we emphasize that the evaluation-binding proof is novel (in particular, not related to the knowledge-soundness proofs of SNARKs at all).

**Circuit Compilation.** Let $\mathcal{C}$ be a polynomial-size arithmetic circuit that, on input $(\boldsymbol{\alpha},\boldsymbol{\beta})$, outputs $\boldsymbol{\xi} = \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) = (\mathcal{F}_i(\boldsymbol{\alpha},\boldsymbol{\beta}))_{i=1}^{\kappa}$. We compile $\mathcal{C}$ to a *compiled circuit* $\mathcal{C}^*$ that implements the same functionality. $\mathcal{C}^*$, see Fig. 1, consists of the public subcircuits $\mathcal{C}_\phi$, $\mathcal{C}_\psi$, $\mathcal{C}_\chi$, and $\mathcal{C}_\xi$ that are combined as follows. In the commitment phase, the committer uses the circuit $\mathcal{C}_\phi$ to compute a number of polynomials $\phi_i(\boldsymbol{\alpha})$ depending on only 1 and $\boldsymbol{\alpha}$. More precisely, $\boldsymbol{\phi}(\boldsymbol{\alpha}) = (\phi_1(\boldsymbol{\alpha}),\ldots,\phi_{\mu_\phi}(\boldsymbol{\alpha}))$ denotes the set of the outputs of all (including intermediate) gates in $\mathcal{C}_\phi$ (the same is the case of other circuits and corresponding polynomials). The commitment depends only on 1, $\boldsymbol{\alpha}$, and $\boldsymbol{\phi}(\boldsymbol{\alpha})$. In the opening phase, the verifier sends $\boldsymbol{\beta}$ to the committer, who uses the circuit $\mathcal{C}_\psi$ to compute some polynomials $\psi_i(\boldsymbol{\beta})$ depending on 1 and $\boldsymbol{\beta}$. This part of the computation is public and can be redone by the verifier.

After that, the committer uses the circuit $\mathcal{C}_\chi$ to compute a number of polynomials $\chi_i(\boldsymbol{\alpha},\boldsymbol{\beta})$ from the inputs and outputs of $\mathcal{C}_\phi$ and $\mathcal{C}_\psi$, that is, from $(1,\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\phi}(\boldsymbol{\alpha}),\boldsymbol{\psi}(\boldsymbol{\beta}))$. $\mathcal{C}_\chi$ has multiplicative depth 1, which guarantees that each $\chi_i(\boldsymbol{\alpha},\boldsymbol{\beta})$ is a product of some $\phi_j(\boldsymbol{\alpha})$ with some $\psi_k(\boldsymbol{\beta})$. Finally, the committer uses $\mathcal{C}_\xi$ to compute the outputs $\mathcal{F}_i(\boldsymbol{\alpha},\boldsymbol{\beta})$ of $\mathcal{C}^*$. We will explain the need for such compilation after Eqs. (7) and (8). We will summarize all actual restrictions on the circuits in Theorem 1. In the introduction, we gave an intuitive explanation of how this compilation reduces the circuit class that we can handle. See Section 4 for an additional discussion on the power of this circuit class.

Next, let $\mathbf{a} \in \mathbb{Z}_p^{\mu}$ be the value of all wires of $\mathcal{C}^*$. We write

$$\mathbf{a} = 1//\boldsymbol{\alpha}//\boldsymbol{\phi}(\boldsymbol{\alpha})//\boldsymbol{\beta}//\boldsymbol{\psi}(\boldsymbol{\beta})//\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta})//\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \ . \tag{3}$$

Here, $\boldsymbol{\alpha} \in \mathbb{Z}_p^{\mu_\alpha}$, $\boldsymbol{\phi}(\boldsymbol{\alpha}) \in \mathbb{Z}_p^{\mu_\phi}$, $\boldsymbol{\beta} \in \mathbb{Z}_p^{\mu_\beta}$, $\boldsymbol{\psi}(\boldsymbol{\beta}) \in \mathbb{Z}_p^{\mu_\psi}$, $\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) \in \mathbb{Z}_p^{\mu_\chi}$, and $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \in \mathbb{Z}_p^{\kappa}$. Thus, $\mu = 1+\mu_\alpha+\mu_\beta+\mu_\phi+\mu_\psi+\mu_\chi+\kappa$. To use the RC1S approach, we construct matrices $U$, $V$, and $W$, such that $U\mathbf{a} \circ V\mathbf{a} = W\mathbf{a}$ iff $\mathcal{C}^*$ is correctly computed. Let $\boldsymbol{\alpha}^* = (1//\boldsymbol{\alpha}//\boldsymbol{\phi}(\boldsymbol{\alpha})) \in \mathbb{Z}_p^{1+\mu_\alpha+\mu_\phi}$ and $\boldsymbol{\beta}^* = (1//\boldsymbol{\beta}//\boldsymbol{\psi}(\boldsymbol{\beta})) \in \mathbb{Z}_p^{1+\mu_\beta+\mu_\psi}$. First, we define matrices $U_\phi, U_\psi, U_\chi, U_\xi, V_\phi, V_\psi, V_\chi$ such that the subcircuits of $\mathcal{C}^*$ and thus $\mathcal{C}^*$ itself are correctly computed iff

$$\begin{gathered} U_\phi\boldsymbol{\alpha}^* \circ V_\phi\boldsymbol{\alpha}^* = \boldsymbol{\phi}(\boldsymbol{\alpha}) \ , \quad U_\psi\boldsymbol{\beta}^* \circ V_\psi\boldsymbol{\beta}^* = \boldsymbol{\psi}(\boldsymbol{\beta}) \ , \\ U_\chi\begin{pmatrix}\boldsymbol{\alpha}^*\\ \boldsymbol{\beta}\\ \boldsymbol{\psi}(\boldsymbol{\beta})\end{pmatrix} \circ V_\chi\begin{pmatrix}\boldsymbol{\alpha}^*\\ \boldsymbol{\beta}\\ \boldsymbol{\psi}(\boldsymbol{\beta})\end{pmatrix} = \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) \ , \quad U_\xi\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) \circ \mathbf{1} = \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \ . \end{gathered} \tag{4}$$

Here, $U_\phi, V_\phi \in \mathbb{Z}_p^{\mu_\phi \times (1+\mu_\alpha+\mu_\phi)}$, $U_\psi, V_\psi \in \mathbb{Z}_p^{\mu_\psi \times (1+\mu_\beta+\mu_\psi)}$, $U_\chi, V_\chi \in \mathbb{Z}_p^{\mu_\chi \times (1+\mu_\alpha+\mu_\beta+\mu_\phi+\mu_\psi)}$, and $U_\xi \in \mathbb{Z}_p^{\kappa \times \mu_\chi}$. In particular,

$$\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^{\mu_\chi} U_{\xi ij} \chi_j(\boldsymbol{\alpha}, \boldsymbol{\beta}) \ , \quad i \in [1, \kappa] \ . \tag{5}$$

Next, we define $U, V, W \in \mathbb{Z}_p^{\nu \times \mu}$, as

$$U = \left(\begin{array}{c|c|c|c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\phi}(\boldsymbol{\alpha}) & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline U_\phi & & & & & & \\ \hline U_\psi & & & U_\psi & & & \\ \hline U_\chi & & & & & & \\ \hline & & & & & U_\xi & \end{array}\right) \ , \ V = \left(\begin{array}{c|c|c|c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\phi}(\boldsymbol{\alpha}) & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline V_\phi & & & & & & \\ \hline V_\psi & & & V_\psi & & & \\ \hline V_\chi & & & & & & \\ \hline \mathbf{1}_\kappa & & & & & & \end{array}\right) \ , \ W = \left(\begin{array}{c|c|c|c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\phi}(\boldsymbol{\alpha}) & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & I_{\mu_\phi} & & & & \\ & & & & I_{\mu_\psi} & & \\ & & & & & I_{\mu_\chi} & \\ & & & & & & I_\kappa \end{array}\right) \ . \tag{6}$$

correspondingly. Clearly, $\nu = \mu_\phi + \mu_\psi + \mu_\chi + \kappa$. Here, we labeled vertically each column of each matrix by the supposed value of the corresponding coefficients of $\mathbf{a} = 1//\boldsymbol{\alpha}// \ldots //\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}, \boldsymbol{\beta})$. Some submatrices ($U_\psi$ and $V_\psi$) are divided between non-continuous areas. The empty submatrices are all-zero in the compiled instance. Clearly, $U\mathbf{a} \circ V\mathbf{a} = W\mathbf{a}$ iff Eq. (4) holds.

**QAP Representation.** Recall that $\ell_i(X) \in \mathbb{Z}_p^{(\leq \nu-1)}[X]$, $i \in [1, \nu]$, interpolates the $\nu$-dimensional unit vector $\boldsymbol{e}_i$. To obtain a QAP representation of the equation $U\mathbf{a} \circ V\mathbf{a} = W\mathbf{a}$, we use interpolating polynomials; e.g., $u_j(X)$ interpolates the $j$th column of $U$. (See Section 2.) To simplify notation, we introduce polynomials like $u_{\phi j}(X)$ and $u_{\chi j}(X)$, where say $u_{\chi j}(X)$ interpolates (*all $\nu$ rows of*) the $j$th column of the $\nu \times (1 + \mu_\alpha + \mu_\beta + \mu_\phi + \mu_\psi)$ submatrix of $U$ that contains $U_\chi$. More precisely, $u_{\chi j}(X) = \sum_{i=1}^{\mu_\chi} U_{\chi ij} \ell_{\mu_\phi+\mu_\psi+i}(X)$.

We divide additively the polynomials $u(X)$ and $v(X)$ into two polynomials: one polynomial ($u_s, v_s$, resp.) that depends on $\boldsymbol{\alpha}$ but not on $\boldsymbol{\beta}$, and another polynomial ($u_p, v_p$, resp.) that depends on public values ($\boldsymbol{\beta}$ and $\{\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta})\}$) but not on $\boldsymbol{\alpha}$ otherwise. Such a division is possible due to the way $\mathcal{C}^*$ is composed from the subcircuits. Thus, $u(X) = \sum_{j=1}^{\mu} \mathsf{a}_j u_j(X) = u_s(X) + u_p(X)$ and $v(X) = \sum_{j=1}^{\mu} \mathsf{a}_j v_j(X) = v_s(X) + v_p(X)$. By Eqs. (3) and (6), we get

$$\begin{aligned} u_s(X) =& \textstyle\sum_{j=2}^{\mu_\alpha+\mu_\phi+1} \mathsf{a}_j u_j(X) \\ =& \textstyle\sum_{j=1}^{\mu_\alpha} \alpha_j (u_{\phi,1+j}(X) + u_{\chi,1+j}(X)) + \sum_{j=1}^{\mu_\phi} \phi_j(\boldsymbol{\alpha})(u_{\phi,1+\mu_\alpha+j}(X) + u_{\chi,1+\mu_\alpha+j}(X)) \ , \\ u_p(X) =& u_1(X) + \textstyle\sum_{j=\mu_\alpha+\mu_\phi+2}^{\mu} \mathsf{a}_j u_j(X) \\ =& u_1(X) + \textstyle\sum_{j=1}^{\mu_\beta} \beta_j (u_{\psi,1+j}(X) + u_{\chi,1+\mu_\alpha+\mu_\phi+j}(X)) + \\ & \textstyle\sum_{j=1}^{\mu_\psi} \psi_j(\boldsymbol{\beta})(u_{\psi,1+\mu_\beta+j}(X) + u_{\chi,1+\mu_\alpha+\mu_\phi+\mu_\beta+j}(X)) + \underbrace{\textstyle\sum_{j=1}^{\mu_\chi} \chi_j(\boldsymbol{\alpha}, \boldsymbol{\beta}) u_{\xi,1+j}(X)}_{= \sum_{i=1}^{\kappa} \mathcal{F}_i(\boldsymbol{\alpha},\boldsymbol{\beta}) \ell_{\nu-\kappa+i}(X)} \ , \end{aligned} \tag{7}$$

where $u_1(X) = u_{\phi 1}(X) + u_{\psi 1}(X) + u_{\chi 1}(X)$, and

$$\begin{aligned} v_s(X) =& \textstyle\sum_{j=2}^{\mu_\alpha+\mu_\phi+1} \mathsf{a}_j v_j(X) \\ =& \textstyle\sum_{j=1}^{\mu_\alpha} \alpha_j (v_{\phi,1+j}(X) + v_{\chi,1+j}(X)) + \sum_{j=1}^{\mu_\phi} \phi_j(\boldsymbol{\alpha})(v_{\phi,1+\mu_\alpha+j}(X) + v_{\chi,1+\mu_\alpha+j}(X)) \ , \\ v_p(X) =& \mathsf{a}_1 v_1(X) + \textstyle\sum_{j=\mu_\alpha+\mu_\phi+2}^{\mu} \mathsf{a}_j v_j(X) \\ =& v_1(X) + \textstyle\sum_{j=1}^{\mu_\beta} \beta_j \left(v_{\psi,1+j}(X) + v_{\chi,1+\mu_\alpha+\mu_\phi+j}(X)\right) + \\ & \textstyle\sum_{j=1}^{\mu_\psi} \psi_j(\boldsymbol{\beta})(v_{\psi,1+\mu_\beta+j}(X) + v_{\chi,1+\mu_\alpha+\mu_\phi+\mu_\beta+j}(X)) \ , \end{aligned} \tag{8}$$

where $v_1(X) = v_{\phi 1}(X) + v_{\psi 1}(X) + v_{\chi 1}(X) + \sum_{i=1}^{\kappa} \ell_{\nu-\kappa+i}(X)$. Recall that $\mathsf{a}_1 = 1$.

In Theorems 2 and 3 (see their claims and proofs), we will need several conditions to hold. Next, we will state and prove that all these conditions hold for $\mathcal{C}^*$. One can observe directly that most of the guarantees, given by $\mathcal{C}^*$ about the shape of $U, V, W$, are actually required by the following conditions. Since the addition of the circuit $\mathcal{C}_\xi$ is essentially for free (it only means the addition of $\kappa$ gates), most of the following conditions are very easy to satisfy; we will denote such conditions by a superscript $+$ as in $(a)^+$. *We emphasize that the only restrictive conditions are Items i and j that basically state that $\mathcal{C}_\chi$ must have multiplicative depth* 1. (See Remark 1 for discussion.) That is, the new SFC scheme will work for all circuits $\mathcal{C}$ that have a polynomial-size compiled circuit $\mathcal{C}^*$, where $\mathcal{C}_\chi$ has multiplicative depth 1.

**Theorem 1.** *Let $\mathcal{C}$ be an arithmetic circuit and let $\mathcal{C}^*$ be its compiled version, so that $U, V, W$ are defined as in Eq.* (6)*. Then the following holds (we will summarize the conditions in the beginning of the proof):*

*(a)$^+$ For $j \in [1, \mu-\kappa]$: if $\boldsymbol{W}^{(j)} = \mathbf{0}$ then $U_{\nu-\kappa+i,j} = 0$ for $i \in [1,\kappa]$.*
*(b)$^+$ For $I \in [1,\kappa]$ and $j \in [1, \mu-\kappa]$, $W_{\nu-\kappa+I,j} = 0$.*
*(c)$^+$ For $j \in [2, 1+\mu_\alpha+\mu_\phi]$, $v_{\phi j}(X)$ and $v_{\chi j}(X)$ are in the span of $(\ell_i(X))_{i=1}^{\nu-\kappa}$.*
*(d)$^+$ $v_1(X) - \sum_{i=1}^{\kappa} \ell_{\nu-\kappa+i}(X)$ and $v_j(X)$, for $j \in [2+\mu_\alpha+\mu_\phi, \mu]$, are in the span of $(\ell_i(X))_{i=1}^{\nu-\kappa}$.*
*(e)$^+$ For $j \in [\mu-\kappa, \mu]$, $\boldsymbol{U}^{(j)} = \mathbf{0}$.*
*(f)$^+$ For $j \in [\mu-\kappa, \mu]$, $\boldsymbol{V}^{(j)} = \mathbf{0}$.*
*(g)$^+$ For $i \in [1,\kappa]$, $w_{\mu-\kappa+i}(X) = \ell_{\nu-\kappa+i}(X)$.*
*(h) The set of non-zero $\boldsymbol{W}^{(j)}$, $j \in [1, \mu-\kappa]$, is linearly independent.*
*(i) For $j \in [\mu-\mu_\chi-\kappa+1, \mu-\kappa]$, $U_{ij} = 0$ if $i \le \nu-\kappa$, while the last $\kappa$ rows of this column range define a matrix $U_\xi$ that satisfies Eq.* (4)*.*
*(j) For $j \in [\mu-\mu_\chi-\kappa+1, \mu-\kappa]$, $\boldsymbol{V}^{(j)} = \mathbf{0}$.*

*Proof.* First, we summarize the requirements, denoting each submatrix of $U$, $V$, and $W$ by the number of condition that ascertains that this submatrix is 0 (or has a well-defined non-zero form); moreover, Item h states that the columns of $W$, that contain identity matrices, are linearly independent. That is,

$$U = \left(\begin{array}{|c|c|c|c|c|c|c|} \hline 1 & \boldsymbol{\alpha} & \boldsymbol{\phi}(\boldsymbol{\alpha}) & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline \multicolumn{3}{|c|}{U_\phi} & & & i & e \\ \hline U_\psi & & & \multicolumn{2}{c|}{U_\psi} & i & e \\ \hline \multicolumn{5}{|c|}{U_\chi} & i & e \\ \hline a & a & & a & & U_\xi i & e \\ \hline \end{array}\right), \quad V = \left(\begin{array}{|c|c|c|c|c|c|c|} \hline 1 & \boldsymbol{\alpha} & \boldsymbol{\phi}(\boldsymbol{\alpha}) & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline \multicolumn{3}{|c|}{V_\phi} & & & j & f \\ \hline V_\psi & & & \multicolumn{2}{c|}{V_\psi} & j & f \\ \hline \multicolumn{5}{|c|}{V_\chi} & j & f \\ \hline \mathbf{1}_\kappa d & c & c & d & d & dj & df \\ \hline \end{array}\right), \quad W = \left(\begin{array}{|c|c|c|c|c|c|c|} \hline 1 & \boldsymbol{\alpha} & \boldsymbol{\phi}(\boldsymbol{\alpha}) & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & I_{\mu_\phi} & & & & g \\ \hline & & & & I_{\mu_\psi} & & g \\ \hline & & & & & I_{\mu_\chi} & g \\ \hline b & b & b & b & b & b & I_\kappa g \\ \hline \end{array}\right).$$

**Item a:** follows since $\boldsymbol{W}^{(j)} = \mathbf{0}$ in the columns labeled by 1, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, and the last rows of $U$ in all these columns are equal to 0, according to Eq. (6).

**Item b:** obvious from $W$ in Eq. (6).

**Item c:** follows since the last rows of $V$, corresponding to columns labeled by $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, are equal to 0.

**Item d:** follows since the last rows of $V$, corresponding to columns labeled by $\boldsymbol{\beta}$, $\boldsymbol{\psi}(\boldsymbol{\beta})$, $\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta})$, and $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta})$, are equal to 0, and the last rows of $\boldsymbol{V}^{(1)}$ are equal to $\mathbf{1}_\kappa$.

**Items e to g, i and j:** follows from direct observation.

**Item h:** follows from the fact that $\boldsymbol{W}^{(j)} = \mathbf{0}$ for some columns $j$, and the submatrix of $W$ that consists of the rest of the columns is an identity matrix. □

*Remark 1.* The compiled circuit $\mathcal{C}^*$ satisfies some additional conditions, not required by Theorem 1. First, by Item h, the set of non-zero $\boldsymbol{W}^{(j)}$ has to be linearly independent (not necessarily an identity matrix), while in Eq. (6), the corresponding columns constitute an identity matrix. Second, by Item a, last rows of $\boldsymbol{U}^{(j)}$ need to be zero only if $\boldsymbol{W}^{(j)}$ is 0; one can insert dummy gates to $\mathcal{C}^*$ such that $W$ has no zero columns. This essentially just corresponds to the fact that we start with an arithmetic circuit and each constraint is about a concrete gate being correctly evaluated. Third, several submatrices of $U, V, W$ are all-zero in our template while there is no actual need for that. For example, $U_\xi$ can be generalized, and $U_\phi$ and $U_\psi$ can also both depend on $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. For the sake of simplicity, we stick to the presented compilation process, and leave the possible generalizations to future work.

**SNARK-Related Techniques.** Next, we follow [Lip19] to derive polynomials related to the SNARK, underlying the new SFC. We simplify the derivation a bit, and refer to [Lip19] for full generality. Let $\mathsf{A}(X,Y) = r_a + u(X)Y$ and $\mathsf{B}(X,Y) = r_b + v(X)Y$ for $r_a, r_b \leftarrow\!\!\$\, \mathbb{Z}_p$. ([Lip19] considered the general case where $\mathsf{A}(X,Y) = r_a Y^\alpha + u(X)Y^\beta$ and $\mathsf{B}(X,Y) = r_b Y^\alpha + v(X)Y^\beta$ for some small integers $\alpha, \beta$ to be fixed later.) The randomizers $r_a$ and $r_b$ are needed to protect the secret information hidden by $\mathsf{A}(X,Y)$ and $\mathsf{B}(X,Y)$, and we use the indeterminate $Y$ to simplify the security proofs. As with $u$ and $w$, we divide the polynomials $\mathsf{A}, \mathsf{B}, \mathsf{C}$ into two addends: (i) a polynomial ($\mathsf{A}_s, \mathsf{B}_s$, $\mathsf{C}_{sp}$), where $\mathsf{A}_s$ and $\mathsf{B}_s$ depend on $\boldsymbol{\alpha}$ but not on $\boldsymbol{\beta}$ while $\mathsf{C}_{sp}$ depends on both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, and (ii) a polynomial ($\mathsf{A}_p, \mathsf{B}_p, \mathsf{C}_p$, resp.) that depends on public values ($\boldsymbol{\beta}$ and $\{\mathcal{F}_i(\boldsymbol{\alpha},\boldsymbol{\beta})\}$) but not on $\boldsymbol{\alpha}$ otherwise. (Such a division was not possible in [Lip19] since there one did not work with a compiled circuit $\mathcal{C}^*$.) Then,

$$\begin{aligned} \mathsf{A}_s(X,Y) =& r_a + u_s(X)Y \ , \quad \mathsf{A}_p(X,Y) = u_p(X)Y \ , \\ \mathsf{B}_s(X,Y) =& r_b + v_s(X)Y \ , \quad \mathsf{B}_p(X,Y) = v_p(X)Y \ . \end{aligned} \tag{9}$$

For integer constants $\delta$ and $\eta$ that we will fix later, define

$$\begin{aligned} \mathsf{C}(X,Y) =& (\mathsf{A}(X,Y) + Y^\delta)(\mathsf{B}(X,Y) + Y^\eta) - Y^{\delta+\eta} \\ =& (r_a + u(X)Y + Y^\delta)(r_b + v(X)Y + Y^\eta) - Y^{\delta+\eta} \\ =& r_a(v(X)Y + Y^\eta) + r_b(\mathsf{A}(X,Y) + Y^\delta) + (u(X)v(X) - w(X))Y^2 + \\ & u(X)Y^{\eta+1} + v(X)Y^{\delta+1} + w(X)Y^2 \\ =& r_a(v(X)Y + Y^\eta) + r_b(\mathsf{A}(X,Y) + Y^\delta) + \mathcal{H}(X)\ell(X)Y^2 + u(X)Y^{\eta+1} + v(X)Y^{\delta+1} + w(X)Y^2 \ , \end{aligned}$$

where the last equation holds iff the committer is honest (see Eq. (2)). Intuitively, we want that a committer must be able to compute $\mathsf{C}(X,Y)$ iff he was honest.

Following [Lip19], the inclusion of $Y^\delta$ and $Y^\eta$ in the definition of $\mathsf{C}(X,Y)$ serves two goals. First, it introduces the addend $u(X)Y^{\eta+1} + v(X)Y^{\delta+1} + w(X)Y^2 = \sum_{j=1}^{\mu} \mathsf{a}_j(u_j(X)Y^{\eta+1} + v_j(X)Y^{\delta+1} + w_j(X)Y^2)$ that makes it possible to verify that $\mathsf{P}$ uses the same coefficients $\alpha_j$ when computing $[\mathsf{A}]_1$, $[\mathsf{B}]_2$, and $[\mathsf{C}]_1$. Second, the coefficient of $Y^2$ is $u(X)v(X) - w(X)$ that divides by $\ell(X)$ iff the committer is honest. That is, the coefficient of $Y^2$ is $\mathcal{H}(X)\ell(X)$ for some polynomial $\mathcal{H}(X)$ iff the prover is honest and thus $\boldsymbol{\xi} = \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta})$.

Let $\gamma$ be another small integer, fixed later. Let $\mathsf{C}(X,Y) = \mathsf{C}_{sp}(X,Y) + \mathsf{C}_p(X,Y)Y^\gamma$, where $\mathsf{C}_p(X,Y)$ depends only on $\boldsymbol{\xi}$. (In [Lip19], $\mathsf{C}_{sp}(X,Y)$ was multiplied with $Y^\alpha$ but here $\alpha = 0$.) The factor $Y^\gamma$ is used to "separate" the public and the secret parts. In the honest case,

$$\begin{aligned} \mathsf{C}_{sp}(X,Y) =& r_a(v(X)Y + Y^\eta) + r_b(\mathsf{A}(X,Y)Y + Y^\delta) + \mathcal{H}(X)\ell(X)Y^2 + \\ & \sum_{j=1}^{\mu-\kappa} \mathsf{a}_j(u_j(X)Y^{\eta+1} + v_j(X)Y^{\delta+1} + w_j(X)Y^2) \ , \\ \mathsf{C}_p(X,Y) =& \sum_{j=\mu-\kappa+1}^{\mu} \mathsf{a}_j(u_j(X)Y^{\eta+1-\gamma} + v_j(X)Y^{\delta+1-\gamma} + w_j(X)Y^{2-\gamma}) \\ =& \sum_{i=1}^{\kappa} \mathcal{F}_i(\boldsymbol{\alpha},\boldsymbol{\beta})(u_{\mu-\kappa+i}(X)Y^{\eta+1-\gamma} + v_{\mu-\kappa+i}(X)Y^{\delta+1-\gamma} + w_{\mu-\kappa+i}(X)Y^{2-\gamma}) \ . \end{aligned} \tag{10}$$

Intuitively, the verifier checks that $\mathsf{C}_{sp}(X,Y)$ is correctly computed by checking that $\mathcal{V}(X,Y) = 0$, where

$$\begin{aligned} \mathcal{V}(X,Y) :=& (\mathsf{A}_s(X,Y) + \mathsf{A}_p(X,Y) + Y^\delta)(\mathsf{B}_s(X,Y) + \mathsf{B}_p(X,Y) + Y^\eta) - \\ & (\mathsf{C}_{sp}(X,Y) + \mathsf{C}_p(X,Y)Y^\gamma) - Y^{\delta+\eta} \ . \end{aligned}$$

Here, $(\mathsf{A}_s, \mathsf{B}_s)$ (the part of $(\mathsf{A}, \mathsf{B})$ that only depends on private information) is the functional commitment, $\mathsf{C}_{sp}$ is the opening, and $\mathsf{A}_p$, $\mathsf{B}_p$, and $\mathsf{C}_p$ can be recomputed by the verifier given public information.

$\mathsf{KC}(1^\lambda, \mathcal{C})$: $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda, \nu)$; Let $\mathcal{C}^*$ be the compiled arithmetic circuit; $\mathcal{C}^*$ defines $\nu, \mu$, and other parameters.
For $\mathsf{tk} = (\chi, y) \leftarrow\!\$\, (\mathbb{Z}_p^*)^2$ such that $\chi^\nu \neq 1$,

$$\mathsf{ck} = \begin{pmatrix} [1, (\chi^i y)_{i=0}^{\nu-1}, y^\eta, (\chi^i \ell(\chi) y^2)_{i=0}^{\nu-2}, (u_j(\chi) y^{\eta+1} + v_j(\chi) y^{\delta+1} + w_j(\chi) y^2)_{j=1}^{\mu-\kappa}]_1, \\ [(u_{\mu-\kappa+i}(\chi) y^{\eta+1-\gamma} + v_{\mu-\kappa+i}(\chi) y^{\delta+1-\gamma} + w_{\mu-\kappa+i}(\chi) y^{2-\gamma})_{i=1}^{\kappa}, y^\delta]_1, \\ [1, (\chi^i y)_{i=0}^{\nu-1}, y^\gamma, y^\eta]_2, [y^{\delta+\eta}]_T \end{pmatrix}.$$

Return $(\mathsf{ck}, \mathsf{tk})$;

$\mathsf{com}(\mathsf{ck}; \boldsymbol{\alpha}; r_a, r_b)$: // $r_a, r_b \leftarrow\!\$\, \mathbb{Z}_p$;
Compute $(\mathsf{a}_j)_{j=2}^{\mu_\alpha+\mu_\phi+1}$ from $\boldsymbol{\alpha}$;
Let $\mathsf{A}_s(X,Y) \leftarrow r_a + \sum_{i=0}^{\nu-1} A_i X^i Y$ be as in Eq. (9);
Let $\mathsf{B}_s(X,Y) \leftarrow r_b + \sum_{i=0}^{\nu-1} B_i X^i Y$ be as in Eq. (9);
For $i \in [1,\kappa]$: $\mathsf{B}_i^{\mathsf{aux}}(X,Y) \leftarrow \ell_{\nu-\kappa+i}(X)\mathsf{B}_s(X,Y)Y$;
$[\mathsf{A}_s]_1 \leftarrow r_a[1]_1 + \sum_{i=0}^{\nu-1} A_i[\chi^i y]_1$; $[\mathsf{B}_s]_2 \leftarrow r_b[1]_2 + \sum_{i=0}^{\nu-1} B_i[\chi^i y]_2$;
For $i \in [1,\kappa]$: $[\mathsf{B}_i^{\mathsf{aux}}]_1 \leftarrow [\mathsf{B}_i^{\mathsf{aux}}(\chi, y)]_1$;
$C \leftarrow ([\mathsf{A}_s, \{\mathsf{B}_i^{\mathsf{aux}}\}_{i=1}^{\kappa}]_1, [\mathsf{B}_s]_2)$; $D \leftarrow (\boldsymbol{\alpha}, r_a, r_b)$; return $(C, D)$;

$\mathsf{open}(\mathsf{ck}; C = ([\mathsf{A}_s, \{\mathsf{B}_i^{\mathsf{aux}}\}_{i=1}^{\kappa}]_1, [\mathsf{B}_s]_2), D = (\boldsymbol{\alpha}, r_a, r_b), \boldsymbol{\beta})$:
Compute $[(\ell_j(\chi) y)_{j=1}^{\nu}]_1$ from $[(\chi^i y)_{i=0}^{\nu-1}]_1$; // Needs to be done once
Compute $[(u_j(\chi) y, v_j(\chi) y)_{j=1}^{\mu}]_1$ from $[(\chi^i y)_{i=0}^{\nu-1}]_1$; // Needs to be done once
Compute $[(w_j(\chi) y)_{j=1}^{\mu}]_1$ from $[(\chi^i y)_{i=0}^{\nu-1}]_1$; // Needs to be done once
Compute $\mathbf{a}$ from $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$;
$u(X) \leftarrow \sum_{j=1}^{\mu} \mathsf{a}_j u_j(X)$; $v(X) \leftarrow \sum_{j=1}^{\mu} \mathsf{a}_j v_j(X)$; $w(X) \leftarrow \sum_{j=1}^{\mu} \mathsf{a}_j w_j(X)$;
$\mathcal{H}(X) \leftarrow (u(X)v(X) - w(X))/\ell(X)$;
$[\mathsf{A}_p]_1 \leftarrow [\mathsf{A}_p(\chi, y)]_1$ where $\mathsf{A}_p(X,Y)$ is as in Eq. (9);
$[\mathsf{C}_{sp}]_1 \leftarrow r_a([v(\chi)y]_1 + [y^\eta]_1) + r_b([\mathsf{A}_s]_1 + [\mathsf{A}_p]_1 + [y^\delta]_1) +$
$\sum_{i=0}^{\nu-2} \mathcal{H}_i[\chi^i \ell(\chi) y^2]_1 + \sum_{j=1}^{\mu-\kappa} \mathsf{a}_j[u_j(\chi) y^{\eta+1} + v_j(\chi) y^{\delta+1} + w_j(\chi) y^2]_1$;
return $\mathsf{op}_{\boldsymbol{\xi}} \leftarrow [\mathsf{C}_{sp}]_1$;

$\mathsf{V}(\mathsf{ck}, C = ([\mathsf{A}_s, \{\mathsf{B}_i^{\mathsf{aux}}\}_{i=1}^{\kappa}]_1, [\mathsf{B}_s]_2), [\mathsf{C}_{sp}]_1, \boldsymbol{\beta}, \{\xi_i\}_{i=1}^{\kappa})$: // $\xi_i =^? \mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta})$
Compute $[(\ell_{\nu-\kappa+i}(\chi) y)_{i=1}^{\kappa}]_1$ from $[(\chi^i y)_{i=0}^{\nu-1}]_1$; // Needs to be done once
Compute $[(\ell_{\nu-\kappa+i}(\chi) y^{2-\gamma})_{i=1}^{\kappa}]_1$ from $[(\chi^i y^{2-\gamma})_{i=0}^{\nu-1}]_1$; // Needs to be done once
Compute $[(\ell_{\nu-\kappa+i}(\chi) y)_{i=1}^{\kappa}]_2$ from $[(\chi^i y)_{i=0}^{\nu-1}]_2$; // Needs to be done once
Compute needed $[u_j(\chi) y]_1$ from $[(\chi^i y)_{i=0}^{\nu-1}]_2$; // Needs to be done once
Compute needed $[v_j(\chi) y]_2$ from $[(\chi^i y)_{i=0}^{\nu-1}]_2$; // Needs to be done once
$[\mathsf{A}_p]_1 \leftarrow [\mathsf{A}_p(\chi, y)]_1$ where $\mathsf{A}_p(X,Y)$ is as in Eq. (9);
$[\mathsf{B}_p]_2 \leftarrow [\mathsf{B}_p(\chi, y)]_2$ where $\mathsf{B}_p(X,Y)$ is as in Eq. (9);
$[\mathsf{C}_p]_1 \leftarrow \sum_{i=1}^{\kappa} \xi_i[\ell_{\nu-\kappa+i}(\chi) y^{2-\gamma}]_1$;
Check $([\mathsf{A}_s]_1 + [\mathsf{A}_p]_1 + [y^\delta]_1) \bullet ([\mathsf{B}_s]_2 + [\mathsf{B}_p]_2 + [y^\eta]_2) = [\mathsf{C}_{sp}]_1 \bullet [1]_2 + [\mathsf{C}_p]_1 \bullet [y^\gamma]_2 + [y^{\delta+\eta}]_T$;
For $i \in [1,\kappa]$: check $[\ell_{\nu-\kappa+i}(\chi) y]_1 \bullet [\mathsf{B}_s]_2 = [\mathsf{B}_i^{\mathsf{aux}}]_1 \bullet [1]_2$;

**Fig. 2.** SNARK-based SFC scheme $\mathsf{FC}_{\mathsf{sn}}^{\mathcal{C}}$ for arithmetic circuit $\mathcal{C}$

**The New SFC Scheme $\mathsf{FC}_{\mathsf{sn}}$: Details.** We are now ready to describe the new succinct functional commitment scheme $\mathsf{FC}_{\mathsf{sn}}$, see Fig. 2. Here, instead of operating with bivariate polynomials like $\mathsf{A}(X,Y)$, one operates with their encodings like $[\mathsf{A}_s(\chi, y)]_\iota$ in the source groups, where $\chi$ and $y$ are secret trapdoors. The commitment key of the SFC scheme contains the minimal amount of information needed to perform commitment, opening, and verification by honest parties. The expression of $\mathsf{ck}$ in $\mathsf{KC}$ has a generic form; one can replace the polynomials $u_j(X)$, $v_j(X)$, $w_j(X)$ with their values evident from Eq. (6). Finally, $\ell_j(X)$ (and thus also $u_j(X)$, $v_j(X)$, and $w_j(X)$) has degree $\nu - 1$ and can thus be computed from $(X^i)_{i=0}^{\nu-1}$, while $\ell(X)$ has degree $\nu$. Here, $[\mathsf{B}_i^{\mathsf{aux}}(\chi, y)]_1$ are additional elements needed to prove evaluation-binding. We explain in the correctness proof of Theorem 3 how to compute $[\mathsf{B}_i^{\mathsf{aux}}(\chi, y)]_1$.

Note that $\mathsf{FC}_{\mathsf{sn}}$ can also be seen as a SNARK proving that $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \boldsymbol{\xi}$, if we let the prover to compute $[\mathsf{A}_p]_1$, $[\mathsf{B}_p]_2$, and $[\mathsf{C}_p]_1$.

**Instantiation.** Let $\mathcal{C}$ be a fixed circuit. Let $\mathcal{R}$ and $\mathcal{S}$ be two sets of bivariate polynomials, such that the commitment key of $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ is equal to $\mathsf{ck} = ([\mathcal{R}(\chi, y)]_1, [\mathcal{S}(\chi, y)]_2)$. Similarly to [Lip19], let

$$\mathsf{Mon}_1 = \{0, 1, 2, 2-\gamma, \delta, \delta+1, \delta+1-\gamma, \eta, \eta+1, \eta+1-\gamma\} \tag{11}$$

be the set of exponents of $Y$ in all polynomials from $\mathcal{R}$. Let

$$\mathsf{Crit} = \{2, \eta+1\} \tag{12}$$

and $\overline{\mathsf{Crit}} = \mathsf{Mon}_1 \setminus \mathsf{Crit}$. For the evaluation-binding proof to hold, we need to fix values of $\gamma, \delta, \eta \in \mathbb{Z}_p$, such that the coefficients from $\mathsf{Crit}$ are unique, that is,

$$2, \eta+1 \notin \{0, 1, 2-\gamma, \delta, \delta+1, \delta+1-\gamma, \eta, \eta+1-\gamma\} \quad \text{and} \quad \eta+1 \neq 2 \ . \tag{13}$$

That is, $\mathsf{Crit} \cap \overline{\mathsf{Crit}} = \emptyset$ and $|\mathsf{Crit}| = 2$. It follows from Theorem 1 that the polynomial $f_i(X, Y) := \ell_{\nu-\kappa+i}(X)Y^{\eta+1}$, $i \in [1, \kappa]$, does not belong to $\mathrm{span}(\mathcal{R})$.

We will later consider two different evaluations for $\gamma$, $\delta$, and $\eta$. Replacing $\gamma$, $\delta$, and $\eta$ with 1, 0, and 3 guarantees that Eq. (13) holds (see Theorem 2, Item 1, for more). Then,

$$\mathsf{ck} = \begin{pmatrix} [1, (\chi^i y)_{i=0}^{\nu-1}, y^3, (\chi^i \ell(\chi) y^2)_{i=0}^{\nu-2}, (u_j(\chi) y^4 + v_j(\chi) y^1 + w_j(\chi) y^2)_{j=1}^{\mu-\kappa}]_1, \\ [(u_{\mu-\kappa+i}(\chi) y^3 + v_{\mu-\kappa+i}(\chi) y^0 + w_{\mu-\kappa+i}(\chi) y^1)_{i=1}^{\kappa}, y^0]_1, \\ [1, (\chi^i y)_{i=0}^{\nu-1}, y^1, y^3]_2, [y^3]_T \end{pmatrix} \ .$$

In this case, the $\mathsf{ck}$ has one element (namely, $[1]_1$) twice, and thus $\mathsf{ck}$ can be shortened by one element.

Alternatively, replacing $\gamma$, $\delta$, and $\eta$ with 4, 0, and 7 (this choice is sufficient for the evaluation-binding reduction to uber-assumption in $\mathbb{G}_T$ to work and will be explained in Theorem 2, Item 2), we get

$$\mathsf{ck} = \begin{pmatrix} [1, (\chi^i y)_{i=0}^{\nu-1}, y^7, (\chi^i \ell(\chi) y^2)_{i=0}^{\nu-2}, (u_j(\chi) y^8 + v_j(\chi) y^1 + w_j(\chi) y^2)_{j=1}^{\mu-\kappa}]_1, \\ [(u_{\mu-\kappa+i}(\chi) y^4 + v_{\mu-\kappa+i}(\chi) y^{-3} + w_{\mu-\kappa+i}(\chi) y^{-2})_{i=1}^{\kappa}, y^0]_1, \\ [1, (\chi^i y)_{i=0}^{\nu-1}, y^4, y^7]_2, [y^7]_T \end{pmatrix} \ .$$

Then, $\mathsf{ck}$ has one element ($[1]_1$) twice, and thus it can be shortened.

**Efficiency.** The CRS length is $1 + \nu + 1 + (\nu - 1) + (\mu - \kappa) + \kappa + 1 = 2\nu + \mu + 2$ elements from $\mathbb{G}_1$, $\nu + 3$ elements from $\mathbb{G}_2$, and 1 element from $\mathbb{G}_T$. In the case of fixed $\gamma$, $\delta$ and $\eta$ in the previous two paragraphs, the CRS length will shorten by 1 element of $\mathbb{G}_1$.

The functional commitment takes $(\nu + 1) + \kappa(\nu + 1) = (\kappa + 1)(\nu + 1)$ exponentiations in $\mathbb{G}_1$ and $\nu + 1$ exponentiations in $\mathbb{G}_2$. The length of the functional commitment is $\kappa + 1$ elements of $\mathbb{G}_1$ and 1 element of $\mathbb{G}_2$.

The opening takes $\mu_\beta + \mu_\psi + \kappa$ (to compute $[\mathsf{A}_p]_1$; note that $u_1(X)$ and other simular polynomials are precomputed), $\mu_\alpha + \mu_\beta + \mu_\phi + \mu_\psi$ (to compute $[v(\chi)y]_1$), and $2 + (\nu - 1) + (\mu - \kappa) = \nu + \mu - \kappa + 1$ (to compute $[\mathsf{C}_{sp}]_1$) exponentiations in $\mathbb{G}_1$; in total, $\nu + \mu + \mu_\alpha + 2\mu_\beta + \mu_\phi + 2\mu_\psi + 1$ exponentiations in $\mathbb{G}_1$. The length of the opening is 1 element of $\mathbb{G}_1$.

The verification takes $(\mu_\beta + \mu_\psi + \kappa) + \kappa = \mu_\beta + \mu_\psi + 2\kappa$ (to compute $[\mathsf{A}_p, \mathsf{C}_p]_1$) exponentiations in $\mathbb{G}_1$, $\mu_\beta + \mu_\psi$ (to compute $[\mathsf{B}_p]_2$) exponentiations in $\mathbb{G}_2$, and $2\kappa + 3$ pairings. Here, we do not count computations (e.g., computation of $[\ell_{\nu-\kappa+i}(\chi)y]_1$ from $[(\chi^i y)_{i=0}^{\nu-1}]_1$) that are only done once per the circuit.

The real efficiency depends of course significantly on the concrete application. We will give some detailed examples in Appendix A.2.

## 4 On the Circuit Class and Example Applications

Next, we study the power of the implementable circuit class $\mathbf{CC}_{\Sigma\Pi\forall}$, and we show that many known functional commitment schemes are for functionalities that belong to this class, and thus can be implemented by $\mathsf{FC}_{\mathsf{sn}}$.

In this section, we assume basic knowledge of the algebraic complexity theory. See [SY10] for necessary background. **VP** is the class of polynomial families $\{f_n\}$, where $f_n$ is an univariate polynomial of $\mathsf{poly}(n)$ variables of $\mathsf{poly}(n)$ degree that has an arithmetic circuit of $\mathsf{poly}(n)$ size [Val79]. $\Sigma\Pi\Sigma$ (resp., $\Sigma\Pi\Sigma\Pi$) is the class of depth-3 (resp., depth-4) circuits composed of alternating levels of sum and product gates with a sum gate at the top [SY10, Section 3.5]. *Sparse polynomials* are $n$-variate polynomials that have $\mathsf{poly}(n)$ monomials.

Recall that a compiled circuit $\mathcal{C}^*$ can evaluate a vector polynomial $\boldsymbol{f}(\boldsymbol{\alpha},\boldsymbol{\beta}) = (f_i(\boldsymbol{\alpha},\boldsymbol{\beta}))_{i=1}^{\kappa}$ iff $\kappa \in \mathsf{poly}(\lambda)$ and each $f_i$ can be written as

$$f_i(\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum \phi_j(\boldsymbol{\alpha})\psi_k(\boldsymbol{\beta}) \ , \tag{14}$$

where all polynomials $\phi_j$ and $\psi_k$ are in the complexity class **VP**, and there are a polynomial number of additions in the representation Eq. (14) (thus, also a polynomial number of polynomials $\phi_j$ and $\psi_k$). We call such representation an *efficient* $\Sigma\Pi\forall$*-representation* (here, $\forall$ denotes "any") of $\boldsymbol{f}$, and we denote by $\mathbf{CC}_{\Sigma\Pi\forall}$ the class of circuits (or vector polynomials) that have an efficient $\Sigma\Pi\forall$-presentation. Clearly, $\mathsf{FC}_{\mathsf{sn}}$ can implement $\boldsymbol{f}$ iff $\boldsymbol{f} \in \mathbf{CC}_{\Sigma\Pi\forall}$.

It is clear that all sparse polynomials in **VP** have an efficient $\Sigma\Pi\forall$-representation, and thus $\mathsf{FC}_{\mathsf{sn}}$ can implement all sparse polynomials. However, we can do more. For example, consider the polynomial $f'(\alpha,\boldsymbol{\beta}) = \prod_{i=1}^{n}(\alpha+\beta_i)$ for $n = \mathsf{poly}(\lambda)$. Since $f'$ has $2^n$ monomials, it is not sparse. However, we can rewrite $f'$ as $f'(\alpha,\boldsymbol{\beta}) = \sum_{d=0}^{n}\alpha^d\sigma_{n-d}(\boldsymbol{\beta})$, where $\sigma_{n-d}(\boldsymbol{\beta}) = \sum_{T\subseteq[1,n],|T|=d}\prod_{i\in T}\beta_i$ is the $(n-d)$th symmetric polynomial. There exists a $\Sigma\Pi\Sigma$ circuit of size $O(n^2)$, due to Ben-Or (see [SY10, Section 3.5]), that computes all $n$ symmetric polynomials in parallel. Thus, $f$ has an efficient $\Sigma\Pi\forall$ -representation, and thus $\mathsf{FC}_{\mathsf{sn}}$ can implement at least one non-sparse polynomial.

On the other hand, $\mathbf{CC}_{\Sigma\Pi\forall} \subseteq \mathbf{VP}$. To see that $\mathbf{CC}_{\Sigma\Pi\forall} \subsetneq \mathbf{VP}$, consider the polynomial $f''(\boldsymbol{\alpha},\boldsymbol{\beta}) = \prod_{i=1}^{n}(\alpha_i+\beta_i)$ for $n = \mathsf{poly}(\lambda)$. Since $f''$ has $2^n$ monomials, it is not sparse. Considering $\beta_i$ as coefficients, it also has $2^n$ monomials in $\boldsymbol{\alpha}$ (the case of considering $\alpha_i$ as coefficients is dual), and thus any $\Sigma\Pi\forall$-representation of $f''$ requires at least $2^n$ addition gates. Since $f''$ can be implemented by a $\Pi\Sigma$ circuit [SY10], it means $\Pi\Sigma \not\subset \mathbf{CC}_{\Sigma\Pi\forall}$; however, clearly, $\mathbf{CC}_{\Sigma\Pi\forall} \not\subset \Pi\Sigma$ so $\mathbf{CC}_{\Sigma\Pi\forall}$ is incomparable to $\Pi\Sigma$. Thus

$$\boxed{\text{the class of sparse polynomials} \subsetneq \mathbf{CC}_{\Sigma\Pi\forall} \subsetneq \mathbf{VP}}$$

Motivated by our analysis of $\alpha''$, it seems we can implement all polynomials $f(\boldsymbol{\alpha},\boldsymbol{\beta})$, where either the dimension $\mu_\alpha$ of $\boldsymbol{\alpha}$ or the dimension $\mu_\beta$ of $\boldsymbol{\beta}$ is logarithmic in $\lambda$. Really, if $\mu_\alpha = O(\log\lambda)$ then there are at most $2^{\mu_\alpha} = \mathsf{poly}(\lambda)$ possible monomials $\phi_i(\boldsymbol{\alpha})$ in $\boldsymbol{\alpha}$, and thus there exists an efficient $\Sigma\Pi\forall$-representation of $f$.

It is an interesting open problem to characterize $\mathbf{CC}_{\Sigma\Pi\forall}$.

**Known Types of SFCs as (Semi-)Sparse Polynomials.** In Table 1, we write down the functionalities of several previous known types of SFCs. This shows that in all such cases, one has a sparse polynomial and thus can use $\mathsf{FC}_{\mathsf{sn}}$ to implement them. In the case of the vector commitment scheme (resp., accumulator), one implements the inner-product scheme with $\boldsymbol{\beta} = \boldsymbol{e}_I$ (resp., $\chi_{\boldsymbol{\alpha}}(X) = \prod(X-\alpha_i)$). In the case of say the polynomial commitment scheme, $\boldsymbol{\beta} = (1,\beta,\ldots,\beta^{n-1})$ and thus $\mu_\beta = n$.

In none of these cases, one needs the power of non-sparse semi-sparse polynomials, and we leave it as another open question to find an application where such power is needed.

**Aggregation.** The next lemma is straightforward.

**Lemma 1.** *Assume that $\mathcal{C}_i \in \mathbf{CC}_{\Sigma\Pi\forall}$, where $i \in [1,Q]$, and $Q = \mathsf{poly}(\lambda)$. Then their parallel composition $\mathcal{C}^{\|} = (\mathcal{C}_1\|\ldots\|\mathcal{C}_Q) \in \mathbf{CC}_{\Sigma\Pi\forall}$.*

**Table 1.** Rewriting the functionalites of various SFC as sparse polynomials

| Type | $\mu_\alpha$ | $\mu_\beta$ | $f_i$ |
|---|---|---|---|
| Inner-product commitment [ILV11,LRY16] | $n$ | $n$ | $\sum_{j=1}^{n} \alpha_j \beta_j$ |
| Polynomial commitment [KZG10] | $n$ | $n$ | $\sum_{j=0}^{n-1} \alpha_j \beta^j$ |
| Vector commitment [CF13] | $n$ | 1 | $\alpha_I = \sum_{j=1}^{n} \alpha_j e_{Ij}$ |
| Accumulator [Bd94,BP97] | 1 | $n$ | $\sum_{j=0}^{\mu_\alpha - 1} \chi_{\boldsymbol{\alpha} j} \beta^j$ |
| Evaluation-point commitment | 1 | $n$ | $\sum_{j=0}^{n-1} \alpha^j \beta_j$ |
| $c$-variate polynomial commitment [PST13,BGH19] | $\binom{n+c}{c}$ | $c$ | $\sum \alpha_j \prod_{k=1}^{c} \beta_k^{j_k}$ |

*Proof.* Obvious since we can just "parallelize" the representation in Eq. (14). □

In practice, Lemma 1 is very important since it means that $\mathsf{FC}_{\mathsf{sn}}$ allows to aggregate a polynomial number of SFCs for which $\mathsf{FC}_{\mathsf{sn}}$ is efficient. It just results in a larger circuit $\mathcal{C}^{\|}$ and thus larger parameters like $\mu$ and $\kappa$. However, as the length of the commitment in $\mathsf{FC}_{\mathsf{sn}}$ depends on $\kappa$, it means that the commitment stays succinct when $Q < |\mathbb{w}|$. On the other hand, the length of the opening will be one group element, independently of $Q$.

As a corollary of Lemma 1, we can construct succinct aggregated inner-product SFCs, accumulators, (multi-point / multi-polynomial) polynomial commitment schemes, vector commitment schemes (including subvector commitment schmes), but also aggregate all these SFC variants with each other. We will give more details and examples in Appendix A.

**Example: Aggregated Succinct Inner-Product Functional Commitment.** In an aggregated SIPFC, the committer commits to $\boldsymbol{\alpha}$ and then opens it simultaneously to $\langle \boldsymbol{\alpha}, \boldsymbol{\beta}_i \rangle = \sum_{j=1}^{n} \alpha_j \beta_{ij}$ for $\kappa$ different verifier-provided vectors $\boldsymbol{\beta}_i$, where $i \in [1, \kappa]$. Assume $\boldsymbol{\alpha}$ and each $\boldsymbol{\beta}_i$ are $n$-dimensional vectors. There is no circuit $\mathcal{C}_\phi$ or $\mathcal{C}_\psi$. Given $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}_i$, $\mathcal{C}_\chi$ computes $\kappa n$ products $\chi_{ij}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \alpha_j \beta_{ij}$, $i \in [1, \kappa]$ and $j \in [1, n]$, and $\mathcal{C}_\xi$ sums them together to obtain $\kappa$ outputs $\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^{n} \alpha_j \beta_{ij}$. Thus, $U_\chi = \mathbf{1}_\kappa \otimes I_n \in \mathbb{Z}_p^{\kappa n \times n}$, $V_\chi = I_{\kappa n}$, $U_\xi = I_n \otimes \mathbf{1}_\kappa^\top \in \mathbb{Z}_p^{n \times \kappa n}$ (note that $\mathcal{C}_\chi$ does not take 1 as an input), and

$$U = \left(\begin{array}{c|c|c|c|c} \underline{1} & \underline{\boldsymbol{\alpha}} & \underline{\boldsymbol{\beta}} & \underline{\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta})} & \underline{\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta})} \\ \hline & U_\chi & & & \\ \hline & & & U_\xi & \end{array}\right), \quad V = \left(\begin{array}{c|c|c|c|c} \underline{1} & \underline{\boldsymbol{\alpha}} & \underline{\boldsymbol{\beta}} & \underline{\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta})} & \underline{\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta})} \\ \hline & & V_\chi & & \\ \hline 1 & & & & \end{array}\right), \quad W = \left(\begin{array}{c|c|c|c|c} \underline{1} & \underline{\boldsymbol{\alpha}} & \underline{\boldsymbol{\beta}} & \underline{\boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta})} & \underline{\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta})} \\ \hline & & & I_{\kappa n} & \\ \hline & & & & I_\kappa \end{array}\right).$$

Here, $\nu = \kappa(n+1)$, $\mu = 1 + n + \kappa n + \kappa n + \kappa = (\kappa+1)n + \kappa + 1$, $\mathsf{A}_s(X,Y) = r_a + \sum_{j=1}^{n} \alpha_j u_{\chi j}(X) Y$, $\mathsf{A}_p(X,Y) = \sum_{i=1}^{\kappa} \langle \boldsymbol{\alpha}, \boldsymbol{\beta}_i \rangle \ell_{\nu-\kappa+i}(X) Y$. Importantly, $\mathsf{B}_s(X,Y) = 0$ (since there is nothing to hide, one can set $r_b \leftarrow 0$; hence, also $\mathsf{B}_i^{\mathsf{aux}}(X,Y) = 0$; thus the commitment is only one group element, $[\mathsf{A}_s]_1$), and $\mathsf{B}_p(X,Y) = \sum_{i=1}^{\kappa} \ell_{\nu-\kappa+i}(X) Y + \sum_{i=1}^{\kappa} \sum_{j=1}^{n} \beta_{ij} v_{\chi,n(i-1)+j}(X) Y$. The verifier has to execute $2\kappa$ exponentiations in $\mathbb{G}_1$ to compute $[\mathsf{A}_p]_1$ and $[\mathsf{C}_p]_1$, $\kappa n$ exponentiations in $\mathbb{G}_2$ to compute $[\mathsf{B}_p]_2$, and 3 pairings. We emphasize that here, both the functional commitment and the opening will consist of a single group element. One obtains IPFC by setting $\kappa \leftarrow 1$; in this case, the verification executes 2 exponentiations in $\mathbb{G}_1$, $n$ exponentiations in $\mathbb{G}_2$, and 3 pairings.

Let us briefly compare the resulting *non-aggregated* IPFC with the IPFC of [ILV11]. Interestingly, while the presented IPFC is a simple specialization of the general SFC scheme, it is only slightly less efficient than [ILV11]. Let $g_\iota$ denote the bitlength of an element of the group $\mathbb{G}_\iota$. The CRS length is $2ng_1 + (n+1)g_2$ in [ILV11], and $(3(\kappa+1) + (4\kappa+1)n)g_1 + (\kappa + \kappa n + 3)g_2 + 1g_T$ (this shortens to $(5n+6)g_1 + (n+4)g_2 + 1g_T$ when $\kappa = 1$) in our case. The commitment takes $n+1$ exponentiations in [ILV11], and $n+2$ in our case. A straightforward [ILV11] opening takes $\Theta(n^2)$ multiplications (this can be probably optimized), while in our case it takes $\Theta(n \log n)$ multiplications. The verifier takes $n$ exponentiations in [ILV11], and $n+3$ here. The

commitment and opening are 1 group elements in both schemes. Thus, our *generic, unoptimized* scheme is essentially as efficient as the most efficient known prior IPFC, losing ground only in the CRS length. On the other hand, we are not aware of *any* previous aggregated IPFC schemes.

## 5 Security of $\mathsf{FC}_{\mathsf{sn}}$

Next, we prove the security of $\mathsf{FC}_{\mathsf{sn}}$. While its correctness and hiding proofs are straightforward, the evaluation-binding proof is far from it. As before, for a fixed $\mathcal{C}$, let $\mathcal{R}$ and $\mathcal{S}$ be two sets of bivariate polynomials, such that $\mathsf{ck} = ([\mathcal{R}(\chi,y)]_1, [\mathcal{S}(\chi,y)]_2)$. For a fixed $\mathcal{C}$, in Theorem 3, we will reduce evaluation-binding of $\mathsf{FC}_{\mathsf{sn}}^{\mathcal{C}}$ to a $(\mathcal{R},\mathcal{S},\{f_i\})$-*span-uber-assumption* in $\mathbb{G}_1$, a new assumption that states that it is difficult to output an element $\sum \Delta_i[f_i(\chi,y)]_1$ together with the coefficient vector $\boldsymbol{\Delta} \neq \mathbf{0}$, where $f_i \notin \mathrm{span}(\mathcal{R})$. Thus, it is a generalization of the $(\mathcal{R},\mathcal{S},\cdot)$-computational uber-assumption in $\mathbb{G}_1$. Importantly, if $\kappa = 1$, then it is equivalent to the latter. To motivate the span-uber-assumption, we will show that it follows from the more conventional $(\mathcal{R},\mathcal{S},f'_I)$-computational uber-assumption (for a related set of polynomials $f'_i$) in $\mathbb{G}_T$ [BBG05]; see Lemma 2. Thus, for the concrete parameters $\mathcal{R},\mathcal{S},\{f_i\}$, and $\{f'_i\}$,

$$\boxed{\text{uber-assumption in } \mathbb{G}_T \Rightarrow \text{span-uber-assumption in } \mathbb{G}_1 \Rightarrow \text{uber-assumption in } \mathbb{G}_1}$$

For the algebraic group model [FKL18] reduction to the PDL in Appendix B to work, we also prove that $f_i \notin \mathrm{span}(\mathcal{R})$ and $f'_i \notin \mathrm{span}(\mathcal{R}\mathcal{S})$; see Theorem 2. (Otherwise, the span-uber-assumption will be trivially insecure in the generic model.) Each concrete proof (e.g., the proof of correctness, the proof of evaluation-binding, and the proofs that $f_i \notin \mathrm{span}(\mathcal{R})$ and $f'_i \notin \mathrm{span}(\mathcal{R}\mathcal{S})$) puts some simple restrictions on the matrices $U$, $V$, $W$. Those restrictions are satisfied in all-but-one examples in Appendix A.2. They can usually be satisfied by slightly modifying the underlying arithmetic circuit.

**Definition 6.** *Let $\mathcal{R}$, $\mathcal{S}$, and $\mathcal{T}$ be three tuples of bivariate polynomials over $\mathbb{Z}_p[X,Y]$. Let $f_i$, $i \in [1,\kappa]$, be bivariate polynomials over $\mathbb{Z}_p[X,Y]$. The $(\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\}_{i=1}^{\kappa})$* computational span-uber-assumption *for $\mathsf{Pgen}$ in group $\mathbb{G}_\iota$, where $\iota \in \{1,2,T\}$, states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{spanuber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, where*

$$\mathsf{Adv}^{\mathrm{spanuber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \chi, y \leftarrow\!\$\, \mathbb{Z}_p^*; \mathsf{ck} \leftarrow ([\mathcal{R}(\chi,y)]_1, [\mathcal{S}(\chi,y)]_2, [\mathcal{T}(\chi,y)]_T); \\ (\boldsymbol{\Delta} \in \mathbb{Z}_p^\kappa, [z]_\iota) \leftarrow \mathcal{A}(\mathsf{ck}) : \boldsymbol{\Delta} \neq \mathbf{0} \wedge [z]_\iota = \sum_{i=1}^{\kappa} \Delta_i [f_i(\chi,y)]_\iota\end{array}\right].$$

If $\kappa = 1$ then the $(\mathcal{R},\mathcal{S},\mathcal{T},\{f\})$ span-uber-assumption is the same as the $(\mathcal{R},\mathcal{S},\mathcal{T},f_1)$ uber-assumption: in this case the adversary is tasked to output $\mathbb{Z}_p \ni \Delta \neq 0$ and $\Delta[f_1(\chi,y)]_\iota$ which is equivalent to outputting $[f_1(\chi,y)]_\iota$.

We will now show that the polynomials used in what follows are linearly independent.

**Theorem 2.** *Write $\mathsf{ck} = ([\varrho(X,Y) : \varrho \in \mathcal{R}]_1, [\sigma(X,Y) : \sigma \in \mathcal{S}]_2)$ as in Fig. 2. For $I \in [1,\kappa]$, let $f_I(X,Y) := \ell_{\nu-\kappa+I}(X)Y^{\eta+1}$ and $f'_I(X,Y) := (\ell_{\nu-\kappa+I}(X))^2 Y^{\eta+2}$.*

1. *Assume $\gamma = 1$, $\delta = 0$, and $\eta = 3$. Assume Items a and h of Theorem 1 hold. Then $f_I(X,Y) \notin \mathrm{span}(\mathcal{R})$ for $I \in [1,\kappa]$.*
2. *Assume $\gamma = 4$, $\delta = 0$, $\eta = 7$, and that Items a, b and h of Theorem 1 hold. Then $f'_I(X,Y) \notin \mathrm{span}(\mathcal{R}\mathcal{S})$ for $I \in [1,\kappa]$. As an obvious corollary, thus also $f'_I(X,Y) \notin \mathrm{span}(\mathcal{R})$ for $I \in [1,\kappa]$.*

*Proof.* **(1: $f_I \notin \mathrm{span}(\mathcal{R})$).** Let $\mathsf{Mon}_1$ be as in Eq. (11) and $\mathsf{Crit} = \{2, \eta+1\}$ as in Eq. (12). For the rest of the proof to make sense, as we will see in a few paragraphs, we need to fix $\gamma$, $\delta$, and $\eta$ so that the coefficients in $\mathsf{Mon}_1$ and in $\mathsf{Mon}_1 \setminus \mathsf{Crit}$ are different (in particular, the coefficients in $\mathsf{Crit}$ are different from each other). A small exhaustive search shows that one can define $\gamma = 1$, $\delta = 0$, $\eta = 3$, as in the claim. This setting can be easily manually verified, by noticing that then $\mathsf{Mon}_1 = \{0,1,2,3,4\}$ and $\mathsf{Crit} = \{2,4\}$, and thus $\mathsf{Mon}_1 \setminus \mathsf{Crit} = \{0,1,3\}$.

Assume that, for some $I \in [1,\kappa]$, $f_I(X,Y) = \ell_{\nu-\kappa+I}(X)Y^{\eta+1}$ belongs to the span of $\mathcal{R}$. We consider the coefficients of $Y^i$, for $i \in \mathsf{Crit}$, in the resulting equality (for some unknown coefficients in front of the polynomials from $\mathcal{R}$), and derive a contradiction from this. We write down an arbitrary linear combination of polynomials in $\mathcal{R}$ as a linear combination of $u_j(X)Y^{\eta+1} + v_j(X)Y^{\delta+1} + w_j(X)Y^2$, $X^i\ell(X)Y^2$, and $T(X,Y)$, where $T(X,Y)$ is some polynomial with monomials that do not have $Y^i$ for $i \in \mathsf{Crit}$. That is,

$$\ell_{\nu-\kappa+I}(X)Y^{\eta+1} = \sum_{j=1}^{\mu-\kappa} t'_j(u_j(X)Y^{\eta+1} + v_j(X)Y^{\delta+1} + w_j(X)Y^2) + t(X)\ell(X)Y^2 + T(X,Y) \tag{15}$$

for some $t(X) \in \mathbb{Z}_p[X]$ (thus $t(X)\ell(X)Y^2$ encompasses all $X^i\ell(X)Y^2$) and integers $t'_j$.

First, considering only the coefficient of $Y^2$ in both the left-hand side and the right-hand side of Eq. (15),

$$\sum_{j=1}^{\mu-\kappa} t'_j w_j(X) + t(X)\ell(X) = 0 \ .$$

Due to Item h of Theorem 1 (the set of non-zero $w_j(X)$ is linearly independent), either $w_j(X) = 0$ or $t'_j = 0$ for $j \in [1, \mu-\kappa]$. Let $\mathcal{J} \subset [1,\mu-\kappa]$ be the set of indices $j \in [1,\mu-\kappa]$ so that $w_j(X) = 0$. Thus, $t'_j = 0$ for $j \notin \mathcal{J}$.

Second, considering only the coefficient of $Y^{\eta+1}$ in Eq. (15),

$$\ell_{\nu-\kappa+I}(X) = \sum_{j=1}^{\mu-\kappa} t'_j u_j(X) = \sum_{j\in\mathcal{J}} t'_j u_j(X) \ .$$

Due to Item a of Theorem 1, $\ell_{\nu-\kappa+I}(X)$ is linearly independent of (the non-zero elements of) $\{u_j(X)\}_{j\in\mathcal{J}}$, a contradiction. Hence, $f_I(X,Y) \notin \mathrm{span}(\mathcal{R})$.

**(Item 2: $f'_I \notin \mathrm{span}(\mathcal{RS})$).** For the proof to make sense, as we will see in a few paragraphs, we need that the set of critical coefficients $\mathsf{Crit}' := \{3, \eta+2\}$ (that is different from $\mathsf{Crit}$ above) is different from the set $\mathsf{Mon}' \setminus \mathsf{Crit}'$ all other coefficients in $\mathcal{RS}$, where $\mathsf{Mon}' :=$

$$\left\{\begin{array}{l} 0,1,2,3,2-\gamma,3-\gamma,\gamma,\gamma+1,\gamma+2,\delta,1+\delta,2+\delta,1-\gamma+\delta,2-\gamma+\delta, \\ \gamma+\delta,1+\gamma+\delta,\eta,2\eta,\eta+1,\eta+2,-\gamma+\eta+1,-\gamma+\eta+2,\gamma+\eta, \\ \gamma+\eta+1,\delta+\eta,1+\delta+\eta,1-\gamma+\delta+\eta,1+2\eta,1-\gamma+2\eta \end{array}\right\} .$$

is defined by $\mathsf{Mon}' = \mathsf{Mon}_1 + \mathsf{Mon}_2$, where $\mathsf{Mon}_1$ is as in Eq. (11) and $\mathsf{Mon}_2 = \{0,1,\gamma,\eta\}$ is the set of exponents of $Y$ in all polynomials from $\mathcal{S}$. A small exhaustive search, performed by using computer algebra, shows that one can define $\gamma = 4$, $\delta = 0$, $\eta = 7$, as in the claim. This setting can be easily manually verified, by noticing that $\mathsf{Mon}' \setminus \mathsf{Crit}' = \{-3,-2,-1,0,1,2,4,5,6,7,8,11,12,14,15\}$ and $\mathsf{Crit}' = \{3,9\}$.

Assume now in contrary that $f'_I \in \mathrm{span}(\mathcal{RS})$. Then, as in Item 1, $(\ell_{\nu-\kappa+I}(X))^2Y^{\eta+2}$ is in the span of the polynomials containing $Y^i$ for $i \in \mathsf{Crit}'$ (and we need to quantify the coefficients of these polynomials) and of all other polynomials. Clearly, the first type of polynomials are in the span of $X^i\ell(X)Y^2$ times $Y^\eta$, $X^i\ell(X)Y^2$ times $X^kY$, $u_j(X)Y^{\eta+1} + v_j(X)Y^{\delta+1} + w_j(X)Y^2$ times $Y^\eta$, and $u_j(X)Y^{\eta+1} + v_j(X)Y^{\delta+1} + w_j(X)Y^2$ times $X^kY$, for properly chosen $i$, $j$, and $k$. Thus,

$$\begin{aligned}(\ell_{\nu-\kappa+I}(X))^2Y^{\eta+2} =& t(X)\ell(X)Y^{\eta+2} + t''(X)\ell(X)Y^3 + \\ & \sum_{j=1}^{\mu-\kappa} t'_j(X)(u_j(X)Y^{2\eta+1} + v_j(X)Y^{\delta+\eta+1} + w_j(X)Y^{\eta+2}) + \\ & \sum_{j=1}^{\mu-\kappa} t^*_j(X)(u_j(X)Y^{\eta+2} + v_j(X)Y^{\delta+2} + w_j(X)Y^3) + T(X,Y) \ ,\end{aligned}$$

where $t'_j(X)$, $t^*_j(X)$, $t(X)$ and $t''(X)$ are univariate polynomials, and $T(X,Y)$ is a polynomial that does not contain monomials with $Y^i$, $i \in \mathsf{Crit}'$. We now consider separately the coefficients of $Y^i$ in this equation for each $i \in \mathsf{Crit}'$ and derive a contradiction.

First, considering the coefficients of $Y^3$, we get $\sum_{j=1}^{\mu-\kappa} t^*_j(X)w_j(X) + t''(X)\ell(X) = 0$. Due to Item h of Theorem 1, either $t^*_j(X) = 0$ or $w_j(X) = 0$ for $1 \le j \le \mu - \kappa$. Let $\mathcal{J} \subset [1, \mu-\kappa]$ be the set of indices $j$ so that $w_j(X) = 0$.

Second, the coefficients of $Y^{\eta+2}$ give us

$$\begin{aligned}(\ell_{\nu-\kappa+I}(X))^2 &= \sum_{j=1}^{\mu-\kappa} t^*_j(X)u_j(X) + \sum_{j=\mu_\alpha+2}^{\mu-\kappa} t'_j(X)w_j(X) + t(X)\ell(X) \\ &= \sum_{j\in\mathcal{J}} t^*_j(X)u_j(X) + \sum_{j\notin\mathcal{J}} t'_j(X)w_j(X) + t(X)\ell(X) \ .\end{aligned}$$

Due to Items a, b and h of Theorem 1 (and of the fact that $(\ell_{\nu-\kappa+i}(X))^2$ has degree $2\nu$), $\{(\ell_{\nu-\kappa+I}(X))^2\} \cup \{u_j(X)\}_{j\in\mathcal{J}} \cup \{w_j(X)\}_{j\notin\mathcal{J}} \cup \{X^i\ell(X)\}_{i=0}^{\nu-2}$ is linearly independent. Contradiction, and thus $f'_I(X,Y) \notin \mathrm{span}(\mathcal{RS})$. □

Next, we show that for the concrete choice of the parameters $\mathcal{R}$, $\mathcal{S}$, $f_i$, and $f'_i$, the span-uber-assumption in $\mathbb{G}_1$ is at least as strong as the uber-assumption in $\mathbb{G}_T$. The new assumption may be weaker since the latter assumption argues about elements in $\mathbb{G}_T$, which may not always be possible [JR10]. However, the proof of Lemma 2 depends crucially on the concrete parameters.

**Lemma 2 (Uber-assumption in $\mathbb{G}_T$ $\Rightarrow$ span-uber-assumption in $\mathbb{G}_1$).** *Assume $\gamma = 4$, $\delta = 0$, and $\eta = 7$. Let $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ be the SFC scheme for arithmetic circuits in Fig. 2. Write $\mathsf{ck} = ([\varrho(X,Y) : \varrho \in \mathcal{R}]_1, [\sigma(X,Y) : \sigma \in \mathcal{S}]_2)$ as in Fig. 2. For $i \in [1,\kappa]$, let $f_i(X,Y) := \ell_{\nu-\kappa+i}(X)Y^{\eta+1}$ and $f'_i(X,Y) := (\ell_{\nu-\kappa+i}(X))^2Y^{\eta+2}$. If the $(\mathcal{R},\mathcal{S},f'_I)$ computational uber-assumption holds in $\mathbb{G}_T$ for each $I \in [1,\kappa]$ then the $(\mathcal{R},\mathcal{S},\{f_i\}_{i=1}^{\kappa})$ computational span-uber-assumption holds in $\mathbb{G}_1$.*

*Proof (Sketch).* Assume $\mathcal{A}$ is an adversary against the $(\mathcal{R},\mathcal{S},\{f_i\}_{i=1}^{\kappa})$ computational span-uber-assumption that has successfully output $\boldsymbol{\Delta} \neq \mathbf{0}$ and $[z]_1 = \sum_{i=1}^{\kappa} \Delta_i[f_i(\chi,y)]_1 = \sum_{i=1}^{\kappa} \Delta_i[\ell_{\nu-\kappa+i}(X)Y^{\eta+1}]_1$.

Since $\boldsymbol{\Delta} \neq \mathbf{0}$, then there exists at least one coordinate $I$ such that $\Delta_I \neq 0$. Let $\mathcal{B}$ be the following adversary against the $(\mathcal{R},\mathcal{S},f'_I)$ computational uber-assumption in $\mathbb{G}_T$. Given $\mathsf{ck}$, $\boldsymbol{\Delta}$, and $[z]_1$, $\mathcal{B}$ computes

$$1/\Delta_I \cdot [z]_1 \bullet [\ell_{\nu-\kappa+I}(\chi)y]_2 = \sum_{i=1}^{\kappa} \Delta_i/\Delta_I \cdot [\ell_{\nu-\kappa+i}(\chi)y^{\eta+1}]_1 \bullet [\ell_{\nu-\kappa+I}(\chi)y]_2 \ .$$

For $i \in [1,\kappa]$, let $d_i(X)$ be the rational function satisfying $d_i(X)\ell(X) = \ell_{\nu-\kappa+i}(X)\ell_{\nu-\kappa+I}(X)$. Clearly, $d_i(X)$ is a polynomial for $i \neq I$. Thus, $d(X) := \sum_{i\neq I} \Delta_i/\Delta_I \cdot d_i(X)$ is a polynomial of degree $\le \nu - 2$. Since $[y^\eta]_2$ is a part of the commitment key, $\mathcal{B}$ can efficiently compute

$$\sum_{i\neq I} \Delta_i/\Delta_I \cdot [\ell_{\nu-\kappa+i}(\chi)y^{\eta+1}]_1 \bullet [\ell_{\nu-\kappa+I}(\chi)y]_2 = \sum_{i\neq I} \Delta_i/\Delta_I \cdot [d_i(\chi)\ell(\chi)y^2]_1 \bullet [y^\eta]_2 = [d(\chi)\ell(\chi)y^2]_1 \bullet [y^\eta]_2 \ .$$

Thus, $\mathcal{B}$ can compute

$$[z^*]_T = [f'_I(\chi,y)]_T \leftarrow [\ell_{\nu-\kappa+I}(\chi)y^{\eta+1}]_1 \bullet [\ell_{\nu-\kappa+I}(\chi)y]_2 = 1/\Delta_I \cdot [z]_1 \bullet [\ell_{\nu-\kappa+I}(\chi)y]_2 - [d(\chi)\ell(\chi)y^2]_1 \bullet [y^\eta]_2$$

and break the $(\mathcal{R},\mathcal{S},f'_I)$-computational uber-assumption in $\mathbb{G}_T$. □

**Theorem 3 (Security of $\mathsf{FC}_{\mathsf{sn}}$).** *Let $\mathcal{C}$ be a fixed circuit and let $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ be the SFC scheme in Fig. 2. Let $\mathsf{ck} = ([\varrho(X,Y) : \varrho \in \mathcal{R}]_1, [\sigma(X,Y) : \sigma \in \mathcal{S}]_2)$ as in Fig. 2. For $i \in [1,\kappa]$, let $f_i(X,Y) := \ell_{\nu-\kappa+i}(X)Y^{\eta+1}$.*

1. *Assume Item c of Theorem 1 holds. Then $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ is correct.*
2. *$\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ is perfectly com-hiding.*

3. $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ *is perfectly open-hiding.*
4. $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ *is perfectly zero-knowledge.*
5. *Assume that either* $\gamma = 1$, $\delta = 0$, *and* $\eta = 3$ *or* $\gamma = 4$, $\delta = 0$, *and* $\eta = 7$. *Assume that Items d to g, i and j of Theorem 1 hold. If the* $(\mathcal{R}, \mathcal{S}, \{f_i\})$*-computational span-uber-assumption holds in* $\mathbb{G}_1$ *then the SFC scheme* $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ *is computationally evaluation-binding.*

*Proof.* **(1: correctness).** We first show that the prover can compute $[\mathsf{B}_i^{\mathsf{aux}}(\chi, y)]_1$, and then that the verification equation holds. Recall that for $i \in [1, \kappa]$, $\mathsf{B}_i^{\mathsf{aux}}(X, Y) = \ell_{\nu-\kappa+i}(X)\mathsf{B}_s(X, Y)Y = \ell_{\nu-\kappa+i}(X)(r_b + v_s(X, Y)Y)Y$, where $v_s(X)$ is as in Eq. (8). First, the addend $r_b\ell_{\nu-\kappa+i}(X)Y$ belongs to the span of $(X^iY)_{i=0}^{\nu-1} \subset \mathcal{R}$. Second, due to Item c of Theorem 1, for all $j \in [2, 1 + \mu_\alpha + \mu_\phi]$,

$$\ell(X) \mid \ell_{\nu-\kappa+i}(X)v_{\phi j}(X) \quad \text{and} \quad \ell(X) \mid \ell_{\nu-\kappa+i}(X)v_{\chi j}(X) \ ,$$

and thus $\mathsf{B}_i^{\mathsf{aux}}(X, Y) - r_b\ell_{\nu-\kappa+i}(X)Y$ is equal to $b'_i(X)\ell(X)Y^2$ for some polynomial $b'_i(X) \in \mathbb{Z}_p^{(\leq \nu-2)}[X]$. Thus, $\mathsf{B}_i^{\mathsf{aux}}(X) \in \mathrm{span}(\mathcal{R})$ and the committer can compute $[\ell_{\nu-\kappa+i}(\chi)\mathsf{B}_s y]_1 = [\mathsf{B}_i^{\mathsf{aux}}(\chi, y)]_1$.

Assume that $\mathsf{ck} \leftarrow \mathsf{KC}(1^\lambda, \mathcal{C})$, $([\mathsf{A}_s, \{\mathsf{B}_i^{\mathsf{aux}}\}_{i=1}^{\kappa}]_1, [\mathsf{B}_s]_2) \leftarrow \mathsf{com}(\mathsf{ck}; \boldsymbol{\alpha}; r_a, r_b)$ and $[\mathsf{C}_{sp}]_1 \leftarrow \mathsf{open}(\mathsf{ck}; ([\mathsf{A}_s, \{\mathsf{B}_i^{\mathsf{aux}}\}_{i=1}^{\kappa}]_1, [\mathsf{B}_s]_2), (\boldsymbol{\alpha}, r_a, r_b), \boldsymbol{\beta})$. It is clear that then the verifier accepts.

**(2: perfect com-hiding).** Follows from the fact that $([\mathsf{A}_s]_1, [\mathsf{B}_s]_2)$ is perfectly masked by uniformly random $r_a, r_b \leftarrow\!\!\$\, \mathbb{Z}_p$. Moreover, $[\mathsf{B}_i^{\mathsf{aux}}]_1$ are publicly verifiable deterministic functions of $[\mathsf{B}_s]_2$.

**(3: perfect open-hiding)**. Due to com-hiding and the fact that $[\mathsf{A}_p]_1$, $[\mathsf{B}_p]_2$, and $[\mathsf{C}_p]_1$ only depend on $(\boldsymbol{\beta}, \{\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta})\})$ (and not on $\boldsymbol{\alpha}$ otherwise), the distribution of all elements in the opening (except possibly $[\mathsf{C}_{sp}]_1$) is the same for any two vectors $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ that satisfy $\mathcal{F}_i(\boldsymbol{\alpha}_1, \boldsymbol{\beta}) = \mathcal{F}_i(\boldsymbol{\alpha}_2, \boldsymbol{\beta})$ for all $i$. Since $[\mathsf{C}_{sp}]_1$ is the unique element that makes the verifier accept, this means that the same claim holds for the whole opening, and $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ is open-hiding.

**(4: perfect zero-knowledge)**. We construct $\mathsf{Sim}$ as follows. It has $(\chi, y)$ as the trapdoor. It samples random $\mathsf{A}_s, \mathsf{B}_s \leftarrow\!\!\$\, \mathbb{Z}_p$, and then sets $[\mathsf{B}_i^{\mathsf{aux}}]_1 \leftarrow [\ell_{\nu-\kappa+i}(\chi)\mathsf{B}_s y]_1$ for all $i$. It computes $\mathsf{B}_p$ (by using the trapdoors), $[\mathsf{A}_p]_1$, and $[\mathsf{C}_p]_1$. It then computes the unique $[\mathsf{C}_{sp}]_1$ that makes the verifier accept,

$$[\mathsf{C}_{sp}]_1 \leftarrow ((\mathsf{A}_s + y^\delta)(\mathsf{B}_s + \mathsf{B}_p) + \mathsf{A}_s y^\eta)[1]_1 + (\mathsf{B}_s + \mathsf{B}_p + y^\eta)[\mathsf{A}_p]_1 - y^\gamma[\mathsf{C}_p]_1 \ .$$

**(5: evaluation-binding).** Assume that $\mathcal{A}$ is an evaluation-binding adversary that, with probability $\varepsilon_{\mathcal{A}}$ and in time $\tau_{\mathcal{A}}$, returns a collision

$$(([\mathsf{A}_s, \{\mathsf{B}_i^{\mathsf{aux}}\}_{i=1}^{\kappa}]_1, [\mathsf{B}_s]_2); \boldsymbol{\beta}; \boldsymbol{\xi}, [\mathsf{C}_{sp}]_1, \tilde{\boldsymbol{\xi}}, [\tilde{\mathsf{C}}_{sp}]_1) \tag{16}$$

with $\boldsymbol{\xi} \neq \tilde{\boldsymbol{\xi}}$, such that (here, $[\mathsf{A}_p, \mathsf{C}_p]_1$ / $[\tilde{\mathsf{A}}_p, \tilde{\mathsf{C}}_p]_1$ is the opening in the collision),

$$[\mathsf{A}_s + \mathsf{A}_p + y^\delta]_1 \bullet [\mathsf{B}_s + \mathsf{B}_p + y^\eta]_2 = [\mathsf{C}_{sp}]_1 \bullet [1]_2 + [\mathsf{C}_p]_1 \bullet [y^\gamma]_2 + [y^{\delta+\eta}]_T \ ,$$
$$[\mathsf{A}_s + \tilde{\mathsf{A}}_p + y^\delta]_1 \bullet [\mathsf{B}_s + \mathsf{B}_p + y^\eta]_2 = [\tilde{\mathsf{C}}_{sp}]_1 \bullet [1]_2 + [\tilde{\mathsf{C}}_p]_1 \bullet [y^\gamma]_2 + [y^{\delta+\eta}]_T \ ,$$

and $[\ell_{\nu-\kappa+i}(\chi)y]_1 \bullet [\mathsf{B}_s]_2 = [\mathsf{B}_i^{\mathsf{aux}}]_1 \bullet [1]_2$ for $i \in [1, \kappa]$. Here we used the fact that by Items f and j of Theorem 1 (see also the definition of $u_p(X)$ and $v_p(X)$ in Eqs. (7) and (8)), the value of $[\mathsf{B}_p]_2$ stays the same in both openings. The latter makes it possible to finish the reduction.

We now construct an adversary $\mathcal{B}$ against the computational uber-assumption in $\mathbb{G}_1$. From the collision, by subtracting the second equation from the first equation, we get

$$[\mathsf{A}_p - \tilde{\mathsf{A}}_p]_1 \bullet [\mathsf{B}_s + \mathsf{B}_p + y^\eta]_2 = [\mathsf{C}_{sp} - \tilde{\mathsf{C}}_{sp}]_1 \bullet [1]_2 + [\mathsf{C}_p - \tilde{\mathsf{C}}_p]_1 \bullet [y^\gamma]_2 \ .$$

Due to the properties of the pairing, this is equivalent to

$$[(\mathsf{A}_p - \tilde{\mathsf{A}}_p)(\mathsf{B}_s + \mathsf{B}_p + y^\eta)]_1 = [\mathsf{C}_{sp}]_1 - [\tilde{\mathsf{C}}_{sp}]_1 + [(\mathsf{C}_p - \tilde{\mathsf{C}}_p)y^\gamma]_1 \ . \tag{17}$$

Denote $\Delta_i := \xi_i - \tilde{\xi}_i$. By Eq. (9), $A_p(X) - \tilde{A}_p(X) = (u_p(X) - \tilde{u}_p(X))Y$. By the definition (see Eq. (7)) of $u_s(X)$, $u_s(X) = \sum_{j=2}^{\mu_\alpha+\mu_\phi+1} \mathsf{a}_j u_j(X)$ for some witness $\mathsf{a}$. After taking into account Items a, e and i of

Theorem 1, we get that $u_p(X) = u_1(X) + \ldots + \sum_{i=1}^{\kappa} \xi_i \ell_{\nu-\kappa+i}(X)$ as in Eq. (7) with $\xi_i$ being the claimed value of $\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta})$. We get a similar formula for $\tilde{u}_p$. Since the same $\boldsymbol{\beta}$ is used in the case of $u_p(X)$ and $\tilde{u}_p(X)$, $\tilde{u}_p(X)$ differs from $u_p(X)$ just by having different output values $\tilde{\xi}_i$. Thus,

$$u_p(X) - \tilde{u}_p(X) = \sum_{i=1}^{\kappa} \Delta_i \ell_{\nu-\kappa+i}(X) \ .$$

(In fact, Item a is not important to get this equality since the corresponding elements in $u_p(X)$ and $\tilde{u}_p(X)$ cancel out after substraction.) Thus, $A_p(X) - \tilde{A}_p(X) = (u_p(X) - \tilde{u}_p(X))Y = (\sum_{i=1}^{\kappa} \Delta_i \ell_{\nu-\kappa+i}(X))Y$.

On the other hand, by Eq. (10), $\mathsf{C}_p - \tilde{\mathsf{C}}_p = \sum_{i=1}^{\kappa} \Delta_i (u_{\mu-\kappa+i}(\chi) y^{\eta+1} + v_{\mu-\kappa+i}(\chi) y^{\delta+1} + w_{\mu-\kappa+i}(\chi) y^2)$. Hence, Eq. (17) is equivalent to

$$\begin{aligned}\left(\sum_{i=1}^{\kappa} \Delta_i \ell_{\nu-\kappa+i}(\chi) y\right)&(\mathsf{B}_s + \mathsf{B}_p + y^\eta) \\ &= (\mathsf{C}_{sp} - \tilde{\mathsf{C}}_{sp}) + \sum_{i=1}^{\kappa} \Delta_i \left(u_{\mu-\kappa+i}(\chi) y^{\eta+1} + v_{\mu-\kappa+i}(\chi) y^{\delta+1} + w_{\mu-\kappa+i}(\chi) y^2\right) \ .\end{aligned}$$

According to Items f and g of Theorem 1, $v_{\mu-\kappa+i}(\chi) = 0$ while $w_{\mu-\kappa+i}(X) = \ell_{\nu-\kappa+i}(\chi)$. Thus, the right-hand side of Eq. (17) is equal to $(\mathsf{C}_{sp} - \tilde{\mathsf{C}}_{sp}) + \sum_{i=1}^{\kappa} \Delta_i \ell_{\nu-\kappa+i}(\chi) y^2$. Going back to the group-based notation,

$$\sum_{i=1}^{\kappa} \Delta_i \left[\ell_{\nu-\kappa+i}(\chi) y (\mathsf{B}_s + \mathsf{B}_p + y^\eta)\right]_1 = [\mathsf{C}_{sp}]_1 - [\tilde{\mathsf{C}}_{sp}]_1 + \sum_{i=1}^{\kappa} \Delta_i [\ell_{\nu-\kappa+i}(\chi) y^2]_1 \ . \tag{18}$$

Now, let

$$[z]_1 := \sum_{i=1}^{\kappa} \Delta_i [\ell_{\nu-\kappa+i}(\chi) y^{\eta+1}]_1 \quad (= \sum_{i=1}^{\kappa} \Delta_i [f_i(\chi, y)]_1) \ .$$

In what follows, we show that $\mathcal{B}$ can compute $[z]_1$ and thus break the span-uber-assumption. Eq. (18) is equivalent to

$$\begin{aligned}[z]_1 + \sum_{i=1}^{\kappa} \Delta_i [\ell_{\nu-\kappa+i}(\chi)(\mathsf{B}_p - y) y]_1 =& [\mathsf{C}_{sp}]_1 - [\tilde{\mathsf{C}}_{sp}]_1 - \sum_{i=1}^{\kappa} \Delta_i [\ell_{\nu-\kappa+i}(\chi) \mathsf{B}_s y]_1 \\ =& [\mathsf{C}_{sp}]_1 - [\tilde{\mathsf{C}}_{sp}]_1 - \sum_{i=1}^{\kappa} \Delta_i [\mathsf{B}_i^{\mathsf{aux}}]_1 \ .\end{aligned}$$

That the last equation holds is guaranteed by $[\ell_{\nu-\kappa+i}(\chi) y]_1 \bullet [\mathsf{B}_s]_2 = [\mathsf{B}_i^{\mathsf{aux}}]_1 \bullet [1]_2$. Note that this is the place where we need the prover to help the verifier by computing the elements $[\mathsf{B}_i^{\mathsf{aux}}]_1$.

Next, we show how to efficiently compute $[\ell_{\nu-\kappa+i}(\chi)(\mathsf{B}_p - y) y]_1$. Define

$$t(X) := v_p(X) - \sum_{i=1}^{\kappa} \ell_{\nu-\kappa+i}(X) \ .$$

Recall $\mathsf{B}_p(X, Y) = v_p(X) Y$. Let $h_i'(X)$ be the rational function that satisfies

$$\begin{aligned}h_i'(X)\ell(X) =& \ell_{\nu-\kappa+i}(X) \left(\mathsf{B}_p(X, Y)/Y - 1\right) \\ =& \ell_{\nu-\kappa+i}(X) \left(t(X) + \sum_{i=1}^{\kappa} \ell_{\nu-\kappa+i}(X) - 1\right) \\ =& \ell_{\nu-\kappa+i}(X)(t(X) + \sum_{j \neq i} \ell_{\nu-\kappa+j}(X)) + \ell_{\nu-\kappa+i}(X)(\ell_{\nu-\kappa+i}(X) - 1) \ .\end{aligned} \tag{19}$$

Due to Item d of Theorem 1 and the definition of $t(X)$ (see also Eqs. (7) and (8)),

$$\ell(X) \mid \ell_{\nu-\kappa+i}(X)t(X) \ .$$

Moreover, $\ell(X) \mid \ell_{\nu-\kappa+i}(X)\ell_{\nu-\kappa+j}(X)$, for $i \neq j$, and $\ell(X) \mid \ell_{\nu-\kappa+i}(X)(\ell_{\nu-\kappa+i}(X)-1)$. Thus, the polynomial on the right-hand side of Eq. (19) divides by $\ell(X)$. Thus, $h'_i(X)$ is a polynomial of degree $\leq \nu-2$ and thus $\mathcal{B}$ can compute efficiently

$$[\ell_{\nu-\kappa+i}(\chi)(\mathsf{B}_p - y)y]_1 = [\ell_{\nu-\kappa+i}(\chi)(\mathsf{B}_p/y - 1)y^2]_1 = [h'_i(\chi)\ell(\chi)y^2]_1 \ ,$$

and then

$$[z]_1 = \sum_{i=1}^{\kappa} \Delta_i [\ell_{\nu-\kappa+i}(\chi)y^{\eta+1}]_1 \leftarrow ([\mathsf{C}_{sp}]_1 - [\tilde{\mathsf{C}}_{sp}]_1) - \sum_{i=1}^{\kappa} \Delta_i \left([\mathsf{B}_i^{\mathsf{aux}}]_1 + [h'_i(\chi)\ell(\chi)y^2]_1\right) \ .$$

Thus, given the collision Eq. (16), $\mathcal{B}$ can compute and output $(\boldsymbol{\Delta}, [z]_1 = \sum \Delta_i [f_i(\chi, y)]_1)$ for $f_i(X,Y) \notin \mathrm{span}(\mathcal{R})$. Thus, $\mathcal{B}$ breaks (with probability $\varepsilon_{\mathcal{A}}$ and time close to $t_{\mathcal{A}}$) the $(\mathcal{R}, \mathcal{S}, \{f_i\})$-computational span-uber-assumption in $\mathbb{G}_1$ in the case $f_i \notin \mathrm{span}(\mathcal{R})$. □

The following Corollary follows from Item 5 in Theorem 3 and Lemma 2.

**Corollary 1.** *Let $\mathcal{C}$ be a fixed circuit. Let $\gamma = 4$, $\delta = 0$, and $\eta = 7$. Let $f'_I(X,Y) := (\ell_{\nu-\kappa+I}(X))^2 Y^{\eta+2}$ for $I \in [1,\kappa]$. If the $(\mathcal{R}, \mathcal{S}, f'_I)$-computational uber-assumption holds in $\mathbb{G}_T$ for all $I \in [1,\kappa]$ then $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ is computationally evaluation-binding.*

*Remark 2.* The indeterminate $Y$ is crucial in establishing the independence of $f_i$ from $\mathcal{R}$. Consider the following example of why this is important. Let $\mathcal{R}^* := \{(X^i)_{i=0}^{\nu-1}, (X^i\ell(X))_{i=0}^{\nu-2}\}$, $\mathcal{S}^* := \{(X^i)_{i=0}^{\nu-1}\}$, and $f_i^* := \ell_{\nu-\kappa+i}(X)$. One can establish that $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ is evaluation-binding under the $(\mathcal{R}^*, \mathcal{S}^*, \{f_i^*\})$-computational span-uber-assumption in $\mathbb{G}_1$. Really, consider the following $(\mathcal{R}^*, \mathcal{S}^*, \{f_i^*\})$-span-uber-assumption adversary $\mathcal{B}^*$ that will create $y$ herself, generate a new $\mathsf{ck}$ based on her input and $y$, and then use $\mathcal{B}$ in Theorem 3 to break the $(\mathcal{R}^*, \mathcal{S}^*, \{f_i^*\})$-computational span-uber-assumption. $\mathcal{B}^*$ will have similar success as $\mathcal{B}$. However, $f_i^* \in \mathrm{span}(\mathcal{R}^*)$ and thus the $(\mathcal{R}^*, \mathcal{S}^*, \{f_i^*\})$-computational span-uber-assumption itself is not secure.

**On the Security of the Span-Uber-Assumption.** It is known that in composite-order bilinear groups, the computational uber-assumption in $\mathbb{G}_T$ holds under appropriate subgroup hiding assumptions [CMM16]. Hence, a composite-order group version of the span-uber-assumption (and also of the new SFC) is secure under a subgroup hiding assumption. In Appendix D, we will use the Déjà Q approach of [CM14] directly to prove that the span-uber-assumption in $\mathbb{G}_\iota$, $\iota \in \{1,2\}$, is secure under a subgroup hiding assumption. More precisely, we establish the following corollary. (See Appendix D for the definition of subgroup hiding and extended adaptive parameter hiding.)

**Theorem 4.** *The $(\mathcal{R}, \mathcal{S}, \{f_i\}_{i=1}^{\kappa})$-computational span-uber-assumption holds in the source group $\mathbb{G}_1$ with all but negligible probability if*

1. *subgroup hiding holds in $\mathbb{G}_1$ with respect to $\mu = \{\mathsf{P}_1^2, \mathsf{P}_2^1\}$,*
2. *subgroup hiding holds in $\mathbb{G}_2$ with respect to $\mu = \{\mathsf{P}_1^1\}$,*
3. *extended adaptive parameter hiding holds with respect to $\mathcal{R} \cup \{f_i\}_{i=1}^{\kappa}$ and $\mathsf{aux} = \{\mathsf{P}_2^{1\,\sigma(\cdot)}\}_{\sigma \in \mathcal{S}}$ for any $\mathsf{P}_2^1 \in \mathbb{G}_2$.*
4. *the polynomials in $\mathcal{R}$ have maximum degree $\mathsf{poly}(\lambda)$.*

Here, $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are additive groups of composite order $N = p_1 p_2$ ($p_1 \neq p_2$) and $\mathsf{P}_\iota^1 \in \mathbb{G}_{\iota,p_1}$, $\mathsf{P}_\iota^2 \in \mathbb{G}_{\iota,p_2}$ are randomly sampled subgroup generators, where $\mathbb{G}_{\iota,p_j}$ is the subgroup of $\mathbb{G}_\iota$ of order $p_j$ and $\mathsf{P}_\iota \in \mathbb{G}_\iota = \mathbb{G}_{\iota,p_1} \oplus \mathbb{G}_{\iota,p_2}$.

The direct proof in Appendix D is simpler than the mentioned two-step proof since it does not rely on the intermediate step of reducing the span-uber-assumption to a uber-assumption in $\mathbb{G}_T$. Moreover, the Déjà

Q approach is more straightforward in case one works in the source group. We will leave it up to future work to reduce *prime-order* span-uber-assumption to a simpler assumption; there has been almost no prior work on reducing prime-order *assumptions*.

Finally, in Appendix B, we will prove that the span-uber-assumption is secure in the algebraic group model [FKL18] under the well-known PDL assumption [Lip12].[1] Following the semi-generic-group model of [JR10], the original AGM of [FKL18] (though later extensions exist) is defined only in the case when the adversary outputs elements in the source groups (but not in $\mathbb{G}_T$), and thus one cannot prove the security of the computational uber-assumption in $\mathbb{G}_T$ using the approach of [Lip19]. Thus, in a well-defined sense, the span-uber-assumption is weaker than the uber-assumption in $\mathbb{G}_T$.

## References


ABLZ17. Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michal Zajac. A subversion-resistant SNARK. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2017. doi:10.1007/978-3-319-70700-6_1. 2, B.1

ALSZ20. Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michal Zajac. On QA-NIZK in the BPK model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 590–620. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45374-9_20. 2

ALSZ21. Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michal Zajac. On Subversion-Resistant SNARKs. *J. Cryptology*, 34(3):1–42, 2021. doi:10.1007/s00145-021-09379-y. 2, B.1

BBF19. Dan Boneh, Benedikt Bünz, and Ben Fisch. Batching techniques for accumulators with applications to IOPs and stateless blockchains. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 561–586. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26948-7_20. 1, 1

BBG05. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 440–456. Springer, Heidelberg, May 2005. doi:10.1007/11426639_26. 1, 2, 1, 2, 5, B

Bd94. Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In Tor Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 274–285. Springer, Heidelberg, May 1994. doi:10.1007/3-540-48285-7_24. 1, 1, 1, A.1

BDFG20. Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Efficient polynomial commitment schemes for multiple points and polynomials. Cryptology ePrint Archive, Report 2020/081, 2020. https://eprint.iacr.org/2020/081. 1, 1, A.2, A.2

BFL20. Balthazar Bauer, Georg Fuchsbauer, and Julian Loss. A classification of computational assumptions in the algebraic group model. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 121–151. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56880-1_5. B.2

BFS16. Mihir Bellare, Georg Fuchsbauer, and Alessandra Scafuro. NIZKs with an untrusted CRS: Security in the face of parameter subversion. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 777–804. Springer, Heidelberg, December 2016. doi:10.1007/978-3-662-53890-6_26. 2, B.1

BGH19. Sean Bowe, Jack Grigg, and Daira Hopwood. Halo: Recursive Proof Composition without a Trusted Setup. Technical report, 2019. Available from https://electriccoin.co/wp-content/uploads/2019/09/Halo.pdf. 1, 1, A.3

BGN05. Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF Formulas on Ciphertexts. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 325–341, Cambridge, MA, USA, February 10–12, 2005. Springer, Heidelberg. 11

Bit17. Nir Bitansky. Verifiable random functions from non-interactive witness-indistinguishable proofs. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pages 567–594. Springer, Heidelberg, November 2017. doi:10.1007/978-3-319-70503-3_19. 1


[1] As a corollary of independent interest, we also show in Appendix B that if $f \notin \text{span}(\mathcal{R})$ then the $(\mathcal{R}, \mathcal{S}, \mathcal{T})$-uber-assumption follows in the AGM under PDL.

Boy08. Xavier Boyen. The uber-assumption family (invited talk). In Steven D. Galbraith and Kenneth G. Paterson, editors, *PAIRING 2008*, volume 5209 of *LNCS*, pages 39–56. Springer, Heidelberg, September 2008. doi:10.1007/978-3-540-85538-5_3. 2, 1, 2

BP97. Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 480–494. Springer, Heidelberg, May 1997. doi:10.1007/3-540-69053-0_33. 1, A.1

Bro01. Daniel R. L. Brown. The exact security of ECDSA. Contributions to IEEE P1363a, January 2001. http://grouper.ieee.org/groups/1363/. B.1

CF13. Dario Catalano and Dario Fiore. Vector commitments and their applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 55–72. Springer, Heidelberg, February / March 2013. doi:10.1007/978-3-642-36362-7_5. 1, 1, 1, A.1

CHM+20. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 738–768. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45721-1_26. 1, 1, A.1, A.2

CK18. Sanjit Chatterjee and R. Kabaleeshwaran. Towards static assumption based cryptosystem in pairing setting: Further applications of DéjàQ and dual-form signature (extended abstract). In Joonsang Baek, Willy Susilo, and Jongkil Kim, editors, *ProvSec 2018*, volume 11192 of *LNCS*, pages 220–238. Springer, Heidelberg, October 2018. doi:10.1007/978-3-030-01446-9_13. D.1, 14

CM14. Melissa Chase and Sarah Meiklejohn. Déjà Q: Using dual systems to revisit q-type assumptions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 622–639. Springer, Heidelberg, May 2014. doi:10.1007/978-3-642-55220-5_34. 1, 5, D, 10, 11, D.1, 12, 13, D.2, 3, 3, D.2

CMM16. Melissa Chase, Mary Maller, and Sarah Meiklejohn. Déjà Q all over again: Tighter and broader reductions of q-type assumptions. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 655–681. Springer, Heidelberg, December 2016. doi:10.1007/978-3-662-53890-6_22. 1, 5, D, 11, 12, 3, 3, D.2

Den02. Alexander W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 100–109. Springer, Heidelberg, December 2002. doi:10.1007/3-540-36178-2_6. 1, B

Fis00. Marc Fischlin. A note on security proofs in the generic model. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 458–469. Springer, Heidelberg, December 2000. doi:10.1007/3-540-44448-3_35. 1, B

FKL18. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018. doi:10.1007/978-3-319-96881-0_2. 1, 5, 5, B, B.1, B.2

Fre10. David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 44–61. Springer, Heidelberg, May / June 2010. doi:10.1007/978-3-642-13190-5_3. D.1

Fuc18. Georg Fuchsbauer. Subversion-zero-knowledge SNARKs. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 315–347. Springer, Heidelberg, March 2018. doi:10.1007/978-3-319-76578-5_11. 2

GGPR13. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Heidelberg, May 2013. doi:10.1007/978-3-642-38348-9_37. 1, 1, 2, 2, 3

Gro10. Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340. Springer, Heidelberg, December 2010. doi:10.1007/978-3-642-17373-8_19. 1, 2

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016. doi:10.1007/978-3-662-49896-5_11. 1, 1, 1, 1, 2, C

GRWZ20. Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 2007–2023. ACM Press, November 2020. doi:10.1145/3372297.3417244. 1, 1

GW11. Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011. doi:10.1145/1993636.1993651. 1, 1

ILV11. Malika Izabachène, Benoît Libert, and Damien Vergnaud. Block-wise P-signatures and non-interactive anonymous credentials with efficient attributes. In Liqun Chen, editor, *13th IMA International Conference on Cryptography and Coding*, volume 7089 of *LNCS*, pages 431–450. Springer, Heidelberg, December 2011. 1, 1, 4, A.2

JR10. Tibor Jager and Andy Rupp. The semi-generic group model and applications to pairing-based cryptography. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 539–556. Springer, Heidelberg, December 2010. doi:10.1007/978-3-642-17373-8_31. 5, 5

KZG10. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194. Springer, Heidelberg, December 2010. doi:10.1007/978-3-642-17373-8_11. 1, 1, 1, A.1

Lip12. Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 169–189. Springer, Heidelberg, March 2012. doi:10.1007/978-3-642-28914-9_10. 1, 1, 2, 5, B

Lip13. Helger Lipmaa. Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 41–60. Springer, Heidelberg, December 2013. doi:10.1007/978-3-642-42033-7_3. 1, 2

Lip19. Helger Lipmaa. Simulation-Extractable ZK-SNARKs Revisited. Technical Report 2019/612, IACR, May 31, 2019. https://ia.cr/2019/612, updated on 8 Feb 2020. 1, 1, 1, 1, 2, 3, 3, 3, 3, 5

LM19. Russell W. F. Lai and Giulio Malavolta. Subvector commitments with application to succinct arguments. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 530–560. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26948-7_19. 1, 1, A.2, A.2

LP20. Helger Lipmaa and Kateryna Pavlyk. Succinct functional commitment for a large class of arithmetic circuits. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 686–716. Springer, Heidelberg, December 2020. doi:10.1007/978-3-030-64840-4_23. ⋆, ⋆

LRY16. Benoît Libert, Somindu C. Ramanna, and Moti Yung. Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *ICALP 2016*, volume 55 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl, July 2016. doi:10.4230/LIPIcs.ICALP.2016.30. 1, 1, 1, 2, 2, 1, A.1, A.2

LY10. Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 499–517. Springer, Heidelberg, February 2010. doi:10.1007/978-3-642-11799-2_30. 1

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, November 2019. doi:10.1145/3319535.3339817. 1

PHGR13. Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013. doi:10.1109/SP.2013.47. 1, 2

PST13. Charalampos Papamanthou, Elaine Shi, and Roberto Tamassia. Signatures of correct computation. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 222–242. Springer, Heidelberg, March 2013. doi:10.1007/978-3-642-36594-2_13. 1, A.3

Set20. Srinath Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 704–737. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56877-1_25. A.3

SY10. Amir Shpilka and Amir Yehudayoff. *Arithmetic Circuits: A Survey of Recent Results and Open Questions*, volume 5 of *Foundations and Trends in Theoretical Computer Science*. Now Publishers Inc, December 2010. 4, 4

TAB+20. Alin Tomescu, Ittai Abraham, Vitalik Buterin, Justin Drake, Dankrad Feist, and Dmitry Khovratovich. Aggregatable subvector commitments for stateless cryptocurrencies. In Clemente Galdi and Vladimir Kolesnikov, editors, *SCN 20*, volume 12238 of *LNCS*, pages 45–64. Springer, Heidelberg, September 2020. doi:10.1007/978-3-030-57990-6_3. 1, 1, A.2

Val79. Leslie G. Valiant. Completeness Classes in Algebra. In *STOC 1979*, pages 249–261, Atlanta, Georgia, USA, 30 April—2 May 1979. 1, 4

WTS+18. Riad S. Wahby, Ioanna Tzialla, Abhi Shelat, Justin Thaler, and Michael Walfish. Doubly-Efficient zkSNARKs Without Trusted Setup. In *IEEE SP 2018*, pages 926–943, San Francisco, California, USA, May 21–23, 2018. IEEE Computer Society. 1

ZGK+17. Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. vSQL: Verifying Arbitrary SQL Queries over Dynamic Outsourced Databases. In *IEEE SP 2017*, pages 863–880, San Jose, CA, USA, May 22–26, 2017. IEEE Computer Society. 1

# A Applications and Efficiency Comparison

## A.1 Applications of Succinct IPFC

The following applications of *inner-product* functional commitment were mentioned in [LRY16].

**Succinct Polynomial Commitment.** Let $n$ be a fixed integer. In a polynomial commitment scheme [KZG10], the committer commits to a polynomial $\alpha(X) \in \mathbb{Z}_p^{(\leq n)}[X]$ of degree $\leq n$. Later, after getting an evaluation point $\beta$, he opens the functional commitment to $\alpha(\beta)$. A polynomial commitment scheme is evaluation-binding if it is hard to open the same commitment to two different evaluations of the same polynomial. Another essential property of polynomial commitment schemes is extractability: namely, it is required that the commitment corresponds to *some* polynomial of degree $\leq n$ that can be extracted [CHM+20]. A succinct polynomial commitment scheme can be implemented as a special case of the succinct IPFC, with $\boldsymbol{\alpha}$ being the vector of coefficients of $\alpha(X)$ and $\boldsymbol{\beta} = (1, \beta, \beta^2, \ldots, \beta^{n-1})$ for some $\beta \in \mathbb{Z}_p$. We leave it as an open question whether such a commitment scheme would be extractable but it seems likely due to its relation to SNARKs.

**Succinct Accumulator.** In an accumulator [Bd94,BP97], the committer commits to a set $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ and later opens the set to any of its elements $\alpha_i$. One can implement a succinct accumulator by defining a polynomial $\chi_{\boldsymbol{\alpha}}(X) := \prod(X - \alpha_i)$ and then using a succinct polynomial commitment scheme for this polynomial. The committer opens to $\beta = \alpha_i$ as in the polynomial commitment scheme.

**Succinct Vector Commitment.** Vector commitment [CF13] is a special case of the functional commitment, where given $I \in [1, n]$, the committer opens the commitment to $\alpha_I$. One can implement succinct vector commitment as a succinct IPFC by letting $\beta = \boldsymbol{e}_I$ to be the $I$th unit vector. This only changes the opening and verification procedures, see Fig. 3. Here, the opening consists of $1 + (n-1) + n + 1 = 2n + 1 = \mu$ exponentiations in $\mathbb{G}_1$ and the verifier has to execute 2 exponentiations in $\mathbb{G}_1$ and 3 pairings.

$\mathsf{open}(\mathsf{ck}; [\mathsf{A}_s]_1, (\boldsymbol{\alpha}, r_a), I)$: $\xi \leftarrow \alpha_I$; $u(X) \leftarrow \sum_{j=1}^{n} \alpha_j \ell_j(X) + \xi \ell_\nu(X)$; $v(X) \leftarrow \ell_I(X) + \ell_\nu(X)$; $w(X) \leftarrow \alpha_I \ell_I(X) + \xi \ell_\nu(X)$; $\mathcal{H}(X) = \sum_{i=0}^{n-2} \mathcal{H}_i X^i \leftarrow (u(X)v(X) - w(X))/\ell(X)$;

return $[\mathsf{C}_{sp}]_1 \leftarrow r_a([\ell_I(\chi)y]_1 + [\ell_\nu(\chi)y]_1 + [y^\eta]_1) + \sum_{i=0}^{n-2} \mathcal{H}_i[\chi^i \ell(\chi) y^2]_1 + \sum_{j=1}^{n} \alpha_j \left[\ell_j(\chi) y^{\eta+1}\right]_1 + \alpha_I \left[\ell_\nu(\chi) y^{\eta+1} + \ell_I(\chi) y^2\right]_1$ ;

$\mathsf{V}(\mathsf{ck}; [\mathsf{A}_s]_1, [\mathsf{C}_{sp}]_1, I, \xi)$: $[\mathsf{A}_p]_1 \leftarrow \xi[\ell_\nu(\chi)y]_1$; $[\mathsf{C}_p]_1 \leftarrow \xi[\ell_\nu(\chi) y^{2-\gamma}]_1$; $[\mathsf{B}_p]_2 \leftarrow [\ell_I(\chi)y]_2 + [\ell_\nu(\chi)y]_2$;
check that $([\mathsf{A}_s]_1 + [\mathsf{A}_p]_1) \bullet [\mathsf{B}_p + y^\eta]_2 = [\mathsf{C}_{sp}]_1 \bullet [1]_2 + [\mathsf{C}_p]_1 \bullet [y^\gamma]_2$;

**Fig. 3.** $\mathsf{open}$ and $\mathsf{V}$ for a VC scheme $\mathsf{FC}^{\mathsf{vc}}_{\mathsf{sn}}$ based on $\mathsf{FC}^{\mathsf{ip}}_{\mathsf{sn}}$

## A.2 Aggregated IPFC, PC and VC from the New SFC

**Aggregation.** In many applications (see, e.g., [LM19,BDFG20,TAB+20]), one needs to use an SFC scheme several times in parallel. Sometimes, the SFC is used for the same circuit $\mathcal{C}$ but on different public data (e.g.,

in the subvector commitment [LM19]). Sometimes, one needs to do it for the same circuit $\mathcal{C}$ but on different private and public data (e.g., in [BDFG20], one needs to commit to different polynomials and open each of them evaluated at a different point). Sometimes, one needs to do it for different circuits (e.g., aggregate vector commitments together with polynomial commitments). Up to now, the best general solution for this is to design a separate aggregated (say) vector commitment and an aggregated (say) polynomial commitment scheme and then run them in parallel. In this section, we will give examples of aggregations of each type.

One can use $\mathsf{FC}_{\mathsf{sn}}$ to arbitrarily aggregate a polynomial number of SFC runs, assuming that each SFC by itself can be implemented by a poly-size compiled circuit. We will not provide general definitions of aggregation (see, e.g., [TAB+20] for detailed definitions of aggregated subvector commitments) or give a detailed construction of general aggregation. However, the underlying aggregation idea is simple: let $\boldsymbol{\alpha}$ be the private data of the committer and let $\boldsymbol{\beta}$ be the private data of the verifier (in all runs of the SFC). Compile the circuit $\mathcal{C}$ that computes all output of all SFC runs to a circuit $\mathcal{C}^*$. Apply then our template to that circuit. Obviously, the aggregated version is just equal to $\mathsf{FC}_{\mathsf{sn}}$ and has its efficiency properties (e.g., it has the opening that of one group element). The CRS has to be extended to $\mathcal{C}$, but the communication will not change significantly.

We emphasize that this means that the prover can aggregate several SFCs into one, while aggregating several SFC commitments and openings by say the verifier is a completely different question.

Next, we will detail some applications of $\mathsf{FC}_{\mathsf{sn}}$. In each case, we will show how to design $U$, $V$, $W$, and the witness vector $\mathbf{a}$, such that $\mathsf{FC}_{\mathsf{sn}}$ works for the desired functions $\mathcal{F}_i$.

**Succinct Aggregated Inner Product (aIP).** Assume that the committer commits to $\boldsymbol{\alpha} \in \mathbb{Z}_p^n$ and then opens it simultaneously to $\langle \boldsymbol{\alpha}, \boldsymbol{\beta}_i \rangle = \sum_{j=1}^n \alpha_j \beta_{ij}$ for $\kappa$ different verifier-provided vectors $\boldsymbol{\beta}_i \in \mathbb{Z}_p^n$, where $i \in [1, \kappa]$. Thus, $\boldsymbol{\beta} \in \mathbb{Z}_p^{\kappa n}$. The circuits $\mathcal{C}_\phi$ or $\mathcal{C}_\psi$ are dummy (they have no gates). Given $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}_i$, $\mathcal{C}_\chi$ computes $\kappa n$ products $\chi_{ij}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \alpha_j \beta_{ij}$, $i \in [1, \kappa]$ and $j \in [1, n]$, and $\mathcal{C}_\xi$ sums them together to obtain $\kappa$ outputs $\mathcal{F}_i(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^n \alpha_j \beta_{ij}$. Thus,

$$\mathbf{a} = (1, \alpha_1, \ldots, \alpha_n, \beta_{11} \ldots, \beta_{\kappa n}, \alpha_1\beta_{11}, \ldots, \alpha_n\beta_{n\kappa}, \textstyle\sum_{j=1}^n \alpha_j\beta_{1j}, \ldots, \sum_{j=1}^n \alpha_j\beta_{\kappa j}) \ ,$$

and $U_\chi = \mathbf{1}_\kappa \otimes I_n \in \mathbb{Z}_p^{\kappa n \times n}$, $V_\chi = I_{\kappa n}$, $U_\xi = \mathbf{1}_n^\top \otimes I_\kappa \in \mathbb{Z}_p^{\kappa \times \kappa n}$ (note that $\mathcal{C}_\chi$ does not take 1 as an input), and

$$U = \left(\begin{array}{c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\beta} & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & U_\chi & & & \\ \hline & & & U_\xi & \end{array}\right), \quad V = \left(\begin{array}{c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\beta} & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & V_\chi & & \\ \hline \mathbf{1}_\kappa & & & & \end{array}\right), \quad W = \left(\begin{array}{c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\beta} & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & & I_{\kappa n} & \\ \hline & & & & I_\kappa \end{array}\right) \ .$$

Here, $\nu = \kappa n + \kappa$, $\mu = 1 + n + \kappa n + \kappa n + \kappa = 2\kappa + n + \kappa + 1$, $\mathsf{A}_s(X, Y) = r_a + \sum_{j=1}^n \alpha_j u_{\chi j}(X) Y$, $\mathsf{A}_p(X, Y) = \sum_{i=1}^\kappa \langle \boldsymbol{\alpha}, \boldsymbol{\beta}_i \rangle \ell_{\nu-\kappa+i}(X) Y$. Importantly, $\mathsf{B}_s(X, Y) = 0$ (since there is nothing to hide, one can set $r_b \leftarrow 0$; hence, also $\mathsf{B}_i^{\mathsf{aux}}(X, Y) = 0$; thus the commitment is only one group element, $[\mathsf{A}_s]_1$), and $\mathsf{B}_p(X, Y) = \sum_{i=1}^\kappa \ell_{\nu-\kappa+i}(X) Y + \sum_{i=1}^\kappa \sum_{j=1}^n \beta_{ij} v_{\chi, n(i-1)+j}(X) Y$.

We will briefly compare the resulting aggregated IPFC with the (non-aggregated) IPFCs of [ILV11,LRY16]. In all cases, both the functional commitment and the opening consist of a single group element. Interestingly, a straightforward [ILV11] opening takes $\Theta(n^2)$ multiplications (this can be probably optimized [2]), while in our case it takes $\Theta(n \log n)$ multiplications.

Summarizing, while the presented non-aggregated IPFC is just a simple specialization of a much more general SFC scheme, it is only slightly less efficient than [ILV11].

[2] One can optimize our IPFC scheme even further, because here, one uses disjoint witnesses $\mathbf{a} = \boldsymbol{\alpha} // \boldsymbol{\chi}(\boldsymbol{\alpha}, \boldsymbol{\beta}) // \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ and $\mathbf{b} = 1 // \boldsymbol{\beta}$ in the R1CS key equation $U\mathbf{a} \circ V\mathbf{b} = W\mathbf{a}$. That is, one does not need to prove that the witnesses $\mathbf{a}$ and $\mathbf{b}$ are the same. Essentially, it means that one can omit the added $Y^\delta$ when defining the polynomial $\mathsf{C}(X, Y)$. We omit further discussion.

**Succinct Aggregated Polynomial Commitment (aPC) [CHM+20,BDFG20].** Assume that the committer commits to the $\kappa$ polynomials $\alpha_i \in \mathbb{Z}_p^{(\le n)}[X]$ (thus, $\boldsymbol{\alpha} \in \mathbb{Z}_p^{\kappa(n+1)}$) and later opens the commitment to the evaluation of each polynomial at a different point $\beta_i \in \mathbb{Z}_p$, $i \in [1,\kappa]$. That is, $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) = (\alpha_i(\beta_i))_{i=1}^{\kappa} = (\sum_{j=0}^{n} \alpha_{ij}\beta_i^j)_{i=1}^{\kappa}$. Here, $\mathcal{C}_\phi$ is an empty circuit, $\mathcal{C}_\psi$ computes all the evaluations of monomials $\boldsymbol{\psi}(\boldsymbol{\beta}) = (\beta_i^{1+j})_{i\le\kappa,1\le j\le n-1} \in \mathbb{Z}_p^{\kappa(n-1)}$, $\mathcal{C}_\chi$ computes products $\boldsymbol{\chi}(\boldsymbol{\beta}) = (\alpha_{ij}\beta_i^j)_{i\le\kappa,0\le j\le n} \in \mathbb{Z}_p^{\kappa(n+1)}$, and $\mathcal{C}_\xi$ computes $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \in \mathbb{Z}_p^{\kappa}$.

Thus, $U_\psi = (I_{\kappa(n-1)}, \mathbf{0}_{\kappa(n-1)\times\kappa}) \in \mathbb{Z}_p^{\kappa(n-1)\times\kappa n}$ (that is, the columns labeled by $\beta_i^n$ are all-zero), $V_\psi = \mathbf{1}_{n-1} \otimes I_\kappa \in \mathbb{Z}_p^{\kappa(n-1)\times\kappa}$, $W_\psi = I_{\kappa(n-1)} \in \mathbb{Z}_p^{\kappa(n-1)\times\kappa(n-1)}$, and

$$U_\psi \begin{pmatrix} \boldsymbol{\beta} \\ \boldsymbol{\psi}(\boldsymbol{\beta}) \end{pmatrix} \circ V_\psi \boldsymbol{\beta} = \boldsymbol{\psi}(\boldsymbol{\beta})$$

as in 4. Also, $U_\chi = I_{\kappa(n+1)} \in \mathbb{Z}_p^{\kappa(n+1)\times\kappa(n+1)}$,

$$V_\chi = \begin{pmatrix} \mathbf{1}_{\kappa-1}, 0_{\kappa-1,\kappa n} \\ I_{(\kappa n+1)} \end{pmatrix} \in \mathbb{Z}_p^{\kappa(n+1)\times(\kappa n+1)}$$

(that is, the first $\kappa-1$ rows of $V_\chi$ are each equal to $\boldsymbol{e}_1^\top$), $W_\chi = I_{\kappa(n+1)} \in \mathbb{Z}_p^{\kappa(n+1)\times\kappa(n+1)}$, and

$$U_\chi \boldsymbol{\alpha} \circ V_\chi \begin{pmatrix} 1 \\ \boldsymbol{\beta} \\ \boldsymbol{\psi}(\boldsymbol{\beta}) \end{pmatrix} = \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta})$$

as in 4. Finally, $U_\xi = \mathbf{1}_{n+1}^\top \otimes I_\kappa \in \mathbb{Z}_p^{\kappa\times\kappa(n+1)}$ and $U_\xi \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) \circ \mathbf{1} = \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta})$ as in 4. Altogether,

$$U = \left(\begin{array}{c|c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & U_\psi & & & \\ \hline & U_\chi & & & & \\ \hline & & & & U_\xi & \end{array}\right), \quad V = \left(\begin{array}{c|c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & V_\psi & & & \\ \hline V_\chi & & V_\chi & & & \\ \hline \mathbf{1}_\kappa & & & & & \end{array}\right), \quad W = \left(\begin{array}{c|c|c|c|c|c} 1 & \boldsymbol{\alpha} & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) \\ \hline & & & I_{\kappa(n-1)} & & \\ \hline & & & & I_{\kappa(n+1)} & \\ \hline & & & & & I_\kappa \end{array}\right).$$

Thus, $\nu = \kappa(n-1)+\kappa(n+1)+\kappa = \kappa(2n+1)$ and $\mu = 1+\kappa(n+1)+\kappa+\kappa(n-1)+\kappa(n+1)+\kappa = 3\kappa(n+1)+1$.

**Succinct Subvector Commitment Scheme (SVC).** Let $\boldsymbol{\alpha} = (\alpha_i)_{i=1}^n \in \mathbb{Z}_p^n$. An SVC [LM19] is a generalization of vector commitment, where $\boldsymbol{\beta} = S = (S_i)_{i=1}^{\kappa}$ is a set of indexes $S_i \in [1,n]$. One opens to $\boldsymbol{\mathcal{F}}(\boldsymbol{\alpha},\boldsymbol{\beta}) = (\alpha_{S_i})_{i=1}^{\kappa}$ for all $i \in \boldsymbol{\beta}$. One can construct a SVC scheme based on an SFC for multi-output inner product (see the previous example), by setting $\beta_i = \boldsymbol{e}_{S_i}$.

Essentially, here we have the same example as in the case of aggregated inner product, except that $\beta_i = \boldsymbol{e}_I$ for some index $I \in [1,n]$. Here, $\mathsf{A}_s(X,Y) = r_a + \sum_{j=1}^{n} \alpha_j u_{\chi j}(X)Y$, $\mathsf{A}_p(X,Y) = \sum_{i=1}^{\kappa} \alpha_{S_i}(\boldsymbol{\alpha},\boldsymbol{\beta})\ell_{\nu-\kappa+i}(X)Y$, $\mathsf{B}_s(X,Y) = 0$ (again, since there is nothing to hide, one can set $r_b \leftarrow 0$; this means that also $\mathsf{B}_i^{\mathsf{aux}}(X,Y) = 0$), and $\mathsf{B}_p(X,Y) = \sum_{i=1}^{\kappa} \ell_{\nu-\kappa+i}(X)Y + \sum_{i=1}^{\kappa} v_{\chi,S_i}(X)Y$. One obtains a vector commitment scheme from this when $\kappa = 1$.

### A.3 Other applications of SFC

**Succinct Evaluation-Point Commitment.** Let $\alpha$ be a point from $\mathbb{Z}_p$ and $\beta_i(X) = \sum_{j=0}^{n} \beta_{ij}X^j \in \mathbb{Z}_p^{(\le n)}[X]$ for $i \in [1,\kappa]$, e.g., $\boldsymbol{\beta} \in \mathbb{Z}_p^{\kappa(n+1)}$. Then $\boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) = (\beta_i(\alpha))_{i=1}^{\kappa} = (\sum_{j=0}^{n} \beta_{ij}\alpha^j)_{i=1}^{\kappa}$ and SFC commits to an evaluation point $\alpha$ and then open it to the evaluation of $\kappa$ different public polynomials of degree $\le n$ at $\alpha$.

In this case, $\mathcal{C}_\phi$ computes $n-1$ different exponents $\phi_j(\alpha) = \alpha^{1+j}$ for $j \in [1,n-1]$, $\mathcal{C}_\psi$ is an empty circuit since no computation is needed over the polynomial $\beta_i$ coefficients to be done, $\mathcal{C}_\chi$ computes $\kappa(n+1)$

products $\chi_{ij}(\alpha,\boldsymbol{\beta}) = \beta_{ij}\alpha^j$ for $i \in [1,\kappa]$, $j \in [0,n]$, and $\mathcal{C}_\xi$ sums up them together to obtain $\kappa$ outputs $\mathcal{F}_i(\alpha,\boldsymbol{\beta}) = \sum_{j=0}^{n} \chi_{ij}(\alpha,\boldsymbol{\beta})$.

Thus, here $\mathbf{a} = 1//\alpha//\boldsymbol{\phi}(\alpha)//\boldsymbol{\beta}//\boldsymbol{\chi}(\alpha,\boldsymbol{\beta})//\boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta})$, $U_\phi = \mathbf{1}_{n-1} \in \mathbb{Z}_p^{(n-1)\times 1}$, $\boldsymbol{\alpha}^* = \alpha$, $V_\phi = (I_{n-1}, \mathbf{0}_{n-1})$ (the last column of $V_\phi$, labeled by $\alpha^n$, is all-zero), $\boldsymbol{\beta}^* = (\alpha, \phi(\alpha))$, and

$$U_\phi \alpha \circ V_\phi \left(\begin{smallmatrix} \alpha \\ \boldsymbol{\phi}(\alpha) \end{smallmatrix}\right) = \boldsymbol{\phi}(\alpha) \ .$$

Also $U_\chi = V_\chi = \mathbf{1}_\kappa \otimes I_{n+1} \in \mathbb{Z}_p^{\kappa(n+1)\times(n+1)}$, and

$$U_\chi \left(\begin{smallmatrix} 1 \\ \alpha \\ \boldsymbol{\phi}(\alpha) \end{smallmatrix}\right) \circ V_\chi \boldsymbol{\beta} = \boldsymbol{\chi}(\boldsymbol{\alpha},\boldsymbol{\beta}) \ .$$

Finally, $U_\xi = \mathbf{1}_\kappa^\top \otimes I_{n+1} \in \mathbb{Z}_p^{(n+1)\times\kappa(n+1)}$, $U_\xi \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) \circ \mathbf{1}_\kappa = \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta})$, and

$$U = \left(\begin{array}{c|c|c|c|c|c} 1 & \alpha & \boldsymbol{\phi}(\alpha) & \boldsymbol{\beta} & \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) \\ \hline & U_\phi & & & & \\ \hline & U_\chi & & & & \\ \hline & & & & U_\xi & \end{array}\right) \ , \ V = \left(\begin{array}{c|c|c|c|c|c} 1 & \alpha & \boldsymbol{\phi}(\alpha) & \boldsymbol{\beta} & \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) \\ \hline & & V_\phi & & & \\ \hline & & & V_\chi & & \\ \hline \mathbf{1}_\kappa & & & & & \end{array}\right) \ , \ W = \left(\begin{array}{c|c|c|c|c|c} 1 & \alpha & \boldsymbol{\phi}(\alpha) & \boldsymbol{\beta} & \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) \\ \hline & I_{n-1} & & & & \\ \hline & & & & I_{\kappa(n+1)} & \\ \hline & & & & & I_\kappa \end{array}\right) \ .$$

Thus, $\mu = 1+1+(n-1)+\kappa(n+1)+\kappa(n+1)+\kappa = (2\kappa+1)n+3\kappa+n+1$ and $\nu = (n-1)+\kappa(n+1)+\kappa = (\kappa+1)n+2\kappa-1$.

Since $U_\phi$ consists of one column $u_\phi$, there is only one polynomial $u_\phi(X) = u_s(X)$. Thus $\mathsf{A}_s(X,Y) = r_a + \alpha u_\phi(X)Y$ and its computation has very low costs. $\mathsf{A}_p(X,Y) = u_1(X) + \sum_{j=n+2}^{\mu} \mathsf{a}_j u_j(X)$, $B_s(X,Y) = r_b + \sum_{j=2}^{n+1} \mathsf{a}_j v_j(X)$, $B_p(X,Y) = v_1(X) + \sum_{j=n+2}^{\mu} \mathsf{a}_j v_j(X)$. Small $\mathsf{A}_s(X,Y)$ makes smaller computational costs of the commitment and shortens the CRS length (in comparison with the general SFC). The CRS length is then shortened by $\nu$ and the commitment takes less (by $\nu$) exponentiations in $\mathbb{G}_1$. Otherwise, the length of the commitment is $\kappa+1$ elements of $\mathbb{G}_1$ and 1 element of $\mathbb{G}_2$. The length of opening is 1 element of $\mathbb{G}_1$.

We are not aware of any previous works on the construction of evaluation-point commitment schemes.

**Succinct Multivariate Polynomial Commitment [PST13,BGH19].** Assume $\boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) = \alpha(\boldsymbol{\beta})$, where $\alpha(\boldsymbol{X}) \in \mathbb{Z}_p[\boldsymbol{X}]$ is a secret $c$-variate polynomial of total degree $d$ and $\boldsymbol{\beta} \in \mathbb{Z}_p^c$ is an evaluation point. Thus, $\alpha(\boldsymbol{\beta}) = \sum_j \alpha_j \prod_{k=1}^{c} \beta_k^{j_k}$, where the sum is taken over each $j = (j_1,\ldots,j_c)$ with $\sum j_c \le d$.

Here, $\boldsymbol{\alpha} = (\alpha_j)_{\sum j_k \le d} \in \mathbb{Z}_p^{\mu_\alpha}$ for $\mu_\alpha = \binom{d+c}{c}$ (the number of monomials of a $c$-variate polynomial of degree $\le d$), $\boldsymbol{\beta} \in \mathbb{Z}_p^c$ (thus, $\mu_\beta = c$), $\mathcal{C}_\phi$ is an empty circuit, $\mathcal{C}_\psi$ computes $\psi_j(\boldsymbol{\beta}) = \prod_{k=1}^{c} \beta_k^{j_k}$ for all $j$ with $1 < \sum j_k \le d$ (thus $\mu_\psi = \mu_\alpha - 1 - c = \binom{d+c}{c} - 1 - c$), $\mathcal{C}_\chi$ computes $\chi_j(\alpha,\boldsymbol{\beta}) = \alpha_j \prod_{k=1}^{c} \beta_k^{j_k}$ (thus $\mu_\chi = \mu_\alpha = \binom{d+c}{c}$), and $\mathcal{C}_\xi$ computes $\sum_j \alpha_j \prod_{k=1}^{c} \beta_k^{j_k}$. Thus,

$$U = \left(\begin{array}{c|c|c|c|c|c|c} 1 & \alpha & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) \\ \hline U_\psi & & U_\psi & & & \\ \hline U_\chi & & & & & \\ \hline U_\xi & & & & U_\xi & \end{array}\right) \ , \ V = \left(\begin{array}{c|c|c|c|c|c|c} 1 & \alpha & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) \\ \hline V_\psi & & V_\psi & & & \\ \hline V_\chi & & V_\chi & & & \\ \hline 1 & & & & & \end{array}\right) \ , \ W = \left(\begin{array}{c|c|c|c|c|c|c} 1 & \alpha & \boldsymbol{\beta} & \boldsymbol{\psi}(\boldsymbol{\beta}) & \boldsymbol{\chi}(\alpha,\boldsymbol{\beta}) & \boldsymbol{\mathcal{F}}(\alpha,\boldsymbol{\beta}) \\ \hline & & & I_{\mu_\psi} & & \\ \hline & & & & I_{\mu_\chi} & \\ \hline & & & & & 1 \end{array}\right) \ .$$

Thus, $\nu = \mu_\psi + \mu_\chi + 1 = 2\binom{d+c}{c} - c$ and $\mu = 1 + \mu_\alpha + \mu_\beta + \mu_\psi + \mu_\chi + 1 = 3\binom{d+c}{c} + 1$.

This scheme is not very efficient if $c$ is large, with the matrix dimensions having size $\Theta((d+c)!/d!/c!) = \Theta(d^c)$ if $d$ is large and $c$ is small. However, if hiding the polynomials $\alpha(\boldsymbol{X})$ were not a part of the task, then one could optimize the matrices depending on the polynomials.

**Succinct Multilinear Polynomial Commitment [Set20].** A multilinear polynomial is a multivariate polynomial that is linear in each of its variables. Assuming the number of $\boldsymbol{\alpha}$-monomials with $\boldsymbol{\beta}$ being the coefficients is $\mathsf{poly}(\lambda)$ (or, the other way around), we can clearly use our techniques. Note that we do not require the total number of monomials to be $\mathsf{poly}(\lambda)$. Moreover, here we do not use in any way the fact that the polynomial is multilinear: we can equally well handle the non-multilinear case assuming that the number of (say) $\boldsymbol{\alpha}$-monomials is $\mathsf{poly}(\lambda)$.

# B Security of Span-Uber-Assumption in the AGM

We will next prove that if $f_i \notin \mathrm{span}(\mathcal{R})$ then $(\mathcal{R}, \mathcal{S}, \{f_i\})$-computational span-uber-assumption holds in the AGM [FKL18] under a PDL assumption [Lip12]. As with other AGM proofs, this can be seen as a heuristic support to the computational span-uber-assumption when $f_i \notin \mathrm{span}(\mathcal{R})$ (as in our case). While a similar result can be possibly derived in the generic group model by following [BBG05], the generic model is known to have problems [Fis00,Den02] not shared with the AGM and thus using the AGM can be seen as a stronger validation.

Let $d_1(\nu), d_2(\nu) \in \mathit{poly}(\lambda)$. Then, $\mathsf{Pgen}$ is $(d_1(\nu), d_2(\nu))$-*PDL (Power Discrete Logarithm, [Lip12] secure* if for any non-uniform PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{pdl}}_{d_1,d_2,\mathsf{Pgen},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, where

$$\mathsf{Adv}^{\mathrm{pdl}}_{d_1,d_2,\mathsf{Pgen},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda, \nu), x \leftarrow\!\$\, \mathbb{Z}_p^* : \mathcal{A}\left(\mathsf{p}; [(x^i)_{i=0}^{d_1(\nu)}]_1, [(x^i)_{i=0}^{d_2(\nu)}]_2\right) = x\right] \ .$$

The $q$-PDL assumption in $\mathbb{G}_1$ (resp., $\mathbb{G}_2$) is equal to the $(q,0)$-PDL (resp., $(0,q)$-PDL) assumption.

## B.1 Algebraic Group Model: Preliminaries

AGM is a new model [FKL18] used to prove the security of a cryptographic assumption, protocol, or a primitive. Essentially, in the AGM, one assumes that each PPT algorithm $\mathcal{A}$ is algebraic in the following sense. Assume $\mathcal{A}$'s input includes $[\boldsymbol{x}_\iota]_\iota$ and no other elements from the group $\mathbb{G}_\iota$. Moreover, assume $\mathcal{A}$ has an access to an oracle $\mathcal{O}$, such that $\mathcal{O}(\iota)$ samples and outputs a random element $[q_{\iota k}]_\iota$ from $\mathbb{G}_\iota$, $\iota \in \{1,2\}$. The oracle access models the ability of $\mathcal{A}$ to create random group elements without knowing their discrete logarithms. Such an oracle is not always included to the AGM but it is necessary since it is usually trivial for the adversary to sample random group elements without knowing their discrete logarithms, [Bro01,BFS16,ABLZ17,ALSZ21]. We assume that if $\mathcal{A}$ outputs group elements $[\boldsymbol{y}_\iota]_\iota$, then $\mathcal{A}$ knows matrices $\boldsymbol{N}_\iota$, such that $\boldsymbol{y}_\iota = \boldsymbol{N}_\iota(\begin{smallmatrix}\boldsymbol{x}_\iota \\ \boldsymbol{q}_\iota\end{smallmatrix})$.

More precisely, a PPT algorithm $\mathcal{A}$ is *(*$\mathsf{Pgen}$*-)algebraic* if there exists an efficient extractor $\mathsf{Ext}_\mathcal{A}$, such that for any PPT sampleable distribution $\mathcal{D}$,

$$\mathsf{Adv}^{\mathrm{agm}}_{\mathsf{Pgen},\mathcal{D},\mathcal{A},\mathsf{Ext}_\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l}\mathsf{p} \leftarrow\!\$\, \mathsf{Pgen}(1^\lambda); \mathbb{x} = ([\boldsymbol{x}_1]_1, [\boldsymbol{x}_2]_2) \leftarrow\!\$\, \mathcal{D}; r \leftarrow\!\$\, \mathsf{RND}_\lambda(\mathcal{A}); \\ ([\boldsymbol{y}_1]_1, [\boldsymbol{y}_2]_2) \leftarrow\!\$\, \mathcal{A}^\mathcal{O}(\mathbb{x}; r); (\boldsymbol{N}_1, \boldsymbol{N}_2) \leftarrow \mathsf{Ext}_\mathcal{A}(\mathbb{x}; r) : \\ (\boldsymbol{y}_1 \neq \boldsymbol{N}_1(\begin{smallmatrix}\boldsymbol{x}_1 \\ \boldsymbol{q}_1\end{smallmatrix}) \vee \boldsymbol{y}_2 \neq \boldsymbol{N}_2(\begin{smallmatrix}\boldsymbol{x}_2 \\ \boldsymbol{q}_2\end{smallmatrix}))\end{array}\right] = \mathsf{negl}(\lambda) \ .$$

$\mathcal{O}$ is an oracle, that given $\iota \in \{1,2\}$ as an input, samples and returns a random element from $\mathbb{G}_\iota$. $[\boldsymbol{q}_\iota]_\iota$ is the list of all elements output by $\mathcal{O}(\iota)$. The AGM states that $\mathsf{Adv}^{\mathrm{agm}}_{\mathsf{Pgen},\mathcal{D},\mathcal{A},\mathsf{Ext}_\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$ for any PPT-sampleable $\mathcal{D}$ and PPT algebraic $\mathcal{A}$.

## B.2 Our Contribution

**Theorem 5.** *Let* $\mathbf{R} = \mathcal{I}_{\mathsf{qap}} = (\mathbb{Z}_p, \mu_0, \{u_j, v_j, w_j\}_{j=0}^{\mu_0})$ *be a QAP instance. Let* $\mathsf{FC}^{\mathcal{C}}_{\mathsf{sn}}$ *be the SFC scheme in Fig. 2 such that* $\mathsf{ck} = ([\mathcal{R}(\chi, y)]_1, [\mathcal{S}(\chi, y)]_2)$. *Let* $f_i(X, Y) \notin \mathrm{span}(\mathcal{R})$ *have degree* $\mathsf{poly}(\lambda)$. *Let* $d' = \max_{h \in \{f_i\} \cup \mathcal{R}}(\deg h)$ *and* $d^* = \max_{h \in \mathcal{S}}(\deg h)$. *If the* $(d', d^*)$*-PDL assumption holds, then the* $(\mathcal{R}, \mathcal{S}, \{f_i\})$*-computational span-uber-assumption holds in* $\mathbb{G}_1$ *in the AGM.*

*Proof.* Assume $\mathcal{A}(\mathsf{p}, \mathbf{R}, \mathsf{ck}; r)$ outputs $\boldsymbol{\Delta} \neq \mathbf{0}$ and $\sum \Delta_i [f_i(\chi, y)]_1$. (Here, $r$ is the random tape of $\mathcal{A}$.) Denote $\boldsymbol{\Gamma} := \{\mathcal{R}(\chi, y)\}$. Since we work in the AGM, there exists an extractor $\mathsf{Ext}_{\mathcal{A}}$ that on the input $(\mathsf{p}, \mathbf{R}, \mathsf{ck}; r)$, with probability $\mathsf{Adv}^{\mathrm{agm}}_{\mathsf{Pgen},\mathcal{D},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) = 1 - \mathsf{negl}(\lambda)$, returns $\boldsymbol{N}$ and $[\boldsymbol{q}]_1$, such that $\sum_i \Delta_i [f_i(\chi, y)]_1 = \boldsymbol{N} \left[ \begin{smallmatrix} \boldsymbol{\Gamma} \\ \boldsymbol{q} \end{smallmatrix} \right]_1$. We abort if the extractor does not succeed.

Taking into account that elements of $\boldsymbol{\Gamma}$ are known polynomials of $X$ and $Y$, given $\boldsymbol{N}$ and $[\boldsymbol{q}]_1$, we can efficiently compute the coefficients of the polynomial $\mathcal{V}(X, Y) := \sum_i \Delta_i f_i(X, Y) - \boldsymbol{N} \left( \begin{smallmatrix} \boldsymbol{\varrho}(\boldsymbol{X}, \boldsymbol{Y}) \\ \boldsymbol{Q} \end{smallmatrix} \right)$. The verification equation guarantees that $\mathcal{V}(\chi, y) = 0$. As common in the AGM proofs, we have to consider the next two cases.

First, assume that $\mathcal{V}(X, Y) = 0$ as a polynomial, that is, $\sum_i \Delta_i f_i(X, Y) = \boldsymbol{N} \left( \begin{smallmatrix} \boldsymbol{\varrho}(\boldsymbol{X}, \boldsymbol{Y}) \\ \boldsymbol{Q} \end{smallmatrix} \right)$. Since $\sum_i \Delta_i f_i(X, Y)$ does not depend on $Q_i$, it means that $\sum \Delta_i f_i(X, Y)$ is in $\mathrm{span}(\mathcal{R})$, contradiction.

Second, assume that $\mathcal{V}(X, Y) \neq 0$ as a polynomial, but $\mathcal{V}(\chi, y) = 0$. Assume that $\mathcal{A}$ is a span-uber-assumption adversary that succeeds with probability $\varepsilon_{uber} := \mathsf{Adv}^{\mathrm{uber}}_{\mathcal{R},\mathcal{A}}(\lambda)$, where $\varepsilon_{uber} > \varepsilon_{agm} := \mathsf{Adv}^{\mathrm{agm}}_{\mathsf{Pgen},\mathcal{D},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$. Note that $\varepsilon_{agm} = \mathsf{negl}(\lambda)$. We construct the following PDL adversary $\mathcal{B}$.

1. The PDL challenger $\mathcal{C}$ samples $\chi \leftarrow\!\!\$\, \mathbb{Z}_p^*$ and gives $\mathbb{x} = ([1, \chi, \ldots, \chi^{d'}]_1, [1, \chi, \ldots, \chi^{d^*}]_2)$ as an input to $\mathcal{B}$.
2. $\mathcal{B}$ samples $y \leftarrow\!\!\$\, \mathbb{Z}_p^*$, and uses it together with $(\mathsf{p}, \mathbf{R})$ and $\mathbb{x}$ to create a correctly distributed $\mathsf{ck}$.
3. $\mathcal{B}$ plays the challenger in the span-uber-assumption game with $\mathcal{A}$: after sending $\mathsf{ck}$ and $r \leftarrow\!\!\$\, \mathsf{RND}_\lambda(\mathcal{A})$ to $\mathcal{A}$, $\mathcal{B}$ obtains $\boldsymbol{\Delta}$ and $\sum \Delta_i [f_i]_1$ from $\mathcal{A}$.
   When $\mathcal{A}$ asks for the $j$th $\mathcal{O}$ query in $\mathbb{G}_\iota$, $\mathcal{B}$ simulates it by sampling and storing random $s_{\iota j}, t_{\iota j} \leftarrow\!\!\$\, \mathbb{Z}_p$, and returning $[q_{\iota j}]_\iota \leftarrow [s_{\iota j} \chi + t_{\iota j}]_\iota$ to $\mathcal{A}$.
4. $\mathcal{B}$ runs the extractor $\mathsf{Ext}_{\mathcal{A}}$, that is guaranteed by the AGM to succeed with probability $1 - \mathsf{negl}(\lambda)$, to obtain matrix $\boldsymbol{N}$ and $[\boldsymbol{q}]_\iota$ (the latter is empty in the non-hashing case).
5. From $\boldsymbol{N}$, she computes the coefficients of $\mathcal{V}(X, Y)$.

Now, $\sum_i \Delta_i [f_i]_1 = \sum_i \Delta_i [f_i(\chi, y)]_1$. Since the verifier accepts, $\mathcal{V}(\chi, y) = 0$; however, $\mathcal{V}(X, Y) \neq 0$ as a polynomial. (Since $q_{\iota j}$ are affine functions of $\chi$, they are not separate indeterminates. Due to their inclusion, however, $\mathcal{V}$ changes slightly to incorporate additional terms. Since this is standard in the AGM, [FKL18], we will not add additional discussion.) For the fixed value of $y \in \mathbb{Z}_p^*$, let $\mathcal{V}^*(X) := \mathcal{V}(X, y)$. Finally, $\mathcal{B}$ does the following:

6. Use an efficient polynomial factorization algorithm to obtain up to $d'$ roots $\chi_i$ of $\mathcal{V}^*(X)$.
7. Return the root $\chi_i$ that satisfies $[\chi_i^j y^k]_1 = [\chi^j y^k]_1$ for some fixed monomial $[\chi^j y^k]_1$ given in the commitment key, where $j \neq 0$. (E.g., $j = 1$ and $k = 1$.)

Clearly, $\mathcal{B}$ has broken the $(d', d^*)$-PDL assumption with probability $\varepsilon_{pdl} = \mathsf{Adv}^{\mathrm{pdl}}_{d',d^*,\mathsf{Pgen},\mathcal{B}}(\lambda) \geq \varepsilon_{uber} - \varepsilon_{agm}$. Thus,

$$\varepsilon_{uber} \leq \mathsf{Adv}^{\mathrm{pdl}}_{d',d^*,\mathsf{Pgen},\mathcal{B}}(\lambda) + \mathsf{Adv}^{\mathrm{agm}}_{\mathsf{Pgen},\mathcal{D},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) \ .$$

Moreover, $\mathcal{B}$'s running time is dominated by the running time of $\mathcal{A}$ and the time to perform polynomial factorization. □

Since in the case $\kappa = 1$, span-uber-assumption is the same as the $(\mathcal{R}, \mathcal{S}, \mathcal{T}, f_1)$ uber-assumption, from Theorem 5, we get the following corollary that is potentially of independent interest.

**Corollary 2.** *Let* $\mathsf{ck} = ([\mathcal{R}(\chi, y)]_1, [\mathcal{S}(\chi, y)]_2)$ *be two sets of bivariate polynomials* $\mathcal{R}$ *and* $\mathcal{S}$*. Let* $\mathcal{D}$ *be the distribution of* $\mathsf{ck}$ *for* $\chi, y \leftarrow\!\!\$\, \mathbb{Z}_p^*$*. Let* $f(X, Y) \notin \mathrm{span}(\mathcal{R})$ *have degree* $\mathsf{poly}(\lambda)$*. Let* $d' = \max_{h \in \{f\} \cup \mathcal{R}}(\deg h)$ *and* $d^* = \max_{h \in \mathcal{S}}(\deg h)$*. If the* $(d', d^*)$*-PDL assumption holds, then the* $(\mathcal{R}, \mathcal{S}, \{f_i\})$*-computational uber-assumption holds in* $\mathbb{G}_1$ *in the AGM.*

We note that the security of the computational uber-assumption in the AGM was independently also proven in [BFL20, Theorem 3.5].

## C SNARK Zero-Knowledge: Formal Security Definitions

Let $\mathcal{R}$ be a relation generator, such that $\mathcal{R}(1^\lambda)$ returns a polynomial-time decidable binary relation $\mathbf{R} = \{(\mathbb{x}, \mathbb{w})\}$. Here, $\mathbb{x}$ is the statement, and $\mathbb{w}$ is the witness. We assume that $\lambda$ is explicitly deductible from the description of $\mathbf{R}$. The relation generator also outputs auxiliary information $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda, n)$ for a well-defined $n$. Because of this, we give $\mathsf{p}$ as an input to all (including honest or adversarial) parties.

As in [Gro16], we define all security notions against a non-uniform adversary. However, since our security reductions are uniform, it is a simple matter to consider only uniform adversaries.

**Definition 7 (Perfect Completeness).** *A non-interactive argument $\Psi$ is* perfectly complete for $\mathcal{R}$*, if for all $\lambda$, all $(\mathsf{p}, \mathbf{R}) \in \mathrm{range}(\mathcal{R}(1^\lambda))$, and $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$,*

$$\Pr\left[(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{K}_{\mathsf{crs}}(\mathsf{p}, \mathbf{R}) : \mathsf{V}(\mathsf{p}, \mathbf{R}, \mathsf{crs}_{\mathsf{V}}, \mathbb{x}, \mathsf{P}(\mathsf{p}, \mathbf{R}, \mathsf{crs}_{\mathsf{P}}, \mathbb{x}, \mathbb{w})) = 1\right] = 1 \ .$$

**Definition 8 (Computational Knowledge-Soundness).** *$\Psi$ is computationally (adaptively)* knowledge-sound for $\mathcal{R}$*, if for every non-uniform PPT $\mathcal{A}$, there exists a non-uniform PPT extractor $\mathsf{Ext}_{\mathcal{A}}$, such that*

$$\mathsf{Adv}^{\mathrm{snd}}_{\mathcal{R},\mathcal{A}}(\lambda) := \Pr\begin{bmatrix}(\mathsf{p}, \mathbf{R}) \leftarrow \mathcal{R}(1^\lambda); (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{K}_{\mathsf{crs}}(\mathsf{p}, \mathbf{R}); r \leftarrow\$ \mathsf{RND}_\lambda(\mathcal{A}); \\ (\mathbb{x}, \pi) \leftarrow \mathcal{A}(\mathsf{p}, \mathbf{R}, \mathsf{crs}; r); \mathbb{w} \leftarrow \mathsf{Ext}_{\mathcal{A}}(\mathsf{p}, \mathbf{R}, \mathsf{crs}; r) : \\ (\mathbb{x}, \mathbb{w}) \notin \mathbf{R} \wedge \mathsf{V}(\mathsf{p}, \mathbf{R}, \mathsf{crs}_{\mathsf{V}}, \mathbb{x}, \pi) = 1\end{bmatrix} \approx_\lambda 0 \ .$$

A knowledge-sound argument system is called an *argument of knowledge*.

**Definition 9 (Statistically Unbounded ZK).** *$\Psi$ is* statistically unbounded Sub-ZK for $\mathcal{R}$*, if for all $\lambda$, all $(\mathsf{p}, \mathbf{R}) \in \mathrm{range}(\mathcal{R}(1^\lambda))$, and all computationally unbounded $\mathcal{A}$, $\varepsilon_0^{unb} \approx_\lambda \varepsilon_1^{unb}$, where*

$$\varepsilon_b^{unb} = \Pr[(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{K}_{\mathsf{crs}}(\mathsf{p}, \mathbf{R}) : \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathsf{p}, \mathbf{R}, \mathsf{crs}) = 1] \ .$$

*Here, the oracle $\mathsf{O}_0(\mathbb{x}, \mathbb{w})$ returns $\perp$ (reject) if $(\mathbb{x}, \mathbb{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{P}(\mathsf{p}, \mathbf{R}, \mathsf{crs}_{\mathsf{P}}, \mathbb{x}, \mathbb{w})$. Similarly, $\mathsf{O}_1(\mathbb{x}, \mathbb{w})$ returns $\perp$ (reject) if $(\mathbb{x}, \mathbb{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{Sim}(\mathsf{p}, \mathbf{R}, \mathsf{crs}, \mathsf{td}, \mathbb{x})$. $\Psi$ is* perfectly unbounded ZK for $\mathcal{R}$ *if one requires that $\varepsilon_0^{unb} = \varepsilon_1^{unb}$.*

## D Subroup Hiding ⇒ Span-Uber-Assumption

In Lemma 2, we proved that, for concrete parameters, span-uber-assumption is implied by a computational uber-assumption in $\mathbb{G}_T$. In this section, we prove that in general, the span-uber-assumption in $\mathbb{G}_1$ holds under subgroup hiding in composite-order bilinear groups.

**Bilinear Composite-Order Groups: Notation.** Let $c$ be the number of indeterminates, $\boldsymbol{X} = (X_1, \ldots, X_c)$. Let $\mathsf{p} = (N, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, \mathsf{P}_1, \mathsf{P}_2)$ be a composite-order bilinear group and let $\mathsf{aux}_{\mathsf{p}} := \{\mathsf{P}_1^1, \mathsf{P}_1^2, \mathsf{P}_2^1, \mathsf{P}_2^2\}$ be auxiliary information about subgroup generators. Here, $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are additive groups of composite order $N = p_1 p_2$ ($p_1 \neq p_2$) and $\mathsf{P}_\iota^1 \in \mathbb{G}_{\iota,p_1}$, $\mathsf{P}_\iota^2 \in \mathbb{G}_{\iota,p_2}$ are randomly sampled subgroup generators, where $\mathbb{G}_{\iota,p_j}$ is the subgroup of $\mathbb{G}_\iota$ of order $p_j$ and $\mathsf{P}_\iota \in \mathbb{G}_\iota = \mathbb{G}_{\iota,p_1} \oplus \mathbb{G}_{\iota,p_2}$. We use $(\mathsf{p}, \mathsf{aux}_{\mathsf{p}}) \leftarrow \mathsf{Pgen}(1^\lambda, 2)$ to denote the algorithm that generates composite-order bilinear groups with two cyclic subgroups and their generates. As explained later, not the whole $\mathsf{aux}_{\mathsf{p}}$ is made public to the adversary $\mathcal{A}$; we specify in each assumption separately which subset $\mu \subseteq \mathsf{aux}_{\mathsf{p}}$ is given to $\mathcal{A}$. Differently from the prime-order case, we denote scalar multiplication of an elliptic curve point $\mathsf{P}$ by an integer $a$ as $[a]\mathsf{P}$ and the pairing of two points $\mathsf{P}_1$ and $\mathsf{P}_2$ by $\hat{e}(\mathsf{P}_1, \mathsf{P}_2)$.

The following definition of the computational uber-assumption in composite-order group source group $\mathbb{G}_\iota$ is from [CM14] (Assumption 4.1). It is essentially the same as Definition 10 but for composite-order groups (and somewhat more general: Definition 10 only considered $c = 2$ and $\mathcal{T} = \emptyset$). Chase *et al.* [CMM16] (Assumption 3.1) defined the computational uber-assumption in composite-order group $\mathbb{G}_T$; however, it is not needed in the current paper.

**Definition 10 (Uber-assumption [CM14] in composite-order groups).** *Let* $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda, 2)$. *Let* $\mathcal{R}$, $\mathcal{S}$, *and* $\mathcal{T}$ *be three tuples of* $c$*-variate polynomials over* $\mathbb{Z}_N[\boldsymbol{X}]$. *Let* $f$ *be a* $c$*-variate polynomial over* $\mathbb{Z}_N[\boldsymbol{X}]$. *The* $(\mathcal{R}, \mathcal{S}, \mathcal{T}, f)$-computational uber-assumption for $\mathsf{Pgen}$ in a composite-order group $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$, *states that for any PPT adversary* $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{uber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},f,\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, *where* $\mathsf{Adv}^{\mathrm{uber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},f,\mathcal{A}}(\lambda) :=$

$$\Pr\begin{bmatrix}(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow\!\$\, \mathsf{Pgen}(1^\lambda, 2); \mathsf{P}_T \leftarrow \hat{e}(\mathsf{P}_1, \mathsf{P}_2); \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;\\ \mathsf{ck} \leftarrow (\{[\varrho(\boldsymbol{x})]\mathsf{P}_1\}_{r\in\mathcal{R}}, \{[\sigma(\boldsymbol{x})]\mathsf{P}_2\}_{\sigma\in\mathcal{S}}, \{[\tau(\boldsymbol{x})]\mathsf{P}_T\}_{\tau\in\mathcal{T}});\\ \mathcal{A}(\mathsf{p}, \mathsf{ck}) = [f(\boldsymbol{x})]\mathsf{P}_\iota\end{bmatrix}.$$

### D.1 Preliminaries on Dèjá Q and Parameter Hiding

We will next give a comprehensive overview of the techniques used in the Dèj'a Q framework. A knowledgeable reader might want to skip to the next subsection.

Two structural properties of composite-order bilinear groups are exploited in the Déjá Q framework, subgroup hiding and parameter hiding. Subgroup hiding is a computational assumption that requires that, if $\mathbb{G}_1$ (respectively $\mathbb{G}_2$) decomposes into two subgroups, then no PPT adversary has a non-negligible chance of distinguishing a random element of one of the subgroups from a random element of the full group.

**Definition 11 (Subgroup Hiding in $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$, with respect to $\mu$ [BGN05,CM14,CMM16]).** *For* $\mu \subseteq \mathsf{aux}_\mathsf{p}$, *define* $\mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_\iota,\mu,\mathcal{A}}(\lambda) = 2\varepsilon^{sh}_{\mathbb{G}_\iota,\mu,\mathcal{A}}(\lambda) - 1$, *where*

$$\varepsilon^{sh}_{\mathbb{G}_\iota,\mu,\mathcal{A}}(\lambda) := \Pr\begin{bmatrix}(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow\!\$\, \mathsf{Pgen}(1^\kappa, 2); b \leftarrow\!\$\, \{0, 1\}; \mathsf{W}_0 \leftarrow\!\$\, \mathbb{G}_\iota;\\ \mathsf{W}_1 \leftarrow\!\$\, \mathbb{G}_{\iota,p_1} : \mathcal{A}(\mathsf{p}, \mu, \mathsf{W}_b) = b\end{bmatrix}.$$

*The* subgroup hiding in $\mathbb{G}_\iota$ *holds* with respect to the auxiliary information $\mu$ *if for all PPT adversaries* $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_\iota,\mu,\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

The auxiliary information $\mu$ captures subgroup generators (i.e., generators of $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$ or its subgroups) that can be given to the adversary. Recall that a cancelling pairing [Fre10] is a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, such that $\hat{e}(\mathbb{G}_{1,p_1}, \mathbb{G}_{2,p_2}) = \hat{e}(\mathbb{G}_{1,p_2}, \mathbb{G}_{2,p_1}) = 1$. By using a cancelling pairing, an adversary $\mathcal{A}$, who has access to $\mathsf{P}_2^2$, can distinguish between $\mathsf{T} = [r]\mathsf{P}_1^1$ and $\mathsf{T} = [r]\mathsf{P}_1$, for $r \leftarrow\!\$\, \mathbb{Z}_N$, by checking whether $\hat{e}(\mathsf{T}, \mathsf{P}_2^2) = 1$. Thus, if $\mathsf{P}_2^2 \in \mu$, then distinguishing between random elements of $\mathbb{G}_{1,p_1}$ and $\mathbb{G}_1$ is easy. Analogously, if $\mathsf{P}_2^1 \in \mu$ then distinguishing between $\mathbb{G}_{1,p_2}$ and $\mathbb{G}_1$ is easy.

*Parameter hiding* is a statistical property of a group that allows certain distributions across subgroups to be independent. Hence, it allows to information-theoretically hide from the adversary some useful information even after the public parameters are revealed. More precisely, parameter hiding in $\mathbb{G}_1$ holds with respect to a family of functions $\mathcal{F}$ if for all $\mathsf{P}_1^1 \in \mathbb{G}_{1,p_1}$ and $\mathsf{P}_1^2 \in \mathbb{G}_{1,p_2}$, the distributions $\{[f(\boldsymbol{x})]\mathsf{P}_1^1 + [f(\boldsymbol{x})]\mathsf{P}_1^2\}_{f\in\mathcal{F}}$ and $\{[f(\boldsymbol{x})]\mathsf{P}_1^1 + [f(\boldsymbol{x}')]\mathsf{P}_1^2\}_{f\in\mathcal{F}}$ are identical, where $\boldsymbol{x}, \boldsymbol{x}' \leftarrow\!\$\, \mathbb{Z}_N^n$ [CM14].

In composite-order groups, parameter hiding takes several different forms depending on the aim of using. First of all, in the original Déjá Q framework [CM14], the proof strategy for the computational uber-assumpion in the source group (as opposed to the proof for the correspondent assumption in the target group) requires only one type of subgroup hiding, namely subgroup hiding between $\mathbb{G}_{1,p_1}$ and $\mathbb{G}_1$. This means that the subgroup generators $\mathsf{P}_2^1$ and $\mathsf{P}_2^2$ on the other side of pairing can be revealed.

In addition, it is desirable to have parameter hiding even if some meaningful information about $\boldsymbol{x}$ is leaked in the group $\mathbb{G}_2$ as well as in $\mathbb{G}_1$. We have to take this into account, *extending* the definition of parameter hiding to allow the case where some additional information $\mathsf{aux}$ about $\boldsymbol{x}$ is revealed.

**Definition 12 (Extended parameter hiding, [CM14,CMM16]).** *For* $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow\!\$\, \mathsf{Pgen}(1^\lambda, 2)$, extended parameter hiding holds in $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$ with respect to a family of functions $\mathcal{F}$, and auxiliary information $\mathsf{aux}$, *if for all* $\mathsf{P}_\iota^1 \in \mathbb{G}_{\iota,p_1}$ *and* $\mathsf{P}_\iota^2 \in \mathbb{G}_{\iota,p_2}$, *the distributions*

$$\{[f(\boldsymbol{x})]\mathsf{P}_\iota^1 + [f(\boldsymbol{x})]\mathsf{P}_\iota^2, a(\boldsymbol{x})\}_{f\in\mathcal{F},a\in\mathsf{aux}} \text{ and } \{[f(\boldsymbol{x})]\mathsf{P}_\iota^1 + [f(\boldsymbol{x}')]\mathsf{P}_\iota^2, a(\boldsymbol{x})\}_{f\in\mathcal{F},a\in\mathsf{aux}}$$

*are identical, for* $\boldsymbol{x}, \boldsymbol{x}' \leftarrow\!\$\, \mathbb{Z}_N^n$.

For our purposes, we need extended parameter hiding to hold for $\mathsf{aux} = \{[\sigma(\cdot)]\mathsf{P}_2^1\}_{\sigma\in\mathcal{R}}$.

For the sake of completeness, next, we replicate the proof that extended parameter hiding holds in composite-order subgroups with $n > 2$ subgroups with respect to all polynomial functions.

**Lemma 3 ([CMM16], Lemma 2.3).** *For all $m, n \in \mathbb{N}$, $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \in \mathsf{Pgen}(1^\lambda, n)$, where $N = p_1 \cdot \ldots \cdot p_n$, $(i_1, i_2)$ such that $1 \le i_1, i_2 \le n$, and for the class of all polynomials $f(\cdot)$ over $\mathbb{Z}_N$, if $\gcd(p_{i_1}, p_{i_2}) = 1$ and if for all $a \in \mathsf{aux}$, $a(\cdot) \in A$ such that $\gcd(|A|, p_{i_2}) = 1$, then the distributions $\{[f(\boldsymbol{x})]\mathsf{P}_{1,i_1}^1 + [f(\boldsymbol{x})]\mathsf{P}_{1,i_2}^1, a(\boldsymbol{x})\}_{f\in\mathcal{F}, a\in\mathsf{aux}}$ and $\{[f(\boldsymbol{x})]\mathsf{P}_{1,i_1}^1 + [f(\boldsymbol{x}')]\mathsf{P}_{1,i_2}^1, a(\boldsymbol{x})\}_{f\in\mathcal{F}, a\in\mathsf{aux}}$ are identical for $\boldsymbol{x}, \boldsymbol{x}' \leftarrow\$ \mathbb{Z}_N^m$. Here, $\mathsf{P}_{1,i}^1 \in \mathbb{G}_{1,p_i}$.*

*Proof.* For any polynomial $f$, one can compute $[f(\boldsymbol{x})]\mathsf{P}_{i_1}^1$ knowing just the value of $x_j \pmod{p_{i_1}}$, for all $j$, $1 \le j \le m$, and $[f(\boldsymbol{x})]\mathsf{P}_{i_2}^1$ knowing just the value of $x_j \pmod{p_{i_2}}$, for all $j$, $1 \le j \le m$. If $\gcd(p_{i_1}, p_{i_2}) = 1$ and the functions in $\mathsf{aux}$ reveal no information about $x_j \pmod{p_{i_2}}$, then by the Chinese Remainder theorem the values of $x_j \pmod{p_{i_2}}$ are independent of all the other values, so this is identical to using an independent $x_j'$ for the $\mathsf{P}_{i_2}^1$ values. □

For a more interactive setting, in which some of the inputs might be provided by an adversary, we can no longer model all inputs to the function $f \in \mathcal{F}$ as uniformly random. For example, the $q$-SDH assumption asks the adversary to compute a pair $(c, [1/(x+c)]\mathsf{P}_\iota)$ with $c \in \mathbb{Z}_p^*$ being chosen by the adversary. Since for certain classes of functions, an unbounded adversary might be able to easily distinguish the two distributions if allowed to query on all possible adversarial inputs, a new definition is required. Informally, this property (of adaptive parameter hiding in $\mathbb{G}_1$) ensures that any unbounded adversary who makes only polynomial number of queries cannot statistically distinguish between the distributions $\{[f(x)]\mathsf{P}_1^1 + [f(x)]\mathsf{P}_1^2\}$ and $\{[f(x)]\mathsf{P}_1^1 + [f'(x)]\mathsf{P}_1^2\}$, for any $f, f'$ from a family of functions $\mathcal{F}$ .

Let $\mathcal{A}$ be a computationally unbounded adversary that is allowed to see only polynomially many evaluations on inputs that it can choose adaptively. Next, in contrary to the extended parameter hiding, we consider not all functions $f$ and uniformly random inputs, but randomly sampled functions $f$ and $f'$ (with potentially different but overlapping domains $\mathsf{dom}(f)$ and $\mathsf{dom}(f')$) applied to adversarially chosen inputs $\boldsymbol{x}$, and require the values $[f(\boldsymbol{x})]\mathsf{P}_1^1 + [f(\boldsymbol{x})]\mathsf{P}_2^2$ and $[f(\boldsymbol{x})]\mathsf{P}_1^1 + [f'(\boldsymbol{x})]\mathsf{P}_2^2$ to be statistically indistinguishable.

**Definition 13 (Adaptive parameter hiding, [CM14]).** *For a group $\mathbb{G}_\iota = \mathbb{G}_{\iota,p_1} \oplus \mathbb{G}_{\iota,p_2}$, $\iota \in \{1, 2\}$, and functions $f$, $f'$ in a family $\mathcal{F}$, let $\mathcal{O}(\cdot)$ return $[f(\boldsymbol{x})]\mathsf{P}_\iota^1 + [f(\boldsymbol{x})]\mathsf{P}_\iota^2$ if the input is in $\mathsf{dom}(f)$ and 1 otherwise, and let $\mathcal{O}'(\cdot)$ return $[f(\boldsymbol{x})]\mathsf{P}_\iota^1 + [f'(\boldsymbol{x})]\mathsf{P}_\iota^2$ if the input is in $\mathsf{dom}(f) \cap \mathsf{dom}(f')$ and 1 otherwise.*

*Then* adaptive parameter hiding *holds* in $\mathbb{G}_\iota$ with respect to $\mathcal{F}$ *if for all $\lambda \in \mathbb{N}$, $\mathsf{P}_\iota^1 \in \mathbb{G}_{\iota,p_1}$, and $\mathsf{P}_\iota^2 \in \mathbb{G}_{\iota,p_2}$, the oracles $\mathcal{O}$ and $\mathcal{O}'$ are statistically indistinguishable if they are queried polynomially many times; i.e., for any (potentially unbounded) distinguisher $\mathcal{D}$ makes $\mathrm{poly}(\lambda)$ queries,*

$$\Pr[f \leftarrow\$ \mathcal{F} : \mathcal{D}^{\mathcal{O}(\cdot)} = 1] - \Pr[f, f' \leftarrow\$ \mathcal{F} : \mathcal{D}^{\mathcal{O}'(\cdot)} = 1] = \mathsf{negl}(\lambda) \ .$$

In the case of the span-uber-assumption, since the adversary chooses coefficients $\boldsymbol{\Delta}$ of the challenge term, we cannot anymore model all inputs to the function as uniformly random. We have the family of functions $\{\sum_{i=1}^{\kappa} \Delta_i[f_i(\boldsymbol{X})]\}_{\boldsymbol{\Delta}=(\Delta_1,\ldots,\Delta_\kappa)\in\mathbb{Z}_p^\kappa}$, where $\boldsymbol{\Delta}$ is chosen by adversary. Thus, we need to use *adaptive parameter hiding* property. Since in the new span-uber assumption $\mathcal{S} \neq \{1\}$, we need as well the *extended parameter hiding* property. Altogether, we make use of the extended adaptive parameter hiding from [CK18].

**Definition 14 (Extended adaptive parameter hiding, [CK18]).** *For a group $\mathbb{G}_\iota = \mathbb{G}_{\iota,p_1} \oplus \mathbb{G}_{\iota,p_2}$, $\iota \in \{1, 2\}$ and functions $f$, $f'$ chosen at random from a family of functions $\mathcal{F}$, let $\mathsf{aux}$ denote the auxiliary information, $\mathcal{O}(\cdot)$ be the oracle that returns $[f(\cdot)]\mathsf{P}_1^1 + [f(\cdot)]\mathsf{P}_1^2$ if the input is in the domain $\mathsf{dom}(f)$ and 1 otherwise. Similarly, let $\mathcal{O}'(\cdot)$ be the oracle that returns $[f(\cdot)]\mathsf{P}_1^1 + [f'(\cdot)]\mathsf{P}_1^2$ if the input is in the domain $\mathsf{dom}(f) \cap \mathsf{dom}(f)$ and 1 otherwise.*

*The* extended adaptive parameter hiding *holds* in $\mathbb{G}_\iota$ with respect to $\mathcal{F}$ and $\mathsf{aux}$ *if for all $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow\$ \mathsf{Pgen}(1^\lambda)$, with $\mu = \{\mathsf{P}_1^1, \mathsf{P}_1^2\}$, the oracles $\mathcal{O}$ and $\mathcal{O}'$ are statistically indistinguishable, if they are given with auxiliary information $\mathsf{aux}$ and queried polynomially many times. That is, for any unbounded $\mathcal{A}$ that makes $\mathsf{poly}(\lambda)$ queries,*

$$\Pr[f \leftarrow\$ \mathcal{F} : \mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{p}, \mathsf{aux}) = 1] - \Pr[f, f' \leftarrow\$ \mathcal{F} : \mathcal{A}^{\mathcal{O}'(\cdot)}(\mathsf{p}, \mathsf{aux}) = 1] = \mathsf{negl}(\lambda) \ .$$

### D.2 Security of Uber-Span-Assumption

Next we prove the security of the uber-span-assumption under the subgroup hiding.

First, we modify Definition 10 to cover span-uber-assumptions in composite-order groups.

**Definition 15 (Span-uber-assumption in composite order groups).** *Let* $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda, 2)$. *Let* $\mathcal{R}$, $\mathcal{S}$, *and* $\mathcal{T}$ *be three tuples of c-variate polynomials over* $\mathbb{Z}_N[\boldsymbol{X}]$. *Let* $f_i$ *be c-variate polynomials over* $\mathbb{Z}_N[\boldsymbol{X}]$ *for* $i \in [1, \kappa]$. *The* $(\mathcal{R}, \mathcal{S}, \mathcal{T}, \{f_i\}_{i=1}^\kappa)$-computational span-uber-assumption for $\mathsf{Pgen}$ in a composite-order group $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$, *states that for any PPT adversary* $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{spanuber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, *where*

$$\mathsf{Adv}^{\mathrm{spanuber}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l}(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow\!\$\, \mathsf{Pgen}(1^\lambda, 2); \mathsf{P}_T \leftarrow \hat{e}(\mathsf{P}_1, \mathsf{P}_2); \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c; \\ \mathsf{ck} \leftarrow (\{[\varrho(\boldsymbol{x})]\mathsf{P}_1\}_{r\in\mathcal{R}}, \{[\sigma(\boldsymbol{x})]\mathsf{P}_2\}_{\sigma\in\mathcal{S}}, \{[\tau(\boldsymbol{x})]\mathsf{P}_T\}_{\tau\in\mathcal{T}}); \\ \mathcal{A}(\mathsf{p}, \mathsf{ck}) = (\boldsymbol{\Delta} \in \mathbb{Z}_N^\kappa, \sum_{i=1}^\kappa \Delta_i[f_i(\boldsymbol{x})]\mathsf{P}_\iota) \wedge \boldsymbol{\Delta} \neq \mathbf{0}_\kappa\end{array}\right] .$$

We denote $r := |\mathcal{R}|$, $s := |\mathcal{S}|$, and $t := |\mathcal{T}|$.

Further, applying the Déjá Q framework of Chase and Meiklejohn [CM14], we show that the $(\mathcal{R}, \mathcal{S}, \{f_i\}_{i=1}^\kappa)$-computational span-uber-assumption is equivalent (under subgroup and parameter hiding) to the following transitioned $q$-assumption.

**Definition 16 (Transitioned span-uber-assumption).** *Let* $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda, 2)$. *Let* $\mathcal{R}$, $\mathcal{S}$, $\mathcal{T}$ *be three tuples of c-variate polynomials over* $\mathbb{Z}_N[\boldsymbol{X}]$. *Let* $f_i$, $i \in [1, \kappa]$ *be c-variate polynomials over* $\mathbb{Z}_N[\boldsymbol{X}]$. *The* transitioned $(\mathcal{R}, \mathcal{S}, \mathcal{T}, \{f_i\}_{i=1}^\kappa)$-computational span-uber-assumption[3] for $\mathsf{Pgen}$ in group $\mathbb{G}_\iota$, $\iota \in \{1, 2\}$, *states that for any PPT adversary,* $\mathsf{Adv}^{\mathrm{spantrans}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$, *where*

$$\mathsf{Adv}^{\mathrm{spantrans}}_{\mathsf{Pgen},\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l}(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow\!\$\, \mathsf{Pgen}(1^\lambda, 2); \mathsf{P}_T^1 \leftarrow \hat{e}(\mathsf{P}_1^1, \mathsf{P}_2^1); \\ r_1, \ldots, r_\ell \leftarrow\!\$\, \mathbb{Z}_N^*; \boldsymbol{x}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell \leftarrow\!\$\, (\mathbb{Z}_N^*)^c; \\ \mathsf{ck} \leftarrow \left(([\varrho(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{j=1}^\ell r_j \varrho(\boldsymbol{x}_j)]\mathsf{P}_1^2)_{\varrho\in\mathcal{R}}, \left([\sigma(\boldsymbol{x})]\mathsf{P}_2^1\right)_{\sigma\in\mathcal{S}}, \mathsf{P}_T^1\right) : \\ \mathcal{A}(\mathsf{p}, \mathsf{ck}) = \left(\boldsymbol{\Delta} \in \mathbb{Z}_{p_1}^\kappa, \sum_{i=1}^\kappa \Delta_i([f_i(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{j=1}^\ell r_j f_i(\boldsymbol{x_j})]\mathsf{P}_1^2)\right) \wedge \\ \boldsymbol{\Delta} \neq \mathbf{0}\end{array}\right] .$$

Here, $\ell = \mathsf{poly}(\lambda)$ is the number of transitions conducted by using the Déjá Q framework; $\ell$ is tied to the number of queries that $\mathcal{A}$ makes to the oracle.

**Theorem 6.** *Let* $(\mathsf{p}, \mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda, 2)$. *If*

1. *the subgroup hiding holds in* $\mathbb{G}_1$ *with respect to* $\mu = \{\mathsf{P}_1^2, \mathsf{P}_2^1\}$ *and in* $\mathbb{G}_2$ *with respect to* $\mu = \{\mathsf{P}_1^1\}$, *and*
2. *the extended adaptive parameter hiding holds in* $\mathbb{G}_1$ *with respect to* $\mathcal{R} \cup \{f_i\}_{i=1}^\kappa$ *and* $\mathsf{aux} = \{[\sigma(\cdot)]\mathsf{P}_2^1\}_{\sigma\in\mathcal{S}}$ *for any* $\mathsf{P}_2^1 \in \mathbb{G}_{2,p_1}$,

*the* $(\mathcal{R}, \mathcal{S}, \{f_i\}_{i=1}^\kappa)$*-computational span-uber-assumption in* $\mathbb{G}_1$ *is implied by the transitioned* $(\mathcal{R}, \mathcal{S}, \{f_i\}_{i=1}^\kappa)$*-computational span-uber-assumption from Definition 16.*

*Proof.* Let $\mathcal{A}$ be a PPT adversary playing the span-uber-assumption game $\mathbf{SUA}^{\mathcal{A}}_{c,\mathcal{R},\mathcal{S},\{f_i\}_{i=1}^\kappa}(\lambda)$ (specified in Fig. 4), and let $\mathsf{Adv}^{\mathrm{final}}_{\mathcal{A}}(\lambda)$ denote its advantage in the final game specified in the statement of the current theorem. We define PPT adversaries $\mathcal{B}_0$, $\mathcal{C}_0$ and a family of PPT adversaries $\mathcal{B}_j$ for all $j$, $1 \le j \le \ell$, such that

$$\begin{aligned}\mathsf{Adv}^{\mathrm{su}}_{c,\mathcal{R},\mathcal{S},\{f_i\}_{i=1}^\kappa,\mathcal{A}}(\lambda) \le & \mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_1,\emptyset,\mathcal{B}_0}(\lambda) + \mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_2,\{\mathsf{P}_1^1\},\mathcal{C}_0}(\lambda) + \\ & \sum_{j=1}^\ell \mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_1,\{\mathsf{P}_1^2,\mathsf{P}_2^1\},\mathcal{B}_j}(\lambda) + \mathsf{Adv}^{\mathrm{final}}_{\mathcal{A}}(\lambda) .\end{aligned} \tag{20}$$

[3] This assumption is based on assumption in Theorem 4.8 of [CM14].

for all $\lambda$, from which the theorem follows. To do this, we construct $\mathcal{B}_0$, $\mathcal{C}_0$ and $\mathcal{B}_j$ for all $j$, $1 \le j \le \ell$, such that

$$\Pr[\mathbf{SUA}^{\mathcal{A}}_{c,\mathcal{R},\mathcal{S},\{f_i\}_{i=1}^{\kappa}}(\lambda)] - \Pr[\mathsf{G}^{\mathcal{A}}_0(\lambda)] \le \mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_1,\emptyset,\mathcal{B}_0}(\lambda) \ , \tag{21}$$

$$\Pr[\mathsf{G}^{\mathcal{A}}_0(\lambda)] - \Pr[\mathsf{G}^{\mathcal{A}}_1(\lambda)] \le \mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_2,\{\mathsf{P}^1_1\},\mathcal{C}_0}(\lambda) \ , \tag{22}$$

$$\Pr[\mathsf{G}^{\mathcal{A}}_j(\lambda)] - \Pr[\mathsf{G}^{\mathcal{A}}_{j,1}(\lambda)] \le \mathsf{Adv}^{\mathrm{sh}}_{\mathbb{G}_1,\{\mathsf{P}^2_1,\mathsf{P}^1_2\},\mathcal{B}_j}(\lambda) \ , \ j \in [1,\ell] \tag{23}$$

$$\Pr[\mathsf{G}^{\mathcal{A}}_{j,1}(\lambda)] - \Pr[\mathsf{G}^{\mathcal{A}}_{j+1}(\lambda)] = 0 \ , \qquad /\!/ \text{ ext. adaptive parameter hiding} \tag{24}$$

$$\Pr[\mathsf{G}^{\mathcal{A}}_{\ell}(\lambda)] \le \mathsf{Adv}^{\mathrm{final}}_{\mathcal{A}}(\lambda) \ , \tag{25}$$

where the various games are depicted in Fig. 4. (We presented the games in a slightly compressed form since it seems to us that it is easier to compare them when they fit to the same page.) In each game, we will highlight the part of game that is changed compared to the previous one.

The game $\mathbf{SUA}^{\mathcal{A}}_{c,\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\}}(\lambda)$ is the original game of the span-uber-assumption. Next, in the game $\mathbf{G}^{\mathcal{A}}_0(\lambda)$, all $\mathbb{G}_1$ elements are shifted to the setting in which they exist only in the subgroup $\mathbb{G}_{1,p_1}$, operating over the original set of variables $\boldsymbol{x}$; this goes unnoticed by the subgroup hiding in $\mathbb{G}_1$ with $\mu = \emptyset$. More precisely, to see that Eq. (21) holds, consider the adversary $\mathcal{B}_0$ in Fig. 5. If $\mathsf{W} \leftarrow\!\!\$\, \mathbb{G}_1$ then the output of $\mathcal{B}_0$ is identical to that in $\mathbf{SUA}^{\mathcal{A}}_{c,\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\}}(\lambda)$. If instead $\mathsf{W} \leftarrow\!\!\$\, \mathbb{G}_{1,p_1}$ then the output of $\mathcal{B}_0$ is identical to that of $\mathbf{G}^{\mathcal{A}}_0(\lambda)$.

In the game $\mathbf{G}^{\mathcal{A}}_1(\lambda)$, all $\mathbb{G}_2$ elements are shifted to a setting in which they exist only in the subgroup $\mathbb{G}_{2,p_1}$; this goes unnoticed by the subgroup hiding in group $\mathbb{G}_2$ with $\mu = \{\mathsf{P}^1_1\}$. More precisely, to see that Eq. (22) holds, consider the adversary $\mathcal{C}_0$ in Fig. 5. If $\mathsf{W} \leftarrow\!\!\$\, \mathbb{G}_2$, then the output of $\mathcal{C}_0$ is identical to that in $\mathbf{G}^{\mathcal{A}}_0(\lambda)$. If instead $\mathsf{W} \leftarrow\!\!\$\, \mathbb{G}_{2,p_1}$, then the output of $\mathcal{C}_0$ is identical to that of $\mathbf{G}^{\mathcal{A}}_1(\lambda)$.

For $j \ge 1$, compared to $\mathbf{G}^{\mathcal{A}}_j(\lambda)$, in $\mathbf{G}^{\mathcal{A}}_{j,1}(\lambda)$, a shadow copy of these elements is added into the subgroup $\mathbb{G}_{1,p_2}$, which goes unnoticed by the subgroup hiding in $\mathbb{G}_1$ with $\mu = \{\mathsf{P}^2_1, \mathsf{P}^1_2\}$. More precisely, to see that Eq. (23) holds, consider the adversary $\mathcal{B}_j$ in Fig. 5. If $\mathsf{W} = \mathsf{P}^1_1 \leftarrow\!\!\$\, \mathbb{G}_{1,p_1}$ then

$$\begin{aligned} \mathsf{V}_k &\leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}^1_1 + [\textstyle\sum_{u=1}^{j} r_u \varrho_k(\boldsymbol{x}_u)]\mathsf{P}^2_1 \ , \\ \mathsf{A} &\leftarrow \textstyle\sum_{i=1}^{\kappa} \Delta_i([f_i(\boldsymbol{x})]\mathsf{P}^1_1 + [\sum_{u=1}^{j} r_u f_i(\boldsymbol{x}_u)]\mathsf{P}^2_1) \ , \end{aligned}$$

which is identical to the values in $\mathbf{G}^{\mathcal{A}}_j(\lambda)$. If instead $\mathsf{W} \leftarrow\!\!\$\, \mathbb{G}_1$, then $\mathsf{W} = \mathsf{P}^1_1 + [r_{j+1}]\mathsf{P}^2_1$ for uniformly distributed $\mathsf{P}^1_1 \in \mathbb{G}_{1,p_1}$, for $r_{j+1} \in \mathbb{Z}^*_N$, and

$$\begin{aligned} \mathsf{V}_k &\leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}^1_1 + [r_{j+1}\varrho_k(\boldsymbol{x}) + \textstyle\sum_{u=1}^{j} r_u \varrho_k(\boldsymbol{x}_u)]\mathsf{P}^2_1 \\ \mathsf{A} &\leftarrow \textstyle\sum_{i=1}^{\kappa} \Delta_i([f_i(\boldsymbol{x})]\mathsf{P}^1_1 + [(r_{j+1} f_i(\boldsymbol{x}) + \sum_{u=1}^{j} r_u f_i(\boldsymbol{x}_u))]\mathsf{P}^2_1) \ , \end{aligned}$$

which is identical to the values in $\mathbf{G}^{\mathcal{A}}_{j,1}(\lambda)$.

After $\mathbf{G}^{\mathcal{A}}_{j,1}(\lambda)$, we move on to $\mathbf{G}^{\mathcal{A}}_{j+1}(\lambda)$. In the game $\mathbf{G}^{\mathcal{A}}_{j+1}(\lambda)$, the shadow copy of $[r_{j+1}\varrho_k(\boldsymbol{x})]\mathsf{P}^2_1$ is switched to operate over a new set of variables $\boldsymbol{x}_{j+1}$. The output of this game is identical to the output of $\mathbf{G}^{\mathcal{A}}_{j,1}(\lambda)$ by the extended adaptive parameter hiding holding in $\mathbb{G}_1$ with respect to $\mathcal{R} \cup \{f_i\}_{i=1}^{\kappa}$ and $\mathsf{aux} = \{[\sigma(\cdot)]\mathsf{P}^1_2\}$ for any $\sigma \in \mathcal{S}$ and $\mathsf{P}^1_2 \in \mathbb{G}_{2,p_1}$. More precisely, to see that Eq. (24) holds, consider the following. If we define $\mathsf{C} = [\sum_{u=1}^{j} r_u \varrho_k(\boldsymbol{x}_u)]\mathsf{P}^2_1$, then $\mathsf{C}$ is independent from $\boldsymbol{x}$ and $\boldsymbol{x}_{j+1}$. We use the extended adaptive parameter hiding property with respect to functions $\mathcal{R} \cup \{f_i\}_{i=1}^{\kappa}$. The distributions

$$(\boldsymbol{\Delta}, \sum_{i=1}^{\kappa} \Delta_i([f_i(\boldsymbol{x})]\mathsf{P}^1_1 + [r_{j+1} f_i(\boldsymbol{x}) + \sum_{u=1}^{j} r_u f_i(\boldsymbol{x}_u)]\mathsf{P}^2_1))$$

and

$$\left(\boldsymbol{\Delta}', \sum_{i=1}^{\kappa} \Delta'_i([f_i(\boldsymbol{x})]\mathsf{P}^1_1 + \sum_{u=1}^{j+1} r_u f_i(\boldsymbol{x}_u))\mathsf{P}^2_1\right)$$

$\mathbf{SUA}^{\mathcal{A}}_{c,\mathcal{R},\mathcal{S},\mathcal{T},\{f_i\}}(\lambda)$

$(\mathsf{p},\mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda,2); \mathsf{P}_1 \leftarrow\!\$\, \mathbb{G}_1; \mathsf{P}_2 \leftarrow\!\$\, \mathbb{G}_2; \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;$
**for** $k \in [1,r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1$; **endfor** ; **for** $k \in [1,s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2$; **endfor**
$\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1,\mathsf{P}_2); (\boldsymbol{\Delta},\mathsf{B}) \leftarrow\!\$\, \mathcal{A}(\mathsf{p},\mathbf{V},\mathbf{Y},\mathsf{Z}); \mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \Delta_i [f_i(\boldsymbol{x})]\mathsf{P}_1;$
**if** $\mathsf{B} = \mathsf{A} \wedge \boldsymbol{\Delta} \neq \mathbf{0}$ **then return** 1; **else return** 0; **fi**

$\mathbf{G}_0^{\mathcal{A}}(\lambda)$

$(\mathsf{p},\mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda,2); \mathsf{P}_1^1 \leftarrow\!\$\, \mathbb{G}_{1,p_1}; \mathsf{P}_2 \leftarrow\!\$\, \mathbb{G}_2; \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;$
**for** $k \in [1,r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1^1$; **endfor** ; **for** $k \in [1,s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2$; **endfor**
$\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1^1,\mathsf{P}_2); (\boldsymbol{\Delta},\mathsf{B}) \leftarrow\!\$\, \mathcal{A}(\mathsf{p},\mathbf{V},\mathbf{Y},\mathsf{Z}); \mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \Delta_i [f_i(\boldsymbol{x})]\mathsf{P}_1^1;$
**if** $\mathsf{B} = \mathsf{A} \wedge \boldsymbol{\Delta} \neq \mathbf{0}$ **then return** 1; **else return** 0; **fi**

$\mathbf{G}_1^{\mathcal{A}}(\lambda)$

$(\mathsf{p},\mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda,2); \mathsf{P}_1^1 \leftarrow\!\$\, \mathbb{G}_{1,p_1}; \mathsf{P}_2^1 \leftarrow\!\$\, \mathbb{G}_{2,p_1}; \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;$
**for** $k \in [1,r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1^1$; **endfor** ; **for** $k \in [1,s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2^1$; **endfor**
$\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1^1,\mathsf{P}_2^1); (\boldsymbol{\Delta},\mathsf{B}) \leftarrow\!\$\, \mathcal{A}(\mathsf{p},\mathbf{V},\mathbf{Y},\mathsf{Z}); \mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \Delta_i [f_i(\boldsymbol{x})]\mathsf{P}_1^1;$
**if** $\mathsf{B} = \mathsf{A} \wedge \boldsymbol{\Delta} \neq \mathbf{0}$ **then return** 1; **else return** 0; **fi** ;

$\mathbf{G}_j^{\mathcal{A}}(\lambda)$

$(\mathsf{p},\mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda,2); \mathsf{P}_1^1 \leftarrow\!\$\, \mathbb{G}_{1,p_1}; \mathsf{P}_1^2 \leftarrow\!\$\, \mathbb{G}_{1,p_2}; \mathsf{P}_2^1 \leftarrow\!\$\, \mathbb{G}_{2,p_1}; \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;$
**for** $u \in [1,j]$ **do** $\boldsymbol{x}_u \leftarrow\!\$\, (\mathbb{Z}_N^*)^c; r_u \leftarrow\!\$\, \mathbb{Z}_N^*$; **endfor**
**for** $k \in [1,r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{u=1}^{j} r_u \varrho_k(\boldsymbol{x}_u)]\mathsf{P}_1^2$; **endfor**
**for** $k \in [1,s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2^1$; **endfor** ; $\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1^1,\mathsf{P}_2^1);$
$(\boldsymbol{\Delta},\mathsf{B}) \leftarrow \mathcal{A}(\mathsf{p},\mathbf{V},\mathbf{Y},\mathsf{Z}); \mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \Delta_i ([f_i(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{u=1}^{j} r_u f_i(\boldsymbol{x}_u)]\mathsf{P}_1^2);$
**if** $\mathsf{B} = \mathsf{A} \wedge \boldsymbol{\Delta} \neq \mathbf{0}$ **then return** 1; **else return** 0; **fi** ;

$\mathbf{G}_{j,1}^{\mathcal{A}}(\lambda)$

$(\mathsf{p},\mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda,2), \mathsf{P}_1^1 \leftarrow\!\$\, \mathbb{G}_{1,p_1}, \mathsf{P}_1^2 \leftarrow\!\$\, \mathbb{G}_{1,p_2}, \mathsf{P}_2^1 \leftarrow\!\$\, \mathbb{G}_{2,p_1}; \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;$
**for** $u \in [1,j]$ **do** $\boldsymbol{x}_u \leftarrow\!\$\, (\mathbb{Z}_N^*)^c; r_u \leftarrow\!\$\, \mathbb{Z}_N^*$; **endfor** ; $r_{j+1} \leftarrow\!\$\, \mathbb{Z}_N^*;$
**for** $k \in [1..r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1^1 + [r_{j+1}\varrho_k(\boldsymbol{x}) + \sum_{u=1}^{j} r_u \varrho_k(\boldsymbol{x}_u)]\mathsf{P}_1^2$; **endfor**
**for** $k \in [1,s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2^1$; **endfor** ; $\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1^1,\mathsf{P}_2^1);$
$(\boldsymbol{\Delta},\mathsf{B}) \leftarrow \mathcal{A}(\mathsf{p},\mathbf{V},\mathbf{Y},\mathsf{Z}); \mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \Delta_i ([f_i(\boldsymbol{x})]\mathsf{P}_1^1 + [r_{j+1} f_i(\boldsymbol{x}) + \sum_{u=1}^{j} r_u f_i(\boldsymbol{x}_u)]\mathsf{P}_1^2);$
**if** $\mathsf{B} = \mathsf{A} \wedge \boldsymbol{\Delta} \neq \mathbf{0}$ **then return** 1; **else return** 0; **fi** ;

$\mathbf{G}_{j+1}^{\mathcal{A}}(\lambda)$

$(\mathsf{p},\mathsf{aux}_\mathsf{p}) \leftarrow \mathsf{Pgen}(1^\lambda,2); \mathsf{P}_1^1 \leftarrow\!\$\, \mathbb{G}_{1,p_1}, \mathsf{P}_1^2 \leftarrow\!\$\, \mathbb{G}_{1,p_2}, \mathsf{P}_2^1 \leftarrow\!\$\, \mathbb{G}_{2,p_1}; \boldsymbol{x} \leftarrow\!\$\, (\mathbb{Z}_N^*)^c;$
**for** $t \in [1, j+1]$ **do** $\boldsymbol{x}_t \leftarrow\!\$\, (\mathbb{Z}_N^*)^c; r_t \leftarrow\!\$\, \mathbb{Z}_N^*$; **endfor**
**for** $k \in [1..r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{u=1}^{j+1} r_u \varrho_k(\boldsymbol{x}_u)]\mathsf{P}_1^2$; **endfor** ;
**for** $k \in [1,s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2^1$; **endfor** ; $\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1^1,\mathsf{P}_2^1);$
$(\boldsymbol{\Delta},\mathsf{B}) \leftarrow\!\$\, \mathcal{A}(\mathsf{p},\mathbf{V},\mathbf{Y},\mathsf{Z}); \mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \Delta_i ([f_i(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{u=1}^{j+1} r_u f_i(\boldsymbol{x}_u)]\mathsf{P}_1^2);$
**if** $\mathsf{B} = \mathsf{A} \wedge \boldsymbol{\Delta} \neq \mathbf{0}$ **then return** 1; **else return** 0; **fi** ;

**Fig. 4.** Games in the proof of Theorem 6

are identical. The auxiliary information $\mathsf{aux}$ contains the elements $\{[\sigma(\cdot)]\mathsf{P}_2^1\}_{\sigma\in\mathcal{S}}$ and these elements do not affect the above distribution as they are defined in different subgroup of $\mathbb{G}_2$ and $\mathbb{G}_T$.

We repeat the process of adding and re-randomizing the original set of variables into the subgroup $\mathbb{G}_{1,p_2}$ polynomially many (say $\ell$) times to get Eq. (20).

**Equation (25)** follows by definition, as the $\mathcal{R}$, $\mathcal{S}$ and $\{f_i\}$ values have now changed to the form specified in the dual-system assumption. □

$\mathcal{B}_0(\mathsf{p}, \emptyset, \mathsf{W})$

---

$\boldsymbol{x} \leftarrow\$ (\mathbb{Z}_N^*)^c$; **for** $k \in [1..r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{W}$; **endfor**
**for** $k \in [1..s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2$; **endfor** ; $\mathsf{Z} \leftarrow \hat{e}(\mathsf{W}, \mathsf{P}_2)$;
$\mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i [f_i(\boldsymbol{x})]\mathsf{W}$; $\mathsf{B} \leftarrow\$ \mathcal{A}(\mathsf{p}, \mathbf{V}, \mathbf{Y}, \mathsf{Z})$; **return** $\mathsf{B} = (\boldsymbol{\Delta}, \mathsf{A})$;

$\mathcal{C}_0(\mathsf{p}, \{\mathsf{P}_1^1\}, \mathsf{W})$

---

$\boldsymbol{x} \leftarrow\$ (\mathbb{Z}_N^*)^c$; **for** $k \in [1..r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{P}_1^1$; **endfor**
**for** $k \in [1..s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{W}$; **endfor** ; $\mathsf{Z} \leftarrow \hat{e}(\mathsf{P}_1^1, \mathsf{W})$;
$\mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i [f_i(\boldsymbol{x})]\mathsf{P}_1^1$; $\mathsf{B} \leftarrow\$ \mathcal{A}(\mathsf{p}, \mathbf{V}, \mathbf{Y}, \mathsf{Z})$; **return** $\mathsf{B} = (\boldsymbol{\Delta}, \mathsf{A})$;

$\mathcal{B}_j(\mathsf{p}, \{\mathsf{P}_1^2, \mathsf{P}_2^1\}, \mathsf{W})$

---

$\boldsymbol{x}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_j, \boldsymbol{x}_{j+1} \leftarrow\$ (\mathbb{Z}_N^*)^c$; $r_1, \ldots, r_j \leftarrow \mathbb{Z}_N^*$;
**for** $k \in [1..r]$ **do** $\mathsf{V}_k \leftarrow [\varrho_k(\boldsymbol{x})]\mathsf{W} + [\sum_{t=1}^{j} r_t \varrho_k(\boldsymbol{x}_t)]\mathsf{P}_1^2$; **endfor**
**for** $k \in [1..s]$ **do** $\mathsf{Y}_k \leftarrow [\sigma_k(\boldsymbol{x})]\mathsf{P}_2^1$; **endfor** ; $\mathsf{Z} \leftarrow \hat{e}(\mathsf{W}, \mathsf{P}_2^1)$;
$\mathsf{A} \leftarrow \sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i [f_i(\boldsymbol{x})]\mathsf{W} + [\sum_{u=1}^{j} r_u f_i(\boldsymbol{x}_u)]\mathsf{P}_1^2$; $\mathsf{B} \leftarrow\$ \mathcal{A}(\mathsf{p}, \mathbf{V}, \mathbf{Y}, \mathsf{Z})$; **return** $\mathsf{B} = (\boldsymbol{\Delta}, \mathsf{A})$;

**Fig. 5.** Adversaries $\mathcal{B}_0, \mathcal{C}_0, \mathcal{B}_j$

*Remark 3 (Tightness of the reduction).* The papers [CM14] and [CMM16] present different reductions from uber-assumptions to subgroup hiding. The newer reduction of [CMM16] requires adding one additional prime to the factorization of $N$, but it achieves logarithmic — rather than linear — tightness. We chose not to give the corresponding tight reduction proof, as it does not improve the efficiency of our scheme (the gain in tightness means the use of smaller security parameter; however, this is offset by the use of a larger $N$); one can clearly combine the techniques of [CMM16] with the current proof to obtain such reduction for span-uber-assumptions. We leave it as an open question whether it is possible to have a reduction which would decrease the number of queries to the oracle without adding another prime factor to $N$.

Theorem 6 states that the original $(\mathcal{R}, \mathcal{S}, \{f_i\}_{i=1}^{\kappa})$-span-uber-assumption is equivalent to a span-uber-assumption in different form in which adversary is given $([\varrho(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{j=1}^{\ell} r_j \varrho(\boldsymbol{x}_j)]\mathsf{P}_1^2)_{\varrho \in \mathcal{R}}$, $([\sigma(\boldsymbol{x})]\mathsf{P}_2^1)_{\sigma \in \mathcal{S}}$, and $\hat{e}(\mathsf{P}_1^1, \mathsf{P}_2^1)$ and asked to compute $\boldsymbol{\Delta}$ and $\sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i([f_i(\boldsymbol{x})]\mathsf{P}_1^1 + [\sum_{j=1}^{\ell} r_j f_i(\boldsymbol{x}_j)]\mathsf{P}_1^2)$. Assume, for the sake of simplicity, that $\mathsf{P}_1^1$ and $\boldsymbol{x}$ are public, so $\mathcal{A}$ can compute the $\mathbb{G}_{1,p_1}$ component of this target, and all she needs to compute is $\sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i(F_i(\boldsymbol{x}) + [\sum_{j=1}^{\ell} r_j f_i(\boldsymbol{x}_j)]\mathsf{P}_1^2)$, where $F_i(\boldsymbol{x}) = [f_i(\boldsymbol{x})]\mathsf{P}_1^1$. Clearly, $[\mathcal{S}(\boldsymbol{x})]\mathsf{P}_2^1$ and $\hat{e}(\mathsf{P}_1^1, \mathsf{P}_2^1)$ provide no advantage, as they operate in different groups over a completely independent set of variables.

Next, in their reasoning, [CM14] and [CMM16] use the so-called bijection argument. Its core is the Lemma that relates the linear independence of polynomials $\mathcal{R}$ with the invertibility of a matrix composed of the adversary's view. (See Lemma 4.4 [CM14].)

We explain the details for our case. In the $(\mathcal{R}, \mathcal{S}, \mathcal{T}, \{f_i\}_{i=1}^{\kappa})$-computational span-uber-assumption, the adversary sees values with exponent of the form $\mathbf{Y} = \boldsymbol{r}V$, where $V$ is the next matrix:

$$\begin{pmatrix} \varrho_1(\boldsymbol{x}_1)\ \varrho_2(\boldsymbol{x}_1)\ \ldots\ \varrho_q(\boldsymbol{x}_1) & \sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i \sum_{j=1}^{\ell} r_j f_i(\boldsymbol{x}_1) \\ \varrho_1(\boldsymbol{x}_2)\ \varrho_2(\boldsymbol{x}_2)\ \ldots\ \varrho_q(\boldsymbol{x}_2) & \sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i \sum_{j=1}^{\ell} r_j f_i(\boldsymbol{x}_2) \\ \ldots\ \ \ldots\ \ \ldots\ \ \ldots & \ldots \\ \varrho_1(\boldsymbol{x}_\ell)\ \varrho_2(\boldsymbol{x}_\ell)\ \ldots\ \varrho_q(\boldsymbol{x}_\ell) & \sum_{i=1}^{\kappa} \boldsymbol{\Delta}_i \sum_{j=1}^{\ell} r_j f_i(\boldsymbol{x}_\ell) \end{pmatrix}$$

According to Lemma 4.4 [CM14], $V$ is invertible with all but negligible probability if the functions in $\mathcal{R}$ are linearly independent and of maximum degree $\mathrm{poly}(\lambda)$ and $\ell = q + 1$ for $q = \mathrm{poly}(\lambda)$.

Since $\boldsymbol{r}$ and $\mathbf{Y} = V\boldsymbol{r}$ are both members of the set $S$ containing all sets of size $q + 1$ over $\mathbb{Z}_N$, so multiplication by $V$ maps $S$ to itself. As $V$ is invertible, the map is invertible as well and is thus a permutation over $S$. Sampling $\boldsymbol{r}$ uniformly by random and then multiplying by $V$ thus yields a vector $\mathbf{Y}$ that is distributed uniformly at random over $\mathbb{Z}_N$. When $V$ is invertible, an adversary $\mathcal{A}$ thus has no advantage in distinguishing

between $\mathbf{Y}$ and a uniformly random vector in $S$, as the distributions over the two are identical, and thus has a negligible overall advantage in the new form of the uber-assumption.

By now, we have proven Theorem 4, which indicates conditions under which the uber-span-assumption is implied by the subgroup hiding.