Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 114 (2024)

Dot-Product Proofs and Their Applications[*]

Nir Bitansky[†] Prahladh Harsha[‡] Yuval Ishai[§] Ron D. Rothblum[¶]
David J. Wu[‖]

Abstract

A *dot-product proof* (DPP) is a simple probabilistic proof system in which the input statement $\mathbf{x}$ and the proof $\boldsymbol{\pi}$ are vectors over a finite field $\mathbb{F}$, and the proof is verified by making a single dot-product query $\langle \mathbf{q}, (\mathbf{x} \| \boldsymbol{\pi}) \rangle$ jointly to $\mathbf{x}$ and $\boldsymbol{\pi}$. A DPP can be viewed as a 1-query fully linear PCP. We study the feasibility and efficiency of DPPs, obtaining the following results:

- **Small-field DPP.** For any finite field $\mathbb{F}$ and Boolean circuit C of size S, there is a DPP for proving that there exists $\boldsymbol{w}$ such that $C(\boldsymbol{x}, \boldsymbol{w}) = 1$ with a proof $\boldsymbol{\pi}$ of length $S \cdot \text{poly}(|\mathbb{F}|)$ and soundness error $\varepsilon = O(1/\sqrt{|\mathbb{F}|})$. We show this error to be asymptotically optimal.

  In particular, and in contrast to the best known PCPs, there exist strictly linear-length DPPs over constant-size fields.

- **Large-field DPP.** If $|\mathbb{F}| \geq \text{poly}(S/\varepsilon)$, there is a similar DPP with soundness error ε and proof length $O(S)$ (in field elements).

The above results do not rely on the PCP theorem and their proofs are considerably simpler. We apply our DPP constructions toward two kinds of applications.

- **Hardness of approximation.** We obtain a simple proof for the NP-hardness of approximating MAXLIN (with dense instances) over any finite field $\mathbb{F}$ up to some constant factor $c > 1$, independent of $\mathbb{F}$. Unlike previous PCP-based proofs, our proof yields *exponential-time* hardness under the exponential time hypothesis (ETH).

- **Succinct arguments.** We improve the concrete efficiency of succinct interactive arguments in the generic group model using input-independent preprocessing. In particular, the communication is comparable to sending two group elements and the verifier's computation is dominated by a single group exponentiation. We also show how to use DPPs together with linear-only encryption to construct succinct commit-and-prove arguments.

[*] A conference version of this paper appeared in *Proc.* 65*th FOCS, 2024* [BHIRW24].

[†] New York University, USA and Tel Aviv University, Israel, `nbitansky@gmail.com`

[‡] Tata Institute of Fundamental Research, India, `prahladh@tifr.res.in`

[§] Technion, Israel, `yuvali@cs.technion.ac.il`

[¶] Technion, Israel, `rothblum@cs.technion.ac.il`

[‖] University of Texas at Austin, USA, `dwu4@cs.utexas.edu`

ISSN 1433-8092

Contents

1 Introduction

A *probabilistically checkable proof* (PCP) is a proof that can be verified by a randomized algorithm that only reads a small part of the proof. The celebrated PCP theorem [AS98, ALMSS98] establishes the remarkable fact that every standard NP proof, say a satisfying assignment for a Boolean circuit, can be efficiently converted into a PCP that can be verified, with a small soundness error, by querying a *constant* number of proof bits. In fact, 3 queries suffice for constant soundness error.

The PCP theorem and its subsequent refinements have found numerous applications in different areas of computer science. In the theory of algorithms, PCPs can be used to establish *hardness of approximation* results for many natural optimization problems under the (minimal) assumption that $\mathrm{P} \neq \mathrm{NP}$ [FGLSS96]. In cryptography, PCPs serve as a building block for *succinct arguments*: low-communication proof systems whose soundness holds against a computationally bounded prover [Kil92, Mic00].

Despite decades of research, the best known PCP constructions still suffer from three limitations:

- First, there are no known "constant-rate" PCPs, whose length grows linearly with the natural NP witness. In particular, the shortest known (constant-query) PCP for the satisfiability of a circuit of size S is of length $S \cdot \mathrm{polylog}(S)$ [BS08, BGHSV05, Din07].[1] Consequently, known PCP-based hardness of approximation results do not scale to the exponential-time regime.
- Second, even the simplest known proofs of the PCP theorem, including the elegant proof of Dinur [Din07], are still quite involved. This may partially explain the poor *concrete efficiency* of existing PCP constructions, which have so far resisted considerable optimization efforts [BCGT13, BBCG$^+$17].
- Finally, an inherent limitation of query-efficient PCPs is that a badly-formed proof may create a noticeable correlation between the verifier's queries and the event of accepting the proof, which is problematic when the same secret queries need to be reused [IKNOS25].

Relaxing the PCP model. Driven by cryptographic applications, the above limitations of classical PCPs have motivated two natural relaxations of the classical PCP model. The first, known as an *interactive oracle proof* (IOP) [KR08, BCS16, RRR21], enables multiple rounds of interaction between the prover and the verifier (see Section 1.2 for discussion). The present work is concerned with a different relaxation of the PCP model, known as a *linear* PCP (LPCP) [IKO07, BCIOP22]. Whereas in a classical PCP each query returns one bit (or symbol) from the proof, in an LPCP, each query returns a *linear combination* of the proof entries. More precisely, the proof π is viewed as a vector over a finite field $\mathbb{F}$, and a query q returns the inner product $\langle q, \pi \rangle$ over $\mathbb{F}$. LPCPs do not suffer from the above limitations of classical PCPs. In particular, over a sufficiently-large field $\mathbb{F}$, the satisfiability of an arithmetic circuit of size S can be proved by a *simple* LPCP with a proof vector π of length $O(S)$ and which can be verified using 3 queries with soundness error of $O(S/|\mathbb{F}|)$ [GGPR13, BCIOP22]. (An even simpler LPCP with $O(S^2)$ proof length, based on the Hadamard code, is implicit in the PCP construction of [ALMSS98].) Similar to classical PCPs, LPCPs can also be used for constructing succinct arguments [IKO07, Gro10, BCIOP22], and in fact, have served as a basis for practical and widely-deployed succinct proof systems [BCGG$^+$14, PHGR16, Gro16]. Here it is often crucial to avoid the third limitation of classical PCPs discussed above, which is possible for LPCPs over large fields $\mathbb{F}$.

[1] Allowing for S^γ queries, a *non-uniform* construction of PCPs of length $2^{O(1/\gamma)} \cdot S$ was given in [BKKMS16].

Single-query LPCP? In contrast to classical PCPs, which require at least 3 queries (or 2 queries over a non-binary alphabet), an LPCP can potentially have only one query. Beyond the intrinsic interest in simplifying proof systems to the extent possible, 1-query LPCPs are motivated by two kinds of applications. First, they can be used to minimize the communication of succinct arguments to just a *single* ciphertext of a suitable public-key encryption scheme (under strong but plausible assumptions) [BCIOP22]. Second, a restricted kind of 1-query LPCP for NP, in which only one answer is accepted by the verifier, implies the NP-hardness of approximating **MAXLIN**, the maximal number of equations that can be simultaneously satisfied in a linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. Known hardness results for this well-studied problem were obtained using heavy PCP machinery [Hås01, HKLT19]. There are three known approaches for constructing 1-query LPCPs:

- The first uses a general compiler to transform any classical PCP into a 1-query LPCP by packing the answers to all PCP queries into a single field element [BCIOP22]. This approach inherits the disadvantages of classical PCPs.

- A second approach uses a more specialized compiler to transform a specific multi-query *linear* PCP into a 1-query LPCP [BIOW20]. While relatively simple, the 1-query LPCPs obtained via this approach are limited in several important ways: Their proof length grows quadratically (rather than linearly) with the circuit size; the field size must grow polynomially with the circuit size; and the resulting LPCP cannot satisfy the "single accepted answer" feature required for the application to hardness of approximation.

- The third approach relies on known hardness of approximation results. Concretely, Barta, Ishai, Ostrovsky and Wu [BIOW20] constructed a 1-query LPCP from the NP-hardness of approximating the nearest codeword problem [KPV14]. Alternatively, one could use the NP-hardness of approximating **MAXLIN**.[2] However, the 1-query LPCP obtained via this approach has a non-negligible completeness error and, similar to the first approach, does not enjoy the efficiency and simplicity benefits of known multi-query LPCP constructions.

To conclude, all known constructions of 1-query LPCPs over constant-size fields build on the PCP theorem and suffer from the associated costs. Moreover, even over large fields, the only known 1-query LPCPs for circuit satisfiability that avoids the PCP theorem incur a quadratic overhead.

1.1 Our Contributions

Motivated by the above limitations of known 1-query LPCPs, we initiate a more systematic study of this simple kind of proof system. In fact, we take simplicity one step further by considering a *fully-linear* variant in which the verifier does not have direct access to the input statement $\boldsymbol{x}$; instead, the inner-product query is applied *jointly* to $(\boldsymbol{x}\|\boldsymbol{\pi})$, the string obtained by concatenating the input $\boldsymbol{x}$ and the proof vector $\boldsymbol{\pi}$. This full linearity requirement, first explicitly considered in [BBCGI19], is motivated by cryptographic applications and will be satisfied by our constructions at no extra cost; see Section 1.2 for additional discussion. We refer to such a 1-query fully linear PCP as a *dot-product proof* (DPP),[3] which we formalize below.

[2] There is in fact a duality relation between these two problems. In both cases, the goal is to find the minimal Hamming weight w of a vector in a given affine subspace of $\mathbb{F}^n$. While in the former case the value of the solution is w, in the latter **MAXLIN** case it is $n - w$.

[3] We used "dot product" instead of "inner product" to avoid an acronym collision with "Interactive Proofs of Proximity" (IPP) [RVW13].

Definition 1.1 (Dot-Product Proof)**.** *A* dot-product proof *(*DPP*), parameterized by a field $\mathbb{F}$, input length n, and proof length m, consists of a probabilistic algorithm $\mathcal{V}$ that outputs a* query vector $\boldsymbol{q} \in \mathbb{F}^{n+m}$ *and an* accepting set $A \subseteq \mathbb{F}$*. A* DPP *for $\mathcal{L} \subseteq \mathbb{F}^n$ satisfies the following properties:*

- *(Completeness:) for every $\boldsymbol{x} \in \mathcal{L}$ there exists a proof vector $\boldsymbol{\pi} \in \mathbb{F}^m$ such that*

  $$\Pr_{(\boldsymbol{q},A)\leftarrow\mathcal{V}} \left[\langle \boldsymbol{q}, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \in A \right] \geq c,$$

  where $c \in [0,1]$ is called the completeness *parameter. By default we assume perfect completeness, that is, $c = 1$.*

- *(Soundness:) for every $\boldsymbol{x} \notin \mathcal{L}$ and all $\boldsymbol{\pi}^* \in \mathbb{F}^m$,*

  $$\Pr_{(\boldsymbol{q},A)\leftarrow\mathcal{V}} \left[\langle \boldsymbol{q}, (\boldsymbol{x} \| \boldsymbol{\pi}^*) \rangle \in A \right] \leq s,$$

  where $s \in [0,1]$ is called the soundness error*.*

Note that the above definition does not refer at all to the computational complexity of the prover, who generates the proof $\boldsymbol{\pi}$ from the input $\boldsymbol{x}$, or the verifier who generates $\boldsymbol{q}$ and A. Indeed, unlike classical PCPs (and similar to PCPs of *proximity* [DR06, BGHSV06]),[4] the notion of DPP is meaningful even when all the algorithms involved are computationally unbounded. In fact, DPPs are nontrivial even for languages with inputs of length 2 over a constant-size field!

Our DPP constructions will all be computationally efficient in the sense that the running time of both the prover and the verifier is polynomial in the proof length. This includes an implicit representation of the accepting set A when the field size is super-polynomial. Moreover, the existence of efficient DPPs for all languages $\mathcal{L} \in \mathrm{P}$ generically implies efficient DPPs for all $\mathcal{L} \in \mathrm{NP}$, where the proof $\boldsymbol{\pi}$ is generated by a polynomial-time prover given the input $\boldsymbol{x}$ and a witness $\boldsymbol{w}$, and the query $\boldsymbol{q}$ is generated by a polynomial-time verifier. Indeed, given the witness $\boldsymbol{w}$, the proof $\boldsymbol{\pi}$ can include $\boldsymbol{w}$ along with a DPP $\boldsymbol{\pi}'$ for a corresponding (polynomial-time verifiable) NP-relation.

We also consider *promise* DPPs where soundness only holds for instances $\boldsymbol{x} \in \{0,1\}^n$ (whereas there is no promise on the proof $\boldsymbol{\pi}^* \in \mathbb{F}^m$). Such DPPs still imply 1-query LPCPs (forgoing *full* linearity, the verifier is given the input $\boldsymbol{x}$ and can check the Booleanity).

We now give a detailed account of our results.

1.1.1 DPP Constructions

Our main contribution is the construction of two kinds of DPPs for Boolean circuits of size S: DPPs over small (constant-size) fields $\mathbb{F}$, achieving constant soundness error that scales optimally with the field size, and promise DPPs over large fields $\mathbb{F}$, of size $|\mathbb{F}| \gg S$. The latter are suitable for cryptographic applications that require super-polynomial field size or negligible soundness. Both kinds of DPP constructions avoid the use of the PCP theorem, and their proof length (in field elements) is $O(S)$ (or in some optimizations $O(S \log p)$). Recall that prior to our work, even when relaxing (promise) DPP to a 1-query LPCP, this asymptotic proof length could not be achieved. Moreover, all known constructions over constant-size fields (regardless of proof length) relied on the full power of the PCP theorem.

[4] However, we do note that unlike PCPs of proximity, the soundness guaranty applies to any $\boldsymbol{x} \notin \mathcal{L}$, rather than instances far from $\mathcal{L}$. This is enabled due to the fact that the query effectively touches the entire input $\boldsymbol{x}$.

A motivating example. It is instructive to give a sense of why the PCP theorem is useful for constructing DPPs over small fields. Consider the following simple construction of a 1-query LPCP over $\mathbb{F}_3$ based on a gap version of the 3-coloring problem: decide whether (1) G is 3-colorable; or (2) every purported 3-coloring of G violates at least a δ-fraction of the edges. The NP-hardness of this problem (for some $\delta > 0$) follows from the PCP theorem [Pet94]. Given a 3-colorable graph G, a 1-query LPCP can proceed as follows. The proof vector $\boldsymbol{\pi}$ consists of the vector of colors. The query algorithm picks a random edge in the graph, and checks that the difference between the two colors is not 0. If $\boldsymbol{\pi}$ contains a valid 3-coloring, the verifier will always accept. On the other hand, if G is δ-far from 3-colorable, every $\boldsymbol{\pi}$ will be rejected with $\geq \delta$ probability.

It is not difficult to modify this construction to satisfy the full linearity requirement of DPP. However, we actually get more than we bargained for: the answer depends only on a constant number of proof entries. Since any such DPP for an NP-hard problem *implies* the PCP theorem, the DPP queries in our (simpler) constructions will need to use *dense* linear combinations of the proof entries.

Small-field DPP. For the case of constant-size fields, we prove the following theorem.

Theorem 1.2 (Optimal-Soundness DPP over Small Fields)**.** *There exists $\varepsilon_q \in O(1/\sqrt{q})$ such that the following holds. For every finite field $\mathbb{F}$ of order $q > 2$ and Boolean circuit C of size S, there is a* DPP *over $\mathbb{F}$ for the language $\{\boldsymbol{x} \in \{0,1\}^n \mid \exists \boldsymbol{w} \in \{0,1\}^m : C(\boldsymbol{x},\boldsymbol{w}) = 1\}$ with proof length $S \cdot \mathrm{poly}(q)$, perfect completeness, soundness error ε_q, and randomness complexity* $\log(S) + \mathrm{poly}(q)$.

For $\mathbb{F}_2$, where perfect completeness is impossible (see Appendix A), we get a constant gap between completeness and soundness. Theorem 1.2 implies that, in contrast to the best known PCPs, there exist strictly linear-length DPPs over constant-size fields. In particular, we improve the proof length of the 3-coloring based DPP construction described above (instantiated with the best known PCPs) by a $\mathrm{polylog}(S)$ factor.

We further prove that there are languages for which *any* DPP must have soundness error $\Omega(1/\sqrt{q})$, thus establishing the optimality of the soundness we achieve.

If we settle for constant soundness error $\varepsilon < 1$, independent of the field size, then the analysis of the construction becomes simpler and has the additional feature of using a constant-size accepting set A. This variant of the construction will be used for the application to hardness of approximation discussed below.

The proof of Theorem 1.2 does rely on highly nontrivial constructions of asymptotically-good families of algebraic-geometric (AG) codes. However, it only uses standard properties of AG codes in a black-box way. If we allow proofs of length $\mathrm{poly}(S)$, we can replace AG codes by a simple "tensoring" approach. This yields a simple self-contained proof for the feasibility of DPPs over constant-size fields, which prior to our work could only be derived from the full PCP theorem.

Large-field DPP. The above construction is only attractive for small fields, since the proof length grows polynomially with $|\mathbb{F}|$, and it cannot be efficiently applied to obtain negligible soundness error ε. For the case of large fields $\mathbb{F}$ where $|\mathbb{F}| \geq \mathrm{poly}(S/\varepsilon)$, we obtain a promise DPP (and in particular a 1-query LPCP) with soundness error ε and proof length $O(S)$ in field elements (or $O(S \cdot \log(S/\varepsilon))$ when optimizing the dependence of $|\mathbb{F}|$ on ε). Unlike the PCP-based 1-query LPCP from [BCIOP22], this DPP can achieve the strong notion of soundness needed for reusing the query.

Theorem 1.3 (DPP over Large Fields)**.** *Let $C\colon \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ be a Boolean circuit of size S and $\varepsilon > 0$ a soundness bound. Then, for a prime $p \geq \tilde{\Omega}(S^9/\varepsilon^6)$, the language $\{\boldsymbol{x} \in \{0,1\}^n :$*

$\exists \boldsymbol{w} \in \{0,1\}^m : C(\boldsymbol{x}, \boldsymbol{w}) = 1\}$ *has a promise* DPP *of proof length* $O(S \cdot \log(S/\varepsilon))$ *over* $\mathbb{F}_p$ *with soundness error* ε*. The proof length can be made* $O(S)$ *at the cost of a worse polynomial dependence of* p *on* S/ε*.*

See Corollary 5.11 for a more concrete version. Theorem 1.3 is obtained by designing a new *general* compiler from fully linear PCP to DPP, and applying it to a 2-query fully linear PCP implicit in [DFGK14] (for which we present a self-contained derivation in Appendix B). We believe that the analysis of our compiler is not tight, and leave open the question of improving the bound on p given by Theorem 1.3.

The above promise DPP, and in fact the corresponding 1-query LPCP, are sufficient for our applications. The theorem also extends to plain (non-promise) DPP at the cost of increasing the proof length to $O(S + n^2)$ and worse polynomial dependence of the field size p on S/ε.

We apply the above DPP constructions toward two kinds of applications: hardness of approximation and succinct arguments.

1.1.2 Application to Hardness of Approximation

In Section 6, we apply our results for DPP over small fields towards obtaining a simple proof for the NP-hardness of approximating **MAXLIN** up to some constant factor.

Theorem 1.4 (Hardness of Approximating **MAXLIN**)**.** *There is a universal constant* $c > 1$ *such that for every field* $\mathbb{F}$ *the following holds. Assuming the exponential time hypothesis (ETH) (Hypothesis 3.7), there does not exist any* $2^{o(n)}$*-time algorithm that can approximate* **MAXLIN**$(\mathbb{F})$*, with* n *variables and* $O(n)$ *equations, to a factor better than* c*.*

Here, **MAXLIN**$(\mathbb{F})$ refers to the constraint satisfaction problem (CSP) where the input is a set of linear equations over the field $\mathbb{F}$ and the goal is to find an assignment to the variables that satisfies the largest number of equations. **MAX-kLIN** is the variant of this problem in which each linear equation involves at most k variables.

The inapproximability factor c we obtain is not the best one could hope for, unlike the optimal inapproximability results of Håstad [Hås01] for **MAX-kLIN** for any $k \geq 3$. In contrast, the $2^{o(n)}$ time bound in Theorem 1.4 is optimal. Indeed, similar to SAT, there is a trivial $2^{O(n)}$-time algorithm for (exactly) solving **MAXLIN** instances with n variables and $O(n)$ equations using Gaussian elimination. A similar exponential hardness result for **MAX-kLIN** would imply a gap version of ETH, whose reduction to standard ETH is a well-known open problem.

Theorem 1.4 is a simple corollary of the constant-soundness variant of Theorem 1.2 (see Theorem 4.3) in which the accepting set is of size 2. Indeed, by guessing one of the two answers at random we get an accepting set of size 1 at the cost of reducing the completeness-soundness gap by a factor of 2.

On dense vs. local CSPs. Theorem 1.4 differs from typical hardness results for CSPs in that it involves *dense* (non-local) constraints. In contrast, most related results refer to CSPs in which each constraint is *local* (e.g., **MAX-kLIN** or **MAX-kCNFSAT** for constant k). While there are CSPs, such as **MAXCircuitSAT** (see, e.g., [App17]) or **MAXQUAD** (see [Gol08, Ex. 10.6]) for which inapproximability results for dense instances are trivial, there are natural classes of *dense* CSPs, such as **MAXLIN** or **MAXCNFSAT**, for which the corresponding questions still seem challenging. In particular, to the best of our knowledge, all existing inapproximability proofs for **MAXLIN** involve the use of the PCP Theorem [HKLT19]. It is conceivable that the inapproximability of these dense CSPs is indeed easier to prove than the PCP Theorem, and our proof of Theorem 1.4 adds credence to this fact in the case of **MAXLIN**. Whether the same holds for **MAXCNFSAT** is left open.

On exponential-time inapproximability. Theorem 1.4 gives a rare example for a non-trivial fully exponential time hardness result for a natural approximation problem obtained under (standard) ETH.[5] Previous exponential hardness results from the literature required an assumption at least as strong as gap-ETH. This is due to the use of the PCP theorem which incurs at least a multiplicative logarithmic overhead (c.f., discussion in [Din16, Sec. 1, Page 4]). Previous reductions used the stronger gap-ETH instead to obtain exponential time hardness. We get around this logarithmic multiplicative overhead by constructing a simpler PCP (more precisely, a DPP or 1-query linear PCP) which avoids the use of the PCP theorem. As noted above, obtaining a similar result for the local variant **MAX-kLIN** would have proved that "ETH implies gap-ETH," thus settling a longstanding open problem.

1.1.3 Application to Succinct Arguments

In Section 7, we show how to leverage DPPs to obtain arguments for NP (i.e., computationally sound proof systems) with a very low communication complexity. First, we observe that using a previous compiler of Bitansky, Chiesa, Ishai, Ostrovsky and Paneth [BCIOP22], we can combine a DPP for the language of Boolean circuit satisfiability with a so-called "linear-only" encryption scheme to obtain a designated-verifier succinct non-interactive argument (SNARG) for NP in the preprocessing model. Notably, the length of the proof in this construction consists of a *single* ciphertext of the underlying linear-only encryption scheme. Previous approaches capable of offering the same level of succinctness (single-ciphertext proofs) were either non-reusable [BCIOP22] (i.e., the scheme is insecure if the prover is able to query the verification oracle) or needed a common reference string (CRS) whose size scales quadratically with the size of the circuit being verified [BIOW20]. By instantiating the [BCIOP22] compiler with a DPP for Boolean circuit satisfiability with linear-size proofs and reusable soundness with a linear-only encryption scheme, we obtain a designated-verifier preprocessing SNARG with reusable soundness and linear-size CRS. In addition, since our approach does not rely on classical PCPs (unlike [BCIOP22]), it can also lead to concretely-efficient instantiations.

Laconic arguments with shorter proofs. We then show how to further reduce the communication cost as well as the verification cost by relying on interaction. Specifically, we construct an interactive *laconic* argument in the preprocessing model. In this model, the verifier starts by sending a long, but *statement-independent*, message. Then, in the statement-dependent online phase, we require that the total communication be small. Our approach is to encode a DPP "in the exponent" of a *pairing-free* group (similar to [BIOW20]). If we instantiate using a DPP with linear-size proofs, then we obtain a laconic argument for Boolean circuit satisfiability with the following properties:

- The verifier's initial (statement-independent) message consists of $\tilde{O}(|C|)$ group elements, where $|C|$ is the size of the Boolean circuit being checked. The same message can be reused for simultaneously proving a *batch* of NP statements of the form: there exist w_i such that $C(x_i, w_i) = 1$.

[5] Recently and independent of our work, Guruswami et al. [GLRSW24] proved the parameterized inapproximability hypothesis (PIH), the PCP hypothesis for parameterized complexity, under ETH.

- In the online phase, the prover-to-verifier communication is essentially two group elements (more precisely, it is a group element and a field element) per proof. The verifier-to-prover communication consists of a short key for a symmetric encryption scheme. Only one key is needed for all proof instances in the batch setting.
- The verification procedure consists of a single group exponentiation together with the DPP verification procedure.
- The soundness holds unconditionally in the generic group model [Nec94, Sho97].

To our knowledge, previous laconic arguments offering a similar (or better) level of succinctness either rely on heavy tools such as indistinguishability obfuscation [SW21, JJ22, WW24a, WZ24, MPV24] or witness encryption [BISW18, MPV24], or require a quadratic-size CRS [BIOW20]. (See Section 1.2 for comparison with the subsequent work [ADI25].) Like the construction from [BIOW20] with a quadratic-size CRS, our DPP-based construction is concretely-efficient. For instance, over a standard 256-bit group, we can support statements with around 2^{10} gates and providing a few bits of soundness (which is suitable in settings where there are high out-of-band costs associated with cheating). The low communication (640 bits, or 512 bits per proof in a batch setting) and the fast verification (essentially a single exponentiation) make this scheme appealing for applications that demand ultra-fast or ultra-low-energy verification. Compared to [BIOW20] (which was also limited to a small number of bits of soundness), our construction reduces the CRS size from quadratic to quasi-linear in the circuit size. For even circuits of modest size with a few thousand gates, this already represents a noticeable reduction in CRS size.

On concrete efficiency. Our current analysis can only yield concrete efficiency advantages for small circuits and with a high soundness error. Thus, we view it more as a proof of concept than a useful end result. However, we believe that this analysis is far from being tight.[6] The potential usefulness of DPPs to practical efficient arguments motivates a tighter analysis of our large-field DPP construction. More ambitiously, it further motivates the question of closing the big polynomial gap between the soundness error we achieve and the optimal $O(1/\sqrt{|\mathbb{F}|})$ soundness that we can achieve over small fields. This might call for an entirely different approach than the one we take here.

Succinct commit-and-prove arguments. The structure of a DPP (and more generally, any fully-linear PCP) also lends itself naturally to give succinct commit-and-prove arguments [Kil92, CLOS02, EG14, CFHK+15]. Here, a prover can commit to an input $\boldsymbol{x} \in \{0,1\}^\ell$ with a short digest σ and later on, provide succinct openings $\pi_1, \ldots, \pi_\ell$ to different functions $f_1, \ldots, f_\ell$ of the committed input (i.e., that $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$ for each $i \in [\ell]$). Both the size of the commitment σ and the size of the proof π_i should be sublinear in both the input length $|\boldsymbol{x}|$ and the size of the Boolean (or arithmetic) circuit computing f.

We show that by applying a simple adaptation of the [BCIOP22] compiler to a DPP, we obtain a succinct commit-and-prove argument system where the size of the commitment and the size of the opening consist of a single ciphertext for a linear-only encryption scheme. The approach exploits the fully-linear property of the DPP: namely, DPP verification corresponds to checking that $\langle \boldsymbol{q}, \boldsymbol{x} \| \boldsymbol{\pi} \rangle \in A$ for some accepting set A, where $\boldsymbol{q}$ is the DPP query, $\boldsymbol{x}$ is the statement, and $\boldsymbol{\pi}$

[6] In particular, we conjecture that the dependence on S (i.e., the circuit size) in our current soundness bound can be improved. This was shown in the simpler case of the Hadamard-based LPCP using a random walk argument [BIOW20], and while it seems to heuristically hold also in our setting, the formal analysis is much more challenging.

is the proof. By linearity, we can write this as

$$\langle \boldsymbol{q}, \boldsymbol{x} \| \boldsymbol{\pi} \rangle = \langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle + \langle \boldsymbol{q}_{\boldsymbol{\pi}} \| \boldsymbol{\pi} \rangle \,,$$

where $\boldsymbol{q} = \boldsymbol{q}_{\boldsymbol{x}} \| \boldsymbol{q}_{\boldsymbol{\pi}}$. To obtain a commit-and-prove argument, we follow the [BCIOP22] approach of encrypting the components of the query vector $\boldsymbol{q}$ using a linear-only encryption scheme. The commitment to an input $\boldsymbol{x}$ is a ciphertext encrypting $\langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle$, and an opening to $\boldsymbol{y} = f(\boldsymbol{x})$ is the ciphertext encrypting $\langle \boldsymbol{q}_{\boldsymbol{\pi}}, \boldsymbol{\pi} \rangle$, where $\boldsymbol{\pi}$ is a DPP proof that $\boldsymbol{y} = f(\boldsymbol{x})$. Both of these are linear functions of the components of the encrypted query vector. To verify the proof, the verifier decrypts the commitment and the opening to learn $z = \langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle + \langle \boldsymbol{q}_{\boldsymbol{\pi}} \| \boldsymbol{\pi} \rangle$ and then checks if $z \in A$. By a similar approach as that used to obtain our laconic argument, we can also obtain an *interactive* commit-and-prove argument where the size of the commitment and the size of the opening each consists of just a single group element.

1.2 Related Works

Interactive oracle proofs. Another relaxation of the classical PCP model, orthogonal to the *linear* variant considered in this work, is an *interactive oracle proof* (IOP) [KR08, BCS16, RRR21]. An IOP is an interactive form of a PCP that allows for multiple rounds of interaction between the prover and the verifier. The IOP model enables simpler and more efficient constructions, which enable even shorter proof length than those known for DPP; see [NR22] and references therein. In the context of cryptographic applications, IOPs can be compiled into succinct arguments without the use of public-key cryptography; on the downside, the communication complexity of the resulting arguments is significantly higher than in LPCP-based constructions because of the higher query complexity. Arnon, Chiesa, and Yogev [ACY22] recently used IOPs to prove hardness of approximation results for stochastic constraint satisfaction problems.

Fully linear proof systems. Fully linear PCPs were formally introduced by Boneh, Boyle, Corrigan-Gibbs, Gilboa, and Ishai [BBCGI19]. As described previously, in a fully linear PCP, the verifier only has *linear* access to *both* the statement and the proof (whereas in a linear PCP, the verifier has full access to the statement itself). The fully linear property is useful for constructing succinct (and possibly, zero-knowledge) arguments on statements that are secret shared across multiple parties or when the statement is encoded using a linearly-homomorphic encryption or commitment scheme. This has been a useful building block in a number of privacy-preserving systems that rely on proofs on distributed data [CB17, ECZB21, BBCGI21]. Fully linear PCPs are also useful for constructing succinct "commit-and-prove" style arguments, where the query part corresponding to the input is used for committing; see Section 7.3. While the communication cost of the former application is typically dominated by the proof length of the fully linear PCP, the cost of the commit-and-prove arguments is dominated by the number of queries, which we minimize in this work.

Constant-rate PCPs. Settling for a super-constant (but sublinear) number of queries, Ben-Sasson, Kaplan, Kopparty, Meir, and Stichtenoth [BKKMS16] construct a PCP of length $O(S)$ for proving the satisfiability of a Boolean circuit of size S. Concretely, with S^{γ} queries the proof length is $2^{O(1/\gamma)} \cdot S$. This construction is not fully uniform, and it is not known to imply exponential-time hardness of approximation results for natural optimization problems. Similarly to our small-field DPP construction, the PCP construction from [BKKMS16] also relies on AG codes. However, it relies on additional structural properties of (certain) families of AG codes beyond the standard properties on which we rely.

On DPP vs. 1-query linear PCP. Recall that a DPP (or 1-query *fully* linear PCP) is stronger than a 1-query linear PCP in that the verifier's decision depends on a single linear query that applies *jointly* to the input and the proof. This has two advantages. First, as discussed above, it is useful for supporting cryptographic applications, such as proofs on distributed data or succinct commit-and-prove arguments. Second, it makes the DPP notion meaningful even for finite languages and even if P = NP. Our results show that this has an inherent price: the soundness error of DPP over $\mathbb{F}$ must be $\Omega(1/\sqrt{|\mathbb{F}|})$, whereas in a 1-query LPCP based on optimal hardness of **MAXLIN** [Hås01],[7] the soundness can be $O(1/|\mathbb{F}|)$. On the other hand, the latter 1-query LPCP construction has several disadvantages compared to our DPP construction: (1) it has a non-negligible completeness error, (2) since it relies on the PCP theorem, it cannot achieve linear proof size nor good concrete efficiency, and (3) it cannot offer the strong notion of soundness required for reusing the queries in the context of succinct non-interactive arguments. An alternative PCP-based 1-query LPCP construction from [BCIOP22] (which can be turned into a DPP) avoids disadvantage (1), but still has disadvantages (2), (3), and moreover has worse soundness than our DPP construction. Finally, a 1-query LPCP from [BIOW20] (which again can be turned into a DPP) requires the field size to be bigger than the circuit size and the proof size to grow quadratically with the circuit size.

Succinctness of group-based arguments. As discussed above, a compiler from [BCIOP22] converts a 1-query LPCP (implied by DPP) to a *designated-verifier* succinct non-interactive argument (dv-SNARG) based on any "linear-only" encryption, with the attractive feature that a proof consists of a single ciphertext for the linear-only encryption scheme; see Section 7.1. However, when applied in the *generic group model* (GGM), this compiler can only produce dv-SNARGs where the proof consists of two group elements and with inverse-polynomial soundness.

In a recent work, Arnon, Dujmovic and Ishai [ADI25] circumvented these limitations by introducing a new compiler from LPCPs to dv-SNARGs in the GGM. The compiler takes a q-query LPCP over $\mathbb{F}$ with soundness ε (satisfying some additional properties) and converts it into a dv-SNARG with similar soundness in which the proof consists of a *single* group element, a hash value, and ℓ additional bits for $\ell \approx q \log|\mathbb{F}|$. Instantiated with our small-field DPP construction, where soundness is amplified via repetition, this yields dv-SNARGs with τ bits of (non-reusable) soundness where $\ell \approx 2\tau$. A similar tradeoff can be obtained via a 2-query Hadamard-based LPCP, but with quadratic-size common reference string, as opposed to linear size required by our small-field DPP.

Settling for the weaker notion of interactive arguments with preprocessing, the compiler from Section 7.2 when combined with our big-field DPP construction has incomparable efficiency features. It is slightly worse than [ADI25] in terms of communication, but has much better verifier complexity (polylog(s) vs. $s^{\Omega(1)}$ for circuit size s) and better prover complexity. However, with 256-bit groups, it can currently provide only a poor level of soundness and cannot accommodate very large circuits due to the sub-optimal parameters of our big-field DPP construction.

[7] The constructions based on [Hås01] has the additional disadvantage of requiring the query to depend on the input, which limits its usefulness for cryptographic applications. However, a similar 1-query LPCP with an input-independent query can be constructed using more recent results on universal factor graphs [FJ12, ABH21].

1.3 Changes from Previous Versions of the Paper

We summarize the main changes we have made to the paper since the last posted version.

Section 5: More efficient embedding and corrected strong soundness analysis. We have augmented the embedding transformation in Section 5.1 to make it more efficient. In a nutshell, now instead of adding a random test query to the underlying FLPCP, we *fold it into an existing query.* We also introduce a simple *binary decomposition optimization* that allows us to tighten the bound of the resulting bounded FLPCP. These two optimizations yield better parameters when packing the queries of the underlying FLPCP.

We adopted the analysis of strong soundness in Section 5.3 according to the changes to Section 5.1. During this revision, we found out that the previous strong soundness analysis overlooked one case, which we have now corrected.

Section 7.2: Reflecting linear combination attacks. Our previous concrete instantiation of laconic arguments in the generic group model in Section 7.2 was inaccurate. The chosen concrete parameters did not account for the probability of finding a non-trivial linear combination of the queries that vanish. We have made this probability explicit in the statements in Section 5 (*hyperplane-evading probability*) and have updated the conclusions to concrete instantiations and how the resulting system compares to existing ones. Since there is ambiguity on what is meant by "128-bits of security" (as it relates to the adversary's running time vs. its advantage), we now explicitly state both the adversary's running time (bounded by 2^{75} calls to the group oracle with 256-bit groups or 2^{100} calls with 302-bit groups) and the soundness provided by our scheme. We also include some discussion on when this is a reasonable parameter setting. Thanks to the more efficient embedding we currently use, the concrete soundness parameters we report are still similar to (or better than) those reported in the previous version.

2 Technical Overview

In this section we give a more detailed overview of our results and the underlying techniques.

2.1 DPPs over Small Fields

We start by describing our constant-rate DPP over small fields. There are two variants of the construction. One works over arbitrarily small fields[8] and achieves a constant soundness error $s < 1$, independent of the field size. The other considers the soundness error as a function of the field size and gets soundness error $O(1/\sqrt{|\mathbb{F}|})$, which we also show is optimal for *any* DPP.

The high-level approach for both constructions takes three main steps:

1. **Outer FLPCP.** We construct a 3-query "constant-rate" fully-linear PCP (FLPCP) over small fields with accepting set $A \subseteq \mathbb{F}^3$ for our target langage $\mathcal{L}$.[9] Recall that an FLPCP [BBCGI19] is a generalization of a DPP where the verifier is allowed more than a single linear query.

[8] Including $\mathbb{F}_2$, though in this case perfect completeness cannot be achieved and one must settle for a constant completeness-soundness gap (see Appendix A).

[9] In the conference version of this paper [BHIRW24], we relied on the fact that $A \subseteq \mathbb{F}^3$ is *simple*. Here we replace this by a more generic approach.

2. **Inner DPP.** We construct a direct DPP "gadget" for membership of a given input $x \in \mathbb{F}^3$ in any set A. In fact, this gadget can be applied directly to any language, but with proof size that scales with the cardinality of the language.

3. **Composition.** We show how to *compose* any FLPCP for language $\mathcal{L}$ in which the accepting set is A, with a DPP for checking membership in A. The result is the desired DPP for $\mathcal{L}$.

We elaborate on each of these steps next.

A constant-rate FLPCP **over small fields.** As noted above, the starting point for our construction is an FLPCP for the target language $\mathcal{L}$, which we consider by default to be circuit satisfiability. There are existing constructions of FLPCPs that have constant rate in the sense of having linear proof length in the witness size [GGPR13, DFGK14], but these require the field size to grow (at least) linearly with the circuit size. Our approach follows their basic template but allows for replacing the underlying Reed-Solomon code with algebraic-geometric (AG) codes.

Actually, we give a simple general framework for constructing FLPCPs using any *multiplication code*, of which the Reed-Solomon code and AG codes are special cases. Loosely speaking, a multiplication code is a linear code E for which the pointwise product of any two codewords lies in some related linear code $E^{\star}$. The canonical example for such a code is the Reed-Solomon code (as the product of low degree polynomials is also a low degree polynomial), but AG codes also have this remarkably useful feature, over small (constant-size) fields.

Thus, we give a generic construction of a 3-query FLPCP from any multiplication code. This construction can be viewed as an abstraction of an FLPCP construction from [BBCGI19], which in turn simplifies a construction based on quadratic span programs implicit in [GGPR13]. In this construction, the verifier's decision algorithm is a quadratic polynomial in the 3 answers, which suffices for hardness of approximating the maximal number of satisfiable *quadratic* equations. Obtaining a similar result for *linear* equations, namely for the standard **MAXLIN** problem, will rely on the DPP construction we describe next.

Composing an FLPCP **with a** DPP. At this point we have a 3-query FLPCP and our goal is to reduce the number of queries to 1.

Inspired by PCP constructions [AS98, BGHSV06], our approach is based on *composition.* In a nutshell, rather than having our verifier actually perform the 3 queries, we would like the prover to append an auxiliary proof, proving that had the verifier made these queries it would accept.

In more detail, for any random string ρ of the original FLPCP verifier, which we refer to as the *outer* verifier, consider the corresponding query matrix $\boldsymbol{Q}_\rho \in \mathbb{F}^{3\times(n+m)}$ (where the rows correspond to the query vectors of the verifier) and accepting set $A_\rho \subseteq \mathbb{F}^3$. Then, given an instance $\boldsymbol{x}$ and FLPCP proof $\boldsymbol{\pi}$ for the outer system, for every string ρ we append a DPP proof $\boldsymbol{\pi}_\rho$ that $\boldsymbol{Q}_\rho \cdot (\boldsymbol{x}\|\boldsymbol{\pi}) \in A_\rho$. Now in order to verify that $\boldsymbol{x} \in \mathcal{L}$ we simply need to select ρ at random and run the "inner" DPP verifier on input $\boldsymbol{Q}_\rho \cdot (\boldsymbol{x}\|\boldsymbol{\pi})$ using the proof $\boldsymbol{\pi}_\rho$. The crucial observation is that linear access to $\boldsymbol{Q}_\rho \cdot (\boldsymbol{x}\|\boldsymbol{\pi})$ can be emulated by linear access to $(\boldsymbol{x}\|\boldsymbol{\pi})$, since the composition of linear functions is itself a linear function.

Thus, to construct a DPP for circuit satifiability over large inputs, it suffices to devise a DPP gadget for the specific accepting set arising from the 3-query FLPCP construction above.

The inner DPP **gadget.** The accepting set in the FLPCP described above turns out to be a simple one. Specifically, it is $A = \{(\alpha, \beta, \gamma) \in \mathbb{F}^3 : \gamma = \alpha \cdot \beta\}$.

In order to get a DPP with imperfect completeness, even over $\mathbb{F}_2$, we can guess a particular input $a \in A$ and check that $\langle r, (\alpha, \beta, \gamma) - a \rangle = 0$. This trivial DPP already achieves some positive gap between the completeness and soundness errors.

Still, we would like a DPP for A with perfect completeness, and where the soundness error vanishes with the field size. Our main approach for achieving this is via a "brute-force" DPP construction that can be applied to *every* language $A \subseteq \mathbb{F}^n$ with asymptotically optimal soundness. The "only" drawback of this DPP is that the proof size scales linearly with $|A|$ (in particular, exponentially with n). However, this is good enough for the crude asymptotic goals of our small-field construction, which allow a poly($|\mathbb{F}|$) multiplicative overhead in the proof size. We also present an alternative route that reduces this overhead by designing a better inner DPP that exploits the simple algebraic structure of A.

We construct the brute-force inner DPP in two steps:

1. We start with a direct construction of a DPP for a seemingly unrelated problem—checking whether a given input is a unit vector (i.e., zero in all but one coordinate which is a 1). The DPP for this "unit-vector" language $\mathbf{UV}$ does not use a proof and is very natural: sample a random set $\Lambda \subseteq \mathbb{F}$ of size t and a query vector $\boldsymbol{\lambda} \in \Lambda^n$, and accept if $\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda$. Completeness is immediate: if $\boldsymbol{x} = \boldsymbol{e}_i \in \mathbf{UV}$ (where $\boldsymbol{e}_i$ denotes the i^{th} standard basis vector), then $\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle = \lambda_i \in \Lambda$. Soundness is somewhat more tricky, where the key step is showing for any non-unit vector $\boldsymbol{x}$, it holds that $\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda$ with at most $O\left(\frac{t}{|\mathbb{F}|-t} + \frac{1}{t}\right)$ probability. Choosing $t = O(\sqrt{|\mathbb{F}|})$ we get the desired soundness error $O(1/\sqrt{|\mathbb{F}|})$. We also show that the soundness error obtained for this problem is optimal, up to constant factors. The rough idea is that for each possible query vector and accepting set of size t, a random input and proof are accepted with $\Omega(t/|\mathbb{F}|)$ probability, and a suitable linear combination of two random unit vectors and their associated proofs is accepted with $\Omega(1/t)$ probability.

2. Using the DPP for $\mathbf{UV}$, we construct a DPP for an arbitrary $A \subseteq \mathbb{F}^n$. For simplicity, we assume here that the vectors $\boldsymbol{x} \in A$ span the vector space $\mathbb{F}^n$, which is indeed the case for the specific $A \subseteq \mathbb{F}^3$ induced by our outer FLPCP. The general case is only slightly more involved. Let $\boldsymbol{M}$ be the matrix whose rows are the vectors $\boldsymbol{x}^\intercal \in A$. Consider first the case where $\boldsymbol{M}$ is a square matrix (and thus, invertible). Then, $\boldsymbol{x} \in A$ if and only if $\boldsymbol{x}^\intercal \boldsymbol{M}^{-1} \in \mathbf{UV}$, which implies a proofless reduction to $\mathbf{UV}$. In the more general case where the vectors $\boldsymbol{x} \in A$ spans $\mathbb{F}^n$ and $|A| = m > n$, the DPP requires a proof. The (honest) proofs are formed by adding $m - n$ columns to $\boldsymbol{M}$ that make the resulting matrix $\hat{\boldsymbol{M}}$ invertible. Parsing each row of $\hat{\boldsymbol{M}}$ as $(\boldsymbol{x}^\intercal \| \boldsymbol{\pi}^\intercal)$, the proof for input $\boldsymbol{x}$ is $\boldsymbol{\pi}$. On an implicit input $(\boldsymbol{x}^\intercal \| (\boldsymbol{\pi}^*)^\intercal)$, the DPP verifier checks that $(\boldsymbol{x}^\intercal \| (\boldsymbol{\pi}^*)^\intercal) \hat{\boldsymbol{M}}^{-1}$ is in $\mathbf{UV}$ using the DPP for $\mathbf{UV}$. Indeed, $\boldsymbol{x} \in A$ if and only if there is $\boldsymbol{\pi}$ such that $(\boldsymbol{x}^\intercal \mid \boldsymbol{\pi}^\intercal)$ is a row of $\hat{\boldsymbol{M}}$ (namely, $(\boldsymbol{x}^\intercal \| \boldsymbol{\pi}^\intercal) = \boldsymbol{e}_i^\intercal \hat{\boldsymbol{M}}$ for some unit vector $\boldsymbol{e}_i \in \mathbf{UV}$).

We summarize the ingredients of our small-field DPP construction in Fig. 1.

2.2 DPPs over Large Fields

In this section, we describe our DPPs over large fields. For this purpose, we provide transformations that transform any FLPCP to a promise DPP. Then plugging-in efficient LPCPs from the literature, which we adapt to the fully linear setting, we obtain our DPPs.

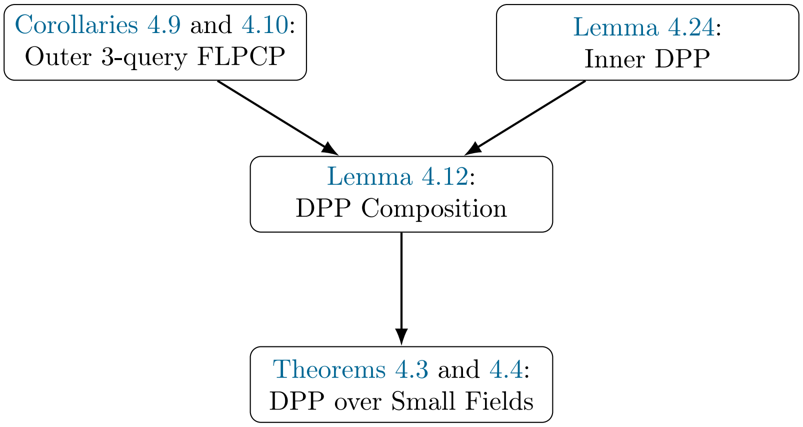

Figure 1: Roadmap to DPPs over small fields.

Our overall approach builds upon and extends previous transformations from [BCIOP22, BIOW20]:

1. **Construct a bounded** FLPCP**.** First, we construct an FLPCP where the magnitude of the answers is substantially smaller than the field size.
2. **Packing** FLPCP **queries into a single query.** Then, we randomly encode the bounded FLPCP queries into a single query.

In [BCIOP22], a classical (Boolean) PCP is used as the underlying bounded LPCP and packing of queries $\boldsymbol{Q} \in \mathbb{F}_p^{k \times m}$ is done by evaluating the linear function $E_{\boldsymbol{Q}}(\boldsymbol{w}) = \boldsymbol{w}^\intercal \boldsymbol{Q}$ at a random point $\boldsymbol{w} \in \mathbb{Z}^k$ from some appropriate rectangle. In [BIOW20], a bounded variant of the Hadamard FLPCP is given, and packing is done similarly, but using a certain multilinear polynomial $E_{\boldsymbol{Q}}(\boldsymbol{w})$ and a random point $\boldsymbol{w}$ over an appropriate rectangle. Compared to the [BCIOP22] transformation, the [BIOW20] transformation has certain concrete efficiency benefits and satisfies a notion known as *strong soundness*. However, it suffers from a quadratic loss in proof length, stemming from the reliance on the Hadamard FLPCP. The special feature of the Hadamard FLPCP on which the packing approach of [BIOW20] crucially relies is that the queries are generated by a *quadratic polynomial* in the randomness. In contrast, in known linear-sized FLPCPs, this query-generation degree is linear in the circuit size.

From FLPCP **to bounded** FLPCP**, generically.** We prove that *any* FLPCP over $\mathbb{F}_p$ can be embedded in a larger field $\mathbb{F}_{p'}$ to yield a bounded FLPCP. This is done by adding a random test query $\boldsymbol{y}$ from $\mathbb{F}_\gamma$, where $\gamma \in [p, p']$ is chosen to optimize the soundness error. The random test query is *folded* into one of the existing queries, say $\boldsymbol{q}_1$, by having $\boldsymbol{q}_1' = \boldsymbol{q}_1 + p\boldsymbol{u}$. This preserves the overall query complexity of the underlying FLPCP, which makes the next step of packing more efficient. The verifier then tests whether the answer $a_1 \mod p'$ falls within a small bound.

While the above embedding transformation is simple, proving soundness is challenging. The challenge stems from the fact that a malicious proof $\boldsymbol{\pi}^*$ may have "unbounded" entries over $\mathbb{F}_{p'}$ rather than $\mathbb{F}_p$ entries. We prove that there are essentially two options for such a malicious proof:

- $p\boldsymbol{\pi}^*$ is *close* to a proof over $\mathbb{F}_p$ up to relatively small p' fractions. That is $p\pi_i^* = \frac{dp'+e}{r}$, where $r \in \mathbb{F}_\gamma$ and e is relatively small. In this case, we show that an answer either 1) exceeds the specified bound, 2) behaves as a random answer $\mod p$, 3) behaves as an answer corresponding to some fixed proof $\boldsymbol{\pi}$ over $\mathbb{F}_p$. In the first case, the verifier rejects due to its bound test, and in the latter two cases, it rejects due to the soundness of the underlying FLPCP.

- It can be *far* from any proof over $\mathbb{F}_p$ (more generally, far from fractions as above), in which case we prove that it will be caught by the added random bound test with high probability.

Fulfilling this high-level approach turns out to require a non-trivial analysis and characterization of closeness to p'-fractions. We believe that our analysis is not tight, and improving it would directly yield an improvement in the DPP's concrete efficiency.

We note that the above description implicitly assumes that unlike the proof $\boldsymbol{\pi}^* \in \mathbb{F}_{p'}^m$, the input $\boldsymbol{x}$ is promised to be in $\{0,1\}^n$ (or $\mathbb{F}_p^n$), where the original language resides. Hence, we only get a promise DPP. By increasing the proof by an additive factor of n^2 and adding two queries, we can turn the corresponding bounded FLPCP to a plain (non-promise) one. This is done using a tensor test akin to the one in the Hadamard FLPCP [BIOW20].

For more details on the above embedding transformation, see Section 5.1.

Packing. For packing of 2-query FLPCPs we use the packing transformation of [BIOW20]. We also generalize the packing transformation of [BCIOP22], from the Boolean setting to the general bounded setting, which yields favorable parameters when packing k-query FLPCPs for $k \geq 3$. We refer the reader to Section 5.2 for details.

3 Preliminaries

Throughout this section we use bold uppercase letters to denote matrices (e.g., $\boldsymbol{M}$, $\boldsymbol{Q}$) and bold lowercase letters to denote vectors (e.g., $\boldsymbol{x}$, $\boldsymbol{w}$). For vectors $\boldsymbol{x}, \boldsymbol{y}$, we write $(\boldsymbol{x} \| \boldsymbol{y})$ to denote the vector obtained from their concatenation.

3.1 Fully Linear PCP

Definition 3.1 (Fully Linear PCP). *A* fully linear PCP *(*FLPCP*), parameterized by a field $\mathbb{F}$, input length n, query complexity q and proof length m, consists of a probabilistic algorithm $\mathcal{V}$ that outputs a query matrix $\boldsymbol{Q} \in \mathbb{F}^{q \times (n+m)}$ and a set $A \subseteq \mathbb{F}^q$. An* FLPCP *for $\mathcal{L} \subseteq \mathbb{F}^n$ satisfies the following properties:*

- *(Completeness:) for every $\boldsymbol{x} \in \mathcal{L}$ there exists $\boldsymbol{\pi} \in \mathbb{F}^m$:*

$$\Pr_{(\boldsymbol{Q},A) \leftarrow \mathcal{V}} \left[\boldsymbol{Q} \cdot (\boldsymbol{x} \| \boldsymbol{\pi}) \in A \right] \geq c,$$

  where $c \in [0,1]$ is called the completeness parameter*.*

- *(Soundness:) for every $\boldsymbol{x} \notin \mathcal{L}$ and all $\boldsymbol{\pi}^* \in \mathbb{F}^m$*

$$\Pr_{(\boldsymbol{Q},A) \leftarrow \mathcal{V}} \left[\boldsymbol{Q} \cdot (\boldsymbol{x} \| \boldsymbol{\pi}^*) \in A \right] \leq s,$$

  where $s \in [0,1]$ is called the soundness error*.*

Thus, a DPP corresponds to a single-query (i.e., $q = 1$) FLPCP. As usual, by default we assume perfect completeness (i.e., $c = 1$). We define the *randomness complexity* of an FLPCP as the (base 2) logarithm of the support size of the verifier's randomness.

Remark 3.2 (Prover and Verifier Complexity)**.** *The above definition of fully linear PCPs (Definition 3.1) does not refer to the computational complexity of the prover, who generates the proof $\boldsymbol{\pi}$ from the input $\boldsymbol{x}$, or the verifier who generates $\boldsymbol{Q}$ and A. Indeed, unlike classical* PCP*s (and similar to* PCP*s of* proximity *[DR06, BGHSV06]), the notion of* FLPCP *(and* DPP*) is meaningful even when all of the algorithms involved are computationally unbounded.*

Eventually, our DPP *constructions will all be computationally efficient in the sense that the running time of both the prover and the verifier is polynomial in the proof length. This includes an implicit representation of the accepting set A when the field size is super-polynomial. Moreover, as we'll show below the existence of efficient* DPP*s for all languages $\mathcal{L} \in P$ generically implies efficient* DPP*s for all $\mathcal{L} \in NP$, where the proof $\boldsymbol{\pi}$ is generated by a polynomial-time prover given the input $\boldsymbol{x}$ and a witness $\boldsymbol{w}$, and the query $\boldsymbol{q}$ is generated by a polynomial-time verifier.*

From deterministic to non-deterministic. As already observed in [BBCGI19], for FLPCPs (and in particular DPPs), constructions for deterministic languages automatically yield a corresponding construction for their non-deterministic counterpart.

Proposition 3.3 (FLPCP for Non-Deterministic Languages)**.** *Let $\mathcal{L} \subseteq \mathbb{F}^n \times \mathbb{F}^m$ be a pair language (i.e., a language consisting of pairs $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbb{F}^{n+m}$. Suppose $\mathcal{L}$ has an* FLPCP *with query complexity $q = q(n, m)$, proof length $m' = m'(n, m)$ soundness error $\delta = \delta(n, m)$ and randomness complexity $r = r(n, m)$. Then, the language $\mathcal{L}' = \{\boldsymbol{x} \in \mathbb{F}^n \mid \exists \boldsymbol{w} \in \mathbb{F}^m : (\boldsymbol{x}, \boldsymbol{w}) \in \mathcal{L}\}$ has a* FLPCP *with query complexity q, proof length $m + m'$, soundness error δ and randomness complexity r.*

Proof. Given an input $\boldsymbol{x} \in \mathcal{L}'$, let $\boldsymbol{w}$ be such that $(\boldsymbol{x}, \boldsymbol{w}) \in \mathcal{L}$ and let $\boldsymbol{\pi}$ be the FLPCP proof. The FLPCP proof that $\boldsymbol{x} \in \mathcal{L}'$ is simply $(\boldsymbol{w} \| \boldsymbol{\pi})$ and the proof is verified by running the underlying FLPCP verifier using $(\boldsymbol{x} \| \boldsymbol{w})$ as the input and $\boldsymbol{\pi}$ as the proof. Completeness and soundness follow immediately from the completeness and soundness of the underlying FLPCP. □

Composition with linear transformation. The following simple proposition shows how to transform an FLPCP for a given language to any linear transformation of that language.

Proposition 3.4 (FLPCP Composition with Linear Transformations)**.** *Let $\mathbb{F}$ be a field, $n' = n'(n)$, and $\boldsymbol{M} \in \mathbb{F}^{n' \times n}$ be a matrix. Suppose the language $\mathcal{L} \subseteq \mathbb{F}^{n'}$ has an* FLPCP *with query complexity $d = d(n')$, proof length $m = m(n')$, soundness error $\delta = \delta(n')$, and randomness complexity $r = r(n')$. Then, the language $\{\boldsymbol{x} \in \mathbb{F}^n : \boldsymbol{M} \cdot \boldsymbol{x} \in \mathcal{L}\}$ has an* FLPCP *with query complexity $d(n')$, proof length $m(n')$, soundness error $\delta(n')$, and randomness complexity $r(n')$.*

Proof. Let $\boldsymbol{x} \in \mathbb{F}^n$ be an input. We construct a proof showing that $\boldsymbol{y} = \boldsymbol{M}\boldsymbol{x} \in \mathcal{L}$, where $\boldsymbol{M} \in \mathbb{F}^{n' \times n}$ is the designated matrix. The FLPCP proof $\boldsymbol{\pi}$ is the FLPCP proof that $\boldsymbol{y} \in \mathcal{L}$. To verify the proof, the verifier generates the query $\boldsymbol{Q} = (\boldsymbol{Q}_{\mathsf{inp}}, \boldsymbol{Q}_{\mathsf{prf}}) \in \mathbb{F}^{d \times n'} \times \mathbb{F}^{d \times m}$ and corresponding accepting set $A \subseteq \mathbb{F}^d$. Given access to $\boldsymbol{x}$ and $\boldsymbol{\pi}$, it now accepts if:

$$\big(\boldsymbol{Q}_{\mathsf{inp}}\boldsymbol{M} \| \boldsymbol{Q}_{\mathsf{prf}}\big)(\boldsymbol{x} \| \boldsymbol{\pi}) \in A,$$

which is indeed a linear query to $\boldsymbol{x} \| \boldsymbol{\pi}$. Completeness and soundness follow from the fact that

$$\begin{aligned}
\big(\boldsymbol{Q}_{\mathsf{inp}}\boldsymbol{M} \| \boldsymbol{Q}_{\mathsf{prf}}\big)(\boldsymbol{x} \| \boldsymbol{\pi}) &= \boldsymbol{Q}_{\mathsf{inp}}\boldsymbol{M}\boldsymbol{x} + \boldsymbol{Q}_{\mathsf{prf}}\boldsymbol{\pi} \\
&= \boldsymbol{Q}_{\mathsf{inp}}\boldsymbol{y} + \boldsymbol{Q}_{\mathsf{prf}}\boldsymbol{\pi} \\
&= (\boldsymbol{Q}_{\mathsf{inp}} \| \boldsymbol{Q}_{\mathsf{prf}})(\boldsymbol{y} \| \boldsymbol{\pi}). \qquad \square
\end{aligned}$$

3.2 Codes

An *error-correcting code*, over an alphabet Σ is an injective mapping $C \colon \Sigma^k \to \Sigma^n$. The *rate* of the code is defined as k/n and the *minimal distance* is defined as the minimal relative Hamming distance of any two distinct codewords. The parameter n, viewed as a function of k, is called the *block length.*

We say that a code $E \colon \mathbb{F}^k \to \mathbb{F}^n$, where the alphabet $\mathbb{F}$ is a finite field, is *linear* if E is a linear map over $\mathbb{F}$. We say that E is systematic if for every $\boldsymbol{m} \in \mathbb{F}^k$, the first k entries of $E(m)$ are equal to $\boldsymbol{m}$.

Uniform code families. Let $n = n(k) \in \mathbb{N}$ and let $\mathbb{F} = (\mathbb{F}_k)_{k \in \mathbb{N}}$ be an ensemble of finite fields. We will sometimes consider code families $\big(E_k : (\mathbb{F}_k)^k \to (\mathbb{F}_k)^{n(k)}\big)_{k \in \mathbb{N}}$. We say that a code family has rate $r = r(k)$ (resp., minimal distance $\delta = \delta(k)$) if E_k has rate $\geq r(k)$ (resp., minimal distance $\geq \delta(k)$), for every $k \in \mathbb{N}$.

We say that the code family is *efficiently computable* if there exists a polynomial-time algorithm that given $m \in (\mathbb{F}_k)^k$ outputs $E_k(m)$. We say that the code has an *efficient implicit encoder* if there exists a polynomial-time algorithm that given $m \in (\mathbb{F}_k)^k$ and an index $i \in [n]$, outputs the i^{th} entry of $E_k(m)$

Finite field representation and operations. For every finite field $\mathbb{F}$, the elements of $\mathbb{F}$ have a canonical representation of size $O(\log(|\mathbb{F}|)$ and there are polylog$(|\mathbb{F}|)$ size Boolean circuits that compute the field operations (addition, subtraction, multiplication, inversion, generating the additive and multiplicative identity elements and sampling of random elements).

3.2.1 Multiplication codes

If $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}^n$ are vectors, we use $\boldsymbol{x} \star \boldsymbol{y}$ to denote their pointwise product. Namely, the vector $\boldsymbol{z} \in \mathbb{F}^n$ such that $z_i = x_i \cdot y_i$, for every $i \in [n]$.

Definition 3.5 (Multiplication Code). *Let $E \colon \mathbb{F}^k \to \mathbb{F}^n$ and $E^\star \colon \mathbb{F}^{k^*} \to \mathbb{F}^n$ be linear codes (with the same block length). We say that $E \colon \mathbb{F}^k \to \mathbb{F}^n$ is a* multiplication code with respect to the square code *$E^\star$ if for every $\boldsymbol{x}, \boldsymbol{x'} \in E$ it holds that $\boldsymbol{x} \star \boldsymbol{x'} \in E^\star$. In such a case, we say that $E^\star$ is the* square code *(of E).*

As the multiplication property only refers to the behaviour of the *image* of the code, and the codes are linear, we can assume without loss of generality that they are also systematic (by choosing the encoding function appropriately).

The canonical example for a multiplication code is the ubiquitous Reed-Solomon code. Here the codewords correspond to degree $k-1$ polynomials and their product is a degree $2(k-1)$ polynomial. However, the Reed-Solomon code requires a large alphabet. Fortunately, using the machinery of AG codes, multiplication codes (with constant rate and distance) are known over any field.

The following theorem builds on a construction of asymptotically-efficient algebraic-geometric codes due to Garcia and Stichtenoth [GS95] and Bassa, Beelen, Garcia, and Stichtenoth [BBGS14], which applies to infinitely many fields. See [VH97, GS01] and in particular [Ran13, Theorem 17] for the furthermore part of the following theorem.

Theorem 3.6 (AG Codes)**.** *There exists a fixed constant $\delta^\star > 0$ such for every finite field $\mathbb{F}$ of order $q > 9$ where q is square, there exists $\rho_q \in (0, 1)$ such that the following holds. There is a polynomial-time constructible family of linear codes $C_n \subseteq \mathbb{F}^n$ such that for all sufficiently large n, the rate of C_n is least ρ_q and the relative distance of the square code $C_n^\star$ (spanned by pointwise products of pairs of codewords from C_n) is at least δ^*.*

Furthermore, there is an infinite sequence $\{n_i\}_{i\in\mathbb{N}}$, where $\frac{n_{i+1}}{n_i} \in O(\sqrt{q})$, and a polynomial-time constructible family of linear codes $\{C_{n_i}\}_{i\in\mathbb{N}} \subseteq \mathbb{F}^{n_i}$ such that for all $i \in \mathbb{N}$, the rate of C_{n_i} is at least $\rho_q \in \Omega(1/\sqrt{q})$ and the relative distance of the square code $C_{n_i}^\star$ is at least $\delta_q^\star \in 1 - O(1/\sqrt{q})$.

The limitations on the choice of q and the density of the sequence n_i in the "furthermore" part will be dealt with using concatenation and padding, which are good enough for meeting our efficiency goals. Similar limitations were also encountered in applications of AG codes to arithmetic secret sharing [CXY20].

3.3 Exponential Time Hypothesis

The exponential time hypothesis (ETH), first formulated by Impagliazzo and Paturi [IP01], states that satisfiability of 3CNF Boolean formulas on n variables requires at least $2^{\varepsilon n}$ time for some $\varepsilon > 0$. The sparsification lemma of Impagliazzo, Paturi, and Zane [IPZ01] shows that to prove the ETH, it suffices to consider 3CNF formula where each variable occurs in at most a constant number of clauses. More precisely, we have the following form of the ETH.

Hypothesis 3.7 (Exponential Time Hypothesis (ETH) [IP01, IPZ01])**.** *There exist constants $C > 1$ and $\varepsilon \in (0, 1)$ such that any algorithm that decides the satisfiability of 3CNF Boolean formulas on n variables with at most Cn clauses requires at least $2^{\varepsilon n}$ time.*

A related and stronger hypothesis (which we will not use) is the gap-exponential time hypothesis (gap-ETH), formulated by Dinur [Din16] and Manurangsi and Raghavendra [MR17], which states that there exist constants $C > 1$ and $\varepsilon \in (0, 1)$ such that then no $2^{\varepsilon n}$-time algorithm can distinguish if a given 3CNF Boolean formulas on n variables with at most Cn clauses is satisfiable or every assignment violates at least 0.01 fraction of clauses.

3.4 MAXLIN and Its Variants

The DPPs, described in this paper, are closely related to two classical NP-hard approximation problems **MAXLIN** [Hås01, HKLT19] and the nearest codeword problem (**NCP**) [KPV14]. In both these problems, the instances are of the form $(\boldsymbol{A}, \boldsymbol{b})$ where $\boldsymbol{A} \in \mathbb{F}^{m\times n}$ is an $m \times n$ matrix with entries from a finite field $\mathbb{F}$ and $\boldsymbol{b} \in \mathbb{F}^m$ is a vector. The goal is to find the "best" vector $\boldsymbol{x} \in \mathbb{F}^m$ that "satisfies" the set of m linear equations $\boldsymbol{Ax} = \boldsymbol{b}$. **MAXLIN** refers to the approximation problem of finding a vector $\boldsymbol{x} \in \mathbb{F}^n$ that satisfies as many linear equations as possible, while **NCP** refers to the dual approximation problem of finding a vector $\boldsymbol{x} \in \mathbb{F}^n$ that minimizes the number of linear equations violated. It will be convenient to work with the related promise problem $\text{gap}_{c,s}$-**MAXLIN**($\mathbb{F}$) for $1/|\mathbb{F}| < s < c < 1$ defined as follows: the YES instances are the set of $(\boldsymbol{A}, \boldsymbol{b})$ pairs such that there exists an $\boldsymbol{x} \in \mathbb{F}^n$ that satisfies at least cm linear equations while the NO instances are the set of $(\boldsymbol{A}, \boldsymbol{b})$ pairs such that every $\boldsymbol{x} \in \mathbb{F}^n$ satisfies at most sm linear equations.

4 DPPs over Small Fields

In this section we construct general-purpose DPPs for verifying that there exists w such that $C(x, w) = 0$. Throughout this section, the field should be thought of as relatively small, since the proof length grows polynomially with the field size.

It will be convenient for us to work with a specific NP-complete language called R1CS.

Definition 4.1 (R1CS)**.** *The* $\mathsf{R1CS}^{\mathbb{F}}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *problem, parameterized by a finite field* $\mathbb{F}$ *and matrices* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k\times(n+m)}$*, consists of all vectors* $\boldsymbol{x} \in \mathbb{F}^n$*, for which there exists* $\boldsymbol{z} \in \mathbb{F}^m$ *such that* $(\boldsymbol{A}\boldsymbol{z}') \star (\boldsymbol{B}\boldsymbol{z}') = \boldsymbol{C}\boldsymbol{z}'$*, where* $\boldsymbol{z}' = (\boldsymbol{x}\|\boldsymbol{z})$*.*[10]

When the field $\mathbb{F}$ is clear from the context, we omit it from the notation.

The language R1CS is closely related to satisfiability for a given arithmetic circuit C. Loosely speaking, one can consider the witness $\boldsymbol{z}$ as being the values of all the multiplication gates and the R1CS relation allows one to check that they were computed correctly. For details, see Thaler's survey [Tha22, Section 8.4].

As a matter of fact, rather than working directly with R1CS, we can work with a deterministic version of the latter. Indeed, recall that DPPs for deterministic languages imply DPPs for their non-deterministic counterparts (see Proposition 3.3). Thus, throughout this section, we work with the deterministic variant of R1CS. For matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k\times n}$ define $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}} = \big\{\boldsymbol{x} \in \mathbb{F}^n : (\boldsymbol{A}\boldsymbol{x}) \star (\boldsymbol{B}\boldsymbol{x}) = \boldsymbol{C}\boldsymbol{x}\big\}$. While we phrase our results as DPPs (and FLPCPs) for dR1CS, these immediately yield corresponding DPPs for R1CS via Proposition 3.3. When translating back to the language of arithmetic circuits, the parameter k corresponds to the number of multiplication gates (and is upper bounded by the circuit size).

We proceed to describe the main theorems proved in this section, which are DPPs in different settings.

Constant-rate DPP **over** $\mathbb{F}_2$**.** The first construction is a specific construction for the field $\mathbb{F}_2$ (i.e., the two element field). This DPP has imperfect completeness, which is inherent over this field (see Appendix A).

Theorem 4.2 (Constant-Rate DPP over $\mathbb{F}_2$)**.** *There exists* $\varepsilon > 0$ *such that for every* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}_2^{k\times n}$*, the language* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *has a* DPP *over* $\mathbb{F}_2$ *with* $O(k)$ *proof length, completeness* $5/8$*, soundness error* $5/8 - \varepsilon$*, and randomness complexity* $\log k + O(1)$*.*

Constant-rate DPP **with small accepting set.** The next construction works over any field of odd characteristic achieving a constant soundness error, and with a linear length proof for constant size fields. In addition, the accepting set is of size 2, which is optimal for DPPs with perfect completeness (see Appendix A).

Theorem 4.3 (Constant-Rate DPP with Constant Soundness)**.** *There is a constant* $\frac{2}{3} < \varepsilon < 1$ *such that for every finite field* $\mathbb{F}$ *of odd characteristic and* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k\times n}$*, the language* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$

[10] Recall that we use $\boldsymbol{a} \star \boldsymbol{b}$ to denote the pointwise product of two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{F}^k$.

has a DPP *over* $\mathbb{F}$ *with proof length* $k \cdot \mathrm{poly}(|\mathbb{F}|)$*, soundness error* ε*, and randomness complexity* $\log(k) + \mathrm{poly}(|\mathbb{F}|)$*. Furthermore, the verifier's accepting set is* $A \subseteq \{0, 1\}$*. For* $\mathbb{F}_{2^k}$ *where* $k > 1$*, the same holds, except that* $A \subseteq \{0, 1, c\}$ *for some* $c \in \mathbb{F} \setminus \{0, 1\}$*.*

Constant-rate DPP **with optimal soundness.** The next result optimizes the soundness error, obtaining $O(1/\sqrt{|\mathbb{F}|})$ soundness error.

Theorem 4.4 (Optimal-Soundness DPP)**.** *There exists* $\varepsilon^*_q \in O(1/\sqrt{q})$ *such that the following holds. For every finite field* $\mathbb{F}$ *of order* $q > 2$*, all sufficiently large* n*, and matrices* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$*, the language* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *has a* DPP *over* $\mathbb{F}$ *with* $k \cdot \mathrm{poly}(q)$ *proof length, soundness error* ε^*_q *and randomness complexity* $\log(k) + \mathrm{poly}(q)$*.*

In Section 4.4 we will show that the $O(1/\sqrt{q})$ soundness error obtained by Theorem 4.4 is optimal (up to a constant factor).

Section outline. We start, in Section 4.1, by giving a construction of a 3-query FLPCP for $\mathsf{dR1CS}$. Then, in Section 4.2 we show a general composition theorem, allowing us to reduce the number of queries in an FLPCP to 1 (thereby making it a DPP), by composing it with an inner DPP gadget. Then, in Section 4.3 we describe constructions of inner gadgets which, in combination with the prior steps, imply Theorems 4.2 to 4.4. Lastly, in Section 4.4 we show an asymptotically tight lower bound on the soundness error of DPPs as a function of the field size.

4.1 FLPCP for dR1CS

We start by constructing efficient FLPCPs for $\mathsf{dR1CS}$. The first result is a 4-query FLPCP.

Theorem 4.5 (4-Query FLPCP)**.** *Let* $\mathbb{F}$ *be a finite field and* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$*. Suppose that* $E \colon \mathbb{F}^k \to \mathbb{F}^\ell$ *is a multiplication code with respect to the square code* $E^\star : \mathbb{F}^{k^\star} \to \mathbb{F}^\ell$*, where* $E^\star$ *has minimal distance* $\delta^\star$*. Then,* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *has an* FLPCP *over* $\mathbb{F}$ *with* 4 *queries, proof length* $k^\star$*, soundness error* $1 - \delta^\star$ *and randomness complexity* $\log_2(\ell)$*. Furthermore, the verifier's accepting set is always* $\{(a, b, c, d) : a \cdot b = c \text{ and } d = 0\}$*.*

Proof. Let $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$ and $\boldsymbol{x} \in \mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$. Let $E \colon \mathbb{F}^k \to \mathbb{F}^\ell$ be a multiplication code with respect to the square code $E^\star \colon \mathbb{F}^{k^\star} \to \mathbb{F}^\ell$ and let $\delta^\star$ denote the distance of $E^\star$. Recall that both E and $E^\star$ are linear and systematic codes. Since E is a multiplication code with respect to $E^\star$, it holds that $E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx}) \in E^\star$, and in particular there exists $\boldsymbol{w} \in \mathbb{F}^{k^\star}$ such that $E^\star(\boldsymbol{w}) = E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})$. The FLPCP proof string is simply $\boldsymbol{w}$.

At a high level, the verification consists of two tests. The first test checks that $\boldsymbol{w}$ was computed correctly from $\boldsymbol{x}$. Since $E^\star(\boldsymbol{w})$ should be equal to $E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})$, and $E^\star$ is a multiplication code, we can test this by checking that $E^\star(\boldsymbol{w})_i = E(\boldsymbol{Ax})_i \cdot E(\boldsymbol{Bx})_i$, for a random index $i \in [\ell]$. Once we know that $\boldsymbol{w}$ was computed correctly, we just need to check that its first k entries, denoted by $\boldsymbol{w}_{\leq k}$ are equal to $\boldsymbol{Cx}$. The latter can be done by taking an inner product with a random vector. However, this is wasteful in terms of randomness (indeed, that test requires $\Omega(k \cdot \log(|\mathbb{F}|))$ randomness). Instead, we check that the encoding of $\boldsymbol{Cx} - \boldsymbol{w}_{\leq k}$ (padded with a sufficient number of zeros) under $E^\star$, is equal to 0 at a random coordinate i. The distance of $E^\star$ guarantees that if the $\boldsymbol{Cx} \neq \boldsymbol{w}_{\leq k}$ then a random coordinate of this codeword is non-zero with high probability. Also, to further save on randomness, we use the same coordinate i for both tests.

Thus, given linear access to the concatenation of the input $\boldsymbol{x}$ and the alleged proof string $\boldsymbol{w}$, the verifier performs the following tests:

1. Sample a random index $i \in [\ell]$ and check that $E^{\star}(\boldsymbol{w})_i = E(\boldsymbol{Ax})_i \cdot E(\boldsymbol{Bx})_i$. The value $E^{\star}(\boldsymbol{w})_i$ is computed as a single linear query to $\boldsymbol{w}$ and each of the values $E(\boldsymbol{Ax})_i$ and $E(\boldsymbol{Bx})_i$ are computed by a linear query to $\boldsymbol{x}$.

2. $E^{*}(\boldsymbol{Cx} - \boldsymbol{w}_{\leq k} \| 0^{k^{\star}-k})_i = 0$ (recall that $\boldsymbol{w}_{\leq k}$ denotes the first k entries of $\boldsymbol{w} \in \mathbb{F}^{k^{*}}$). This consists of a single linear query to $(\boldsymbol{x}\|\boldsymbol{w})$.

Completeness. Suppose $\boldsymbol{x} \in \mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$, that is, $(\boldsymbol{Ax}) \star (\boldsymbol{Bx}) = \boldsymbol{Cx}$. Let $\boldsymbol{w}$ be the proof vector as described above.

By construction $E^{\star}(\boldsymbol{w}) = E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})$. Thus, for every $i \in [\ell]$ it holds that $E^{\star}(\boldsymbol{w})_i = E(\boldsymbol{Ax})_i \cdot E(\boldsymbol{Bx})_i$ and so the verifier's first test passes with probability 1.

Since $E^{\star}$ is a multiplication code (and systematic), $\boldsymbol{w}_{\leq k}$ (the k-length prefix of $\boldsymbol{w}$) is equal to $(\boldsymbol{Ax}) \star (\boldsymbol{Bx})$, which in turn is equal to $\boldsymbol{Cx}$. Thus, we have that $\boldsymbol{w}_{\leq k} = \boldsymbol{Cx}$ and so the verifier's second test always passes.

Soundness. Suppose $\boldsymbol{x} \notin \mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$, that is, $(\boldsymbol{Ax}) \star (\boldsymbol{Bx}) \neq \boldsymbol{Cx}$ and fix an alleged proof string $\boldsymbol{w}$. We show that the verifier rejects with probability at least $\delta^{\star}$.

Suppose first that $E^{\star}(\boldsymbol{w}) \neq E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})$. Since E is a multiplication code with respect to $E^{\star}$, there exists $\boldsymbol{c}^{\star} \in E^{\star}$ such that $\boldsymbol{c}^{\star} = E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})$. Since $\boldsymbol{c}^{\star} \neq E^{\star}(\boldsymbol{w})$, by the distance of $E^{\star}$, with probability at least $\delta^{\star}$ over $i \in [\ell]$, it holds that:

$$E^{\star}(\boldsymbol{w})_i \neq c_i^{\star} = \big(E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})\big)_i = E(\boldsymbol{Ax})_i \cdot E(\boldsymbol{Bx})_i,$$

and so the verifier's first test rejects with probability at least $\delta^{\star}$.

Thus, we may assume that $E^{\star}(\boldsymbol{w}) = E(\boldsymbol{Ax}) \star E(\boldsymbol{Bx})$. As the codes are systematic, this means that $\boldsymbol{w}_{\leq k} = (\boldsymbol{Ax}) \star (\boldsymbol{Bx})$, where $\boldsymbol{w}_{\leq k}$ is the k-length prefix of $\boldsymbol{w}$. Since $(\boldsymbol{Ax}) \star (\boldsymbol{Bx}) \neq \boldsymbol{Cx}$, we conclude that $\boldsymbol{w}_{\leq k} \neq \boldsymbol{Cx}$ and so, by the distance of E^{*}, the verifier rejects in its second test with probability $\delta^{\star}$. □

The FLPCP of Theorem 4.5 has 4 queries. This is good enough for our main small-field DPP theorem, since the inner DPP we present next can efficiently accommodate any language with constant input length. However, for other applications it will be useful to eliminate one of the queries. This can be done at a very mild cost in soundness error and randomness complexity as follows.

Theorem 4.6 (3-Query FLPCP)**.** *Let $\mathbb{F}$ be a finite field and $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$. Suppose that $E \colon \mathbb{F}^k \to \mathbb{F}^{\ell}$ is a multiplication code with respect to the square code $E^{\star} \colon \mathbb{F}^{k^{\star}} \to \mathbb{F}^{\ell}$, where $E^{\star}$ has minimal distance $\delta^{\star}$. Then, for every parameter $\varepsilon \geq 1/|\mathbb{F}|$, the language $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ has an* FLPCP *over $\mathbb{F}$ with 3 queries, proof length $k^{\star}$, soundness error $1 - \delta^{\star} + \varepsilon$ and randomness complexity $\log_2(\ell) + \log(1/\varepsilon)$. Furthermore, the verifier's accepting set is always $\{(\alpha, \beta, \gamma) : \alpha \cdot \beta = \gamma\}$.*

Proof. The verifier generates the four queries $\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3, \boldsymbol{q}_4$ of the verifier of Theorem 4.5 and also chooses a random field element λ from a fixed subset of $\mathbb{F}$ of size $1/\varepsilon$. Given an input $\boldsymbol{x}$ and proof $\boldsymbol{\pi}$, it accepts if $\langle \boldsymbol{q}_1, (\boldsymbol{x}\|\boldsymbol{\pi})\rangle \cdot \langle \boldsymbol{q}_2, (\boldsymbol{x}\|\boldsymbol{\pi})\rangle = \langle \boldsymbol{q}_3 + \lambda \cdot \boldsymbol{q}_4, (\boldsymbol{x}\|\boldsymbol{\pi})\rangle$.

Completeness. Let $\boldsymbol{x} \in \mathcal{L}$ and let $\boldsymbol{\pi}$ be the proof guaranteed by Theorem 4.5 with respect to input $\boldsymbol{x}$. Since the original FLPCP verifier checks that the answer to the 4th query is 0, by perfect completeness we have that $\langle \boldsymbol{q}_4, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle = 0$. Since it checks that the product of the first and second answers is equal to the third, we also have that $\langle \boldsymbol{q}_1, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \cdot \langle \boldsymbol{q}_2, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle = \langle \boldsymbol{q}_3, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle$, and so the verifier always accepts.

Soundness. Let $\boldsymbol{x} \notin \mathcal{L}$ and fix a proof vector $\boldsymbol{\pi}$. By the soundness of the FLPCP, with probability $\delta^\star$ we have that either (1) $\langle \boldsymbol{q}_1, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \cdot \langle \boldsymbol{q}_2, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \neq \langle \boldsymbol{q}_3, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle$ or (2) $\langle \boldsymbol{q}_4, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \neq 0$. Suppose first that $\langle \boldsymbol{q}_4, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle = 0$. In such a case:

$$\langle \boldsymbol{q}_1, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \cdot \langle \boldsymbol{q}_2, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle \neq \langle \boldsymbol{q}_3, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle = \langle \boldsymbol{q}_3 + \lambda \cdot \boldsymbol{q}_4, (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle ,$$

and so the verifier rejects. On the other hand, if $\langle \boldsymbol{q}_4, \boldsymbol{x} \| \boldsymbol{\pi} \rangle \neq 0$ then $\langle \boldsymbol{q}_3 + \lambda \cdot \boldsymbol{q}_4, \boldsymbol{x} \| \boldsymbol{\pi} \rangle = \langle \boldsymbol{q}_3, \boldsymbol{x} \| \boldsymbol{\pi} \rangle + \lambda \cdot \langle \boldsymbol{q}_4, \boldsymbol{x} \| \boldsymbol{\pi} \rangle$ is a uniformly random value (independent of $\boldsymbol{q}_1, \boldsymbol{q}_2$) in a set of size $1/\varepsilon$, and so the probability that it is equal to $\langle \boldsymbol{q}_1, \boldsymbol{x} \| \boldsymbol{\pi} \rangle \cdot \langle \boldsymbol{q}_2, \boldsymbol{x} \| \boldsymbol{\pi} \rangle$ is at most ε. □

Remark 4.7 (Efficiency and Uniformity). *We remark that the prover and verifier in Theorems 4.5 and 4.6 can be implemented as size* $\mathrm{poly}(n, m, k)$ *arithmetic circuits over* $\mathbb{F}$ *that use oracle gates to an implicit encoding function of* E*.*

We now derive several FLPCPs in various settings, based on different multiplication codes. First, recall that for every parameter $\ell \in [k, |\mathbb{F}|]$, there is a Reed-Solomon code, which is a multiplication code with rate k/ℓ (resp., $2k/\ell$) and distance $1 - \frac{k+1}{\ell}$ (resp., $1 - \frac{2k+1}{\ell}$) for the base code (resp., square code). Using this code with Theorem 4.6 (and $\varepsilon = 1/\ell$) we get the following corollary.

Corollary 4.8 (Reed-Solomon-Based FLPCP). *Let* $\mathbb{F}$ *be a finite field and* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$*. Then, for every parameter* $\ell \in [k, |\mathbb{F}|]$*, the language* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *has an* FLPCP *over* $\mathbb{F}$ *with* 3 *queries, proof length* $2k$*, soundness error* $(2k+2)/\ell$*, and randomness complexity* $2\log(\ell)$*. Furthermore, the verifier's accepting set is always* $\{(\alpha, \beta, \gamma) : \alpha \cdot \beta = \gamma\}$*.*

The main drawback of the Reed-Solomon based FLPCP is that it requires a large alphabet, namely $|\mathbb{F}| > k$. Using AG codes (see Theorem 3.6) we can obtain a FLPCP over general fields. We consider here two settings. In the first setting we optimize the proof length but settle for constant soundness error. We use here the AG code of Theorem 3.6 with constant rate and distance.

Corollary 4.9 (Constant-Rate FLPCP over Small Fields). *There exists a fixed* $\delta > 0$ *such that the following holds. Let* $\mathbb{F}$ *be a finite field and* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$*. Then, the language* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *has an* FLPCP *over* $\mathbb{F}$ *with* 3 *queries,* $O(k)$ *proof length, soundness error* δ*, and randomness complexity* $\log_2(k) + O(1)$*. Furthermore, the verifier's accepting set is always* $\{(\alpha, \beta, \gamma) : \alpha \cdot \beta = \gamma\}$*.*

Next, we use AG codes to derive an FLPCP in which we optimize the soundness error. This uses the furthermore part of Theorem 3.6.

Corollary 4.10 (Soundness-Optimized FLPCP over Small Fields). *There exists* $\delta_q^* \in 1 - O(1/\sqrt{q})$ *such that the following holds. For every finite field* $\mathbb{F}$ *of order* $q > 9$ *such that* q *is a square, all sufficiently large* n *and matrices* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{F}^{k \times n}$*, the language* $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ *has an* FLPCP *over* $\mathbb{F}$ *with* 3 *queries,* $k \cdot \mathrm{poly}(q)$ *proof length, soundness error* $1 - \delta_q^*$ *and randomness complexity* $\log(k) + \log(1 - \delta_q^*) + O(\log(q))$*. Furthermore, the verifier's accepting set is always* $\{(\alpha, \beta, \gamma) : \alpha \cdot \beta = \gamma\}$*.*

Note that Corollary 4.10 is restricted to cases in which the field order is a square. This is due to our use of the multiplication code stated in the "furthermore" part of Theorem 3.6. Jumping ahead, we remark that when establishing Theorem 4.4 (which is not restricted to square order) we will still use Corollary 4.10 over the field of order q^2, which can be viewed as a 6-query FLPCP over $\mathbb{F}_q$. This can be composed with our inner DPP gadget to yield a DPP over $\mathbb{F}_q$ (see Appendix C). We also remark that while the furthermore part of Theorem 3.6 is restricted to infinite input lengths (rather than all sufficiently long inputs), this restriction does not carry over to Corollary 4.10 since we can pad the input to the next valid input length, while using the fact that the valid sequence of inputs is not too far apart (see the furthermore part of Theorem 3.6 for details).

Remark 4.11 (Alternative Approach for Soundness-Optimized FLPCP)**.** *As an alternative to the use of the furthermore part of Theorem 3.6, one can start off with Corollary 4.9 (which only relied on the main part of Theorem 3.6) and amplify the soundness to* $1/\mathrm{poly}(q)$ *by repeating the basic procedure* $O(\log(q))$ *times. Using a randomness efficient procedure (e.g., via a random walk on an expander graph, see [Vad12, Section 4.2]) this can be done using only* $O(\log(q))$ *random bits.*

This results in an FLPCP *with* $O(\log(q))$ *queries and randomness complexity, and soundness error* $1/\mathrm{poly}(q)$*. While the number of queries is larger than that in Corollary 4.10, jumping ahead, we remark that this construction can be used instead of Corollary 4.10 to obtain our high soundness DPP (though with a bigger* $\mathrm{poly}(q)$ *overhead to the proof size).*

Additional FLPCP constructions. By generalizing the multiplication code framework to enable a multiplication of two different codes, one can capture the "Hadamard FLPCP" from [ALMSS98, IKO07] as an instance of our general construction. This yields a 3-query FLPCP with near-optimal soundness error $O(1/|\mathbb{F}|)$ and randomness complexity $O(\log(|\mathbb{F}|))$, but at the cost of a quadratic proof length. For $\mathbb{F}$ of an odd characteristic, the query complexity of the Hadamard FLPCP can be improved to 2 [BIOW20]. Finally, if we replace $\mathsf{dR1CS}$ by the satisfiability of *Boolean* circuits, a 2-query FLPCP with *linear* proof length can be obtained from multiplication codes via the technique from [DFGK14]. See Appendix B for details.

4.2 DPP Composition

Our second main step is a generic composition lemma, which compiles an outer FLPCP for a language $\mathcal{L}$ with an inner DPP gadget, to produce a DPP for $\mathcal{L}$. This is similar to and simpler than the robust PCP + PCP of proximity composition of Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan [BGHSV06], with DPPs playing a role similar to that of PCPs of proximity.

Lemma 4.12 (FLPCP Composition with DPP)**.** *Suppose that* $\mathcal{L}$ *has an* FLPCP *verifier* $\mathcal{V}$*, over the field* $\mathbb{F}$*, with query complexity* q*, proof length* m*, completeness* c*, soundness error* s*, and randomness complexity* r*. Suppose furthermore that for every choice of randomness* $\rho \in \{0,1\}^r$ *the verification predicate* $A^{(\rho)} \subseteq \mathbb{F}^q$ *of* $\mathcal{V}$ *has a* DPP *with completeness* c'*, soundness error* s'*, proof length* m'*, and randomness complexity* r'*. Then,* $\mathcal{L}$ *has a* DPP *with completeness* $c \cdot c'$*, soundness error* $1-(1-s)\cdot(1-s')$*, proof length* $m + 2^r \cdot m'$*, and randomness complexity* $r + r'$*.*

Proof. Let $\mathcal{V}_{\mathsf{outer}}$ be the FLPCP verifier and for every choice of randomness $\rho \in \{0,1\}^\ell$ for $\mathcal{V}_{\mathsf{outer}}$, let $\mathcal{V}^{(\rho)}_{\mathsf{inner}}$ be the DPP verifier for the predicate $A^{(\rho)}$ as in the lemma's statement (the nomenclature is chosen based on the usage of these verifiers below). We construct a DPP verifier for $\mathcal{L}$, which we refer to as the "composed" verifier.

We start by describing the proof string. Fix an input $\boldsymbol{x} \in \mathcal{L}$ and let $\boldsymbol{\pi}$ be the corresponding FLPCP proof-string for $\mathcal{V}_{\mathsf{outer}}$. For every choice of randomness $\rho \in \{0,1\}^r$ for $\mathcal{V}_{\mathsf{outer}}$, let $\boldsymbol{Q}^{(\rho)} \in \mathbb{F}^{q \times (n+m)}$ denote the linear queries that $\mathcal{V}_{\mathsf{outer}}$ performs when using randomness ρ. Denote by $A^{(\rho)} \subseteq \mathbb{F}^q$ the decision predicate of V_{outer} for that choice of ρ and let $\alpha^{(\rho)} = \boldsymbol{Q}^{(\rho)} \cdot (\boldsymbol{x} \| \boldsymbol{\pi})$ denote the answers to the queries.

To check the proof string, the composed DPP verifier first generates randomness ρ for $\mathcal{V}_{\mathsf{outer}}$. It also generates a query $\boldsymbol{z} \in \mathbb{F}^{q+m'}$ for $\mathcal{V}^{(\rho)}_{\mathsf{inner}}$ and a corresponding decision predicate $A \subseteq \mathbb{F}$. We parse the query vector as $\boldsymbol{z} = \boldsymbol{z}_{\mathsf{inp}} \| \boldsymbol{z}_{\mathsf{prf}}$, where $\boldsymbol{z}_{\mathsf{inp}} \in \mathbb{F}^q$ and $\boldsymbol{z}_{\mathsf{prf}} \in \mathbb{F}^{m'}$. The verifier accepts if

$$\left\langle (\boldsymbol{Q}^{(\rho)})^{\intercal} \cdot \boldsymbol{z}_{\mathsf{inp}}, \boldsymbol{x} \| \boldsymbol{\pi} \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle \in A,$$

which is indeed a (single) linear query to $\big((\boldsymbol{x} \| \boldsymbol{\pi}) \| (\boldsymbol{\pi}^{(\rho)})_{\rho \in \{0,1\}^r} \big)$.

Our analysis of the composed DPP will rely on the following claim.

Claim 4.13. *For every $\boldsymbol{x} \in \mathbb{F}^n$, $\boldsymbol{\pi} \in \mathbb{F}^m$, $\rho \in \{0,1\}^r$, $\boldsymbol{\pi}^{(\rho)} \in \mathbb{F}^{m'}$, $\boldsymbol{z} = (\boldsymbol{z}_{\mathsf{inp}}, \boldsymbol{z}_{\mathsf{prf}}) \in \mathbb{F}^q \times \mathbb{F}^{m'}$ and $\boldsymbol{\alpha}^{(\rho)} = \boldsymbol{Q}^{(\rho)} \cdot (\boldsymbol{x} \| \boldsymbol{\pi})$, it holds that:*

$$\left\langle (\boldsymbol{Q}^{(\rho)})^{\intercal} \cdot \boldsymbol{z}_{\mathsf{inp}}, (\boldsymbol{x} \| \boldsymbol{\pi}) \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle = \left\langle \boldsymbol{z}, \left(\boldsymbol{\alpha}^{(\rho)} \| \boldsymbol{\pi}^{(\rho)} \right) \right\rangle.$$

Proof. This follows by linearity:

$$\begin{aligned}
\left\langle (\boldsymbol{Q}^{(\rho)})^{\intercal} \cdot \boldsymbol{z}_{\mathsf{inp}}, \boldsymbol{x} \| \boldsymbol{\pi} \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle &= \left\langle \boldsymbol{z}_{\mathsf{inp}}, \boldsymbol{Q}^{(\rho)} \cdot (\boldsymbol{x} \| \boldsymbol{\pi}) \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle \\
&= \left\langle \boldsymbol{z}_{\mathsf{inp}}, \boldsymbol{\alpha}^{(\rho)} \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle \\
&= \left\langle \boldsymbol{z}, \left(\boldsymbol{\alpha}^{(\rho)} \| \boldsymbol{\pi}^{(\rho)} \right) \right\rangle.
\end{aligned}$$

□

Completeness. Fix $\boldsymbol{x} \in \mathcal{L}$ and the proof string $\big(\boldsymbol{\pi}, (\boldsymbol{\pi}^{(\rho)})_\rho\big)$ as defined above. By the completeness of $\mathcal{V}_{\mathsf{outer}}$, with probability c over the choice of ρ, it holds that $\boldsymbol{\alpha}^{(\rho)} \in A^{(\rho)}$, where $\boldsymbol{\alpha}^{(\rho)} = \boldsymbol{Q}^{(\rho)} \cdot (\boldsymbol{x} \| \boldsymbol{\pi})$. Assume that ρ is selected as above.

Since $\boldsymbol{\alpha}^{(\rho)} \in A^{(\rho)}$, by the completeness of $\mathcal{V}_{\mathsf{inner}}$, with probablity c', we have that $\big\langle \boldsymbol{z}, \big(\boldsymbol{\alpha}^{(\rho)} \| \boldsymbol{\pi}^{(\rho)}\big) \big\rangle \in A$, where $\boldsymbol{z}$ is the query vector of $\mathcal{V}_{\mathsf{inner}}$ and A is its decision predicate. Parsing $\boldsymbol{z}$ as $\boldsymbol{z} = \boldsymbol{z}_{\mathsf{inp}} \| \boldsymbol{z}_{\mathsf{prf}}$, with $\boldsymbol{z}_{\mathsf{inp}} \in \mathbb{F}_q$ and $\boldsymbol{z}_{\mathsf{prf}} \in \mathbb{F}^{m'}$, by Claim 4.13 we have that:

$$\left\langle (\boldsymbol{Q}^{(\rho)})^{\intercal} \cdot \boldsymbol{z}_{\mathsf{inp}}, (\boldsymbol{x} \| \boldsymbol{\pi}) \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle = \left\langle \boldsymbol{z}, \left(\boldsymbol{\alpha}^{(\rho)} \| \boldsymbol{\pi}^{(\rho)} \right) \right\rangle \in A,$$

and so the composed verifier accepts in this case.

Soundness. Fix $\boldsymbol{x} \notin \mathcal{L}$ and an alleged proof string $\big(\boldsymbol{\pi}, (\boldsymbol{\pi}^{(\rho)})_\rho\big)$. By the soundness of $\mathcal{V}_{\mathsf{outer}}$ with probability $1-s$ over the choice of ρ, it holds that $\boldsymbol{\alpha}^{(\rho)} \notin A^{(\rho)}$, where $\boldsymbol{\alpha}^{(\rho)} = \boldsymbol{Q}^{(\rho)} \cdot (\boldsymbol{x} \| \boldsymbol{\pi})$. Assume that a ρ as above is indeed chosen.

Observe that in this case $\boldsymbol{\alpha}^{(\rho)}$ is a NO instance and by the soundness of $\mathcal{V}_{\mathsf{inner}}$, for every $\boldsymbol{\pi}'$, with probability $1-s'$ it holds that $\big\langle \boldsymbol{z}, \big(\boldsymbol{\alpha}^{(\rho)} \| \boldsymbol{\pi}'\big) \big\rangle \notin A$, where $\boldsymbol{z}$ is $\mathcal{V}_{\mathsf{inner}}$'s query vector and A is its decision predicate. In particular, this is true if we set $\boldsymbol{\pi}' = \boldsymbol{\pi}^{(\rho)}$, and so we have that $\big\langle \boldsymbol{z}, \big(\boldsymbol{\alpha}^{(\rho)} \| \boldsymbol{\pi}^{(\rho)}\big) \big\rangle \notin A$ with probability s'.

Thus, by Claim 4.13, in this case

$$\left\langle (\boldsymbol{Q}^{(\rho)})^\intercal \cdot \boldsymbol{z}_{\mathsf{inp}}, (\boldsymbol{x}\|\boldsymbol{\pi}) \right\rangle + \left\langle \boldsymbol{z}_{\mathsf{prf}}, \boldsymbol{\pi}^{(\rho)} \right\rangle = \left\langle \boldsymbol{z}, \left(\boldsymbol{\alpha}^{(\rho)}\|\boldsymbol{\pi}^{(\rho)}\right) \right\rangle \notin A,$$

and so the composed verifier rejects. Overall, the composed verifier rejects with probability at least $(1-s)\cdot(1-s')$. □

4.3 DPP Gadgets

In order to transform the FLPCP of Theorem 4.6 into a DPP, via the Composition Lemma 4.12, we need to construct an inner "DPP gadget" for the multiplicative relation $\{(\alpha,\beta,\gamma)\in\mathbb{F}^3 : \gamma = \alpha\cdot\beta\}$. In this section we present two *generic* constructions that apply to any language.

The first (and simpler) construction, presented in Section 4.3.1, has imperfect completeness, and in fact a poor completeness-soundness gap that degrades with the language size. Our second and main construction has perfect completeness and asymptotically optimal soundness, and takes two steps:

1. We consider the language $\mathbf{UV} = \{\boldsymbol{e}_1,\ldots,\boldsymbol{e}_n\}$ consisting of all n unit vectors. In other words, $\mathbf{UV}$ is the set of all vectors that are 0 in all coordinates except for a single coordinate, which is a 1. Our first step (Section 4.3.2) is a direct (proof-less) DPP for this problem.

2. Using the DPP for $\mathbf{UV}$, the second step (Section 4.3.3) constructs a "brute-force DPP" for any $\mathcal{L}$ where proof length scales linearly with $|\mathcal{L}|$. This can be used to obtain a poly($|\mathbb{F}|$)-size DPP for the multiplicative relation (in fact, for any $\mathcal{L}\subseteq\mathbb{F}^{O(1)}$).

4.3.1 "Simplest DPP" with Imperfect Completeness

Most of our DPP constructions achieve perfect completeness. However, perfect completeness cannot be generally achieved over $\mathbb{F}_2$ (see Lemma A.1). Settling for imperfect completeness, and for a completeness-soundness gap that vanishes with $|\mathcal{L}|$, we have the following simple DPP construction. The construction can be viewed as applying a special case of the randomization technique of Razborov and Smolensky [Raz87, Smo87] to the truth-table representation of $\mathcal{L}$. Additional features of this construction are that it does not require a proof π (i.e., $m = 0$) and that it has an accepting set of size 1.

Proposition 4.14 (Truth-Table DPP). *For every $\mathcal{L}\subseteq\mathbb{F}^n$, there exists a proof-less* DPP *for $\mathcal{L}$ over $\mathbb{F}$ with soundness error $s = 1/|\mathbb{F}|$, completeness $c = s + \frac{1}{2|\mathcal{L}|}$, and an accepting set A of size* 1.

Proof. The DPP verifier proceeds as follows: Pick uniformly random $\boldsymbol{r}\leftarrow\mathcal{L}$ and $\boldsymbol{q}\leftarrow\mathbb{F}^n$, and check that $\langle \boldsymbol{q}, \boldsymbol{x}-\boldsymbol{r}\rangle = 0$. Note that the latter condition can be written in the form $\langle \boldsymbol{q},\boldsymbol{x}\rangle = a$ for a which is determined by $\boldsymbol{q}$ and $\boldsymbol{r}$. Hence, this is a "proof-less" DPP with an accepting set A of size 1.

Soundness follows from the fact that if $\boldsymbol{x}\notin\mathcal{L}$, then conditioned on every choice of $\boldsymbol{r}\in\mathcal{L}$ we have $\boldsymbol{x}-\boldsymbol{r}\neq 0$, and hence the inner product $\langle \boldsymbol{q},\boldsymbol{x}-\boldsymbol{r}\rangle$ is uniformly distributed over $\mathbb{F}$.

Completeness follows from the fact that when $\boldsymbol{x}\in\mathcal{L}$, there is exactly one choice of $\boldsymbol{r}\in\mathcal{L}$, occurring with $1/|\mathcal{L}|$ probability, conditioned on which $\langle \boldsymbol{q},\boldsymbol{x}-\boldsymbol{r}\rangle$ is identically 0. Conditioned on other choices of $\boldsymbol{r}$, the answer $\langle \boldsymbol{q},\boldsymbol{x}-\boldsymbol{r}\rangle$ is uniformly distributed. Hence, for $\boldsymbol{x}\in\mathcal{L}$ we have:

$$\Pr\left[\langle \boldsymbol{q},\boldsymbol{x}-\boldsymbol{r}\rangle = 0\right] = \frac{1}{|\mathcal{L}|}\cdot 1 + \left(1-\frac{1}{|\mathcal{L}|}\right)\cdot\frac{1}{|\mathbb{F}|} \geq \frac{1}{|\mathbb{F}|} + \frac{1}{2|\mathcal{L}|}.$$ □

Applying this result over the binary field $\mathbb{F}_2$ and the multiplicative relation over this field, we get that:

Corollary 4.15. *There exists a proof-less* DPP *for* $\{(\alpha, \beta, \gamma) \in (\mathbb{F}_2)^3 : \gamma = \alpha \cdot \beta\}$ *with soundness error* $s = 1/2$*, completeness* $c = 5/8$*, and an accepting set* A *of size* 1.

Proof of Theorem 4.2. To prove Theorem 4.2 we combine Corollary 4.9 with Corollary 4.15, by following the composition lemma Lemma 4.12, with the following adjustment.

Denote the soundness error of the outer FLPCP (i.e., of Corollary 4.9) by s. Then, the probability that the composed verifier accepts a false statement is if either (1) the outer verifier accepts a false statement and the inner verifier accepts the true statement, or (2) if the outer verifier rejects and the inner verifier accepts a false statement. Overall, the probability to accept is: $s \cdot \frac{5}{8} + (1-s) \cdot \frac{1}{2} = \frac{1}{2} + \frac{s}{8} < \frac{5}{8}$. □

4.3.2 Unit Vectors

We turn to our main inner DPP construction. Our first step is an optimal DPP for a simple "universal" language. Let $\mathbf{UV} \subseteq \mathbb{F}^n$ be the language of all unit vectors. In other words, $\mathbf{UV}$ contains all vectors that are zero, except for one coordinate which is equal to 1.

We start with a simpler DPP for $\mathbf{UV}$ that has constant soundness error and a minimal accepting set A. We then present a construction with a more complicated analysis that achieves (asymptotically optimal) soundness error of $O(1/\sqrt{|\mathbb{F}|})$.

Proposition 4.16 (DPP for $\mathbf{UV}$)**.** *Let* $\mathbb{F}$ *be a field of odd characteristic. Then, the language* $\mathbf{UV}$ *has a proof-less* DPP *with soundness error* $4/5$*, randomness complexity* $O(n)$*, and accepting set of size at most two.*

Proof of Proposition 4.16. The verifier, given access to an input $\boldsymbol{x} \in \mathbb{F}^n$, with probability $4/5$ samples a random vector $\boldsymbol{q} \in \{0,1\}^n$ and accepts if $\langle \boldsymbol{x}, \boldsymbol{q} \rangle \in \{0,1\}$, and with probability $1/5$ sets $\boldsymbol{q} = 1^n$, and accepts if $\langle \boldsymbol{x}, \boldsymbol{q} \rangle = 1$.

Completeness. If $\boldsymbol{x} \in \mathbf{UV}$, $\langle \boldsymbol{x}, \boldsymbol{q} \rangle = q_i$ and the verifier accepts with probability 1.

Soundness. Let $\boldsymbol{x} \notin \mathbf{UV}$. Suppose first that $\boldsymbol{x}$ has at least two non-zero coordinates and assume without loss of generality that $x_1, x_2 \neq 0$. Fix $q_3, \ldots, q_n$ and denote $\alpha = \sum_{i=3}^{n} x_i \cdot q_i$. Notice that for $q_1, q_2 \in \{0,1\}$ we get four possible values for $q_1 \cdot x_1 + q_2 \cdot x_2$, namely: $0, x_1, x_2, x_1 + x_2$. Since $x_1, x_2 \neq 0$ and the field is of odd characteristic, at least three of these four values are distinct. It follows that for at least one value of $(q_1, q_2) \in \{0,1\}^2$ it holds that $\langle \boldsymbol{x}, \boldsymbol{q} \rangle = q_1 \cdot x_1 + q_2 \cdot x_2 + \alpha \notin \{0,1\}$. Thus, in this case, the verifier rejects with probability at least $\frac{4}{5} \cdot \frac{1}{4} = \frac{1}{5}$.

Assume that $\boldsymbol{x} \notin \mathbf{UV}$ has at most one non-zero coordinate. Assume without loss of generality that $x_1 \neq 1, x_2 = \cdots = x_n = 0$ (note that it may be the case that $x_1 = 0$). Then, when $\boldsymbol{q} = 1^n$, $\langle \boldsymbol{x}, \boldsymbol{q} \rangle = x_1$. Hence, the verifier rejects with probability at least $1/5$. □

We note that an accepting set of size $|A| = 2$ is the best possible for a DPP with perfect completeness (see Appendix A). For fields of characteristic 2, we prove a similar result with $|A| = 3$.

Proposition 4.17 (DPP for **UV** over Fields of Characteristic 2)**.** *Let $\mathbb{F} = \mathbb{F}_{2^k}$ for $k \geq 2$. Then, the language* **UV** $\subseteq \mathbb{F}^n$ *has a proof-less* DPP *with soundness error (at most)* 9/10*, randomness complexity $O(n)$, and accepting set A of size 3.*

Proof of Proposition 4.17. Let $c \in \mathbb{F} \setminus \{0, 1\}$. The verifier, given access to an input $\boldsymbol{x} \in \mathbb{F}^n$, with probability 9/10 samples a random vector $\boldsymbol{q} \in \{0, 1, c\}^n$ and accepts if $\langle \boldsymbol{x}, \boldsymbol{q} \rangle \in \{0, 1, c\}$, and with probability 1/10 sets $\boldsymbol{q} = 1^n$, and accepts if $\langle \boldsymbol{x}, \boldsymbol{q} \rangle = 1$. The proof is similar to the proof of Proposition 4.16. Completeness is identical, we focus on soundness.

Soundness. Let $\boldsymbol{x} \notin \mathbf{UV}$. As before, suppose first that $\boldsymbol{x}$ has at least two non-zero coordinates and assume without loss of generality that $x_1, x_2 \neq 0$. Fix $q_3, \ldots, q_n$ and denote $\alpha = \sum_{i=3}^{n} x_i \cdot q_i$.

Now consider two sub-cases. If $x_1 = x_2$, then for $q_1, q_2 \in \{0, 1, c\}$ we get a multiset of 9 possible values for $q_1 \cdot x_1 + q_2 \cdot x_2$, which includes (in particular) $0, x_1, x_1 \cdot c, x_1 \cdot (c + 1)$. Since $c \notin \{0, 1\}$, all 4 field elements are distinct. In particular, for at least one value of q_1, q_2 it holds that $\langle \boldsymbol{x}, \boldsymbol{q} \rangle = q_1 \cdot x_1 + q_2 \cdot x_2 + \alpha \notin \{0, 1, c\}$. In this case, the verifier rejects with probability at least $\frac{9}{10} \cdot \frac{1}{9} = \frac{1}{10}$. If $x_1 \neq x_2$, then the multiset of 9 possible values for $q_1 \cdot x_1 + q_2 \cdot x_2$ includes $0, x_1, x_2, x_1 + x_2$. Since x_1, x_2 are distinct nonzero elements, the above 4 elements are distinct. Thus, as in the previous sub-case, the verifier rejects with probability at least 1/10.

We are left with the case that $\boldsymbol{x} \notin \mathbf{UV}$ has a single non-zero coordinate, where similarly to the previous proof, the verifier rejects with probability at least 1/10 (when $\boldsymbol{q} = 1^n$). □

Now we show a more general result in which the error can be as small as $O(1/\sqrt{|\mathbb{F}|})$. While the construction is simple, the analysis is somewhat more involved.

Proposition 4.18 (DPP for **UV** with Soundness $O(1/\sqrt{|\mathbb{F}|})$)**.** *The language* **UV** *has a proof-less* DPP *over the field of order q with soundness error $O(1/\sqrt{q})$ and randomness complexity $\sqrt{q} \cdot \log(q) + n \cdot \log(q)$.*

Proof. Let $\mathbb{F}$ be a field of order q and $t \in [q]$ a parameter to be determined below. The verifier, given access to an input $x \in \mathbb{F}^n$, samples a random subset $\Lambda \subseteq \mathbb{F}$ of size $|\Lambda| = t$ uniformly at random and further samples $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n) \in \Lambda^n$. The verifier accepts if $\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda$.

Completeness. Let $\boldsymbol{x} \in \mathbf{UV}$ and let $i \in [n]$ be such that $\boldsymbol{x} = \boldsymbol{e}_i$ is the i^{th} unit vector. Then $\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle = \lambda_i \in \Lambda$ and so the verifier always accepts.

Soundness. The soundness of the DPP is implied directly by the following lemma.

Lemma 4.19. *If $\boldsymbol{x} \in \mathbb{F}^n$ and $\boldsymbol{x} \notin$* **UV***, then*

$$\Pr[\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda] \leq O\left(\frac{1}{t} + \frac{t}{|\mathbb{F}| - t}\right),$$

where $\Lambda \subseteq \mathbb{F}$ is a random set of size t and $\boldsymbol{\lambda}$ is random in Λ^n.

Proof. First note that the distribution of $(\Lambda, \boldsymbol{\lambda})$ can be sampled as follows, first sample a random function $\pi \colon [n] \to [t]$, namely a partition of $[n]$ to t buckets. Then sample a random injective function $\gamma \colon [t] \to \mathbb{F}$, and set $\Lambda = \gamma([t])$ and $\lambda_i = \gamma(\pi(i))$ for all $i \in [n]$. Throughout, for any partition π and $j \in [t]$, we denote the sum of x_i in each bucket j by $\alpha_j^\pi \coloneqq \sum_{i \in \pi^{-1}(j)} x_i$.

Claim 4.20. *Fix any partition π such that there exist $1 \le i < j \le t$ where $\alpha_i^\pi \neq 0$, $\alpha_j^\pi \neq 0$. Then,*

$$\Pr_{\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda\right] \le \frac{t+1}{|\mathbb{F}| - t + 1} \quad .$$

Proof. Assume without loss of generality that $\alpha_{t-1}^\pi \neq 0, \alpha_t^\pi \neq 0$. Now fix any (distinct) $\gamma(1), \dots, \gamma(t-2)$. First, sampling $\gamma(t-1) \leftarrow \mathbb{F} \setminus \gamma([t-2])$, and denoting $c_j := \sum_{i \in [j]} \gamma(i)\alpha_i^\pi$, it holds that

$$\Pr_{\gamma(t-1)}[c_{t-1} = 0] = \Pr_{\gamma(t-1)}[\alpha_{t-1}^\pi \gamma(t-1) + c_{t-2} = 0] \le \frac{1}{|\mathbb{F}| - (t-2)} \le \frac{1}{|\mathbb{F}| - t + 1} \quad .$$

Next, fix $\gamma(t-1)$ such that $c_{t-1} \neq 0$. Then, sampling $\gamma(t) \leftarrow \mathbb{F} \setminus \gamma([t-1])$, it holds that

$$\begin{aligned}
&\Pr_{\gamma(t)}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda \mid c_{t-1} \neq 0\right] \\
&\quad = \Pr_{\gamma(t)}\left[\alpha_t^\pi \gamma(t) + c_{t-1} \in \Lambda\right] \\
&\quad = \sum_{i \in [t-1]} \Pr_{\gamma(t)}\left[\alpha_t^\pi \gamma(t) + c_{t-1} = \gamma(i)\right] + \Pr_{\gamma(t)}\left[(\alpha_t^\pi - 1)\gamma(t) + c_{t-1} = 0\right] \\
&\quad \le (t-1) \cdot \frac{1}{|\mathbb{F}| - (t-1)} + \frac{1}{|\mathbb{F}| - (t-1)} \\
&\quad \le \frac{t}{|\mathbb{F}| - t + 1} \quad .
\end{aligned}$$

Overall,

$$\Pr_{\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda\right] \le \Pr_{\gamma(t-1)}\left[c_{t-1} = 0\right] + \Pr_{\gamma(t)}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x} \rangle \in \Lambda \mid c_{t-1} \neq 0\right] \le \frac{t+1}{|\mathbb{F}| - t + 1} \quad .$$

□

Claim 4.21. *Assume $\boldsymbol{x}$ has Hamming weight $h \ge 2$. Then,*

$$\Pr_{\pi}\left[\exists i \neq j : \alpha_i^\pi \neq 0, \alpha_j^\pi \neq 0\right] \ge 1 - \frac{4}{t} \quad .$$

Proof. Assume without loss of generality that $x_1, \dots, x_h \neq 0$ and that $x_{h+1} = \dots = x_n = 0$. Fix any $\pi(1), \dots, \pi(h-2) \in [t]$, and consider the sum of x_i in each bucket j so far:

$$\alpha_j^{\pi_{h-2}} := \sum_{i \in \pi^{-1}(j) \cap [h-2]} x_i \quad .$$

Let $S = \left\{ j : \alpha_j^{\pi_{h-2}} \neq 0 \right\}$. We consider two cases:

- Assume $|S| \ge 2$, and fix some $i \neq j$ in S. Then

$$\Pr_{\pi(h-1),\pi(h)}\left[\alpha_i^\pi = 0 \vee \alpha_j^\pi = 0\right] \le \Pr_{\pi(h-1),\pi(h)}\left[\pi(h-1) \in \{i,j\} \vee \pi(h) \in \{i,j\}\right] \le \frac{2}{t} + \frac{2}{t} \le \frac{4}{t} \quad .$$

- Assume $|S| \le 1$. Then,

$$\begin{aligned}
&\Pr_{\pi(h-1),\pi(h)}\left[\alpha_{\pi(h-1)}^\pi = 0 \vee \alpha_{\pi(h)}^\pi = 0 \vee \pi(h-1) = \pi(h)\right] \le \\
&\Pr_{\pi(h-1),\pi(h)}\left[\pi(h-1) \in S \vee \pi(h) \in S \vee \pi(h-1) = \pi(h)\right] \le \frac{|S|}{t} + \frac{|S|}{t} + \frac{1}{t} \le \frac{3}{t} \quad .
\end{aligned}$$

Indeed, if $\pi(h-1), \pi(h) \notin S$ and $\pi(h-1) \neq \pi(h)$, then

$$\alpha^{\pi}_{\pi(h-1)} = \alpha^{\pi_{h-2}}_{\pi(h-1)} + x_{h-1} = x_{h-1} \neq 0 \ ,$$
$$\alpha^{\pi}_{\pi(h)} = \alpha^{\pi_{h-2}}_{\pi(h)} + x_h = x_h \neq 0 \ .$$

This completes the proof of the claim. □

Claim 4.22. *Assume $\boldsymbol{x}$ has Hamming weight $h = 1$, but is not a unit vector. Then,*

$$\Pr\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] \leq \frac{t+1}{|\mathbb{F}| - t + 1} \ .$$

Proof. Assume without loss of generality $x_1 \notin \{0,1\}$ and $x_2 = \cdots = x_n = 0$. Fix any π, let $i = \pi(1)$, and fix any distinct $\gamma(1), \ldots, \gamma(i-1), \gamma(i+1), \ldots, \gamma(t)$. Then sampling $\gamma(i) \leftarrow \mathbb{F} \setminus \cup_{j \neq i} \gamma(j)$,

$$\begin{aligned}
\Pr_{\gamma(i)}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] &= \Pr_{\gamma(i)}\left[x_1\gamma(i) \in \Lambda\right] \\
&= \sum_{j \neq i} \Pr_{\gamma(i)}\left[x_1 \gamma(i) = \gamma(j)\right] + \Pr_{\gamma(i)}\left[(x_1 - 1)\gamma(i) = 0\right] \\
&\leq \frac{t}{|\mathbb{F}| - t + 1} \ .
\end{aligned}$$

□

Claim 4.23. *Assume $\boldsymbol{x} = \boldsymbol{0}$. Then,*

$$\Pr\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] = \frac{t}{|\mathbb{F}|} \ .$$

Proof. In this case $\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle = 0$. The probability that a random set $\Lambda \subseteq \mathbb{F}$ of size t includes any specific element (in particular 0) is $t/|\mathbb{F}|$. □

Putting the claims together, we consider the following cases:

- If $\boldsymbol{x}$ has Hamming weight $h \geq 2$, then by Claim 4.20 and Claim 4.21,

$$\begin{aligned}
\Pr_{\pi,\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] &\leq \Pr_{\pi}\left[\nexists i \neq j : \alpha^{\pi}_i \neq 0, \alpha^{\pi}_j \neq 0\right] + \Pr_{\pi,\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda \mid \exists i \neq j : \alpha^{\pi}_i \neq 0, \alpha^{\pi}_j \neq 0\right] \\
&\leq O\left(\frac{1}{t}\right) + O\left(\frac{t}{|\mathbb{F}| - t}\right) \ .
\end{aligned}$$

- If $\boldsymbol{x}$ has Hamming weight $h = 1$ but is not a unit vector, then by Claim 4.22,

$$\Pr_{\pi,\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] \leq O\left(\frac{t}{|\mathbb{F}| - t}\right) \ .$$

- If $\boldsymbol{x} = \boldsymbol{0}$, then by Claim 4.23,

$$\Pr_{\pi,\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] = \frac{t}{|\mathbb{F}|} \ .$$

Overall, it follows that unless $\boldsymbol{x}$ is a unit vector $\Pr_{\pi,\gamma}\left[\langle \boldsymbol{\lambda}, \boldsymbol{x}\rangle \in \Lambda\right] \leq O\left(\frac{1}{t} + \frac{t}{|\mathbb{F}|-t}\right)$. This completes the proof of soundness (Lemma 4.19). □

Proposition 4.18 follows. □

4.3.3 A Brute-Force DPP

In this section we use the DPP for **UV** to construct a DPP for any language $\mathcal{L}$, where the proof size scales with $|\mathcal{L}|$.

Lemma 4.24 (Simple DPP Constructions)**.** *Let $\mathbb{F}$ be of size $q > 2$ and $\mathcal{L} \subseteq \mathbb{F}^n$ of size ℓ. Then $\mathcal{L}$ has perfectly-complete* DPP*s with proof length $O(\ell)$ and the following additional parameters:*

- ***Small Accepting Set*** DPP*: If q is odd, soundness error at most $\frac{4}{5}$ and accepting set of size at most 2. If $q = 2^k$, $k > 1$, soundness error at most $\frac{9}{10}$ and accepting set of size at most 3. In either one, the randomness complexity is $O(\ell)$.*
- ***Small Soundness Error*** DPP*: Soundness error $O(1/\sqrt{q})$ and randomness complexity $O((\sqrt{q} + \ell)\log q)$.*

Proof. Let $\boldsymbol{X} \in \mathbb{F}^{n\times\ell}$ be a matrix whose columns $\boldsymbol{x}_1, \dots, \boldsymbol{x}_\ell$ are all such that $\boldsymbol{x}_i \in \mathcal{L}$, and consider the matrix

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{X} & \boldsymbol{I}_{n\times n} \\ \boldsymbol{I}_{\ell\times\ell} & \boldsymbol{0}_{\ell\times n} \end{bmatrix} \in \mathbb{F}^{(\ell+n)\times(\ell+n)} \quad .$$

Then $\boldsymbol{M}$ is invertible and for any (column) vector $(\boldsymbol{x}\|\boldsymbol{w}) \in \mathbb{F}^{n+\ell}$ and unit (column) vector $\boldsymbol{e}_i \in \{0,1\}^\ell$, $\boldsymbol{M}^{-1}(\boldsymbol{x}\|\boldsymbol{w}) = (\boldsymbol{e}_i\|0^n)$ if and only if $(\boldsymbol{x}\|\boldsymbol{w}) = (\boldsymbol{x}_i\|\boldsymbol{e}_i)$. In the following, we write $\boldsymbol{M}^{-\intercal}$ to denote the matrix $\boldsymbol{M}^{-\intercal} := (\boldsymbol{M}^{-1})^\intercal$.

Accordingly, in the DPP for $\mathcal{L}$, a proof for $\boldsymbol{x}_i$ is the corresponding vector $\boldsymbol{e}_i$. Given linear query access to $(\boldsymbol{x}\|\boldsymbol{w})$, the DPP verifier samples a query $\boldsymbol{q} \in \mathbb{F}^\ell$ along with accepting set A for the **UV** DPP, samples a uniformly random vector $\boldsymbol{\lambda} \in \mathbb{F}^n$, and outputs the query $\boldsymbol{M}^{-\intercal}(\boldsymbol{q}||\boldsymbol{\lambda})$ with accepting set A.

Completeness. If $(\boldsymbol{x}\|\boldsymbol{w}) = (\boldsymbol{x}_i\|\boldsymbol{e}_i)$, then

$$\left\langle \boldsymbol{M}^{-\intercal}(\boldsymbol{q}||\boldsymbol{\lambda}), (\boldsymbol{x}\|\boldsymbol{w}) \right\rangle = \left\langle (\boldsymbol{q}||\boldsymbol{\lambda}), \boldsymbol{M}^{-1}(\boldsymbol{x}\|\boldsymbol{w}) \right\rangle = \langle (\boldsymbol{q}||\boldsymbol{\lambda}), (\boldsymbol{e}_i\|0^n) \rangle = \langle \boldsymbol{q}, \boldsymbol{e}_i \rangle \in A \ ,$$

where the membership in A follows from the completeness of the DPP for **UV**.

Soundness. Assume $\boldsymbol{x} \notin \mathcal{L}$, and let $\boldsymbol{M}^{-1}(\boldsymbol{x}\|\boldsymbol{w}) = (\boldsymbol{e}^*\|\boldsymbol{z}^*) \in \mathbb{F}^\ell \times \mathbb{F}^n$. We consider two cases:

- If $\boldsymbol{z}^* \neq \boldsymbol{0}$, then $\langle \boldsymbol{M}^{-\intercal}(\boldsymbol{q}||\boldsymbol{\lambda}), (\boldsymbol{x}\|\boldsymbol{w}) \rangle = \langle (\boldsymbol{q}||\boldsymbol{\lambda}), (\boldsymbol{e}^*\|\boldsymbol{z}^*) \rangle$ is uniformly random and falls in A with probability at most $|A|/q$.
- If $\boldsymbol{z}^* = \boldsymbol{0}$, but $\boldsymbol{e}^* \notin$ **UV**, then $\langle \boldsymbol{M}^{-\intercal}(\boldsymbol{q}||\boldsymbol{\lambda}), (\boldsymbol{x}\|\boldsymbol{w}) \rangle = \langle \boldsymbol{q}, \boldsymbol{e}^* \rangle$ and we reject whenever the **UV** verifier rejects.

The concrete parameters now follow from the guaranteed soundness and upper bound on the size of accepting sets given by Propositions 4.16 to 4.18. □

Remark 4.25 (Faster Proving Time)**.** *In the* DPP *of Lemma 4.24 the verifier needs to compute the matrix-vector product $\boldsymbol{M}^{-1}\boldsymbol{q}$, which in general takes $(\ell + n)^2$ time. However, for specific languages, it may be possible to do this computation faster. For example, when applying the lemma to the relation $\{(1, \alpha, \alpha^2, \alpha^{q-1}) : \alpha \in \mathbb{F}\}$ the corresponding multiplication corresponds to an FFT, which can be done in quasi-linear time. The latter* DPP *can be used to derive a* DPP *for the squaring relation $\{(1, \alpha, \alpha^2) : \alpha \in \mathbb{F}\}$ in which the verifier runs in quasi-linear time.*

We are now ready to prove the main small-field DPP theorems, obtaining constant soundness (with constant-size accepting set) and optimal soundness, respectively.

Proof of Theorems 4.3 and 4.4. To prove the theorems we combine Corollaries 4.9 and 4.10 (respectively) with the two parts of Lemma 4.24 (providing constant soundness and optimal soundness, respectively) via Composition Lemma 4.12. We note that Corollary 4.10 assumes square q. However, viewing a 3-query FLPCP over $\mathbb{F}_{q^2}$ as a 6-query FLPCP over $\mathbb{F}_q$, we can still use the brute-force DPP of Lemma 4.24. See Appendix C for details. □

4.4 Lower Bound on the Soundness Error

In this section we show that the $O(1/\sqrt{|\mathbb{F}|})$ soundness error of our DPP construction is asymptotically tight. In fact, we show this for the language $\mathbf{UV}$ of unit vectors in $\mathbb{F}^n$. We start by assuming our default notion of DPP with perfect completeness, and then discuss the extension to the general case.

Consider a (perfectly complete) DPP and a query $\boldsymbol{q}$ that has an accepting set $A \subset \mathbb{F}$ of size t. We will describe two kinds of attacks. The first attack picks a uniformly random $(\boldsymbol{x}\|\boldsymbol{\pi})$, which leads to $\approx t/|\mathbb{F}|$ soundness error. The second attack uses input vectors of weight 2, leading to $\approx 1/t$ soundness error. Randomly choosing one of the two attacks will give us the desired $\Omega(1/\sqrt{|\mathbb{F}|})$ lower bound. We proceed with the formal analysis.

Lemma 4.26 (Random Attack)**.** *For any input length $n \geq 1$, proof length $m \geq 0$, nonzero query vector $\boldsymbol{q} \in \mathbb{F}^{n+m}$, and accepting set $A \subset \mathbb{F}$ of size t, a uniformly random choice of $\boldsymbol{x} \in \mathbb{F}^n \setminus \mathbf{UV}$ and $\boldsymbol{\pi} \in \mathbb{F}^m$ satisfies* $\Pr[\langle \boldsymbol{q}, (\boldsymbol{x}\|\boldsymbol{\pi})\rangle \in A] \geq (t-1)/|\mathbb{F}|$.

Proof. Since $\boldsymbol{q}$ is nonzero, if *both* $\boldsymbol{x}$ and $\boldsymbol{\pi}$ are chosen uniformly at random, the answer $\langle \boldsymbol{q}, (\boldsymbol{x}\|\boldsymbol{\pi})\rangle$ is uniformly random in $\mathbb{F}$, and hence is accepted (i.e., in A) with $t/|\mathbb{F}|$ probability. Since $|\mathbf{UV}|/|\mathbb{F}^n| \leq 1/|\mathbb{F}|$, conditioning on the event that $\boldsymbol{x} \in \mathbb{F}^n \setminus \mathbf{UV}$ reduces this acceptance probability by at most $1/|\mathbb{F}|$. Hence, random $\boldsymbol{x} \in \mathbb{F}^n \setminus \mathbf{UV}$ and $\pi \in \mathbb{F}^m$ are accepted with at least $t/|\mathbb{F}| - 1/|\mathbb{F}| = (t-1)/|\mathbb{F}|$ probability, as required. □

For the next attack, consider any DPP for $\mathbf{UV}$, and let $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_n \in \mathbb{F}^m$ be valid DPP proof vectors for the inputs $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n \in \mathbf{UV}$, respectively, such that $\boldsymbol{v}_i = (\boldsymbol{e}_i\|\boldsymbol{\pi}_i)$ is always accepted. We show a distribution over weight-2 vectors and proofs which is accepted with at least $\approx 1/t$ probability.

Lemma 4.27 (Weight-2 Attack)**.** *Let $|\mathbb{F}| > 2$ and $n \geq 2$. Let $\alpha \in \mathbb{F} \setminus \{0,1\}$ and let $\beta = 1 - \alpha$. Let $\boldsymbol{v}_i$ be as defined above. Then, for any nonzero $\boldsymbol{q} \in \mathbb{F}^m$ and $A \subseteq \mathbb{F}$ of size t generated by the* DPP *verifier, a uniformly random choice of* distinct $i, j \in [n]$ *satisfies* $\Pr[\langle \boldsymbol{q}, \alpha\boldsymbol{v}_i + \beta\boldsymbol{v}_j\rangle \in A] \geq 1/t - 1/n$.

Proof. By completeness, we have $a_i = \langle \boldsymbol{q}, \boldsymbol{v}_i\rangle \in A$ for every $i \in [n]$. If $a_i = a_j$, we have $\langle \boldsymbol{q}, \alpha\boldsymbol{v}_i + \beta\boldsymbol{v}_j\rangle = \alpha a_i + \beta a_j = (\alpha + \beta)a_i = a_i \in A$. It thus suffices to lower bound the probability that for a random choice of distinct i, j, we have $a_i = a_j$. This is the same as the probability of drawing two distinct balls of the same color from an urn containing n balls of t colors. Letting n_c be the number of balls of color c, $1 \leq c \leq t$, and conditioning on the color of the first ball, this probability can be written as:

$$\begin{aligned} p_{n_1,\ldots,n_t} &= \sum_{c=1}^{t} \frac{n_c}{n} \cdot \frac{n_c - 1}{n} \\ &= \frac{\sum_{c\in[t]} n_c^2}{n^2} - \frac{1}{n^2} \sum_{c\in[t]} n_c \\ &= \frac{\sum_{c\in[t]} n_c^2}{(\sum_{c\in[t]} n_c)^2} - \frac{1}{n} \\ &\geq \frac{1}{t} - \frac{1}{n}, \end{aligned}$$

where the inequality follows from the Cauchy-Shwartz inequality. □

Combining the above two lemmas, we get an asymptotically tight lower bound on the soundness error that can be obtained by general DPPs over $\mathbb{F}$.

Theorem 4.28 (DPP Soundness Lower Bound)**.** *For every finite field $\mathbb{F}$, there is n and a language $\mathcal{L} \subseteq \mathbb{F}^n$ such that every* DPP *for $\mathcal{L}$ has soundness error $\varepsilon \geq \Omega(1/\sqrt{|\mathbb{F}|})$.*

Proof. Let $\mathcal{L} = \mathbf{UV}$ where $n \geq 2\sqrt{|\mathbb{F}|}$. Consider a DPP for $\mathcal{L}$ over $\mathbb{F}$ with proof length m. We may assume without loss of generality that the query $q = 0$ never occurs (eliminating such a query does not increase the soundness error).

Consider the following randomized prover strategy: pick a random bit b, and then sample $\boldsymbol{v} = (\boldsymbol{x}\|\boldsymbol{\pi})$ as follows:

- If $b = 0$, pick $\boldsymbol{x}$ at random from $\mathbb{F}^n \setminus \mathbf{UV}$ and $\boldsymbol{\pi}$ at random from $\mathbb{F}^m$, as in Lemma 4.26.
- If $b = 1$, let $\boldsymbol{v} = \alpha\boldsymbol{v}_i + \beta\boldsymbol{v}_j$ for random distinct $i, j \in [n]$, as in Lemma 4.27.

We argue that, conditioned on every possible $\boldsymbol{q}$ and A picked by the DPP verifier, the above attack strategy makes the verifier accept an input $\boldsymbol{x} \notin \mathbf{UV}$ with $\Omega(1/\sqrt{|\mathbb{F}|})$ probability. Indeed, letting $t = |A|$, if $t \geq \sqrt{|\mathbb{F}|}$ then Lemma 4.26 guarantees that conditioned on $b = 0$, the attack succeeds with at least $(t-1)/|\mathbb{F}| \geq \Omega(1/\sqrt{|\mathbb{F}|})$ probability. On the other hand, if $t < \sqrt{|\mathbb{F}|}$, then Lemma 4.27 guarantees that conditioned on $b = 1$, the attack succeeds with at least $1/t - 1/n \geq 1/2t \geq \Omega(1/\sqrt{|\mathbb{F}|})$ probability. This implies that conditioned on any choice of $(\boldsymbol{q}, A)$, the attack succeeds with $\varepsilon \geq \Omega(1/\sqrt{|\mathbb{F}|})$ probability. □

Remark 4.29 (DPP with Imperfect Completeness)**.** *While the above analysis assumes perfect completeness, it can be easily extended to* DPP *with a small completeness error. Indeed, the success probability of the random attack is insensitive to the completeness error, whereas the weight-2 attack applies to all "good" choices of $(\boldsymbol{q}, A)$ for which at least (say) $n/2$ of the $\boldsymbol{v}_i$ satisfy $\langle \boldsymbol{q}, \boldsymbol{v}_i \rangle \in A$. By Markov inequality, the probability that $(\boldsymbol{q}, A)$ is good tends to 1 when the completeness error tends to 0. It follows that the soundness error must be $\Omega(1/\sqrt{|\mathbb{F}|})$ even with a sufficiently small constant completeness error.*

5 From FLPCP to DPP Over Large Fields

The FLPCP to DPP transformation from the previous sections leads to proofs of polynomial length in the field size (and hence inverse polynomial in the soundness error). In this section, we provide transformations that preserve the number of field elements in the original FLPCP proof, while enlarging the field size polynomially. Using FLPCPs from the literature (or from Section 4.1), for field size $p > \mathrm{poly}(S/\varepsilon)$, and any arithmetic circuit of size S, we get a DPP for proving that there exists $\boldsymbol{w}$ such that $C(\boldsymbol{x}, \boldsymbol{w}) = 1$, where the proof has $O(S \log \varepsilon^{-1})$ field elements and soundness error ε.

The most efficient version of our transformation results in a *promise* DPP, where soundness only holds for instances taken from the promise subset $\{0,1\} \supseteq \mathcal{L}$ (whereas there is no promise on the proof $\boldsymbol{\pi}^*$).

Definition 5.1 (Promise FLPCP)**.** *Let $\mathbb{F}$ be a field and let $\mathcal{L} \subseteq \{0,1\}^n \subseteq \mathbb{F}^n$. An* FLPCP *for $\mathcal{L}$ is a promise* FLPCP *if it only satisfies the following relaxed soundness requirement:*

- *(Soundness in $\{0,1\}^n$) for every $\boldsymbol{x} \in \{0,1\} \setminus \mathcal{L}$ and all $\boldsymbol{\pi}^* \in \mathbb{F}^m$*

$$\Pr_{(\boldsymbol{Q},A) \leftarrow \mathcal{V}} \left[\boldsymbol{Q} \cdot (\boldsymbol{x} \| \boldsymbol{\pi}^*) \in A \right] \leq s,$$

where s is the soundness error.

Looking ahead, we will start with an FLPCP over $\mathbb{F}_p$ for a language $\mathcal{L} \subseteq \{0,1\}^n$ and transform it to a promise DPP (namely, a 1-query promise FLPCP) for $\mathcal{L}$ over a larger field $\mathbb{F}_{p'} \supseteq \mathbb{F}_p \supseteq \{0,1\}$. We note that since membership in $\{0,1\}$ is efficiently testable, our promise DPPs imply 1-query LPCPs, where the verifier, given the input $\boldsymbol{x}$ can check the promise.

We also obtain a plain (non-promise) DPP for Boolean languages at the cost of increasing the proof length from $O(S \log \varepsilon^{-1})$ to $O(S \log \varepsilon^{-1} + n^2)$ as well as the dependence of p' on S/ε.

Overall approach. Our overall approach builds upon and extends previous transformations from [BCIOP22, BIOW20]:

1. **Construct a bounded** FLPCP**.** First, we construct a promise FLPCP where the magnitude of the answers is substantially smaller than the field size.

2. **Packing** FLPCP **queries into a single query.** Then, we randomly encode the bounded FLPCP queries into a single query.

In [BCIOP22], a classical (Boolean) PCP is used as the underlying bounded LPCP and packing of queries $\boldsymbol{Q} \in \mathbb{F}_p^{k \times m}$ is done by evaluating the linear function $E_{\boldsymbol{Q}}(\boldsymbol{w}) = \boldsymbol{w}^{\intercal}\boldsymbol{Q}$ at a random point $\boldsymbol{w} \in \mathbb{Z}^k$ from some appropriate rectangle. In [BIOW20], a bounded variant of the Hadamard FLPCP is given, and packing is done similarly, but using a certain multilinear polynomial $E_{\boldsymbol{Q}}(\boldsymbol{w})$ and a random point $\boldsymbol{w}$ over an appropriate rectangle. Compared to the [BCIOP22] transformation, the [BIOW20] transformation has certain concrete efficiency benefits and satisfies a notion known as *strong soundness*. However, the proof length suffers from a quadratic loss in the circuit size, stemming from the reliance on the Hadamard FLPCP.

We start by showing that any FLPCP over $\mathbb{F}_p$ can be embedded in a larger field $\mathbb{F}_{p'}$ to yield a bounded promise FLPCP by adding a single random test query, which is folded into one of the existing queries. We also generalize the packing transformation from [BCIOP22], which for 3-query FLPCPs yields better parameters than the variant in [BIOW20]. We note that all of our transformations apply equally well to LPCPs that are not fully linear (namely, we can start with a multi-query LPCP, transform it to a bounded multi-query LPCP, and then to a 1-query LPCP).

Notation and conventions. Throughout this section we rely on the following:

- For $a \in \mathbb{Q}$ and $p, p' \in \mathbb{N}$, we denote by $[a]_p$, the representative of a mod p in $[-p/2, p/2)$. We denote by $[a]_{p',p} = [[a]_{p'}]_p$ (note that the order counts).
- We identify the field $\mathbb{F}_p$ of prime characteristic $p > 2$ with the integers $\left\{-\frac{p-1}{2}, \ldots, \frac{p-1}{2}\right\} = \{[1]_p, \ldots, [p-1]_p\}$. Accordingly for any prime $p' \geq p$, $\mathbb{F}_p \subseteq \mathbb{F}_{p'}$.
- For a matrix $\boldsymbol{Q}$ and vector $\boldsymbol{a}$ of appropriate dimensions, we denote by $[\boldsymbol{Q}\boldsymbol{a}]_{\mathbb{Z}}$ their product over the integers.
- For vectors $\boldsymbol{u} = (u_1, \ldots, u_k)$ and $\boldsymbol{b} = (b_1, \ldots, b_k)$, we write $|\boldsymbol{u}| \leq \boldsymbol{b}$ if $|u_i| \leq b_i$ for every $i \leq k$.
- For matrices $\boldsymbol{Q} \in \mathbb{F}_p^{n \times m}, \boldsymbol{Q}' \in \mathbb{F}_p^{n' \times m}$, we write $(\boldsymbol{Q} \| \boldsymbol{Q}') \coloneqq \begin{pmatrix} \boldsymbol{Q} \\ \boldsymbol{Q}' \end{pmatrix}$ to denote their (vertical) concatenation. This is in particular the case when $m = 1$, where both $\boldsymbol{Q}, \boldsymbol{Q}'$ are vectors, or when $n' = 1$, where $\boldsymbol{Q}'$ is a row vector.
- For a vector $\boldsymbol{b} = (b_1, \ldots, b_k)$, and $i \leq k$ we denote $\boldsymbol{b}^i \coloneqq b_1 \times b_2 \times \cdots \times b_i$.
- We denote by $\mathbf{0}^t$ the zero vector of dimension t.
- We consider FLPCPs (resp., DPPs) for relations where there is an explicit prover $\mathcal{P}$ that takes as input a statement x and a witness w, and outputs a FLPCP (resp., a DPP) proof π. We write $\langle \mathcal{P}, \mathcal{V} \rangle$ to denote such a FLPCP (resp., DPP), where $\mathcal{P}$ denotes the prover and $\mathcal{V}$ denotes the verifier.

Definition 5.2 (Bounded FLPCP)**.** *Let $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^h$ be a relation, and let $\boldsymbol{b} \in \mathbb{Z}_+^k$ be a bound. A k-query* FLPCP *$\langle \mathcal{P}, \mathcal{V} \rangle$ over $\mathbb{F}_p$ for $\mathcal{R}$ with proof length m is $\boldsymbol{b}$-bounded, if for any $(\boldsymbol{x}, \boldsymbol{w}) \in \mathcal{R}$, proof $\boldsymbol{\pi}$ in the support of $\mathcal{P}(\boldsymbol{x}, \boldsymbol{w})$, and every verifier query $\boldsymbol{Q} \in \mathbb{F}_p^{k \times (n+m)}$, it holds that $|[\boldsymbol{Q}(\boldsymbol{x} \| \boldsymbol{\pi})]_{\mathbb{Z}}| \leq \boldsymbol{b}$ (namely, their product over the integers is strictly smaller in absolute value than $\boldsymbol{b}$, coordinate-wise).*

We note that any FLPCP is without loss of generality $\boldsymbol{b}$-bounded for

$$\boldsymbol{b} = \left((n+m) \left(\frac{p-1}{2} \right)^2, \ldots, (n+m) \left(\frac{p-1}{2} \right)^2 \right) .$$

We observe that any FLPCP can be made to have (honest) Boolean proofs by a simple binary-encoding transformation, at the cost of a logarithmic factor in the proof size.

Claim 5.3 (Boolean Proofs)**.** *Assume $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^h$ has a k-query promise* FLPCP *over $\mathbb{F}_p$ with proof length m, then $\mathcal{R}$ has a k-query promise* FLPCP *over $\mathbb{F}_p$ with proofs in $\{0,1\}^{m \log p}$, and the same accepting set. In particular, the resulting* FLPCP *is $\boldsymbol{b}$-bounded for bound*

$$\boldsymbol{b} = \left((n+m)\frac{p-1}{2}, \ldots, (n+m)\frac{p-1}{2} \right) .$$

Proof. Any proof $\boldsymbol{\pi} \in \mathbb{F}_p^m$ is translated to a new proof $\widehat{\boldsymbol{\pi}} \in \{0,1\}^{m \log p} \subseteq \mathbb{F}_p^{m \log p}$, which is the binary encoding of $\boldsymbol{\pi}$. Any verifier query $\boldsymbol{q} = (\boldsymbol{q}^x, \boldsymbol{q}^\pi) \in \mathbb{F}_p^n \times \mathbb{F}_p^m$ is replaced by a query $\widehat{\boldsymbol{q}} = (\boldsymbol{q}^x, \widehat{\boldsymbol{q}}^\pi) \in \mathbb{F}_p^n \times \mathbb{F}_p^{m \log p}$ where $\widehat{\boldsymbol{q}}^\pi = [\boldsymbol{q}^\pi \boldsymbol{M}]_p$ and $\boldsymbol{M} \in \mathbb{Z}^{m \times m \log p}$ is a matrix that given a binary encoding of a vector $\widehat{v} \in \{0,1\}^{m \log p}$ of a vector $v \in \mathbb{F}_p^m$, reconstructs v.

It follows readily that this transformation preserves completeness, soundness, and satisfies the claimed boundedness. □

While below we favor better soundness at the cost of logarithmic blowup in proof size, it is possible in all of our constructions to avoid this logarithmic blowup at the cost of worst soundness.

Evasive accepting sets. We rely on the following *evasive accepting set* property.

Definition 5.4 (Evasive Accepting Set). *A k-query ε-sound FLPCP over $\mathbb{F}_p$ has an evasive accepting set if for any no-instance and any fixing of answers $a_2, \ldots, a_k$, sampling $a_1 \leftarrow \mathbb{F}_p$ uniformly at random leads to acceptance with probability at most ε.*

We note that an evasive accepting set is quite natural. Any ε-sound DPP trivially has an evasive accepting set. Also, all FLPCPs constructed in this paper have an evasive accepting set. In particular the sets $\{(ab, a, b) \in \mathbb{F}_p^3\}$ and $\{(a^2, a) \in \mathbb{F}_p^2\}$ we consider are evasive for $\varepsilon > 1/p$.

We now prove that any FLPCP over any prime order $\mathbb{F}_p$ with Boolean (honest) proofs and an evasive accepting set can be turned into a $\boldsymbol{b}$-bounded promise FLPCP over $\mathbb{F}_{p'}$ for a (non-trivial) bound $\|\boldsymbol{b}\|_\infty \ll p'$.

Hyperplane-evading probability. One quantity we will be interested in later on for the sake of DPP and 1-query LPCP applications is the *hyperplane evading probability* of the resulting FLPCP queries, which intuitively bounds the probability of gaining information about a query $\boldsymbol{q}$ by learning linear equations $\langle \boldsymbol{q}, \boldsymbol{v} \rangle$ for arbitrarily chosen $\boldsymbol{v}$.

Definition 5.5 (Hyperplane-Evading Probability). *We say that the hyperplane-evading probability of an* FLPCP *query $\boldsymbol{q}_i = (\boldsymbol{q}^x \| \boldsymbol{q}^\pi) \in \mathbb{F}_p^n \times \mathbb{F}_p^m$ is δ if conditioned on any fixing of the rest of the queries $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{i-1}, \boldsymbol{q}_{i+1}, \ldots, \boldsymbol{q}_k$ and any affine hyperplane $(\boldsymbol{v}, \alpha) \in (\mathbb{F}_p^m \setminus \{\boldsymbol{0}^m\}) \times \mathbb{F}_p$ the probability that $\langle \boldsymbol{q}^\pi, \boldsymbol{v} \rangle = \alpha$ is at most δ.*

We say that a DPP *(or 1-query* LPCP*) is δ-evasive if its (single) query has hyperplane-evading probability (at most) δ.*

We now state our main theorem regarding bounded FLPCPs from unbounded ones.

Theorem 5.6 (Bounded Embedding). *Let $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^h$ be a relation. Let $p, p', m, \mu, \gamma, k \in \mathbb{N}, \varepsilon \in [p^{-1}, 1)$ be such that p, p' are primes, $\mu = m + n$, $p' \geq \mu^2 \gamma p^2 \varepsilon^{-2}$, $\gamma \geq 100 \mu p \varepsilon^{-1}$, and $\mu \geq 400$. Then any k-query ε-sound promise* FLPCP *over $\mathbb{F}_p$ for $\mathcal{R}$ with Boolean (honest) proofs of length m and an evasive accepting set, can be transformed into a k-query 1.01ε-sound, $\boldsymbol{b}$-bounded promise* FLPCP *over $\mathbb{F}_{p'}$ for $\mathcal{R}$ with the same proof length m and bound $\boldsymbol{b} = \left(\frac{\mu(p\gamma - 1)}{2}, \frac{\mu(p-1)}{2}, \ldots, \frac{\mu(p-1)}{2} \right)$. The proof length and prover complexity are preserved and the verifier complexity is preserved up to a $\mathrm{poly}(\log p, k)$ additive term. The hyperplane-evading probability of the first query is γ^{-1}.*

Packing FLPCP. The gap between the bound and field size allows us to pack the FLPCP queries into one query. The following theorem is a generalization of a Boolean packing theorem from [BCIOP22] and a variant of a non-Boolean packing theorem from [BIOW20].

Theorem 5.7 (Packing). *Let $\boldsymbol{b} \in \mathbb{N}^k$ and let $p > ((2\boldsymbol{b})^k)^2/\varepsilon$ be a prime. Any k-query $(\boldsymbol{b}-\mathbf{1})$-bounded (promise)* FLPCP *over $\mathbb{F}_p$ with soundness error ε can be transformed into a (promise)* DPP *over $\mathbb{F}_p$ with soundness error 2ε. The proof length and prover complexity are preserved and the verifier complexity is also preserved up to an additive term of* $\mathrm{poly}(\log p, k)$*.*

We also recall the packing theorem from [BIOW20] for the case $k = 2$, for which it is favorable.

Theorem 5.8 (Two-Query Packing [BIOW20]). *Let $\boldsymbol{b} = (\bar{b}, \underline{b})$ where $\underline{b} \leq \bar{b}$ and let $p > 8\underline{b}^2\bar{b}/\varepsilon$ be a prime. Any 2-query $\boldsymbol{b}$-bounded (promise)* FLPCP *over $\mathbb{F}_p$ with soundness error ε can be transformed into a (promise)* DPP *over $\mathbb{F}_p$ with soundness error $3\varepsilon/2$. If the first query of the* FLPCP *has hyperplane-evading probability δ, the resulting DPP is δ-evasive. The proof length and prover complexity are preserved and the verifier complexity is also preserved up to an additive term of* $\mathrm{poly}(\log p)$*.*

We note that the evasiveness property is not explicit in [BIOW20], but follows readily from the fact that the [BIOW20] 2-query packing has the form $\boldsymbol{q}_1 + \alpha\boldsymbol{q}_2$ for some appropriately sampled scalar α.

Remark 5.9 (Embedding and Packing for LPCPs). *Theorems 5.6 to 5.8 also hold for the case of* LPCP*s (that are not fully linear). In this case, the parameter μ from Theorem 5.6 can be replaced with the proof length m. Additionally, in this case Theorem 5.6 can be applied to LPCPs for relations $\mathcal{R} \in \mathbb{F}_p^n \times \mathbb{F}_p^h$ rather than just Boolean relations. This, in particular, allows starting from* FLPCP*s for arithmetic, rather than Boolean, circuits over $\mathbb{F}_p$.*

Corollaries for DPPs. We now derive several corollaries regarding the construction of DPPs from FLPCPs.

The next corollary starts from a 2-query FLPCP and relies on the packing transformation of [BIOW20] from Theorem 5.8.

Corollary 5.10 (Evasive DPP from 2-FLPCP). *Assume there exists a 2-query promise* FLPCP *over $\mathbb{F}_p$ for $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^h$ with an evasive accepting set, proofs of length m, and soundness error $\varepsilon = \frac{c\mu}{p}$, where $\mu = n + m \geq \frac{400}{\log p}$ and $c \geq 1$ is a constant. Then for $\gamma > \frac{100}{c}\mu^2 p^2 \log p$, there exists a promise* DPP *over $\mathbb{F}_{p'}$ with soundness error $\varepsilon' = \frac{1.52c\mu}{p}$ for $p' > \mu^2 p^4 \gamma (\log p)^3/c$ and proof length $m \log p$. The* DPP *is γ^{-1}-evasive. The prover complexity is preserved and the verifier complexity is preserved up to a* $\mathrm{poly}(\log p', k)$ *additive term. The same also holds for* LPCP *and yields a single-query* LPCP *rather than a* DPP*. Here $\mu = n + m$ is replaced with m.*

Proof. By Claim 5.3, we can assume the proof is Boolean of length $\hat{\mu} = \mu \log p$, and note that $\hat{\mu} \geq 400$. Set $\gamma = 100\hat{\mu}p\varepsilon^{-1}$. Then, provided p' is large enough, the 2-query FLPCP resulting from Theorem 5.6 has soundness error 1.01ε and is bounded by $\left(\frac{\hat{\mu}p\gamma}{2}, \frac{\hat{\mu}p}{2}\right)$. Applying Theorem 5.8, we obtain a DPP with soundness error 1.52ε over any field of size

$$p' \geq 8\frac{\hat{\mu}p\gamma}{2}\left(\frac{\hat{\mu}p}{2}\right)^2 \varepsilon^{-1} = \mu^2 p^4 \gamma (\log p)^3/c \ .$$

Note that since $p \geq \varepsilon^{-1}$, $p' \gg \hat{\mu}^2 p^2 \gamma \varepsilon^{-2}$, hence Theorem 5.6 applies.

Since the bounded FLPCP we obtain from Theorem 5.6 has a first query with hyperplane-evading probability γ^{-1}, it follows that the DPP is γ^{-1}-evasive. □

Plugging in the 2-query FLPCP from Corollary B.6 (implicit in [DFGK14]) we obtain the following corollary.

Corollary 5.11 (Evasive DPP for Boolean Circuit Satisfiability). *Let $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit of size $s > 40$, counting both inputs and fan-in 2* NAND *gates, and let $\varepsilon > 0$ be a soundness bound. Let $\gamma > 3850\varepsilon^{-2}(n+2s)^4 \log \frac{3.1(n+2s)}{\varepsilon}$ and let $p > 93\gamma\varepsilon^{-4}(n+2s)^6 \left(\log \frac{3.1(n+2s)}{\varepsilon}\right)^3$ be a prime. Then, the language $\left\{\boldsymbol{x} \in \{0,1\}^n : \exists \boldsymbol{w} \in \{0,1\}^h : C(\boldsymbol{x}, \boldsymbol{w}) = 1\right\}$ has a promise* DPP *of proof length $2s \log \frac{3.1(n+2s)}{\varepsilon}$ over $\mathbb{F}_p$ with soundness error ε. The DPP is γ^{-1}-evasive. For the case of a 1-query LPCP, $n+2s$ can be replaced with $2s$.*

We also state a more general corollary that starts with any k-query FLPCP and applies the packing from Theorem 5.7, which is favorable for $k \geq 3$.

Corollary 5.12 (DPP from FLPCP). *Assume there exists a k-query promise* FLPCP *over $\mathbb{F}_p$ for $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^h$ with an evasive accepting set, proofs of length m, and soundness error $\varepsilon = \frac{c\mu}{p}$, where $\mu = n + m \geq \frac{400}{\log p}$ and $c \geq 1$ is a constant. Then there exists a promise* DPP *over $\mathbb{F}_{p'}$ with soundness error $\varepsilon' = \frac{2.02c\mu}{p}$ for $p' > 10^4(\mu p \log p)^{2k+5}/c^3$ and proof length $m \log p$. The prover complexity is preserved and the verifier complexity is preserved up to a* $\operatorname{poly}(\log p', k)$ *additive term. The same also holds for* LPCP *and yields a single-query* LPCP *rather than a* DPP. *Here $\mu = n+m$ is replaced with m.*

Proof. By Claim 5.3, we can assume the proof is Boolean of length $\hat{\mu} = \mu \log p$, and note that $\hat{\mu} \geq 400$. Set $\gamma = 100\hat{\mu}p\varepsilon^{-1}$. Then, provided p' is large enough, the k-query FLPCP resulting from Theorem 5.6 has soundness error 1.01ε and is bounded by $\left(\frac{\hat{\mu}(p\gamma-1)}{2}, \frac{\hat{\mu}(p-1)}{2}, \ldots, \frac{\hat{\mu}(p-1)}{2}\right)$. In particular, it is $(\boldsymbol{b}-\boldsymbol{1})$-bounded for $\boldsymbol{b} = \left(\frac{\hat{\mu}p\gamma}{2}, \frac{\hat{\mu}p}{2}, \frac{\hat{\mu}p}{2}, \ldots, \frac{\hat{\mu}p}{2}\right)$. Applying Theorem 5.7, we obtain a DPP with soundness error 2.02ε over any field of size

$$p' \geq \left(\hat{\mu}^k p^k \gamma\right)^2 \varepsilon^{-1} = 10^4(\hat{\mu}p)^{2(k+1)}\varepsilon^{-3} = 10^4(\hat{\mu}p)^{2k+5}/c^3 = 10^4(\mu p \log p)^{2k+5}/c^3 \enspace .$$

Note that $p' \gg \hat{\mu}^2 p^2 \gamma \varepsilon^{-2}$, hence Theorem 5.6 applies. □

5.1 Bounded Embedding (Proof of Theorem 5.6)

In this section we prove Theorem 5.6. We start by describing the construction, which is essentially identical to the original construction, except that the verifier adds to one of the queries a random *bound test* and checks appropriate boundedness of the answer corresponding answer.

Construction 5.13 (FLPCP Embedding). *Given an ε-sound* FLPCP $\langle \mathcal{P}, \mathcal{V} \rangle$ *for $\mathcal{R}$ over $\mathbb{F}_p$ with an evasive accepting set, we construct a new* FLPCP $\langle \mathcal{P}', \mathcal{V}' \rangle$ *for $\mathcal{R}$ over $\mathbb{F}_{p'}$.*

- *Each proof $\boldsymbol{\pi} \in \mathbb{F}_p^m$ for $\boldsymbol{x} \in \mathbb{F}_p^n$ is interpreted as a proof over $\mathbb{F}_{p'} \supseteq \mathbb{F}_p$.*

- *The verifier $\mathcal{V}'$ samples queries $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k \in \mathbb{F}_p^\mu$ according to the original verifier $\mathcal{V}$, as well a random vector $\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m$, and sets $\boldsymbol{u}' = (\boldsymbol{0}_n \| \boldsymbol{u}) \in \mathbb{F}_{p'}^\mu$. The new queries are then $\boldsymbol{q}_1' = \boldsymbol{q}_1 + p\boldsymbol{u}', \boldsymbol{q}_2' = \boldsymbol{q}_2, \ldots, \boldsymbol{q}_k' = \boldsymbol{q}_k$.*
- *The verifier $\mathcal{V}'$ accepts answer $\boldsymbol{a}' \in \mathbb{F}_{p'}^k$ iff $a_1' \leq b_1 := \frac{\mu(p\gamma-1)}{2}$, for all other i, $a_i' \leq b_2 := \frac{\mu(p-1)}{2}$ and $\mathcal{V}$ accepts $\boldsymbol{a} = \boldsymbol{a}' \bmod p$.*

5.1.1 Analysis of Construction 5.13

We prove that Construction 5.13 satisfies the requirements of Theorem 5.6.

Completeness and boundedness. The completeness of the construction follows directly from the completeness of the underlying FLPCP and the fact that it has Boolean proofs. Indeed, for any proof $\boldsymbol{x}, \boldsymbol{\pi} \in \{0,1\}^n \times \{0,1\}^m$ and any $\boldsymbol{q}' \in \mathbb{F}_p^\mu + p\mathbb{F}_\gamma^\mu$,

$$|\langle \boldsymbol{q}', (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle| \leq \mu \left(\frac{p-1}{2} + p\frac{\gamma-1}{2} \right) = \frac{\mu(p\gamma-1)}{2} \quad ,$$

and for any $\boldsymbol{q}' \in \mathbb{F}_p^\mu$,

$$|\langle \boldsymbol{q}', (\boldsymbol{x} \| \boldsymbol{\pi}) \rangle| \leq \mu \left(\frac{p-1}{2} \right) \quad .$$

Hyperplane evasiveness. To see that the first query $\boldsymbol{q}_1'$ has hyperplane evading probability γ^{-1}, recall that $\boldsymbol{q}' = (\boldsymbol{q}^x, \boldsymbol{q}^\pi)$ where $\boldsymbol{q}^\pi = \boldsymbol{q}_1^\pi + p\boldsymbol{u}'$, where $\boldsymbol{u}'$ is uniformly random in $\mathbb{F}_\gamma^m$, independently of all other queries, and p is invertible modulo p'. Hence for any $\boldsymbol{v} \neq \boldsymbol{0}$ and α, $\langle \boldsymbol{q}^\pi, \boldsymbol{v} \rangle = \alpha$ with probability at most γ^{-1}.

Soundness. The challenging part is proving soundness of the transformation. The challenge stems from the fact that a malicious proof may have "unbounded" entries over $\mathbb{F}_{p'}$ whereas soundness is only a-priori guaranteed for proofs with $\mathbb{F}_p$ entries (the input, in contrast, is promised to be in $\{0,1\}^n$). We prove that there are essentially two options for such a malicious proof:

1. It can be *close* to a proof over $\mathbb{F}_p$ up to relatively small p' fractions. In this case, we show that answers either exceed the specified bound, behave as an answer according to some fixed proof over $\mathbb{F}_p$, or behave uniformly at random over $\mathbb{F}_p$.
2. It can be far from any such proof, in which case we prove that it will be caught by the added random tests.

Following this high-level approach requires a careful analysis and characterization of closeness to p'-fractions. We proceed to the proof.

Definition 5.14 (Δ-Closeness)**.** *For $\Delta, \zeta \in \mathbb{N}$, we say that $a \in \mathbb{Z}$ is Δ-close to a (p', ζ)-fraction if there exists an integer $1 \leq r < \zeta$, such that $|[ar]_{p'}| \leq \Delta$. We refer to r as the denominator of the fraction. We say that a vector $\boldsymbol{\pi}^* \in \mathbb{F}_{p'}^\mu$ is Δ-close to a (p', ζ)-fraction if each of its entries is Δ-close to a (p', ζ)-fraction.*

Interpreting Δ-closeness. The above definition intuitively says that the integer a is close to a "p'-fraction" $\frac{d}{r}p'$, for some small r and enumerator $d < r$. Intuitively, by choosing some of its proof coordinates to be close to such fractions, a malicious proof may still pass as bounded (for instance if $r = 2$, this will happen with probability 1/2 over a random query). As we shall see, we will be able to characterize whether the proof behaves essentially as an $\mathbb{F}_p$ proof or fails the random test according to its closeness to p'-fractions.

We next show that if the scaled (malicious) proof vector $p\boldsymbol{\pi}^*$ is Δ-close to a (p', p)-fraction and satisfies a certain "small-lcm" condition, then the probability of acceptance is bounded by the soundness error. We then show that any other case is discovered with high probability by the added random test.

Proofs close to (p', p)-fractions with a small lowest common multiple (lcm). We prove the following claim:

Claim 5.15. *Assume $p\boldsymbol{\pi}^*$ is Δ-close to a (p', p)-fraction, with denominators $r_1, \dots, r_m$, and further assume that $L \coloneqq \mathsf{lcm}(r_1, \dots, r_m) \le \frac{p'}{\mu\gamma p\Delta}$ and $p < \Delta$. Then for any no-instance $\boldsymbol{x} \in \mathbb{F}_p^n$ and proof $\boldsymbol{\pi}^* \in \mathbb{F}_p^m$, the verifier $\mathcal{V}'$ accepts $(\boldsymbol{x}||\boldsymbol{\pi}^*)$ with probability at most $\varepsilon + \frac{p}{\gamma}$.*

Proof. Throughout, let $p\pi_i^* = \frac{d_i p' + e_i}{r_i}$, where $1 \le r_i < p$, $|e_i| \le \Delta$. It will be convenient to think of all k queries as having the form $\boldsymbol{q}' = (\boldsymbol{q}^x \| \boldsymbol{q}^\pi + p\boldsymbol{u})$ where for $\boldsymbol{q}_1'$, $\boldsymbol{u}_1 \leftarrow \mathbb{F}_\gamma^m$ is random, whereas for all other $\boldsymbol{q}_i'$, $\boldsymbol{u}_i = \boldsymbol{0}$. Then we can write:

$$\langle \boldsymbol{q}', \boldsymbol{x} \| \boldsymbol{\pi}^* \rangle = \langle \boldsymbol{q}^x, \boldsymbol{x} \rangle + \frac{\alpha}{pL} p' + \sum_{i \in [m]} (q_i^\pi + p u_i) \cdot \frac{e_i}{p r_i} \ ,$$

where $\alpha = \alpha(\boldsymbol{q}^\pi, \boldsymbol{u})$ is an integer.

First note that if $\alpha / pL \notin \mathbb{Z}$, then

$$|\, [\langle \boldsymbol{q}', (\boldsymbol{x} \| \boldsymbol{\pi}^*) \rangle]_{p'} \,| \ge p'/pL - \mu \frac{p\gamma}{2} \frac{\Delta}{p} \ge \frac{\mu\gamma\Delta}{2} > \frac{\mu\gamma p}{2} \ , \qquad \square$$

in which case the query exceeds the bound b_1, and the verifier rejects.
Otherwise, $\alpha / pL \in \mathbb{Z}$ and

$$\langle \boldsymbol{q}', (\boldsymbol{x} \| \boldsymbol{\pi}^*) \rangle = \langle \boldsymbol{q}^x, \boldsymbol{x} \rangle + \sum_{i \in [m]} (q_i^\pi + p u_i) \cdot \frac{e_i}{p r_i} \bmod p' \ .$$

Moreover,

$$\left| \langle \boldsymbol{q}^x, \boldsymbol{x} \rangle + \sum_{i \in [m]} (q_i^\pi + p u_i) \cdot \frac{e_i}{p r_i} \right| \le \frac{\mu\gamma\Delta}{2} < p'/2 \ .$$

Thus, defining $\pi_i \coloneqq e_i r_i^{-1} \bmod p$, where $r_i^{-1} \in \mathbb{F}_p^*$ is the multiplicative inverse of r_i, we have:

$$\begin{aligned}
\left[\left\langle \boldsymbol{q}', (\boldsymbol{x}\|\boldsymbol{\pi}^*)\right\rangle\right]_{p',p} &= \left[\langle \boldsymbol{q}^x, \boldsymbol{x}\rangle + \sum_{i\in[m]} (q_i^\pi + pu_i)\cdot \frac{e_i}{pr_i}\right]_{p',p} \\
&= \left[\langle \boldsymbol{q}^x, \boldsymbol{x}\rangle + \sum_{i\in[m]} (q_i^\pi + pu_i)\cdot \frac{e_i}{pr_i}\right]_{p} \\
&= \left[\langle \boldsymbol{q}^x, \boldsymbol{x}\rangle + \left(\sum_{i\in[m]} (q_i^\pi + pu_i)\cdot \frac{e_i}{pr_i}\right)\cdot \prod_{i=1}^{m} (r_i r_i^{-1})\right]_p \\
&= \left[\langle \boldsymbol{q}^x, \boldsymbol{x}\rangle + \sum_{i\in[m]} (q_i^\pi + pu_i)\cdot \frac{e_i r_i^{-1}}{p}\cdot \prod_{j\neq i}^{m}(r_j r_j^{-1})\right]_p \\
&= \left[\langle \boldsymbol{q}^x, \boldsymbol{x}\rangle + \sum_{i\in[m]} q_i^\pi \pi_i/p + u_i\pi_i\right]_p \\
&= [\langle (\boldsymbol{q}^x||\boldsymbol{q}^\pi), (\boldsymbol{x}\|\boldsymbol{\pi}/p)\rangle + \langle \boldsymbol{u},\boldsymbol{\pi}\rangle]_p \ .
\end{aligned}$$

Accordingly, considering all k queries $\boldsymbol{q}'_1, \ldots, \boldsymbol{q}'_k$ and defining $a_j = \left[\left\langle (\boldsymbol{q}_j^x||\boldsymbol{q}_j^\pi), (\boldsymbol{x}\|\boldsymbol{\pi}/p)\right\rangle + \langle \boldsymbol{u}_j, \boldsymbol{\pi}\rangle\right]_p$, it suffices to show that $\boldsymbol{a}$ is accepted by the underlying verifier $\mathcal{V}$ with probability at most $\varepsilon + \frac{p}{\gamma}$. We consider three cases:

1. $\boldsymbol{\pi} = \boldsymbol{0} \bmod p$: in this case, $[\boldsymbol{\pi}/p]_p$ is a proof in $\mathbb{F}_p^m$ and $a_j = \left[\left\langle (\boldsymbol{q}_j^x||\boldsymbol{q}_j^\pi), (\boldsymbol{x}\|\boldsymbol{\pi}/p)\right\rangle\right]_p$. Hence, the bound follows from the ε-soundness of the underlying FLPCP.
2. $\boldsymbol{\pi} \neq \boldsymbol{0} \bmod p$ and $\langle (\boldsymbol{q}_1^x||\boldsymbol{q}_1^\pi), (\boldsymbol{x}\|\boldsymbol{\pi}/p)\rangle \in \mathbb{Z}$: in this case, since $\boldsymbol{u}_1$ is uniform over $\mathbb{F}_\gamma$, $\langle \boldsymbol{u}_1, \boldsymbol{\pi}\rangle$ is $\frac{p}{\gamma}$-close to uniform over $\mathbb{F}_p$, and thus so is a_1. Since the accepting set of the underlying FLPCP is evasive, the verifier overall accepts with probability at most $\varepsilon + \frac{p}{\gamma}$.
3. For some $j \in [k]$, $\left\langle (\boldsymbol{q}_j^x||\boldsymbol{q}_j^\pi), (\boldsymbol{x}\|\boldsymbol{\pi}/p)\right\rangle \notin \mathbb{Z}$: in this case the corresponding $\alpha(\boldsymbol{q}_j^\pi, \boldsymbol{u}_j) \notin \mathbb{Z}$, in which case we have shown that the boundedness condition is violated.

This concludes the proof of Claim 5.15. □

We now move on to prove that in all cases not covered by the above, answer a'_1 will fall outside the threshold bound $b_1 = \frac{\mu(p\gamma-1)}{2}$ with high probability. In the following claims, fix any $\boldsymbol{q}^x, \boldsymbol{q}^\pi$, and consider the random variable $\boldsymbol{q}'(\boldsymbol{u}) := (\boldsymbol{q}^x\|\boldsymbol{q}^\pi + p\boldsymbol{u})$ given by sampling $\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m$.

Catching proofs that are far from (p', γ)-fractions. We show that if $p\boldsymbol{\pi}^*$ is not close to a (p', γ)-fraction, then the corresponding answers fall almost always outside the threshold bounds.

Claim 5.16. *If $p\boldsymbol{\pi}^*$ is* **not** *Δ-close to a (p', γ)-fraction, then:*

$$\Pr_{\boldsymbol{u}\leftarrow\mathbb{F}_\gamma}\left[\left|\, \left[\left\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\right\rangle\right]_{p'} \right| \leq b_1\right] \leq \frac{\mu p}{\Delta} + \frac{1}{\gamma} \ .$$

Proof. Assume $p\boldsymbol{\pi}^*$ has an entry that is not Δ-close to a (p',γ)-fraction, and assume without loss of generality it is $p\pi_1^*$. Then the p'-modular distance between $p\pi_1^* u$ and $p\pi_1^* u'$, for any $u \neq u' \in \mathbb{F}_\gamma$, is at least Δ; i.e., $|\,[p\pi_1^*(u-u')]_{p'}\,| > \Delta$. This implies that, considering $\boldsymbol{u} = (u_1,\dots,u_m) \leftarrow \mathbb{F}_\gamma^m$, and conditioning on any $v = \langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\rangle - pu_1\pi_1^*$, there exist at most $\lceil 2b_1/\Delta \rceil$ values for u_1 such that

$$|\,[\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\rangle]_{p'}\,| = |\,[v + pu_1\pi_1^*]_{p'}\,| \le b_1 \ .$$

Since u_1 is uniform on $\mathbb{F}_\gamma$ independently of v, the above occurs with probability at most

$$\left(1 + \frac{2b_1}{\Delta}\right)/\gamma \le \frac{\mu p}{\Delta} + \frac{1}{\gamma} \ .$$

□

From here on, we assume that $p\boldsymbol{\pi}^*$ is Δ-close to a (p',γ)-fraction with denominators $r_1,\dots,r_m$.

Catching proofs with large denominators. We show that if the denominators $r_1,\dots,r_m$ are large, this will be discovered by the bound test.

Claim 5.17. *Let $r = \max_i\{r_i\}$ and assume $r \ge 2$, $p' > \mu\Delta\gamma p$, then:*

$$\Pr_{\boldsymbol{u}\leftarrow\mathbb{F}_\gamma}\left[|\,[\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\rangle]_{p'}\,| \le b_1\right] \le \frac{1}{r} + \frac{1}{\gamma} + \frac{4}{\Delta} \ .$$

Proof. Assume without loss of generality $r_1 = r$, and recall that $p\pi_1^* = \frac{d_1 p' + e_1}{r_1}$, for some $1 \le r_1 < \gamma$, $|e_1| \le \Delta$. We can assume without loss of generality that $\gcd(d_1, r_1) = 1$, indeed since $p\pi^* \in \mathbb{Z}$, $\gcd(d_1, r_1)$ divides also e_1, so we can reduce the fraction while keeping $e_1 < \Delta$. Next, note that the p'-modular distance between any two multiples $\alpha\frac{p'}{r_1}$ and $\alpha'\frac{p'}{r_1}$, such that $\alpha \neq \alpha' \bmod r_1$, is at least p'/r_1. Now, recalling that $|u_1\frac{e_1}{r_1}| \le \frac{\gamma}{2}\frac{\Delta}{r_1}$ for any $u_1 \in \mathbb{F}_\gamma$, it follows that, conditioned on any $v = \langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\rangle - pu_1\pi_1^*$, there are at most $\lceil 2b_1/\left(\frac{p'}{r_1} - \frac{\gamma\Delta}{2r_1}\right)\rceil$ values for $\alpha := u_1 d_1 \bmod r_1$ such that

$$|\,[\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\rangle]_{p'}\,| = \left|\left[v + \frac{u_1 d_1 p' + e_1}{r_1}\right]_{p'}\right| \le b_1 \ .$$

However, recalling that d_1 is invertible mod r_1, we have:

$$\Pr_{u_1\leftarrow\mathbb{F}_\gamma}\left[u_1 = \alpha d_1^{-1} \bmod r_1\right] \le \frac{\lfloor\frac{\gamma}{r_1}\rfloor + 1}{\gamma} \le \frac{1}{r_1} + \frac{1}{\gamma} \ .$$

Overall,

$$\begin{aligned}
\Pr_{\boldsymbol{u}\leftarrow\mathbb{F}_\gamma}\left[|\,[\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x}\|\boldsymbol{\pi}^*\rangle]_{p'}\,| \le b_1\right] &\le \left(\frac{4b_1 r_1}{2p' - \gamma\Delta} + 1\right)\left(\frac{1}{r_1} + \frac{1}{\gamma}\right) \\
&\le \left(\frac{2\mu p\gamma r_1}{2\mu\Delta\gamma p - \gamma\Delta} + 1\right)\left(\frac{1}{r_1} + \frac{1}{\gamma}\right) \\
&\le \left(\frac{2r_1}{\Delta} + 1\right)\left(\frac{1}{r_1} + \frac{1}{\gamma}\right) \\
&\le \frac{4}{\Delta} + \frac{1}{r_1} + \frac{1}{\gamma} \ .
\end{aligned}$$

□

Catching proofs with small denominators and large lcm. Next, we show that if $\max_i\{r_i\}$ is small, but the lcm is large, then this will be caught by the bound test with high probability.

Claim 5.18. *Let ρ be such that $\max_i\{r_i\} \leq \rho \leq L := \mathsf{lcm}(r_1, \ldots, r_m)$, and assume $p' \geq \mu\Delta\gamma\rho^2$, $\Delta \geq 2p$. Then:*

$$\Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma}\left[\left|\,\left[\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x} \| \boldsymbol{\pi}^* \rangle\right]_{p'}\right| \leq b_1\right] \leq \frac{m\rho}{\gamma} + \frac{1}{\rho} \enspace .$$

Proof. First, since $\max_i\{r_i\} \leq \rho \leq L$, we can choose a subset $S \subseteq \{r_1, \ldots, r_m\}$ with $\ell = \mathsf{lcm}(S)$, such that $\rho \leq \ell \leq \rho^2$. Let us assume without loss of generality that $S = \{r_1, \ldots, r_s\}$ for some $s \leq m$, and that $\gcd(r_i, d_i) = 1$, where d_i is such that $p\pi_i^* = \frac{d_i p' + e_i}{r_i}$, and $|e_i| \leq \Delta$. Now, note that the p'-modular distance between any two multiples $\alpha\frac{p'}{\ell}$ and $\alpha'\frac{p'}{\ell}$, such that $\alpha \neq \alpha' \bmod \ell$, is at least $p'/\ell \geq p'/\rho^2$. In addition, for any $\boldsymbol{u} \in \mathbb{F}_\gamma^m$,

$$\left|\sum_{r_i \in S} u_i \cdot \frac{e_i}{r_i}\right| \leq s\frac{\gamma}{2}\Delta \leq m\frac{\gamma}{2}\Delta \enspace .$$

Also, note that

$$2b_1 \leq \mu p\gamma \leq \frac{\mu\Delta\gamma}{2} \leq \frac{p'}{\rho^2} - m\frac{\gamma}{2}\Delta \enspace .$$

It follows that, conditioned on any $v = \langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x} \| \boldsymbol{\pi}^* \rangle - \sum_{r_i \in S} p u_i \pi_i^*$, there exists at most one value for $\alpha(\boldsymbol{u}) := \ell \sum_{r_i \in S} \frac{u_i d_i}{r_i} \bmod \ell$ such that

$$\left|\langle \boldsymbol{q}'(\boldsymbol{u}), \boldsymbol{x} \| \boldsymbol{\pi}^* \rangle]_{p'}\right| = \left|\left[v + \sum_{r_i \in S} u_i \cdot \frac{d_i p' + e_i}{r_i}\right]_{p'}\right| = \left|\left[v + \frac{\alpha(\boldsymbol{u})}{\ell}p' + \sum_{r_i \in S} u_i \cdot \frac{e_i}{r_i}\right]_{p'}\right| \leq b_1 \enspace .$$

We will now show that, for $\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m$, the distribution of $\alpha(\boldsymbol{u})$ is $\frac{m\rho}{\gamma}$-statistically close to being uniform modulo ℓ, which will conclude the proof, since $\ell \geq \rho$.

Recall that

$$\alpha(\boldsymbol{u}) := \ell \sum_{r_i \in S} \frac{u_i d_i}{r_i} \enspace ,$$

and consider the factorization $\ell = f_1 \times \cdots \times f_s$, such that:

1. $f_1, \ldots, f_s$ are pairwise co-prime,
2. $\gcd\left(f_i, \frac{\ell d_i}{r_i}\right) = 1$,
3. $f_i \leq r_i$.

To find such a factorization, just factor ℓ to primes $p_1^{a_s}, \ldots, p_s^{a_s}$, and take f_i to be all the prime powers that are maximal in r_i, and were not already taken for $j < i$ (we assume without loss of generality that each r_i has such maximal power, or we can drop it without changing the lcm). The fact that $\gcd\left(f_i, \frac{\ell d_i}{r_i}\right) = 1$ then follows from maximality, as well as the fact that $\gcd(r_i, d_i) = 1$.

We will now show that $(\alpha(\boldsymbol{u}) \bmod f_1, \ldots, \alpha(\boldsymbol{u}) \bmod f_s)$ is $\frac{s\rho}{\gamma}$-statistically close to uniform, from which the result will follow by the Chinese Remainder Theorem. Indeed, $\alpha(\boldsymbol{u}) \bmod f_i = v + u_i \cdot \frac{\ell d_i}{r_i}$, where u_i is independent of v and $\frac{\ell d_i}{r_i}$ is invertible $\bmod f_i$. Thus, it suffices to show that $u_i \bmod f_i$ is $\frac{\rho}{\gamma}$-statistically close to being uniform $\bmod f_i$, and since $u_1, \ldots, u_s$ are independent, we will get an overall bound of $\frac{s\rho}{\gamma}$ as required. Indeed, for any $\varphi \bmod f_i$:

$$\Pr_{u_i \leftarrow \mathbb{F}_\gamma}[u_i \bmod f_i = \varphi] \in \left(\frac{\lfloor \frac{\gamma}{f_i} \rfloor}{\gamma}, \frac{\lfloor \frac{\gamma}{f_i} \rfloor + 1}{\gamma}\right) \subseteq \left(\frac{1}{f_i} - \frac{1}{\gamma}, \frac{1}{f_i} + \frac{1}{\gamma}\right) \quad .$$

Thus, $u_i \bmod f_i$ is at most $\frac{f_i}{\gamma} \leq \frac{r_i}{\gamma} \leq \frac{\rho}{\gamma}$ far from uniform. This concludes the proof. □

Putting it all together. We now complete the proof of Theorem 5.6. In what follows let $\Delta = \lceil \mu p / \varepsilon \rceil$. We consider the following cases:

- If $p\boldsymbol{\pi}^*$ is Δ-close to a (p', p)-fraction, with denominators $r_1, \ldots, r_m$ and $L \coloneqq \mathsf{lcm}(r_1, \ldots, r_m) \leq \varepsilon^{-1}$, then relying on

  $$p' \geq \mu^2 \gamma p^2 \varepsilon^{-2} \geq \mu \gamma \Delta p L \ ,$$

  it follows from Claim 5.15 that the soundness error is at most

  $$\varepsilon + \frac{p}{\gamma} \leq \varepsilon + \frac{\varepsilon}{100} \leq 1.01\varepsilon \ ,$$

  where we rely on $\gamma \geq 100\mu p \varepsilon^{-1} \geq 100 p \varepsilon^{-1}$.

- If $p\boldsymbol{\pi}^*$ is Δ-far from any (p', γ)-fraction, then by Claim 5.16, the soundness error is at most

  $$\frac{\mu p}{\Delta} + \frac{1}{\gamma} \leq \varepsilon + \frac{\varepsilon}{100} \leq 1.01\varepsilon \ ,$$

  where we rely on $\Delta \geq \mu p \varepsilon$, $\gamma \geq 100\mu p \varepsilon^{-1} \geq 100\varepsilon^{-1}$.

- If $p\boldsymbol{\pi}^*$ is Δ-close to a (p', γ)-fraction, with denominators $r_1, \ldots, r_m$ and $r = \max\{r_i\} \geq \varepsilon$, then relying on

  $$p' \geq \mu^2 \gamma p^2 \varepsilon^{-2} \geq \mu \gamma \Delta p \ ,$$

  it follows from Claim 5.17 that the soundness error is at most

  $$\frac{4}{\Delta} + \frac{1}{r} + \frac{1}{\gamma} \leq \frac{\varepsilon}{200} + \varepsilon + \frac{\varepsilon}{200} \leq 1.01\varepsilon \ ,$$

  where we rely on $\Delta \geq \mu p \varepsilon^{-1}$, $\mu \geq 400$, $p \geq 2$, $\gamma \geq 100\mu p \varepsilon^{-1} \geq 200\varepsilon^{-1}$.

- Otherwise $p\boldsymbol{\pi}^*$ is Δ-close to a (p', γ)-fraction, with denominators $r_1, \ldots, r_m$, $\max\{r_i\} < \varepsilon^{-1} < L \coloneqq \mathsf{lcm}(r_1, \ldots, r_m)$. Then letting $\rho = \varepsilon^{-1}$ and relying on

  $$p' \geq \mu^2 \gamma p^2 \varepsilon^{-2} \geq \mu \gamma \Delta \rho^2 \ ,$$

  it follows from Claim 5.18 that the soundness error is at most

  $$\frac{\mu \rho}{\gamma} + \frac{1}{\rho} \leq \frac{\varepsilon}{10} + \varepsilon \leq 1.01\varepsilon \ ,$$

  where we rely on $\gamma \geq 100\mu p \varepsilon^{-1} \geq 100\mu \varepsilon^{-2}$.

Remark 5.19 (Non-promise FLPCP)**.** *The above transformation results in a bounded* promise *FLPCP. In particular for instances $\boldsymbol{x} \in \mathbb{F}_{p'}^n \setminus \{0,1\}^n$ there is no soundness guarantee. We can turn the construction to a non-promise one, where any input $\boldsymbol{x} \notin \{0,1\}^n$ is rejected with probability $1 - O(1/p)$, at the cost of increasing the proof size additively by n^2 and adding two queries. This is done using a Hadamard tensor test.*

Specifically we append the vector $\boldsymbol{x} \otimes \boldsymbol{x}$ to the proof. Then, using two additional queries we obtain $\langle \boldsymbol{x}, \boldsymbol{r} \rangle$ and $\langle \boldsymbol{x} \otimes \boldsymbol{x}, \boldsymbol{r} \otimes \boldsymbol{r} \rangle$ for a random $\boldsymbol{r} \leftarrow \mathbb{F}_p^n$, which allows testing the tensor structure. We can then use another random linear test to check that $x_i = x_i^2$ to verify Booleanity. The result of this test in the honest case is always meant to be 0 and hence can be folded into the other queries (similarly to the proof of Theorem 4.6). Note that the additional (honest) answers are all n^2p-bounded. A more detailed analysis of this Hadamard test can be found in [BIOW20].

5.2 Query Packing (Proof of Theorem 5.7)

In this section we prove Theorem 5.7. We start by describing the construction, which is a generalization of the packing construction from [BCIOP22]. In particular, we pack the queries $\boldsymbol{Q}$ by evaluating the linear function $E_{\boldsymbol{Q}}(\boldsymbol{w}) = \boldsymbol{w}^\intercal \boldsymbol{Q}$. The only adaptation is in the choice of the rectangle from which $\boldsymbol{w}$ is chosen at random, which is derived from the bound vector $\boldsymbol{b}$. At the end of the section, we explain how the construction differs from that of [BIOW20].

Fact 5.20. *There is an efficient algorithm for the following problem:*

- ***Input:*** *Vectors $\boldsymbol{w}, \boldsymbol{b} \in \mathbb{Z}_+^k$ and integer $a \in \mathbb{Z}$ such that each $w_i > 2\sum_{j<i}(b_j - 1)w_j$.*
- ***Output:*** *A vector $\boldsymbol{a} \in \mathbb{Z}^k$ such that $|a_i| \le b_i - 1$ and $a = [\langle \boldsymbol{a}, \boldsymbol{w} \rangle]_{\mathbb{Z}}$ if such $\boldsymbol{a}$ exists.*

Construction 5.21 (Query Packing for FLPCPs)**.** *Let $(\mathcal{P}, \mathcal{V})$ be a k-query $(\boldsymbol{b} - \boldsymbol{1})$-bounded* FLPCP *over $\mathbb{F}_p$ with proof length m and let $\ell \le p/(2\boldsymbol{b})^k$. Define $(\mathcal{P}', \mathcal{V}')$ over $\mathbb{F}_p$ as follows.*

- *The verifier $\mathcal{V}'$ runs the* FLPCP *verifier $\mathcal{V}$ to obtain queries $\boldsymbol{Q} \in \mathbb{F}_p^{k \times (n+m)}$, picks a sequence of k random field elements $\boldsymbol{w} = (w_1, \ldots, w_k)$ where*

  $$w_i \leftarrow [\underline{w}_i, \overline{w}_i] \coloneqq [((2\boldsymbol{b})^{i-1} - 1)\ell + 1, (2\boldsymbol{b})^{i-1}\ell] \enspace .$$

  The query vector is $[\boldsymbol{Q}^\intercal \boldsymbol{w}]_{\mathbb{Z}}$.

- *The prover's $\mathcal{P}'$ proof is the proof $\boldsymbol{\pi}$ generated the underlying $\mathcal{P}$.*

- *The verifier $\mathcal{V}'$ obtains an answer $a \in \mathbb{F}_p$, applies the subset sum algorithm of Fact 5.20 to find $\boldsymbol{a}$ such that $a = [\langle \boldsymbol{a}, \boldsymbol{w} \rangle]_{\mathbb{Z}}$ (if none exists it rejects) and accepts if the verifier $\mathcal{V}$ accepts $\boldsymbol{a}$.*

5.2.1 Analysis of Construction 5.21

We start by proving completeness and then prove soundness.

Claim 5.22 (Completeness)**.** *For $p > 2\langle \boldsymbol{b} - \boldsymbol{1}, \overline{\boldsymbol{w}} \rangle$, the verifier $\mathcal{V}'$ accepts any honest $(\boldsymbol{x} \| \boldsymbol{\pi})$. Furthermore, $(2\boldsymbol{b})^k \ell \ge 2\langle \boldsymbol{b} - \boldsymbol{1}, \overline{\boldsymbol{w}} \rangle$.*

Proof. First, our choice of $\boldsymbol{w}$ satisfies $w_i > 2\sum_{j<i}(b_j - 1)w_j$. For this, we prove by induction that $\underline{w}_i > 2\sum_{j<i}(b_j - 1)\overline{w}_j$. The base of the induction is that $\underline{w}_1 > 0$. As for the induction step,

$$\begin{aligned}
2\sum_{j<i+1}(b_j - 1)\overline{w}_j &= \left(2\sum_{j<i}(b_j - 1)\overline{w}_j\right) + 2(b_i - 1)\overline{w}_i \\
&< \underline{w}_i + 2(b_i - 1)\overline{w}_i \\
&= ((2\boldsymbol{b})^{i-1} - 1)\ell + 1 + 2(b_i - 1)(2\boldsymbol{b})^{i-1}\ell \\
&= (2\boldsymbol{b})^{i-1}\ell - \ell + 1 + (2\boldsymbol{b})^{i}\ell - 2(2\boldsymbol{b})^{i-1}\ell \\
&< ((2\boldsymbol{b})^{i} - 1)\ell + 1 \\
&= \underline{w}_{i+1}
\end{aligned}$$

By the boundedness of the underlying FLPCP, it follows that for any honest proof $(\boldsymbol{x}\|\boldsymbol{\pi})$, and $\boldsymbol{w}$ chosen by the verifier V',

$$[\langle \boldsymbol{Q}^{\intercal}\boldsymbol{w}, (\boldsymbol{x}\|\boldsymbol{\pi})\rangle]_p = [\langle \boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}), \boldsymbol{w}\rangle]_p = [\langle \boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}), \boldsymbol{w}\rangle]_{\mathbb{Z}}, \ ,$$

which follows from the fact that $|\boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi})| \leq \boldsymbol{b} - \boldsymbol{1}$, $|\boldsymbol{w}| \leq \overline{\boldsymbol{w}}$, and $p > 2\langle \boldsymbol{b} - \boldsymbol{1}, \overline{\boldsymbol{w}}\rangle$.

Accordingly, the verifier $\mathcal{V}'$ applies the subset sum algorithm and obtains $[\boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi})]_{\mathbb{Z}}$. □

Claim 5.23 (Soundness)**.** *The construction has soundness error at most $\varepsilon + \boldsymbol{b}^k\ell^{-1}$, where ε is the soundness error of the underlying* FLPCP*.*

Proof. Fix any $\boldsymbol{x}$ and prover strategy $\boldsymbol{\pi}^* \in \mathbb{F}_p^m$. We say that $\boldsymbol{Q}$ is *invalid* if it does *not* hold that $|[\boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}^*)]_p| \leq \boldsymbol{b} - \boldsymbol{1}$. We first show that conditioned on any choice of invalid queries $\boldsymbol{Q}$, the probability, over the choice of $\boldsymbol{w}$ that $\mathcal{V}'$ accepts is bounded by $\boldsymbol{b}^k/\ell$. Fix any candidate answers $\boldsymbol{a}$ such that $|\boldsymbol{a}| \leq \boldsymbol{b} - \boldsymbol{1}$. Then,

$$\Pr_{\boldsymbol{w}}[\langle \boldsymbol{Q}^{\intercal}\boldsymbol{w}, (\boldsymbol{x}\|\boldsymbol{\pi}^*)\rangle = \langle \boldsymbol{a}, \boldsymbol{w}\rangle] = \Pr_{\boldsymbol{w}}[\langle \boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}^*), \boldsymbol{w}\rangle = \langle \boldsymbol{a}, \boldsymbol{w}\rangle] = \Pr_{\boldsymbol{w}}[\langle \boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}^*) - \boldsymbol{a}, \boldsymbol{w}\rangle = 0] \leq \ell^{-1} \ .$$

Indeed, since $\boldsymbol{Q}$ is invalid $\langle \boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}^*) - \boldsymbol{a}, \boldsymbol{w}\rangle$ is a non-trivial degree-one polynomial in $\boldsymbol{w} = (w_1, \dots, w_k)$ and each w_i is picked uniformly at random from a set of size ℓ.

By a union bound, the probability that there *exists* $\boldsymbol{a}$ such that $|\boldsymbol{a}| \leq \boldsymbol{b} - \boldsymbol{1}$ and $\langle \boldsymbol{Q}^{\intercal}\boldsymbol{w}, (\boldsymbol{x}\|\boldsymbol{\pi}^*)\rangle = \langle \boldsymbol{a}, \boldsymbol{w}\rangle$ is at most $(2\boldsymbol{b})^k/\ell$. Accordingly, for invalid queries, the subset sum algorithm will fail to find a solution and $\mathcal{V}'$ will reject except with probability $(2\boldsymbol{b})^k/\ell$.

It is left to note that conditioned on valid queries $\boldsymbol{Q}$ the corresponding answers $\boldsymbol{Q}(\boldsymbol{x}\|\boldsymbol{\pi}^*)$ are correctly decoded by the subset sum algorithm, in which case the soundness of $(\mathcal{P}, \mathcal{V})$ kicks in. (In the case that $(\mathcal{P}, \mathcal{V})$ is a promise FLPCP, then this holds within the promise and the resulting DPP is a promise DPP). □

Overall, choosing $\ell = (2\boldsymbol{b})^k/\varepsilon$, concludes the proof of Theorem 5.7.

Remark 5.24 (Packing in [BIOW20])**.** *The packing transformation in [BIOW20] also deals with general bounded* FLPCP*s, but is different. First, rather than using the linear function $E_{\boldsymbol{Q}}(\boldsymbol{w}) = \boldsymbol{w}^{\intercal}\boldsymbol{Q}$, it uses the multi-linear function $E_{\boldsymbol{Q}}(\boldsymbol{w}) = (\boldsymbol{w}^0, \boldsymbol{w}^1, \dots, \boldsymbol{w}^{k-1})\boldsymbol{Q}$, where $\boldsymbol{w}$ is sampled from an appropriate rectangle. The corresponding lower bound on the field size p is $\approx \min\{b_1, \dots, b_k\} \cdot ((2\boldsymbol{b})^k/\varepsilon)^{k-1}$, compared to our $((2\boldsymbol{b})^k)^2/\varepsilon$. In particular, for $k \leq 2$, their transformation is superior by a factor of $\approx \max\{b_1, \dots, b_k\}$. In contrast, for any $k \geq 3$, our transformation is superior by a factor of $((2\boldsymbol{b})^k)^{k-3}/\varepsilon^{k-2}$.*

5.3 Strong Soundness

The notion of strong soundness (for LPCPs) introduced in [BCIOP22] essentially says that any proof is either very close to accepting or very far from accepting. This feature intuitively guarantees that the verifier's acceptance is almost uncorrelated to its specific queries, which is in turn important for establishing *reusability* of verifier queries in the setting of privately verifiable succinct arguments.

In the context of bounded FLPCPs, we follow [BIOW20] and require that strong soundness holds for verifiers that enforce the bound. In particular, whether or not the bound is satisfied is almost uncorrelated to the specific verifier queries and almost only depends on the proof itself. This guarantees that strong soundness is preserved by the packing transformation.

Definition 5.25 (Strong Soundness). *A $\boldsymbol{b}$-bounded* FLPCP *has strong soundness error ε if its verifier only accepts answers within the bound $\boldsymbol{b}$, and any input-proof pair $(\boldsymbol{x}\|\boldsymbol{\pi})$ is either accepted with probability at least $1-\varepsilon$ or at most ε.*

Claim 5.26 (Strong Soundness). *The packing transformation of bounded* FLPCP*s into* DPP*s given by Theorem 5.7 preserves strong soundness up to a factor of* 2.

An analogous claim is proven in [BIOW20] for the packing transformation there. The proof in our setting is essentially identical, we give it here for completeness.

Proof of Claim 5.26. Fix any instance-proof pair $(\boldsymbol{x}\|\boldsymbol{\pi}^*)$. We consider two cases:

- **The underlying** FLPCP **verifier $\mathcal{V}$ accepts with probability at least** $1-\varepsilon$**:** Then, with probability at least $1-\varepsilon$ it both accepts and the answers are $\boldsymbol{b}$-bounded. In this case, the DPP verifier $\mathcal{V}'$ also accepts the corresponding packed answer.

- **The underlying** FLPCP **verifier $\mathcal{V}$ accepts with probability at most** ε**:** This means that except with probability ε, either the answer exceeds the bound $\boldsymbol{b}$ or $\mathcal{V}$ rejects. However, whenever the answer exceeds the bound $\boldsymbol{b}$, the corresponding query is *invalid* (as termed in the proof of Claim 5.23) and the DPP verifier $\mathcal{V}'$ accepts with probability at most ε, so overall in this case $\mathcal{V}'$ accepts with probability at most 2ε. □

Does Theorem 5.6 imply strong soundness? The only question left is whether the bounded (promise) FLPCP from Theorem 5.6 has strong soundness. As is, it is in fact not strongly sound. For instance, if the prover takes an honest proof $\boldsymbol{\pi}$ and resets $\pi_1^* = \lceil \frac{p'+\pi_1}{2} \rceil$, then an answer to query $\boldsymbol{q}$ will fall outside/inside the bound according to the parity of q_1. Nevertheless, we show a variant of the transformation that adds $O(\log \varepsilon^{-1})$ queries is strongly sound, assuming that the (unbounded) FLPCP we start from has strong soundness ε with a strongly evasive accepting set.

Definition 5.27 (Strongly Evasive Accepting Set). *A k-query ε-sound FLPCP over $\mathbb{F}_p$ has a strongly evasive accepting set if for any (yes or no) instance and any fixing of answers $a_2, \ldots, a_k$, sampling $a_1 \leftarrow \mathbb{F}_p$ uniformly at random leads to acceptance with probability at most ε.*

The only difference between an evasive accepting set and a strongly evasive one is that the latter is evasive also for yes-instances. In fact, it is not hard to see that for FLPCP evasiveness implies strong evasiveness, since the accepting set is sampled independently of the instance and verification only depends on the answers (and not explicitly on the instance).

Claim 5.28 (Bounded Embedding with Strong Soundness)**.** *Let $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^h$ be a relation. Let $\mu, \gamma, k, p' \in \mathbb{N}, \varepsilon \in [p^{-1}, 1/2)$ be such that $p' \geq \mu^2\gamma p^2\varepsilon^{-2}$, $\gamma \geq 100\mu p\varepsilon^{-1}$, and $(p-1)/2 > \mu \geq 400$. Then any k-query ε-strongly-sound* FLPCP *over $\mathbb{F}_p$ for $\mathcal{R}$ with (honest) Boolean proofs of length m and a strongly evasive accepting set, can be transformed into a $(k+k')$-query ε'-strongly-sound, $\boldsymbol{b}$-bounded promise* FLPCP *over $\mathbb{F}_{p'}$ for $\mathcal{R}$, with the same proof length m, bound $\boldsymbol{b} = \left(\frac{\mu p\gamma}{2}, \ldots, \frac{\mu p\gamma}{2}\right)$, and $\varepsilon' = \max\{\varepsilon, 0.51^{k'}\}$.*

The prover complexity is preserved and the verifier complexity is preserved up to a $(k+k') \cdot$ $\mathrm{poly}(\log p)$ *additive term.*

Corollary 5.29 (DPP with Strong Soundness)**.** *Any constant-query* FLPCP *over $\mathbb{F}_p$ with strong soundness ε can be transformed into a* DPP *with strong soundness error ε over $\mathbb{F}_{p'}$ where $p' = p^{O(\log \varepsilon^{-1})}$.*

Proof of Claim 5.28. The construction is the same as Construction 5.13, with several adaptations:

- Whereas in the previous construction, we assumed that in the underlying FLPCP only honest proofs are Boolean, here we explicitly test Booleanity. That is, we assume that the underlying FLPCP rejects non-Boolean proofs except with probability ε. This can be done using at most two extra queries and linear proof overhead using Corollary B.2.
- We add k' random bound tests. Namely, we set $\boldsymbol{q}'_{k+i} = (\boldsymbol{0} \| p\boldsymbol{u}_i)$ for $i \in [k']$ and where $\boldsymbol{u}_i \leftarrow \mathbb{F}_\gamma^m$.
- The verifier accepts $(\boldsymbol{a}\|\boldsymbol{a}')$ if all the bound tests pass, namely, $|(\boldsymbol{a}\|\boldsymbol{a}')| \leq \boldsymbol{b}$, and the underlying verifier accepts $\boldsymbol{a}$ mod p.

Remark 5.30 (Alternative Embedding)**.** *We remark that since we allow more queries (compared to the previous construction), the above could be simplified. We can have the first k queries be simply the underlying FLPCP queries (without embedding a random query into the first), and the k' tests be random tests that are not scaled by p. However, for compatibility with the analysis of Theorem 5.6, we stick to the above description.*

Going back to the proof of Theorem 5.6, we consider three cases:

1. If $p\boldsymbol{\pi}^*$ is not Δ-close to a (γ, p')-fraction, then by Claim 5.16 and our choice of parameters,
$$\Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m}\left[\left|[\langle p\boldsymbol{u}, \boldsymbol{\pi}^*\rangle]_{p'}\right| \leq b\right] \leq \Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m}\left[\left|[\langle p\boldsymbol{u}, \boldsymbol{\pi}^*\rangle]_{p'}\right| \leq b_1\right] \leq \frac{\mu p}{\Delta} + \frac{1}{\gamma} \leq 1.01\varepsilon \leq 0.51 \ .$$
2. If $p\boldsymbol{\pi}^*$ is Δ-close to a (γ, p')-fraction with denominators $r_1, \ldots, r_m$, and $r := \max_i \{r_i\} \geq 2$, then by Claim 5.17 and our choice of parameters,
$$\Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m}\left[\left|[\langle p\boldsymbol{u}, \boldsymbol{\pi}^*\rangle]_{p'}\right| \leq b\right] \leq \Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m}\left[\left|[\langle p\boldsymbol{u}, \boldsymbol{\pi}^*\rangle]_{p'}\right| \leq b_1\right] \leq \frac{1}{r} + \frac{1}{\gamma} + \frac{4}{\Delta} \leq \frac{1}{2} + 0.01\varepsilon \leq 0.51 \ .$$
3. Otherwise, $p\boldsymbol{\pi}^*$ is Δ-close to a (γ, p')-fraction, and $\max_i \{r_i\} = 1$, meaning that $p\boldsymbol{\pi}^* = \boldsymbol{d}p' + \boldsymbol{e}$ where $\|\boldsymbol{e}\|_\infty \leq \Delta$.

In the first two cases, the probability of the bound test passing is at most $0.51^{k'}$.

We now consider several sub-cases of Case 3:

3.1. If $\boldsymbol{e} \neq 0 \bmod p$, then following the analysis of Claim 5.15 and the strong evasiveness of the underlying FLPCP, $\boldsymbol{q}'_1$ fails either the bound test or the underlying FLPCP verification, except with probability ε.

From hereon, we assume $\boldsymbol{e} = 0 \bmod p$, in which case also $\boldsymbol{d} = 0 \bmod p$ and accordingly, we can write $\boldsymbol{\pi}^* = \boldsymbol{d}'p' + \boldsymbol{e}'$ where $\|\boldsymbol{e}'\|_\infty \leq \Delta/p$. Note that in this case for $\boldsymbol{q}'_1, \ldots, \boldsymbol{q}'_k$ it holds that $|\langle \boldsymbol{q}'_i, (\boldsymbol{x}\|\boldsymbol{e}')\rangle| \leq p'/2$ and hence

$$\left[\langle \boldsymbol{q}'_i, (\boldsymbol{x}\|\boldsymbol{\pi}^*)\rangle\right]_{p',p} = \left[\langle \boldsymbol{q}'_i, (\boldsymbol{x}\|\boldsymbol{e}')\rangle\right]_{p',p} = \left[\langle \boldsymbol{q}'_i, (\boldsymbol{x}\|\boldsymbol{e}')\rangle\right]_{p}$$

3.2. If $(\boldsymbol{e}' \bmod p) \notin \{0,1\}^m$, the verifier of the underlying FLPCP rejects except with probability ε. From hereon we assume $(\boldsymbol{e}' \bmod p) \in \{0,1\}^m$ and write $\boldsymbol{e}' = \boldsymbol{t} + p\boldsymbol{w}$ where $\boldsymbol{t} \in \{0,1\}^m$.

3.3. If $\boldsymbol{w} \neq 0$, we show that

$$\Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m}\left[\left|[\langle p\boldsymbol{u}, \boldsymbol{\pi}^*\rangle]_{p'}\right| \leq b\right] = \Pr_{\boldsymbol{u} \leftarrow \mathbb{F}_\gamma^m}\left[|\langle p\boldsymbol{u}, \boldsymbol{t} + p\boldsymbol{w}\rangle| \leq b\right] \leq 0.51 \ .$$

Assume without loss of generality that $w_1 \neq 0$ and fix any $v = \sum_{i=2}^m pu_i(t_i + pw_i)$. Note that for any two distinct $u, u' \in \mathbb{F}_\gamma$,

$$|p(u - u')(t_1 + pw_1)| \geq p^2|w_1| - p|t_1| \geq p(p-1) \ .$$

Hence, there are at most $\lceil 2b/p(p-1)\rceil = \lceil \mu\gamma/(p-1)\rceil$ values u_1 such that $|v + pu_1(t_1 + pw_1)| \leq b$. This occurs with probability at most

$$\frac{1 + (\mu\gamma/(p-1))}{\gamma} \leq \frac{1}{\gamma} + \frac{\mu}{p-1} \leq 0.51 \ ,$$

under the assumption that $p > 2\mu + 1$.

It follows that in this case, the bounds tests fail except with probability $0.51^{k'}$.

3.4. We are left with the case that $\boldsymbol{e}' = \boldsymbol{t} \in \{0,1\}^m$. In this case, for any query $\boldsymbol{q}'$, $|\langle \boldsymbol{q}', (\boldsymbol{x}\|\boldsymbol{e}')\rangle| \leq \mu p\gamma$ and the bound test always passes. Furthermore, since $[\langle \boldsymbol{q}', (\boldsymbol{x}\|\boldsymbol{\pi}^*)\rangle]_{p',p} = [\langle \boldsymbol{q}', (\boldsymbol{x}\|\boldsymbol{e}')\rangle]_p$, the strong soundness of the underlying FLPCP implies that we either accept with probability at most ε or at least $1 - \varepsilon$. □

6 From DPP to Hardness of Approximation

Our hardness results stem from the simple observation that 1-query linear PCPs are essentially equivalent to **MAXLIN** (the problem of finding an assignment that maximizes the number of linear equations satisfied). More precisely, we have the following proposition.

Proposition 6.1 (1-Query Linear PCPs and **MAXLIN**). *Suppose a language $\mathcal{L}$ has a 1-query linear PCP over a field $\mathbb{F}$ with proof length $m(\cdot)$, randomness complexity $r(\cdot)$, completeness c, soundness error s, and accepting set of size t. Then there exists a $O(t \cdot 2^{r(n)})$-time reduction from $\mathcal{L}$ to gap$_{c/t,s/t}$-***MAXLIN**$(\mathbb{F})$ *that maps instance x of $\mathcal{L}$ of size n to instances $(\boldsymbol{A}, \boldsymbol{b})$ of gap$_{c/t,s/t}$-***MAXLIN**$(\mathbb{F})$ *with $m(n)$ variables and $t \cdot 2^{r(n)}$ equations (i.e, $A \in \mathbb{F}^{m(n)\times t\cdot 2^{r(n)}}$ and $b \in \mathbb{F}^{t\cdot 2^{r(n)}}$).*

*Conversely, if $\mathcal{L}$ is reducible in polynomial time to gap$_{c,s}$-***MAXLIN**$(\mathbb{F})$ *where an instance x of size n is mapped to an instance $(\boldsymbol{A}, \boldsymbol{b})$ with $m(n)$ variables and $2^{r(n)}$ equations, then $\mathcal{L}$ has a 1-query linear PCP over the field $\mathbb{F}$ with proof length $m(\cdot)$, randomness complexity $r(\cdot)$, completeness c, soundness error s and accepting set of size 1.*

Proof. Suppose a language $\mathcal{L}$ has a 1-query linear PCP over field $\mathbb{F}$ with proof length $m(\cdot)$, randomness complexity $r(\cdot)$, completeness c, soundness error s, and accepting set of size t. The reduction from $\mathcal{L}$ to gap$_{c/t,s/t}$-**MAXLIN**$(\mathbb{F})$ runs as follows: On input $\boldsymbol{x}$ (an instance of $\mathcal{L}$) of length n, run the 1-query linear PCP verifier $\mathcal{V}$ for each random string $R \in \{0,1\}^{r(n)}$ to obtain the linear query $q(\boldsymbol{x}, R)$ and answer set $A(\boldsymbol{x}, R)$. The output gap$_{c/t,s/t}$-**MAXLIN**$(\mathbb{F})$ instance Φ consists of all equations of the form $\langle q(\boldsymbol{x}, R), z\rangle = b$ for each $b \in A(\boldsymbol{x}, R)$ and $R \in \{0,1\}^{r(n)}$. It is not hard to see that if $\boldsymbol{x} \in \mathcal{L}$ then Φ is at least c/t-satisfiable and if $\boldsymbol{x} \notin \mathcal{L}$ then Φ is at most s/t-satsifiable. □

Furthermore, If we start with a DPP for $\mathcal{L}$ instead of 1-query linear PCP, the above reduction produces **MAXLIN** instances $\boldsymbol{A} \cdot \boldsymbol{z} = \boldsymbol{b}$ where only $\boldsymbol{b}$ depends on the input instance $\boldsymbol{x}$ and the matrix A only depends on the input length $n = |\boldsymbol{x}|$. We summarize this in the following proposition.

Proposition 6.2 (DPP and **MAXLIN**)**.** *Suppose a language $\mathcal{L}$ has a* DPP *over a field $\mathbb{F}$ with proof length $m(\cdot)$, randomness complexity $r(\cdot)$, completeness c, soundness error s, and accepting set of size t. Then there exists a $O(t \cdot 2^{r(n)})$-time reduction from $\mathcal{L}$ to gap$_{c/t,s/t}$-***MAXLIN**$(\mathbb{F})$ *that maps instance x of $\mathcal{L}$ of size n to instances $(\boldsymbol{A}, \boldsymbol{b})$ of gap$_{c/t,s/t}$-***MAXLIN**$(\mathbb{F})$ *with $m(n)$ variables and $t \cdot 2^{r(n)}$ equations (i.e, $A \in \mathbb{F}^{m(n)\times t\cdot 2^{r(n)}}$ and $b \in \mathbb{F}^{t\cdot 2^{r(n)}}$) and $\boldsymbol{A}$ is only a function of the input length $|\boldsymbol{x}|$ and only $\boldsymbol{b}$ is a function of the input $\boldsymbol{x}$*

Such reductions are said to be reductions with universal factor graphs [FJ12, ABH21].

For a constant $C > 1$, let $\mathbf{3SAT}_C$ refer to the set of satisfiable 3CNF formulas over n variables with at most Cn clauses. The constant-soundness variant of our small-field DPP construction (Theorem 4.3) implies that there is $\varepsilon < 1$ such that over every finite field $\mathbb{F}$ of order $q > 2$, $\mathbf{3SAT}_C$ has a DPP with proof length $O_q(n)$, randomness complexity $\log n + O_q(1)$, perfect completeness, soundness error ε and accepting set A of size 2. For $\mathbb{F}_2$ we have a constant gap with A of size 1. Plugging this into the above proposition yields the following theorem.

Theorem 6.3 (Hardness of Approximation for **MAXLIN**)**.** *There is a constant $1/3 < \delta < 1/2$ such that for every finite field $\mathbb{F}$ other than $\mathbb{F}_2$ the following holds. There is a polynomial time reduction from $\mathbf{3SAT}_C$ to gap$_{1/2,\delta}$-***MAXLIN**$(\mathbb{F})$ *that transforms a 3CNF formula Φ over n variables and at most Cn clauses into a* **MAXLIN** *instance Φ with a universal factor graph with over $N = O(n)$ variables and $M = O(n)$ equations.*

*For $\mathbb{F}_2$, we obtain a polynomial reduction from $\mathbf{3SAT}_C$ to gap$_{5/8,5/8-\varepsilon}$-***MAXLIN**$(\mathbb{F}_2)$ *for some constant $\varepsilon \in (0,1)$ that transforms a 3CNF formula Φ over n variables and at most Cn clauses into a* **MAXLIN** *instance Φ with over $N = O(n)$ variables and $M = O(n)$ equations.*

We remark that this theorem immediately implies that there is some constant $c > 1$ such that it is NP-hard to approximate **MAXLIN**$(\mathbb{F})$ to within a factor better than c, and similarly for the nearest codeword problem **NCP**$(\mathbb{F})$.

We note that these are *not* the best inapproximability results for either of these problems. For instance, for **MAXLIN**, Håstad [Hås01] and Austrin, Brown-Cohen, and Håstad [ABH21] obtain the optimal inapproximability factor of $|\mathbb{F}|(1+\varepsilon)$ for every $\varepsilon > 0$ even for **MAXLIN** instances which have at most 3 variables per equation.

This theorem is nevertheless interesting as it produces **MAXLIN** instances with $O(n)$ variables while all previously known reductions involve at least a multiplicative logarithmic overhead. This in particular yields the following corollary.

Corollary 6.4 (Exponential-Time Hardness of Approximation for **MAXLIN** under ETH)**.** *There is a universal constant $c > 1$ such that for every field $\mathbb{F}$, the following holds. Assuming the exponential time hypothesis (ETH) (Hypothesis 3.7), there does not exist any $2^{o(n)}$-time algorithm that can approximate* **MAXLIN**$(\mathbb{F})$ *to a factor better than c.*

To the best of our knowledge, this is the first (fully) exponential time hardness of an approximation problem obtained assuming the ETH. All previous exponential hardness results required an assumption at least as strong as the gap-ETH. This is because all prior reductions use the PCP Theorem which incurs at least a multiplicative logarithmic overhead (c.f., discussion in [Din16, Section 1, Page 4]). Previous reductions used the stronger gap-ETH instead to obtain exponential time hardness instead. We get around this logarithmic multiplicative overheard by constructing a simpler PCP (more precisely, DPPs or 1-query linear PCPs) which avoids the use of the PCP Theorem.

7 From DPP to Succinct Arguments

In this section, we present several compilers from DPPs to different kinds of efficient proof systems. Most of the results of this section in fact apply even to the more general notion of LPCP (which follows also from promise DPP), though in the rest of this section we stick to the DPP terminology.

We discuss three kinds of compilers. First, in Section 7.1, we use the previous compiler from [BCIOP22] to convert a DPP into a designated-verifier succinct non-interactive arguments (SNARG) in the preprocessing model, where the proof consists of just a *single* ciphertext of a suitable encryption scheme. Then, in Section 7.2, we present a new compiler that has even better succinctness features in the generic group model, at the cost of requiring interaction and *non-reusable* (but input-independent) preprocessing. Finally in Section 7.3, we leverage the "fully linear" feature of DPPs to obtain succinct commit-and-prove arguments.

7.1 From DPP to Single-Ciphertext SNARGs

In this section we use the compiler of Bitansky, Chiesa, Ishai, Ostrovsky and Paneth [BCIOP22] to combine a DPP for a language $\mathcal{L}$ with a "linear-only" encryption scheme, obtaining a designated-verifier SNARG for $\mathcal{L}$ in the preprocessing model, where the proof consists of a *single* ciphertext (of the linear-only encryption scheme). We recall the formal definition of SNARGs and linear-only encryption schemes in Appendix D.

We state the main implication below from [BCIOP22], and then compare our DPP-based instantiation with the previous approaches from [BCIOP22] based on traditional PCPs and the approach of Barta, Ishai, Ostrovsky, and Wu [BIOW20] based on the Hadamard linear PCP. Throughout this section, we consider DPPs for NP where there is an explicit polynomial-time (uniform) prover $\mathcal{P}$ that takes as input a statement x and a witness w, and outputs a DPP proof π. We write $\langle \mathcal{P}, \mathcal{V} \rangle$ to denote such a DPP, where $\mathcal{P}$ denotes the prover and $\mathcal{V}$ denotes the verifier. Similarly, we require that there is an efficient polynomial-time algorithm for deciding membership in the accepting set A of the DPP.

Theorem 7.1 (SNARGs from Linear-Only Encryption [BCIOP22]). *Let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda \in \mathbb{N}}$ be an NP relation and let $\langle \mathcal{P}, \mathcal{V} \rangle = \{\langle \mathcal{P}_\lambda, \mathcal{V}_\lambda \rangle\}_{\lambda \in \mathbb{N}}$ be a* DPP *for $\mathcal{R}$ over $\mathbb{F} = \{\mathbb{F}_\lambda\}_{\lambda \in \mathbb{N}}$ with completeness $c = c(\lambda)$ and soundness error $s = s(\lambda)$ against* affine *strategies (see Remark 7.2). Suppose there exists a linear-only encryption scheme over $\mathbb{F}$. Then, there exists an adaptively-sound single-theorem designated-verifier preprocessing SNARG with completeness $c(\lambda)$ and soundness error $s(\lambda) + \mathsf{negl}(\lambda)$. The SNARG proof consists of a* single *ciphertext for the linear-only encryption scheme.*

Moreover, if the underlying DPP *satisfies strong soundness (Definition 5.25) against affine strategies and the encryption scheme is linear-only with interactive extraction (see [BCIOP22, Definition C.6]), then there exists a multi-theorem preprocessing SNARG with completeness $c(\lambda)$ and soundness error $Q(\lambda) \cdot s(\lambda) + \mathsf{negl}(\lambda)$, where $Q = Q(\lambda)$ is the number of verification queries the adversary makes in the multi-theorem soundness experiment.*

Remark 7.2 (Soundness for Affine Strategies). *The [BCIOP22] compiler (Theorem 7.1) requires that the underlying linear interactive proof (i.e., the* DPP *in our setting) is sound against* affine *strategies. Let $\mathcal{V}$ be a* DPP *over $\mathbb{F}$ with input length n and proof length m. Soundness against affine strategies then says that for every $\boldsymbol{x} \notin \mathcal{L}$ and every affine strategy $\boldsymbol{\pi}^* \in \mathbb{F}^m$, $t^* \in \mathbb{F}$,*

$$\Pr_{(\boldsymbol{q}, A) \leftarrow \mathcal{V}} [\langle \boldsymbol{x} \| \boldsymbol{\pi}^*, \boldsymbol{q} \rangle + t^* \in A] \leq s,$$

where $s \in [0, 1]$ is the soundness error. We note that we can transform any DPP *with soundness error s into a* DPP *with soundness error $s + \delta$ against affine strategies, where $\delta = \Pr_{(\boldsymbol{q}, A) \leftarrow \mathcal{V}}[\boldsymbol{q} = \boldsymbol{0}]$. First, we observe that*

$$\Pr_{\substack{(\boldsymbol{q}, A) \leftarrow \mathcal{V} \\ r \leftarrow \mathbb{F} \setminus \{0\}}} [r \in A \mid \boldsymbol{q} \neq \boldsymbol{0}] \leq \frac{s}{1 - \delta}.$$

The idea now is to take $(\boldsymbol{q}, A) \leftarrow \mathcal{V}$ and define the new query $\boldsymbol{q}' \leftarrow \alpha \boldsymbol{q}$ where $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and answer set $A' \leftarrow \alpha A \coloneqq \{\alpha r : r \in A\}$. To see that this provides soundness against affine strategies, take any $\boldsymbol{x} \notin \mathcal{L}$ and consider any affine strategy $(\boldsymbol{\pi}^*, t^*)$. We consider two possibilities:

- *If $t^* = 0$, then soundness of the* DPP *says that $\Pr_{(\boldsymbol{q}, A) \leftarrow \mathcal{V}}[\langle \boldsymbol{x} \| \boldsymbol{\pi}^*, \boldsymbol{q} \rangle \in A] \leq s$. Correspondingly, for any $\alpha \in \mathbb{F} \setminus \{0\}$, $\Pr_{(\boldsymbol{q}, A) \leftarrow \mathcal{V}}[\langle \boldsymbol{x} \| \boldsymbol{\pi}^*, \alpha \boldsymbol{q} \rangle \in A'] \leq s$.*
- *If $t^* \neq 0$, then*

  $$\begin{aligned} \Pr[\langle \boldsymbol{x} \| \boldsymbol{\pi}^*, \alpha \boldsymbol{q} \rangle + t^* \in A'] &\leq \Pr[\boldsymbol{q} = \boldsymbol{0}] + \Pr[\langle \boldsymbol{x} \| \boldsymbol{\pi}^*, \alpha \boldsymbol{q} \rangle + t^* \in A' \mid \boldsymbol{q} \neq \boldsymbol{0}] \cdot \Pr[\boldsymbol{q} \neq \boldsymbol{0}] \\ &= \delta + (1 - \delta) \Pr[\langle \boldsymbol{x} \| \boldsymbol{\pi}^*, \boldsymbol{q} \rangle + \alpha^{-1} t^* \in A \mid \boldsymbol{q} \neq \boldsymbol{0}] \\ &\leq \delta + s. \end{aligned}$$

  where all probabilities are taken over $(\boldsymbol{q}, A) \leftarrow \mathcal{V}$ and $\alpha \leftarrow \mathbb{F} \setminus \{0\}$.

If the DPP *satisfies strong soundness (Definition 5.25), then the same construction and analysis can be used to obtain a* DPP *with strong soundness against affine strategies (and strong soundness error $s + \delta$).*

Remark 7.3 (Comparison with Previous SNARGs). *The designated-verifier preprocessing SNARG from Theorem 7.1 has the appealing property that the proof consists of just a single ciphertext of the underlying linear-only encryption scheme. We briefly compare against previous approaches for obtaining similar SNARGs with single-ciphertext proofs:*

- *The [BCIOP22] approach using classical PCPs. The construction of Bitansky, Chiesa, Ishai, Ostrovsky and Paneth [BCIOP22] applies a packing transformation to a classical PCP to obtain a 1-query (input-oblivious) linear PCP. This construction has high concrete cost (due to the reliance on classical PCPs) and more significantly, does not satisfy* strong *soundness (Definition 5.25). The lack of strong soundness means that the resulting SNARG obtained by applying the [BCIOP22] compiler does not satisfy* reusable *soundness (i.e., soundness no longer holds if the prover has oracle access to the verifier).*

- *The [BIOW20] approach based on packing the Hadamard linear PCP. Barta, Ishai, Ostrovsky, and Wu [BIOW20] apply a packing transformation to the Hadamard linear PCP to obtain a* DPP *that satisfies strong soundness. However, since the [BIOW20] construction relies on the Hadamard linear PCP as its starting point, the size of the query in the resulting* DPP *is* quadratic *in the size of the circuit. Correspondingly, this leads to a succinct argument where the common reference string is* quadratic *in the circuit size.*

In this work, we show how to construct a DPP with linear-size proofs by combining the 2-query FLPCP *for Boolean circuit satisfiability (Corollary B.6) with our embedding and packing transformations (Corollary 5.12). In conjunction with an encryption scheme that satisfies an interactive linear-only assumption (see [BCIOP22, Appendix C]), this yields a SNARG with* reusable *soundness and a linear-size CRS.*

7.2 From DPP to Laconic Arguments with Preprocessing

A disadvantage of the generic compiler from Theorem 7.1 is that it requires a linear-only encryption scheme. Concretely, this leads to constructions with longer proofs (when instantiated with candidate linear-only encryption schemes based on decisional composite residuosity [Pai99] or lattice assumptions [Reg09]), or the construction imposes requirements on the size of the DPP response (when instantiated with group-based assumptions). While the group-based instantiations yields the most concretely-succinct constructions (c.f., [BIOW20]), this instantiation require that the DPP response be sufficiently small so that computing discrete logarithms is feasible (either via the Pollard kangaroo algorithm [Pol00] or through precomputing a lookup table). This small-response property does not hold for the DPP with linear-size proofs from Corollary 5.11.

Here, we show an alternative approach to use a DPP to directly construct an interactive laconic argument in the preprocessing model where the prover's message consists of a single group element and a single field element. Similar to [BIOW20], we obtain this construction by embedding a DPP "in the exponent" of a *pairing-free* group. Security of our construction follows in the generic group model.

Definition 7.4 (Laconic Argument). *Let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of NP relations indexed by a security parameter $\lambda \in \mathbb{N}$, and let $\mathcal{L} = \{\mathcal{L}_\lambda\}_{\lambda \in \mathbb{N}}$ be the associated family of NP languages. A laconic argument for $\mathcal{R}$ is an interactive protocol $\langle \mathcal{P}, \mathcal{V} \rangle$ between an efficient prover $\mathcal{P}$ and an efficient verifier $\mathcal{V}$. For a statement x and a witness w, we write $\langle \mathcal{P}(1^\lambda, x, w), \mathcal{V}(1^\lambda, x) \rangle$ to denote the verifier's output in an execution of the protocol where the prover's input is $(1^\lambda, x, w)$ and the verifier's input is $(1^\lambda, x)$. The laconic argument should satisfy the following properties:*

- *(Completeness:) for every $\lambda \in \mathbb{N}$ and every $(x, w) \in \mathcal{R}_\lambda$,*

$$\Pr[\langle \mathcal{P}(1^\lambda, x, w), \mathcal{V}(1^\lambda, x) \rangle = 1] \geq c(\lambda)$$

where $c \in [0,1]$ is called the completeness parameter. *By default, we assume perfect correctness, that is, $c = 1$.*

- *(Soundness:) for every $\lambda \in \mathbb{N}$, every $x \notin \mathcal{L}_\lambda$, and every efficient (and possibly non-uniform) prover $\mathcal{P}^*$*
$$\Pr[\langle \mathcal{P}^*(1^\lambda, x), \mathcal{V}(1^\lambda, x)\rangle = 1] \le \varepsilon(\lambda),$$
where ε is the soundness error.

- *(Laconic prover:) There exists a universal polynomial* poly$(\cdot)$ *such that for all security parameters $\lambda \in \mathbb{N}$ and every $(x, w) \in \mathcal{R}_\lambda$, the total communication from the prover to the verifier in an execution of $\langle \mathcal{P}(1^\lambda, x, w), \mathcal{V}(1^\lambda, x)\rangle$ is* $\mathrm{poly}(\lambda + |x| + \log |w|)$.

Definition 7.5 (Laconic Argument with Preprocessing)**.** *A laconic argument $\langle \mathcal{P}, \mathcal{V}\rangle$ supports (instance-independent) preprocessing if the protocol can be decomposed into two phases:*

- *(Offline preprocessing:) In the offline preprocessing step, the verifier computes a long* instance-independent *message* crs*, which we refer to as a common reference string (CRS). The length of the CRS should satisfy* $|\mathsf{crs}| = \mathrm{poly}(\lambda + |x| + |w|)$.

- *(Online verification:) After the offline preprocessing step, the total size of all subsequent message (i.e., the "online" phase of the protocol) must be short (i.e.,* $\mathrm{poly}(\lambda + |x| + \log |w|)$*).*

If the initial message crs *can be* reused *across an arbitrary polynomial number of protocol invocations (without breaking soundness), we say the laconic argument supports* reusable *preprocessing. Otherwise, we say it supports* non-reusable *preprocessing.*

Construction 7.6 (Laconic Argument from DPP)**.** *Let* $\mathsf{GroupGen}$ *be a prime-order group generator, and let $p = p(\lambda)$ be the group order output by* $\mathsf{GroupGen}$*. Let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda \in \mathbb{N}}$ be an NP relation and let $\mathcal{L} = \{\mathcal{L}_\lambda\}_{\lambda \in \mathbb{N}}$ be the associated NP language. Let $(\mathcal{P}, \mathcal{V}) = \{(\mathcal{P}_\lambda, \mathcal{V}_\lambda)\}_{\lambda \in \mathbb{N}}$ be a* DPP *for $\mathcal{L}$ over $\mathbb{F}_p = \{\mathbb{F}_{p(\lambda)}\}_{\lambda \in \mathbb{N}}$ with input length $n = n(\lambda)$ and proof length $m = m(\lambda)$. We construct a four-message preprocessing laconic argument for $\mathcal{L}$ as follows:*

- **Verifier preprocessing:** *On input the security parameter λ, the verifier samples $(\mathbb{G}, p, g) \leftarrow$ $\mathsf{GroupGen}(1^\lambda)$ and $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda(\rho)$, where ρ is the verifier randomness for the* DPP*. The verifier parses $\boldsymbol{q} = (q_1, \ldots, q_{n+m})$ and samples $\alpha \leftarrow \mathbb{F}_p$. For $i \in [m]$, let $h_i \leftarrow g^{\alpha q_{n+i}}$. It outputs the message* $\mathsf{crs} = ((\mathbb{G}, p, g), h_1, \ldots, h_m)$.

- **Prover commitment:** *On input the initial message* $\mathsf{crs} = ((\mathbb{G}, p, g), h_1, \ldots, h_m)$*, the statement $\boldsymbol{x}$, and a witness $\boldsymbol{w}$, the prover constructs a DPP proof $\boldsymbol{\pi} \leftarrow \mathcal{P}_\lambda(\boldsymbol{x}, \boldsymbol{w})$. It outputs the message $h = \prod_{i \in [m]} h_i^{\pi_i}$.*

- **Verifier challenge:** *On input the prover message $h \in \mathbb{G}$, the verifier replies with the DPP randomness ρ.*

- **Prover response:** *On input ρ, the prover computes $\boldsymbol{q} = (q_1, \ldots, q_{n+m}) \leftarrow \mathcal{V}_\lambda(\rho)$ and replies with $t = \sum_{i \in [m]} \pi_i q_{n+i} \in \mathbb{F}_p$.*

- **Verification:** *On input the statement $\boldsymbol{x}$ and the prover response $t \in \mathbb{F}_p$, the verifier first computes $z = t + \sum_{i \in [n]} x_i q_i$ and accepts if $h = g^{\alpha t}$ and $z \in A$.*

Theorem 7.7 (Laconic Argument from DPP)**.** *Suppose $(\mathcal{P}, \mathcal{V})$ is a* DPP *for $\mathcal{L}$ over $\mathbb{F}_p$ with input length $n = n(\lambda)$, proof length $m = m(\lambda)$, completeness $c = c(\lambda)$, and soundness error $s = s(\lambda)$. Suppose moreover that $(\mathcal{P}, \mathcal{V})$ is δ-evasive (Definition 5.5); that is, for all fixed non-zero vectors $\mathbf{0}^m \neq \boldsymbol{y} \in \mathbb{F}_p^m$,*

$$\Pr_{(\boldsymbol{q},A) \leftarrow \mathcal{V}_\lambda} [\langle \boldsymbol{q}, \mathbf{0}^n \| \boldsymbol{y} \rangle = 0] \leq \delta.$$

Then, modeling $\mathsf{GroupGen}$ *as a generic group (Definition E.1), for all (malicious) provers $\mathcal{P}^*$ making at most $Q = Q(\lambda)$ queries to the generic group oracle, Construction 7.6 is a laconic argument in the non-reusable preprocessing model with parameter error c and soundness error s' where*

$$s' \leq s + \delta Q + 2Q/p.$$

Proof. We separately analyze the completeness and soundness of Construction 7.6.

Completeness. Completeness essentially follows by completeness of the underlying DPP. Consider an execution of the protocol on input $(\boldsymbol{x}, \boldsymbol{w}) \in \mathcal{R}$. The verifier's initial message $\mathsf{crs} = ((\mathbb{G}, p, g), h_1, \ldots, h_m)$, where $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda(\rho)$ and $h_i = g^{\alpha q_{n+i}}$. The prover's message h is then $h = \prod_{i \in [m]} h_i^{\pi_i} = g^{\alpha \sum_{i \in [m]} \pi_i q_{n+i}}$ where $\boldsymbol{\pi} \leftarrow \mathcal{P}_\lambda(\boldsymbol{x}, \boldsymbol{w})$. The verifier replies with the randomness ρ and the prover responds with $t = \sum_{i \in [m]} \pi_i q_{n+i}$. Consider now the final verification procedure. The verifier now computes

$$z = t + \sum_{i \in [n]} x_i q_i = \sum_{i \in [m]} \pi_i q_{n+i} + \sum_{i \in [n]} x_i q_i = \langle \boldsymbol{x} \| \boldsymbol{\pi}, \boldsymbol{q} \rangle .$$

By completeness of the DPP, $z = \langle \boldsymbol{x} \| \boldsymbol{\pi}, \boldsymbol{q} \rangle \in A$ with probability at least c. Moreover, by construction, $h = g^{\alpha t}$, and so the verifier accepts with probability at least c.

Soundness. Take any instance $\boldsymbol{x} \notin \mathcal{L}_\lambda$. Let $\mathcal{P}^*$ be a malicious prover. We proceed via a hybrid argument:

- Hyb_0: This is the real soundness experiment where we replace the group $(\mathbb{G}, p, g)$ with oracle access to a generic group $\mathcal{G}$. Specifically, in this experiment, the challenger proceeds as follows:

  1. Run $(\mathsf{pp}, \mathsf{sk}, p) \leftarrow \mathsf{GGM.Setup}(1^\lambda)$. It samples $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda(\rho)$, $\alpha \leftarrow \mathbb{F}_p$, and computes $g \leftarrow \mathsf{GGM.Encode}(\mathsf{sk}, 1)$ and $h_i \leftarrow \mathsf{GGM.Encode}(\mathsf{sk}, \alpha q_{n+i})$ for each $i \in [m]$. The challenger sends $\mathsf{crs} = (\mathsf{pp}, p, g, h_1, \ldots, h_m)$ to $\mathcal{P}^*$.
  2. Algorithm $\mathcal{P}^*$ replies with an encoding $h \in \{0,1\}^\lambda$. The challenger then sends ρ to $\mathcal{P}^*$ and $\mathcal{P}^*$ replies with $t \in \mathbb{F}_p$.
  3. The challenger computes $z = t + \sum_{i \in [n]} x_i q_i$.

  The output of the experiment is 1 if h and $\mathsf{GGM.Encode}(\mathsf{sk}, \alpha t)$ encode the same value (checked using $\mathsf{GGM.Add}$ and $\mathsf{GGM.Test}$) and moreover, $z \in A$.

- Hyb_1: Same as Hyb_0 except the challenger in this experiment will explicitly maintain the mapping $\mathsf{T} \colon \{0,1\}^\lambda \to \mathbb{F}_p$ of encodings to scalars (i.e., the challenger will implement the behavior of $\mathcal{G}$). After the malicious prover outputs the encoding $h \in \{0,1\}^\lambda$, the challenger first checks if there exists an entry $(h \mapsto c)$ in T for some $c \in \mathbb{F}_p$. If not, the challenger halts with output 0. Otherwise, if there is a mapping $(h \mapsto c)$ in T, then the output of the experiment is 1 if $\alpha^{-1} c + \sum_{i \in [n]} x_i q_i \in A$ and 0 otherwise.

- Hyb_2: In this experiment, we change how the generic group queries are implemented. The challenger starts by sampling $(\mathsf{pp}, \mathsf{sk}, p) \leftarrow \mathsf{GGM.Setup}(1^\lambda)$ as in Hyb_1. For each $i \in [m]$, the challenger samples $h_i \leftarrow \{0,1\}^\lambda$ and adds the mapping $(h_i \mapsto \hat{q}_{n+i})$ to the table T. It also samples $g \leftarrow \{0,1\}^\lambda$ and adds $(g \mapsto 1)$ to T. Here, $\hat{q}_{n+i}$ is a *formal variable.* The challenger gives $\mathsf{crs} = (\mathsf{pp}, p, g, h_1, \ldots, h_m)$ to $\mathcal{P}^*$. The challenger then simulates the generic group oracles as follows:

  – $\mathsf{GGM.Setup}$: The challenger always replies with $\perp$.

  – $\mathsf{GGM.Encode}$: The challenger always replies with $\perp$.

  – $\mathsf{GGM.Add}$: On input a key k and handles $\xi_1, \xi_2 \in \{0,1\}^\lambda$, the challenger checks that $k = \mathsf{pp}$ and that the handles ξ_1, ξ_2 are present in T and mapped to formal polynomials f_1, f_2 over $\mathbb{F}_p$. If not, the challenger replies with $\perp$. If the checks pass, the challenger samples a fresh handle $\xi \leftarrow \{0,1\}^\lambda$ and adds the entry $\xi \mapsto (f_1 + f_2)$ to T and replies with ξ.

  – $\mathsf{GGM.Test}$: On input a key k and a handle ξ, the oracle checks that $k = \mathsf{pp}$, that ξ is present in T and mapped to a formal polynomial f over $\mathbb{F}_p$. If not, the challenger replies with $\perp$. If the checks pass, the challenger outputs "zero" if $f \equiv 0$ is the identically-zero polynomial over $\mathbb{F}_p$ and "non-zero" otherwise.

  In this experiment, every encoding in T maps onto a formal polynomial f in the variables $\hat{q}_{n+1}, \ldots, \hat{q}_{n+m}$. After the prover outputs the encoding $h \in \{0,1\}^\lambda$, the prover proceeds as in Hyb_1. Specifically, the challenger checks that there exists a mapping $(h \mapsto f)$ in T, and moreover, that $\alpha^{-1} \cdot f(\alpha q_{n+1}, \ldots, \alpha q_{n+m}) + \sum_{i \in [n]} x_i q_i \in A$ where $\alpha \leftarrow \mathbb{F}_p$ and $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$. If both checks pass, the output of the experiment is 1, and otherwise, it is 0.

For a malicious prover $\mathcal{P}^*$, we write $\mathsf{Hyb}_i(\mathcal{P}^*)$ to denote the output distribution of experiment Hyb_i with $\mathcal{P}^*$. We now argue that the output distributions of each pair of adjacent distributions are statistically indistinguishable.

Lemma 7.8. *For all adversaries $\mathcal{P}^*$ that make at most Q queries to $\mathcal{G}$,*

$$|\Pr[\mathsf{Hyb}_0(\mathcal{P}^*) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{P}^*) = 1]| \leq Q/p.$$

Proof. By construction, experiments Hyb_0 and Hyb_1 are identical unless the following events occur:

- Algorithm $\mathcal{P}^*$ outputs an encoding $h \in \{0,1\}^\lambda$ that does *not* exist in the table T.
- After outputting h, algorithm $\mathcal{P}^*$ makes a sequence of generic group oracle queries such that at the end of the experiment, there exists a mapping $(h \mapsto c)$ in the table T for some value c where $\alpha^{-1} c + \sum_{i \in [n]} x_i q_i \in A$.

After $\mathcal{P}^*$ chooses h, the generic group oracle adds encodings to T only if $\mathcal{P}^*$ makes successful queries to $\mathsf{GGM.Encode}$ and $\mathsf{GGM.Add}$. If such a query succeeds, then the challenger samples an encoding $\xi \leftarrow \{0,1\}^\lambda$ and adds ξ to T. If $\mathcal{P}^*$ makes at most Q generic group queries, the probability that the challenger samples $\xi = h$ and adds h to T is at most $Q/2^\lambda < Q/p$, and the claim follows. □

Lemma 7.9. *For all adversaries $\mathcal{P}^*$ that make at most Q queries to $\mathcal{G}$,*

$$|\Pr[\mathsf{Hyb}_1(\mathcal{P}^*) = 1] - \Pr[\mathsf{Hyb}_2(\mathcal{P}^*) = 1]| \leq \delta Q + Q/p.$$

Proof. For each $i \in [Q]$, we define a sequence of intermediate hybrids

- $\mathsf{Hyb}_{1,i}$: Same as Hyb_1 except the challenger answers the first i queries to $\mathcal{G}$ according to the specification of Hyb_2.

By construction, $\mathsf{Hyb}_{1,0}$ is Hyb_1 while $\mathsf{Hyb}_{1,Q}$ is Hyb_2. We now argue that for all $i \in [Q]$, the statistical distance between $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ is $\delta + 1/2^\lambda + 1/p$. The only difference between $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ is in how the challenger answers the i^{th} query:

- $\mathsf{GGM.Setup}$: In both $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$, the challenger responds to a setup query with $\perp$.

- $\mathsf{GGM.Encode}$: In both experiments, the output of the $\mathsf{GGM.Encode}$ query is $\perp$ unless the adversary queries the oracle on $k = \mathsf{sk}$. Since the view of $\mathcal{A}$ in the first $i-1$ queries in $\mathsf{Hyb}_{1,i-1}$ is independent of sk (by construction) and sk is uniform over $\{0,1\}^\lambda$, the challenger in $\mathsf{Hyb}_{1,i-1}$ responds with $\perp$ with probability $1 - 1/2^\lambda$. In $\mathsf{Hyb}_{1,i}$, the challenger always responds with $\perp$. In this case, the challenger's response on the i^{th} query in $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ differ with probability at most $1/2^\lambda < 1/p$.

- $\mathsf{GGM.Add}$: The behavior of the addition oracle is identical in $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$.

- $\mathsf{GGM.Test}$: Suppose an adversary makes a query to $\mathsf{GGM.Test}$ on $k = \mathsf{pp}$ and a handle $\xi \in \{0,1\}^\lambda$. First, if $k \neq \mathsf{pp}$ or $\xi \notin \mathsf{T}$, then the output in both $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ is $\perp$. It thus suffices to consider the case where $k = \mathsf{pp}$ and $\xi \in \mathsf{T}$. Let $(\xi \mapsto f)$ be the mapping in T. By construction, in $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$, the polynomial f is an affine polynomial in the formal variables $\hat{q}_{n+1}, \dots, \hat{q}_{n+m}$. Let $f_0, f_1, \dots, f_m \in \mathbb{F}_p$ be the coefficients. Namely, write

$$f(\hat{q}_{n+1}, \dots, \hat{q}_{n+m}) \coloneqq f_0 + \sum_{i \in [m]} f_i \hat{q}_{n+i}.$$

If $f \equiv 0$ is the identically-zero polynomial, then the output in both $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ is "zero." Thus, consider the case where $f \not\equiv 0$. In this case, the output in $\mathsf{Hyb}_{1,i}$ is always "non-zero," whereas in $\mathsf{Hyb}_{1,i-1}$, the output is "zero" if $f(\alpha q_{n+1}, \dots, \alpha q_{n+m}) = 0$, where $\alpha \leftarrow \mathbb{F}_p$ and $(\boldsymbol{q}, A) \leftarrow \mathcal{V}$ are the query components the challenger sampled at the beginning. First, observe that if $f_0 \neq 0$ and $f_1 = \cdots = f_m = 0$, then $f(\hat{q}_{n+1}, \dots, \hat{q}_{n+m}) = f_0 \neq 0$ and the behavior in the two experiments is again identical. It thus suffices to only consider the setting where at least one of $f_1, \dots, f_m$ is non-zero. Now, the key observation is that the challenger's behavior in the first $i-1$ queries of $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ are independent of α and $\boldsymbol{q}$. Indeed, the values of α and $\boldsymbol{q}$ can be sampled *after* the malicious prover outputs its i^{th} query. In particular, this means that the coefficients $f_0, f_1, \dots, f_m$ are *independent* of α and $q_1, \dots, q_m$.

Since the DPP is δ-evasive and at least one of $f_1, \dots, f_m$ is non-zero, we conclude that

$$\Pr_{(\boldsymbol{q},A) \leftarrow \mathcal{V}_\lambda}[\langle \boldsymbol{q}, \mathbf{0}^n \| \boldsymbol{y} \rangle = 0] = \Pr_{(\boldsymbol{q},A) \leftarrow \mathcal{V}_\lambda}\left[\sum_{i \in [m]} f_i q_{n+i} = 0\right] \leq \delta.$$

Now, $f(\alpha q_{n+1}, \dots, \alpha q_{n+m}) = f_0 + \alpha \cdot \langle \boldsymbol{q}, \mathbf{0}^n \| \boldsymbol{y} \rangle$. With probability $1 - \delta$, $\langle \mathbf{0}^n \| \boldsymbol{q}, \boldsymbol{y} \rangle \neq 0$. Thus, $f(\alpha q_1, \dots, \alpha q_{n+m}) = 0$ only if $\alpha = -f_0 / \langle \mathbf{0}^n \| \boldsymbol{q}, \boldsymbol{y} \rangle$. However, since $\alpha \leftarrow \mathbb{F}_p$, this occurs with probability $1/p$. Thus, we conclude that over the choice of $\boldsymbol{q}$ and α,

$$\Pr_{\substack{(\boldsymbol{q},A)\leftarrow\mathcal{V}_\lambda \\ \alpha\leftarrow\mathbb{F}_p}}[f(\alpha q_{n+1},\ldots,\alpha q_{n+m})=0]\le\delta+\frac{1}{p}.$$

We conclude that the challenger's response on the i^{th} query in $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ differ with probability at most $\delta+1/p$.

The above analysis shows that the challenger's response to the i^{th} query in $\mathsf{Hyb}_{1,i-1}$ and $\mathsf{Hyb}_{1,i}$ only differs with probability at most $\delta+1/p$. The claim now follows by a hybrid argument. □

Lemma 7.10. *For all adversaries $\mathcal{P}^*$,* $\Pr[\mathsf{Hyb}_2(\mathcal{P}^*)=1]\le s$.

Proof. By construction, in hybrid Hyb_2, the view of the prover $\mathcal{P}^*$ (prior to $\mathcal{P}^*$ outputting its commitment h) is independent of α and $\boldsymbol{q}$. Namely, the components of $\mathsf{crs}=(\mathsf{pp},p,g,h_1,\ldots,h_m)$ are independent of α and $\boldsymbol{q}$, and likewise for the prover's queries to the generic group oracle. This means that in Hyb_2, the challenger can *defer* sampling $\alpha\leftarrow\mathbb{F}_p$ and $(\boldsymbol{q},A)\leftarrow\mathcal{V}_\lambda$ until *after* the prover outputs $h\in\{0,1\}^\lambda$. Suppose there exists a mapping $(h\mapsto f)$ in T. Otherwise, the output of the experiment is 0. Write $f(\hat{q}_{n+1},\ldots,\hat{q}_{n+m})\coloneqq f_0+\sum_{i\in[m]}f_i\hat{q}_{n+i}$. By assumption, the coefficients $(f_0,\ldots,f_m)$ are independent of $\boldsymbol{q}$ and α. Let $\boldsymbol{\pi}^*=(f_1,\ldots,f_m)$. Then, the output of the experiment is 1 only if

$$\alpha^{-1}f(\alpha q_{n+1},\ldots,\alpha q_{n+m})+\sum_{i\in[n]}x_iq_i=\alpha^{-1}f_0+\langle\boldsymbol{x}\|\boldsymbol{\pi}^*,\boldsymbol{q}\rangle\in A.$$

Recall that $\boldsymbol{x}$ is a false statement. We now consider two cases:

- Suppose $f_0=0$. Since $\boldsymbol{\pi}^*$ is independent of $\boldsymbol{q}$, we can appeal to soundness of the DPP to conclude that $\Pr_{(\boldsymbol{q},A)\leftarrow\mathcal{V}_\lambda}[\langle\boldsymbol{x}\|\boldsymbol{\pi}^*,\boldsymbol{q}\rangle\in A]\le s$.
- Suppose $f_0\ne0$. Since $\alpha\leftarrow\mathbb{F}_p$, the distribution of $\alpha^{-1}f_0$ is uniform over $\mathbb{F}_p$ (and independent of $\langle\boldsymbol{x}\|\boldsymbol{\pi}^*,\boldsymbol{q}\rangle$). Thus,

$$\Pr_{\substack{(\boldsymbol{q},A)\leftarrow\mathcal{V}_\lambda \\ \alpha\leftarrow\mathbb{F}_p}}[\alpha^{-1}f_0+\langle\boldsymbol{q},\boldsymbol{x}\|\boldsymbol{\pi}^*\rangle\in A]\le\frac{|A|}{p}\le s.$$

  Note that $|A|/p\le s$ since otherwise, a uniform random strategy would break soundness with probability greater than s.

We conclude that in Hyb_2, the experiment outputs 1 with probability at most s. □

Soundness now follows by combining Lemmas 7.8 to 7.10.

Laconic prover. The total prover communication consists of a single group element $h\in\mathbb{G}$ and a single field element $t\in\mathbb{F}_p$. Since $\log|\mathbb{G}|,\log p=\operatorname{poly}(\lambda)$, the overall proof size is $\operatorname{poly}(\lambda)$.

Preprocessed verification. This follows by construction. Namely, the verifier's initial (long) message is statement-independent. The subsequent (online) communication consists of the prover's messages (a group element and a field element) and the randomness used in the preprocessing step. The query randomness can be taken to be $\mathrm{poly}(\lambda)$ bits[11] without loss of generality by having the verifier sample it using a seed for any secure cryptographic pseudorandom generator (PRG). □

Concrete instantiations. We now describe how to instantiate Construction 7.6 for the language of Boolean circuit satisfiability.

- Let $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit of size s and consisting of fan-in-2 NAND gates. By Corollary 5.11, there exists a δ-evasive DPP of length $m = 2s \log \frac{3.1(n+2s)}{\varepsilon}$ over $\mathbb{F}_p$ with soundness error ε as long as the following conditions hold:

$$\frac{1}{\delta} > \frac{3850}{\varepsilon^2}(n+2s)^4 \log \frac{3.1(n+2s)}{\varepsilon} \quad \text{and} \quad p > \frac{93}{\delta\varepsilon^4}(n+2s)^6 \left(\log \frac{3.1(n+2s)}{\varepsilon}\right)^3 .$$

- From Theorem 7.7, we obtain a laconic argument with soundness error $\varepsilon + \delta Q + 2Q/p$ in the generic group model (as formulated in Definition E.1). As discussed in Remark E.2, the analogous theorem statement in Shoup's generic group model (where encodings are unique) would bound the soundness error by $\varepsilon + \delta(Q+m+1)^2 + 2(Q+m+1)^2/p$.

  Concretely, any adversary $\mathcal{A}$ that makes Q queries in the generic group model with unique encodings can be converted into an adversary $\mathcal{B}$ that makes $(Q+m+1)^2$ queries in the generic group model with the zero-test oracle (Definition E.1). Adversary $\mathcal{B}$ would implement $\mathsf{GGM.Add}$ by additionally performing $\mathsf{GGM.Test}$ between the output encoding and all of the previous encodings (and outputting the previous encoding whenever $\mathsf{GGM.Test}$ outputs 1). The number of encodings is $Q+m+1$ (Q encodings from the adversary's Q generic group oracle queries, and $m+1$ encodings from the common reference string).

Suppose we instantiate the group $\mathbb{G}$ in Construction 7.6 with a 256-bit elliptic curve group (e.g., P-256 or Curve25519). In this case $\log p = 256$. In Table 1, we describe the soundness achieved by our construction for circuits of different sizes. We fix the statement size to be $n = 128$ and assume the adversary makes at most $Q = 2^{75}$ queries to the generic group oracle. We set $\delta = 2^{-157}$. In this case, $\delta(Q+m+1)^2 + 2(Q+m+1)^2/p \approx 2^{-7}$.

Note that while 2^{75} group operations may be feasible in practice (we estimate that a supercomputer can complete this computation in a few years), it is arguably reasonable in the context of lightweight cryptography, where even block ciphers with 80-bit keys are sometimes considered acceptable [ISO19]. Such a bound on the number of group operations is even more conservative if the prover is limited to "online" attacks that must take place in a reasonable amount of time, say one hour. The latter can be achieved by keeping the encoded DPP query encrypted until shortly before they are used (and have the verifier provide the decryption key as the first message). Finally, we can get a more standard level of security with a small increase to the group size (e.g., a 2^{100}-query bound with 302-bit groups).

Our laconic argument gives a concretely-efficient laconic argument for proving relations on simple Boolean circuits with very low communication and verifier computation. The size of the initial

[11] In practice, a PRG seed is typically 128-bits long.

Circuit Size s	**Soundness Error** ε	**Total Online Communication**
2^8	$2^{-5.5}$	640 bits
2^{10}	$2^{-3.4}$	640 bits
2^{12}	$2^{-0.6}$	640 bits

Table 1: Laconic argument for Boolean circuit satisfiability (Construction 7.6) for proving satisfiability of a Boolean circuit of size s, counting inputs and NAND gates. We fix the statement size to be $n = 128$ and instantiate the group $\mathbb{G}$ using a 256-bit group (e.g., P-256 or Curve25519). For each instantiation, we report the minimum soundness error ε achieved by our construction against adversaries performing at most 2^{75} group operations. In the laconic argument, the verifier's initial message consists of $O(s \log s)$ group elements and the verifier's online message is at most 128 bits (corresponding to either the DPP randomness or a PRG seed). The prover's message consists of a group element and a field element; we assume that each can be represented by a 256-bit string. We also note that if we instead work over a 302-bit group, then we achieve comparable levels of soundness against adversaries that can perform up to 2^{100} group operations. Using a 302-bit group increases the total online communication by 14% from 640 bits to 732 bits.

(offline) verifier message is *linear* in the size of the circuit, and the total online communication consists of a group element, a field element, and a short PRG seed (or the DPP verification randomness). Moreover, the computational cost of verification is extremely small: just a single group exponentiation. This makes our approach particularly well-suited for verifiers that are running on weak or energy-constrained devices. While the current construction does not achieve a high level of soundness, this can still be useful in settings where there are out-of-band mechanisms for incentivizing honest behavior (e.g., strong penalties if a prover is caught behaving maliciously). This is similar to the notion of covert security in the context of multiparty computation [CO99, AL10]. We conclude with a brief comparison against previous approaches here:

- The most succinct *pairing-based* SNARGs [Gro16, Lip24, DMS24] have longer proofs. The current state-of-the-art is Pari [DMS24] which has proofs of size 1280 bits at the 128-bit security level. In all of these constructions, the verifier also needs to compute more expensive pairings. The total communication is 2× smaller for our construction (640 bits) and the only cryptographic operation for our verifier is a single exponentiation in a *pairing-free* group. On the flip side, the pairing-based constructions are non-interactive, publicly-verifiable, and have negligible soundness error.

- If we consider concretely-efficient schemes with proofs that are shorter than those in [Gro16], the best previous construction is due to Barta, Ishai, Ostrovsky, and Wu [BIOW20].[12] They give a designated-verifier SNARG where the proof consists of two group elements in a pairing-free group. Concretely, this yields a construction where the proof size is just 512 bits (20% shorter than our construction). However, the drawback of their construction is the size of the CRS scales *quadratically* with the size of the circuit. For instance, for a circuit with 2^{12} gates, the size of the CRS in their construction would be 256 MB, whereas in our scheme, it is just 5.5 MB. Note that the non-interactive version of the [BIOW20] scheme either storing a large verification state or requiring a large verification time (for computing discrete logarithms). One

[12] See Section 1.2 for comparison with the subsequent work of Arnon, Dujmovic, and Ishai [ADI25].

may also consider an interactive variant (analogous to Construction 7.6) with slightly longer proofs (same as Construction 7.6) which would achieve comparable verification complexity to our construction.

Remark 7.11 (Proving Batch Statements)**.** *Construction 7.6 operates in the setting where the prover is trying to convince the verifier of a single NP statement x. The same construction and analysis can also be used in a batch setting, where there are N statements $x_1, \dots, x_n$, and the prover simultaneously proves that there are witnesses $w_1, \dots, w_N$ such that $C(x_i, w_i) = 1$ for all $i \in [N]$. In this case, we can use the same (offline) statement-independent message to verify all online statements. Importantly, this offline message length is independent of the number of statements. Moreover, the* same *short verifier-to-prover message can be used for proving all statements.*

Remark 7.12 (Reducing Communication with a Hashed Commitment)**.** *In Construction 7.6, the prover commits to its DPP proof $\boldsymbol{\pi}$ by publishing a group element $h \in \mathbb{G}$. In this case, to provide λ bits of security, the size of the group element h is at least 2λ bits. One approach to reduce the communication is for the prover to commit using a hash $\mathcal{H}(h)$ of the group element rather than the group element. Here, the output length of the hash function can be exactly λ-bits. When the hash function $\mathcal{H} \colon \mathbb{G} \to \{0,1\}^\lambda$ is modeled as a random oracle, we can argue that the construction still preserves soundness with a $\operatorname{poly}(\lambda, \log Q)$-increase in the soundness error, where Q is the number of random oracle queries the adversary makes. The increase in the soundness error is due to the fact that a malicious prover can potentially find multiple vectors $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \dots, \boldsymbol{\pi}_T$ where $\mathcal{H}(\boldsymbol{\pi}_i) = \mathcal{H}(\boldsymbol{\pi}_1)$ for all $i \in [T]$. If* any *of these strategies convinces the verifier, then the prover succeeds. By a union bound, if the underlying DPP has soundness error s, then with probability Ts, none of the strategies $\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_T$ are successful. Finally, if we model $\mathcal{H}$ as a random oracle, then we can show that an adversary making at most Q queries to the random oracle will only be able to find $T = \lambda \log Q$ vectors with the same hash, except perhaps with probability $2^{-\Omega(\lambda)}$. To summarize this strategy reduces the size of the prover commitment by roughly a factor of 2 at the expense of a modest increase to the soundness error.*

Using a hashed commitment is beneficial when the soundness error of the underlying DPP is at most $s \le 1/(\lambda \log Q)$. This is due to the increase in soundness incurred from replacing the group element with a hash of the group element. Our current concrete instantiations of DPPs over large fields in Table 1 have high soundness error, so this approach does not help improve the concrete efficiency of our instantiations. However, as we see no barrier to constructing DPPs over large fields with better soundness, we believe this approach for reducing prover communication could enable new concretely-efficient laconic arguments in the future.

7.3 From DPP to Succinct Commit-and-Prove Arguments

DPPs (and, more broadly, fully linear PCPs) are also useful for constructing succinct *commit-and-prove arguments* [Kil92, CLOS02, EG14, CFHK+15]. In this setting, a prover first commits to an input $\boldsymbol{x} \in \{0,1\}^\ell$ with a short digest σ and later on, can provide succinct openings $\pi_1, \dots, \pi_\ell$ to different functions $f_1, \dots, f_\ell$ of the committed input (i.e., that $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$ for each $i \in [\ell]$). The primary efficiency requirement is that the size of the commitment σ as well as the size of each proof π_i be sublinear in both the input length $|\boldsymbol{x}|$ and the size of the Boolean (or arithmetic) circuit computing f. The security requirement is knowledge soundness which requires that for every efficient adversary that is able to produce a commitment σ together with openings $\pi_1, \dots, \pi_\ell$ for function-value pairs $(f_1, \boldsymbol{y}_1), \ldots, (f_\ell, \boldsymbol{y}_\ell)$, there exists an efficient extractor that can output an input $\boldsymbol{x} \in \{0,1\}^\ell$ such that $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$ for all $i \in [\ell]$. We provide the formal definition below:

Definition 7.13 (Succinct Commit-and-Prove Argument)**.** *Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of Boolean functions indexed by a security parameter, where each $\mathcal{F}_\lambda$ is a set of Boolean functions $f \colon \{0,1\}^\ell \to \{0,1\}^t$ on inputs of length $\ell = \ell(\lambda)$ and outputs of length $t = t(\lambda)$. A succinct commit-and-prove argument system for $\mathcal{F}$ is a tuple of probabilistic polynomial-time algorithms* ($\mathsf{Setup}, \mathsf{Commit}, \mathsf{Prove}, \mathsf{Verify}$) *with the following syntax:*

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{crs}, \mathsf{vk})$*: On input the security parameter, the setup algorithm outputs a common reference string* crs *together with a verification key* vk*.*
- $\mathsf{Commit}(\mathsf{crs}, \boldsymbol{x}) \to (\sigma, \mathsf{st})$*: On input the common reference string* crs *and an input $\boldsymbol{x} \in \{0,1\}^\ell$, the commit algorithm outputs a commitment σ together with a commitment state* st*.*
- $\mathsf{Prove}(\mathsf{st}, f) \to \pi$*: On input the commitment state* st *and a function $f \colon \{0,1\}^\ell \to \{0,1\}^t$, the prove algorithm outputs a proof π. The proof π is sometimes also referred to as an* opening*.*
- $\mathsf{Verify}(\mathsf{vk}, \sigma, f, \boldsymbol{y}, \pi) \to b$*: On input the verification key* vk*, the commitment σ, a function $f \colon \{0,1\}^\ell \to \{0,1\}^t$, a value $\boldsymbol{y} \in \{0,1\}^t$, and a proof π, the verification algorithm outputs a bit $b \in \{0,1\}$.*

Moreover, the commit-and-prove argument should satisfy the following properties:

- *(Completeness:) for every $\lambda \in \mathbb{N}$, every input $\boldsymbol{x} \in \{0,1\}^{\ell(\lambda)}$, and every function $f \in \mathcal{F}_\lambda$,*

$$\Pr\left[\mathsf{Verify}(\mathsf{vk}, \sigma, f, f(\boldsymbol{x}), \pi) = 1 : \begin{array}{c} (\mathsf{crs}, \mathsf{vk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ (\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \boldsymbol{x}) \\ \pi \leftarrow \mathsf{Prove}(\mathsf{st}, f) \end{array}\right] \geq c(\lambda)$$

  where $c \in [0,1]$ is called the completeness parameter*. By default, we assume perfect correctness, that is, $c = 1$.*

- *(Knowledge soundness:) for every efficient adversary $\mathcal{A}$, there exists an efficient extractor $\mathcal{E}$ such that for all $\lambda \in \mathbb{N}$,*

$$\Pr\left[\begin{array}{c} \exists i \in [k] : \mathsf{Verify}(\mathsf{vk}, \sigma, f_i, \boldsymbol{y}_i, \pi_i) = 1 \\ \textit{and } f_i(\boldsymbol{x}) \neq \boldsymbol{y}_i \end{array} : \begin{array}{c} (\mathsf{crs}, \mathsf{vk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \left(\sigma, \{(f_i, \boldsymbol{y}_i, \pi_i)\}_{i \in [k]}\right) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}; r_\mathcal{A}) \\ \boldsymbol{x} \leftarrow \mathcal{E}(1^\lambda, \mathsf{crs}; r_\mathcal{A}) \end{array}\right] \leq \varepsilon(\lambda),$$

  where $\varepsilon \in [0,1]$ is called the knowledge soundness error *and $r_\mathcal{A}$ denotes the (uniform) random coins used by adversary $\mathcal{A}$.*

- *(Succinctness:) There exists a universal polynomial* $\mathrm{poly}(\cdot)$ *such that for all security parameters $\lambda \in \mathbb{N}$, every input $x \in \{0,1\}^{\ell(\lambda)}$, every function $f \in \mathcal{F}_\lambda$, every (σ, st) in the support of $\mathsf{Commit}(\mathsf{crs}, \boldsymbol{x})$, and every proof π in the support of $\mathsf{Prove}(\mathsf{st}, f)$, it holds that $|\sigma|, |\pi| = \mathrm{poly}(\lambda + \log|\boldsymbol{x}| + \log|f|)$.*

Remark 7.14 (Public Verification vs. Designated Verifier)**.** *Similar to Remark D.2, we can also consider a publicly-verifiable commit-and-prove argument where the verification state is simply the common reference string.*

Remark 7.15 (Relationship to Functional Commitments). *Succinct commit-and-prove arguments are also referred to as functional commitments [IKO07, BC12, LRY16]. In the setting of (non-interactive) functional commitments, the knowledge soundness requirement is often relaxed to a weaker* evaluation binding *property which requires that a computationally-bounded adversary cannot produce a commitment σ together with accepting proofs π_1, π_2 to distinct values $\boldsymbol{y}_1 \neq \boldsymbol{y}_2$ with respect to a single function f. However, even if the adversary opens a commitment σ to $\boldsymbol{y}$ with respect to a function f, evaluation binding does not require that there exist any $\boldsymbol{x}$ where $\boldsymbol{y} = f(\boldsymbol{x})$. On the flip side, the weaker evaluation binding property is a falsifiable property [GW11], and a recent line of works have shown how to construct (publicly-verifiable) functional commitments for general functions from falsifiable number-theoretic assumptions [CP23, WW23b, BCFL23, WW23a, WW24b, ABF24].*

Succinct commit-and-prove arguments from DPPs **and linear-only encryption.** With a simple adaptation of the [BCIOP22] compiler, we can combine any DPP (and more generally, any FLPCP) with a linear-only encryption scheme to obtain a succinct commit-and-prove argument. The commitment and the opening in our scheme consists of a single ciphertext for the underlying linear-only encryption scheme. We describe the construction below:

Construction 7.16 (Succinct Commit-and-Prove Argument). *Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of Boolean functions $f\colon \{0,1\}^\ell \to \{0,1\}^t$. Suppose that each function $f \in \mathcal{F}_\lambda$ can be described by a bit-string of length $s = s(\lambda)$. Define the language*

$$\mathcal{L} = \{(\boldsymbol{x}, f, \boldsymbol{y}) \in \{0,1\}^\ell \times \{0,1\}^s \times \{0,1\}^t : \boldsymbol{y} = f(\boldsymbol{x})\}.$$

Let $n = n(\lambda) = \ell(\lambda) + s(\lambda) + t(\lambda)$ be the length of a statement $(\boldsymbol{x}, f, \boldsymbol{y})$, Let $(\mathcal{P}, \mathcal{V}) = \{(\mathcal{P}_\lambda, \mathcal{V}_\lambda)\}_{\lambda \in \mathbb{N}}$ be a DPP *for $\mathcal{L}$ over a finite field $\mathbb{F}$ with proof length $m = m(\lambda)$. Let $\Pi_{\mathsf{Enc}} = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add}, \mathsf{ImVer})$ be a linear-only encryption scheme with plaintext space $\mathbb{F}$ (see Definition D.3). We construct a succinct commit-and-prove argument for $\mathcal{F}$ as follows:*

- $\mathsf{Setup}(1^\lambda)$*: On input the security parameter λ, the setup algorithm samples a* DPP *query $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$, a public/private key-pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and a random scalar $\alpha \leftarrow \mathbb{F} \setminus \{0\}$. For $i \in [\ell]$, it computes $\mathsf{ct}_{\boldsymbol{x},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \alpha q_i)$, and for $i \in [m]$, it computes $\mathsf{ct}_{\boldsymbol{\pi},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, q_{n+i})$. It outputs $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$ along with the verification key $\mathsf{vk} = (\mathsf{sk}, \boldsymbol{q}', \alpha, A)$, where $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$.*

- $\mathsf{Commit}(\mathsf{crs}, \boldsymbol{x})$*: On input $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$ and the input $\boldsymbol{x} \in \{0,1\}^\ell$, the commitment algorithm views $\mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}$ as an encryption of $\boldsymbol{q}_{\boldsymbol{x}} \in \mathbb{F}^\ell$ and uses* Add *to homomorphically compute a ciphertext $\mathsf{ct}_{\boldsymbol{x}}$ that encrypts $\langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle$. It outputs the commitment $\sigma = \mathsf{ct}_{\boldsymbol{x}}$ and the commitment state $\mathsf{st} = (\mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}, \boldsymbol{x})$.*

- $\mathsf{Prove}(\mathsf{st}, f)$*: On input the commitment state $\mathsf{st} = (\mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}, \boldsymbol{x})$ and a function f, the prover algorithm computes $\boldsymbol{y} = f(\boldsymbol{x})$. Then, it construct a* DPP *proof $\boldsymbol{\pi} \leftarrow \mathcal{P}_\lambda((\boldsymbol{x}, f, \boldsymbol{y}), \perp)$. It views $\mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}$ as an encryption of $\boldsymbol{q}_{\boldsymbol{\pi}} \in \mathbb{F}^m$ and uses* Add *to homomorphically compute a ciphertext $\mathsf{ct}_{\boldsymbol{\pi}}$ that encrypts $\langle \boldsymbol{q}_{\boldsymbol{\pi}}, \boldsymbol{\pi} \rangle$. It outputs the proof $\pi = \mathsf{ct}_{\boldsymbol{\pi}}$.*

- $\mathsf{Verify}(\mathsf{vk}, \sigma, f, \boldsymbol{y}, \pi)$*: On input the verification key $\mathsf{vk} = (\mathsf{sk}, \boldsymbol{q}', \alpha, A)$, a commitment $\sigma = \mathsf{ct}_{\boldsymbol{x}}$, a function $f \in \{0,1\}^s$, an output $\boldsymbol{y} \in \{0,1\}^t$, and a proof $\pi = \mathsf{ct}_{\boldsymbol{\pi}}$, the verification algorithm first checks that $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1$ and $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}) = 1$. If either check fails, then the verification algorithm outputs 0. Otherwise, it computes $z_{\boldsymbol{x}} \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}})$, $z_{\boldsymbol{\pi}} \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}})$, and $z = \alpha^{-1} z_{\boldsymbol{x}} + z_{\boldsymbol{\pi}} + \langle f \| \boldsymbol{y}, \boldsymbol{q}' \rangle$. The algorithm outputs 1 if $z \in A$ and 0 otherwise.*

Theorem 7.17 (Succinct Commit-and-Prove Argument)**.** *Suppose $(\mathcal{P}, \mathcal{V})$ has completeness c and soundness error s against affine strategies. Then Construction 7.16 has completeness c and soundness error $k \cdot \left(s + \frac{2}{|\mathbb{F}|-1}\right) + \mathsf{negl}(\lambda)$, where k is a bound on the number of openings the adversary outputs.*

Proof. We show each requirement separately.

Completeness. Take any security parameter $\lambda \in \mathbb{N}$, input $\boldsymbol{x} \in \{0,1\}^\ell$, and function $f \in \mathcal{F}_\lambda$. Let $(\mathsf{crs}, \mathsf{vk}) \leftarrow \mathsf{Setup}(1^\lambda)$. By construction, the Setup algorithm samples $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and $\alpha \leftarrow \mathbb{F} \setminus \{0\}$. Then, $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$ and $\mathsf{vk} = (\mathsf{sk}, \boldsymbol{q}', \alpha, A)$, where $\mathsf{ct}_{\boldsymbol{x},i}$ is an encryption of αq_i, $\mathsf{ct}_{\boldsymbol{\pi},i}$ is an encryption of q_{n+i}, and $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$. Next, let $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \boldsymbol{x})$. Then, $\sigma = \mathsf{ct}_{\boldsymbol{x}}$ and $\mathsf{st} = (\mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}, \boldsymbol{x})$. By correctness of the linear-only encryption scheme, we have that

$$\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = \alpha \langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle \wedge \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1,$$

where $\boldsymbol{q}_{\boldsymbol{x}} = [q_1, \ldots, q_\ell]$. Next, let $\pi \leftarrow \mathsf{Prove}(\mathsf{st}, f)$. By construction, the Prove algorithm first constructs a DPP proof $\boldsymbol{\pi} \leftarrow \mathcal{P}_\lambda((\boldsymbol{x}, f, \boldsymbol{y}), \perp)$ and homomorphically computes the proof $\pi = \mathsf{ct}_{\boldsymbol{\pi}}$. By correctness of the linear-only encryption scheme, we have that

$$\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}) = \langle \boldsymbol{q}_{\boldsymbol{\pi}}, \boldsymbol{\pi} \rangle \wedge \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}) = 1,$$

where $\boldsymbol{q}_{\boldsymbol{\pi}} = [q_{n+1}, \ldots, q_{n+m}]$. Finally, consider the output of $\mathsf{Verify}(\mathsf{vk}, \sigma, f, \boldsymbol{y}, \pi)$. As argued above, $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1$ and $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}) = 1$. Similarly, $z_{\boldsymbol{x}} = \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = \alpha \langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle$ and $z_{\boldsymbol{\pi}} = \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}) = \langle \boldsymbol{q}_{\boldsymbol{\pi}}, \boldsymbol{\pi} \rangle$. Moreover,

$$z = \alpha^{-1} z_{\boldsymbol{x}} + z_{\boldsymbol{\pi}} + \left\langle f \| \boldsymbol{y}, \boldsymbol{q}' \right\rangle = \langle \boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{x} \rangle + \langle \boldsymbol{q}_{\boldsymbol{\pi}}, \boldsymbol{\pi} \rangle + \left\langle \boldsymbol{q}', f \| \boldsymbol{y} \right\rangle = \langle \boldsymbol{q}, \boldsymbol{x} \| f \| \boldsymbol{y} \| \boldsymbol{\pi} \rangle,$$

since $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$. By completeness of DPP, this means that $z \in A$ with probability at least c. Correctness follows.

Knowledge soundness. Let $\mathcal{A}$ be any efficient adversary for the knowledge soundness security game. We now describe how to construct an efficient extractor algorithm $\mathcal{E}$ from $\mathcal{A}$. First, we define a "wrapper algorithm" $\mathcal{A}'$ as follows:

- $\mathcal{A}'(\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}})$: On input a public key, a collection of ciphertexts $\mathsf{ct}_{\boldsymbol{x},i}$ and $\mathsf{ct}_{\boldsymbol{\pi},i}$, and randomness $r_{\mathcal{A}}$, let $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$. Invoke $\left(\sigma, \{(f_i, \boldsymbol{y}_i, \pi_i)\}_{i \in [k]}\right) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$ and output $(\sigma, \pi_1, \ldots, \pi_k)$.

The wrapper algorithm $\mathcal{A}'$ conforms to the syntactic requirements for the linear-only security game. Let $\mathcal{E}_{\mathsf{LO}}$ be the linear-only extractor associated with $\mathcal{A}'$. We use $\mathcal{E}_{\mathsf{LO}}$ to construct the knowledge extractor $\mathcal{E}$ for $\mathcal{A}$ as follows:

- $\mathcal{E}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$: On input $\lambda \in \mathbb{N}$, $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$, and randomness $r_{\mathcal{A}}$, run $(\boldsymbol{\Pi}, \boldsymbol{b}) \leftarrow \mathcal{E}_{\mathsf{LO}}(\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}})$. Let $\boldsymbol{\Pi}_1^\intercal \in \mathbb{F}^{\ell+m}$ be the first row of $\boldsymbol{\Pi}$, and parse $\boldsymbol{\Pi}_1^\intercal = \boldsymbol{x} \| \boldsymbol{z}$ where $\boldsymbol{x} \in \mathbb{F}^\ell$ and $\boldsymbol{z} \in \mathbb{F}^m$. If $\boldsymbol{x} \in \{0,1\}^\ell$, output $\boldsymbol{x}$; otherwise, output $\perp$.

We now argue that the extractor $\mathcal{E}$ satisfies the required property. We begin by defining a sequence of hybrid experiments:

- Hyb_0: This is the real knowledge soundness experiment:
  - The challenger first samples a DPP query $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$, a public/private key-pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and a random scalar $\alpha \leftarrow \mathbb{F} \setminus \{0\}$. For $i \in [\ell]$, it computes $\mathsf{ct}_{\boldsymbol{x},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \alpha q_i)$, and for $i \in [m]$, it computes $\mathsf{ct}_{\boldsymbol{\pi},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, q_{n+i})$. It sets $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$.
  - The challenger runs $\big(\sigma, \{(f_i, \boldsymbol{y}_i, \pi_i)\}_{i \in [k]}\big) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$, where $r_{\mathcal{A}}$ is a uniform random string. It parses $\sigma = \mathsf{ct}_{\boldsymbol{x}}$ and $\pi_i = \mathsf{ct}_{\boldsymbol{\pi}}^{(i)}$ for each $i \in [k]$.
  - The challenger computes $\boldsymbol{x} \leftarrow \mathcal{E}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$. In addition, the challenger computes $z_{\boldsymbol{x}} \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}})$, and for each $i \in [k]$, $z_{\boldsymbol{\pi}}^{(i)} \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)})$. Finally, it sets $z_i = \alpha^{-1} z_{\boldsymbol{x}} + z_{\boldsymbol{\pi}}^{(i)} + \langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \rangle$ where $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$.
  - The output of the experiment is 1 if there exists an index $i \in [k]$ such that $f_i(\boldsymbol{x}) \neq \boldsymbol{y}_i$, $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1 = \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)}) = 1$, and $z_i \in A$.
- Hyb_1: Same as Hyb_0, except the challenger computes $z_{\boldsymbol{x}}$ and $z_{\boldsymbol{\pi}}^{(i)}$ using the linear-only extractor:
  - The challenger computes $(\boldsymbol{\Pi}, \boldsymbol{b}) \leftarrow \mathcal{E}_{\mathsf{LO}}(\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}})$, where $\boldsymbol{\Pi} \in \mathbb{F}^{(k+1)\times(\ell+m)}$ and $\boldsymbol{b} \in \mathbb{F}^{k+1}$
  - Next, it sets
  $$\begin{bmatrix} \hat{z}_{\boldsymbol{x}} \\ \hat{z}_{\boldsymbol{\pi}}^{(1)} \\ \vdots \\ \hat{z}_{\boldsymbol{\pi}}^{(k)} \end{bmatrix} = \boldsymbol{\Pi} \cdot \begin{bmatrix} \alpha q_1 \\ \vdots \\ \alpha q_\ell \\ q_{n+1} \\ \vdots \\ q_{n+m} \end{bmatrix} + \boldsymbol{b}. \tag{7.1}$$

  The output of the experiment is 1 if there exists an index $i \in [k]$ such that $f_i(\boldsymbol{x}) \neq \boldsymbol{y}_i$ and $\hat{z}_i \in A$, where $\hat{z}_i = \alpha^{-1}\hat{z}_{\boldsymbol{x}} + \hat{z}_{\boldsymbol{\pi}}^{(i)} + \langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \rangle$. Notably, in this experiment, the challenger does not check if $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1 = \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)}) = 1$.
- Hyb_2: Same as Hyb_1, except the challenger now computes $\mathsf{ct}_{\boldsymbol{x},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, 0)$ for all $i \in [\ell]$. Similarly, for all $i \in [m]$, the challenger computes $\mathsf{ct}_{\boldsymbol{\pi},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, 0)$. In this experiment, the challenger can defer the sampling of $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$ until *after* it computes $(\boldsymbol{\Pi}, \boldsymbol{b})$.

We write $\mathsf{Hyb}_i(\mathcal{A}, \mathcal{E})$ to denote the random variable corresponding to the output of an execution of Hyb_i with adversary $\mathcal{A}$ and extractor $\mathcal{E}$. We now analyze each pair of adjacent distributions.

Lemma 7.18. *If Π_{Enc} is linear-only, then there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Hyb}_1(\mathcal{A}, \mathcal{E}) = 1] \geq \Pr[\mathsf{Hyb}_0(\mathcal{A}, \mathcal{E}) = 1] - \mathsf{negl}(\lambda)$.*

Proof. Suppose that $\mathsf{Hyb}_0(\mathcal{A}, \mathcal{E}) = 1$. This means that there exists an index $i \in [k]$ where

$$f_i(\boldsymbol{x}) \neq \boldsymbol{y}_i \quad \text{and} \quad \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1 = \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)}) = 1 \quad \text{and} \quad z_i \in A,$$

where $z_i = \alpha^{-1} z_{\boldsymbol{x}} + z_{\boldsymbol{\pi}}^{(i)} + \langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \rangle$, $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$, $z_{\boldsymbol{x}} \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}})$ and $z_{\boldsymbol{\pi}}^{(i)} \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)})$. We argue that in this case, the output in Hyb_1 is also 1 with all but negligible probability. Suppose otherwise: namely, that with non-negligible probability ε, the output in Hyb_1 is 0. By construction then, it must be the case that in Hyb_1, $\hat{z}_i \notin A$. By construction of $\hat{z}_i$, this means either

$$\hat{z}_{\boldsymbol{x}} \neq z_{\boldsymbol{x}} \quad \text{or} \quad z_{\boldsymbol{\pi}}^{(i)} \neq \hat{z}_{\boldsymbol{\pi}}^{(i)}. \tag{7.2}$$

We claim that this breaks the linear-only property of Π_{Enc}. Specifically, define a message sampler $\mathcal{M}$ that takes as input a public key pk, samples $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$, and outputs $(\alpha q_1, \ldots, \alpha q_\ell, q_{n+1}, \ldots, q_{n+m})$. Then let E be the following event:

- Sample $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $(\alpha q_1, \ldots, \alpha q_\ell, q_{n+1}, \ldots, q_{n+m}) \leftarrow \mathcal{M}(\mathsf{pk})$.
- Let $\mathsf{ct}_{\boldsymbol{x},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, q_i)$ for all $i \in [\ell]$ and $\mathsf{ct}_{\boldsymbol{\pi},i} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, q_{n+i})$ for all $i \in [m]$.
- Let $(\mathsf{ct}'_1, \ldots, \mathsf{ct}'_k) \leftarrow \mathcal{A}'(\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}})$, where $r_{\mathcal{A}}$ is a uniform random string. By construction of $\mathcal{A}'$, the output is derived by running $\mathcal{A}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$, where $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$.
- Let $(\boldsymbol{\Pi}, \boldsymbol{b}) \leftarrow \mathcal{E}(\mathsf{pk}, \mathsf{ct}_1, \ldots, \mathsf{ct}_m; r_{\mathcal{A}})$ and compute $\hat{z}_{\boldsymbol{x}}, \hat{z}_{\boldsymbol{\pi}}^{(1)}, \ldots, \hat{z}_{\boldsymbol{\pi}}^{(k)}$ according to Eq. (7.1).
- We say E occurs if $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) = 1$ and $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{x}}) \neq \hat{z}_{\boldsymbol{x}}$ or if $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)}) = 1$ and $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_{\boldsymbol{\pi}}^{(i)}) \neq \hat{z}_{\boldsymbol{\pi}}^{(i)}$.

By construction, if $\mathsf{Hyb}_0(\mathcal{A}, \mathcal{E}) = 1$ and Eq. (7.2) holds, then event E occurs. But this event precisely coincides with the winning condition in the linear-only security game, so algorithm $\mathcal{A}'$ breaks linear-only security with the same advantage ε. Thus, we conclude that $\varepsilon = \mathsf{negl}(\lambda)$. In other words, whenever $\mathsf{Hyb}_0(\mathcal{A}, \mathcal{E}) = 1$, with all but $\varepsilon = \mathsf{negl}(\lambda)$ probability, $\mathsf{Hyb}_1(\mathcal{A}, \mathcal{E}) = 1$, and the claim holds.

Lemma 7.19. *If Π_{Enc} is semantically secure, then there exists a negligible function such that for all $\lambda \in \mathbb{N}$, $|\Pr[\mathsf{Hyb}_2(\mathcal{A}, \mathcal{E}) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, \mathcal{E}) = 1]| = \mathsf{negl}(\lambda)$.*

Proof. Suppose $|\Pr[\mathsf{Hyb}_2(\mathcal{A}, \mathcal{E}) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, \mathcal{E}) = 1]| \geq \varepsilon(\lambda)$ for some non-negligible $\varepsilon = \varepsilon(\lambda)$. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the semantic security experiment for Π_{Enc}:

- On input the public key pk, algorithm $\mathcal{B}$ samples $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$. It submits the vectors $(\alpha q_1, \ldots, \alpha q_\ell, q_{n+1}, \ldots, q_{n+m})$ and $\mathbf{0}^{\ell+m}$ to $\mathcal{A}$.
- The challenger replies with a collection of ciphertexts $(\mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m})$.
- Algorithm $\mathcal{B}$ sets $\mathsf{crs} = (\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}})$ and runs $(\sigma, \{(f_i, \boldsymbol{y}_i, \pi_i)\}_{i \in [k]}) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$, where $r_{\mathcal{A}}$ is a uniform random string.
- Algorithm $\mathcal{B}$ computes the extracted input $\boldsymbol{x} \leftarrow \mathcal{E}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$ and the extracted linear function $(\boldsymbol{\Pi}, \boldsymbol{b}) \leftarrow \mathcal{E}_{\mathsf{LO}}(\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}})$.
- Algorithm $\mathcal{B}$ computes $\hat{z}_{\boldsymbol{x}}, \hat{z}_{\boldsymbol{\pi}}^{(1)}, \ldots, \hat{z}_{\boldsymbol{\pi}}^{(k)}$ according to Eq. (7.1) using $(\boldsymbol{\Pi}, \boldsymbol{b})$, α, and $\boldsymbol{q}$. Finally, algorithm $\mathcal{B}$ outputs 1 if there exists an index $i \in [k]$ such that $f_i(\boldsymbol{x}) \neq \boldsymbol{y}_i$ and $\hat{z}_i \in A$, where $\hat{z}_i = \alpha^{-1} \hat{z}_{\boldsymbol{x}} + \hat{z}_{\boldsymbol{\pi}}^{(i)} + \langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \rangle$ and $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$.

By construction, if the challenger encrypts the vector $(\alpha q_1, \ldots, \alpha q_\ell, q_{n+1}, \ldots, q_{n+m})$, then algorithm $\mathcal{B}$ computes the output according to the specification of $\mathsf{Hyb}_1(\mathcal{A}, \mathcal{E})$ whereas if the challenger encrypts the vector $\mathbf{0}^{\ell+m}$, then algorithm $\mathcal{B}$ computes the output according to the specification of $\mathsf{Hyb}_2(\mathcal{A}, \mathcal{E})$. Correspondingly, algorithm $\mathcal{B}$ breaks semantic security with the same advantage ε. □

Lemma 7.20. *Suppose* DPP *has soundness error s. Then, for all adversaries $\mathcal{A}$ that output at most k possible openings,* $\Pr[\mathsf{Hyb}_2(\mathcal{A}, \mathcal{E}) = 1] \le k \cdot \left(s + \frac{2}{|\mathbb{F}|-1}\right)$.

Proof. In Hyb_2, the adversary's view is independent of the DPP query $\boldsymbol{q}$ (and the random scalar α), so we can defer the sampling of α and $\boldsymbol{q}$ until *after* running $\mathcal{A}$ and $\mathcal{E}$. Consider such an execution of Hyb_2. Then, we have the following:

- Let $\left(\sigma, \{(f_i, \boldsymbol{y}_i, \pi_i)\}_{i \in [k]}\right) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}; r_{\mathcal{A}})$.
- Let $(\boldsymbol{\Pi}, \boldsymbol{b}) \leftarrow \mathcal{E}_{\mathsf{LO}}\left(\mathsf{pk}, \mathsf{ct}_{\boldsymbol{x},1}, \ldots, \mathsf{ct}_{\boldsymbol{x},\ell}, \mathsf{ct}_{\boldsymbol{\pi},1}, \ldots, \mathsf{ct}_{\boldsymbol{\pi},m}; r_{\mathcal{A}}\right)$. Let $\boldsymbol{\Pi}_1^{\mathsf{T}} \in \mathbb{F}^{\ell+m}$ be the first row of $\boldsymbol{\Pi}$, and parse $\boldsymbol{\Pi}_1^{\mathsf{T}} = \boldsymbol{x}_\sigma \| \boldsymbol{z}_\sigma$ where $\boldsymbol{x}_\sigma \in \mathbb{F}^\ell$ and $\boldsymbol{z}_\sigma \in \mathbb{F}^m$. If $\boldsymbol{x}_\sigma \notin \{0,1\}^\ell$, then let $\boldsymbol{x} = \bot$; otherwise, let $\boldsymbol{x} = \boldsymbol{x}_\sigma$; note that this is consistent with the specification of the extractor $\mathcal{E}$.
- Let $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and let $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda$. Compute $\hat{z}_{\boldsymbol{x}}, \hat{z}_{\boldsymbol{\pi}}^{(1)}, \ldots, \hat{z}_{\boldsymbol{\pi}}^{(k)}$ according to Eq. (7.1) using $(\boldsymbol{\Pi}, \boldsymbol{b})$, α, and $\boldsymbol{q}$. For each $i \in [k]$, let $\hat{z}_i = \alpha^{-1} \hat{z}_{\boldsymbol{x}} + \hat{z}_{\boldsymbol{\pi}}^{(i)} + \langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \rangle$ where $\boldsymbol{q}' = [q_{\ell+1}, \ldots, q_n]$.

Take any index $i \in [k]$ where $\hat{z}_i \in A$. Let $\boldsymbol{\Pi}_{i+1}^{\mathsf{T}} \in \mathbb{F}^{\ell+m}$ be the $(i+1)^{\text{st}}$ row of $\boldsymbol{\Pi}$ (corresponding to the i^{th} proof $\mathsf{ct}_{\boldsymbol{\pi}}^{(i)}$). Parse $\boldsymbol{\Pi}_{i+1}^{\mathsf{T}} = \boldsymbol{x}_\pi \| \boldsymbol{z}_\pi$ where $\boldsymbol{x}_\pi \in \mathbb{F}^\ell$ and $\boldsymbol{z}_\pi \in \mathbb{F}^m$. Let $\boldsymbol{q}_1 = [q_1, \ldots, q_\ell]$ and $\boldsymbol{q}_2 = [q_{n+1}, \ldots, q_{n+m}]$. Then, $\boldsymbol{q} = \boldsymbol{q}_1 \| \boldsymbol{q}' \| \boldsymbol{q}_2$. From Eq. (7.1),

$$\begin{aligned} \hat{z}_{\boldsymbol{x}} &= \alpha \langle \boldsymbol{x}_\sigma, \boldsymbol{q}_1 \rangle + \langle \boldsymbol{z}_\sigma, \boldsymbol{q}_2 \rangle + b_1 \\ \hat{z}_{\boldsymbol{\pi}}^{(i)} &= \alpha \langle \boldsymbol{x}_\pi, \boldsymbol{q}_1 \rangle + \langle \boldsymbol{z}_\pi, \boldsymbol{q}_2 \rangle + b_{i+1}. \end{aligned}$$

This means that

$$\begin{aligned} \hat{z}_i &= \alpha^{-1} \hat{z}_{\boldsymbol{x}} + \hat{z}_{\boldsymbol{\pi}}^{(i)} + \left\langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \right\rangle \\ &= \langle \boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi, \boldsymbol{q}_1 \rangle + \left\langle f_i \| \boldsymbol{y}_i, \boldsymbol{q}' \right\rangle + \left\langle \alpha^{-1} \boldsymbol{z}_\sigma + \boldsymbol{z}_\pi, \boldsymbol{q}_2 \right\rangle + b_1 + b_{i+1} \\ &= \left\langle (\boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi) \| f_i \| \boldsymbol{y}_i \| (\alpha^{-1} \boldsymbol{z}_\sigma + \boldsymbol{z}_\pi), \boldsymbol{q} \right\rangle + b_1 + b_{i+1}. \end{aligned}$$

As argued above, the DPP query $\boldsymbol{q}$ is sampled after the values of α, $\boldsymbol{x}_\sigma$, $\boldsymbol{x}_\pi$, $\boldsymbol{z}_\sigma$, $\boldsymbol{z}_\pi$, f_i, $\boldsymbol{y}_i$, b_1, and b_{i+1} are determined. Thus, by soundness of DPP (against affine strategies), if $\hat{z}_i \in A$, then with probability at least $1 - s$, it must be the case that $(\boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi) \| f_i \| \boldsymbol{y}_i \in \mathcal{L}$. Otherwise, the affine strategy $(\alpha^{-1} \boldsymbol{z}_\sigma + \boldsymbol{z}_\pi, b_1 + b_{i+1})$ breaks soundness of the DPP for the statement $(\boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi) \| f_i \| \boldsymbol{y}_i$. This means that

$$\boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi \in \{0,1\}^\ell \quad \text{and} \quad \boldsymbol{y}_i = f_i(\boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi). \tag{7.3}$$

Next, we show that if Eq. (7.3) holds, then $\boldsymbol{x}_\pi = \mathbf{0}^\ell$. Suppose otherwise. Since $\alpha \leftarrow \mathbb{F}$ and is sampled independently of $\boldsymbol{x}_\sigma, \boldsymbol{x}_\pi$,

$$\Pr[\boldsymbol{x}_\sigma + \alpha \boldsymbol{x}_\pi \in \{0,1\}^\ell : \alpha \leftarrow \mathbb{F} \setminus \{0\}] \le \frac{2}{|\mathbb{F}| - 1}.$$

Thus, if Eq. (7.3) holds, then with probability $1 - 2/(|\mathbb{F}| - 1)$, we have that $\boldsymbol{x} = \boldsymbol{x}_\pi \in \{0,1\}^\ell$ and $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$. Thus, we conclude that whenever $\hat{z}_i \in A$, then $f_i(\boldsymbol{x}) = \boldsymbol{y}_i$ except with probability $s + 2/(|\mathbb{F}| - 1)$. The claim now holds by a union bound over all k openings. □

Knowledge soundness now follows from Lemmas 7.18 to 7.20 and a hybrid argument. □

Succinctness. Both the commitment and the opening in Construction 7.16 consist of a single ciphertext for the linear-only encryption scheme, which has size $\mathrm{poly}(\lambda + \log |\mathbb{F}|)$. □

Remark 7.21 (Extending to NP Relations). *Construction 7.16 extends directly to proving NP relations on the committed input $\boldsymbol{x}$. Namely, instead of showing that $\boldsymbol{y} = f(\boldsymbol{x})$ for a committed input $\boldsymbol{x}$, the prover can also prove that there exists a $\boldsymbol{w}$ such that $\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{w})$. To do so, we instantiate Construction 7.16 with a* DPP *for the NP language*

$$\mathcal{L} = \{(\boldsymbol{x}, f, \boldsymbol{y}) \mid \exists \boldsymbol{w} : \boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{w})\}.$$

Laconic commit-and-prove arguments. Construction 7.16 leverages a linear-only encryption scheme to construct a succinct commit-and-prove argument where commitments and openings each consist of a single ciphertext for a linear-only encryption scheme. By adapting our laconic argument (Construction 7.6), we can achieve better succinctness using an interactive approach and working in the generic group model. In this setting, the commitment and opening consist of a single *group element*. The construction is the analog of Construction 7.16, and we sketch it below.

We use the same conventions as in Construction 7.16. Specifically, let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of Boolean functions $f \colon \{0,1\}^\ell \to \{0,1\}^t$ and that each function $f \in \mathcal{F}_\lambda$ can be described by a bit-string of length $s = s(\lambda)$. Define the language

$$\mathcal{L} = \{(\boldsymbol{x}, f, \boldsymbol{y}) \in \{0,1\}^\ell \times \{0,1\}^s \times \{0,1\}^t : \boldsymbol{y} = f(\boldsymbol{x})\}.$$

Let $n = n(\lambda) = \ell(\lambda) + s(\lambda) + t(\lambda)$ be the length of a statement $(\boldsymbol{x}, f, \boldsymbol{y})$, Let $(\mathcal{P}, \mathcal{V}) = \{(\mathcal{P}_\lambda, \mathcal{V}_\lambda)\}_{\lambda \in \mathbb{N}}$ be a DPP for $\mathcal{L}$ over a finite field $\mathbb{F}$ with proof length $m = m(\lambda)$. We work over a prime-order group $\mathsf{GroupGen}$. The construction then proceeds as follows:

- **Verifier preprocessing:** On input the security parameter λ, the verifier samples $(\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda)$ and $(\boldsymbol{q}, A) \leftarrow \mathcal{V}_\lambda(\rho)$, where ρ is the verifier randomness for the DPP. The verifier parses $\boldsymbol{q} = (q_1, \dots, q_{n+m})$ and samples $\alpha_1, \alpha_2 \leftarrow \mathbb{F}_p$. For each $i \in [\ell]$, let $h_{\boldsymbol{x},i} \leftarrow g^{\alpha_1 q_i}$. For each $i \in [m]$, let $h_{\boldsymbol{\pi},i} \leftarrow g^{\alpha_2 q_{n+i}}$. It outputs the message
$$\mathsf{crs} = ((\mathbb{G}, p, g), h_{\boldsymbol{x},1}, \dots, h_{\boldsymbol{x},\ell}, h_{\boldsymbol{\pi},1}, \dots, h_{\boldsymbol{\pi},m}).$$
- **Prover commitment:** On input the message $\mathsf{crs} = ((\mathbb{G}, p, g), h_{\boldsymbol{x},1}, \dots, h_{\boldsymbol{x},\ell}, h_{\boldsymbol{\pi},1}, \dots, h_{\boldsymbol{\pi},m})$ and an input $\boldsymbol{x} \in \{0,1\}^\ell$, the prover commits to it by computing $\sigma = \prod_{i \in [\ell]} h_{\boldsymbol{x},i}^{x_i}$.
- **Prover opening:** To construct an opening to a function f, the prover first evaluates $\boldsymbol{y} = f(\boldsymbol{x})$, and constructs a DPP proof $\boldsymbol{\pi} \leftarrow \mathcal{P}_\lambda((\boldsymbol{x}, f, \boldsymbol{y}), \bot)$. It outputs the message $\pi = \prod_{i \in [m]} h_{\boldsymbol{\pi},i}^{\pi_i}$.
- **Verifier challenge:** On input a commitment $\sigma \in \mathbb{G}$ and an opening $(f, \boldsymbol{y}, \pi)$, the verifier replies with the DPP randomness ρ.
- **Prover response:** On input ρ, the prover computes $\boldsymbol{q} = (q_1, \dots, q_{n+m}) \leftarrow \mathcal{V}_\lambda(\rho)$. Then, it computes $t_{\boldsymbol{x}} = \sum_{i \in [\ell]} q_i x_i$ and $t_{\boldsymbol{\pi}} = \sum_{i \in [m]} \pi_i q_{n+i}$
- **Verification:** On input the prover response $(t_{\boldsymbol{x}}, t_{\boldsymbol{\pi}})$ the verifier computes $z = t_{\boldsymbol{x}} + \langle f \| \boldsymbol{y}, \boldsymbol{q}' \rangle + t_{\boldsymbol{\pi}}$ and accepts if $\sigma = g^{\alpha_1 t_{\boldsymbol{x}}}$, $\pi = g^{\alpha_2 t_{\boldsymbol{\pi}}}$, and $z \in A$.

The above approach also generalizes directly to the setting where the prover constructs multiple openings of the commitment with respect to different functions $f_1, \dots, f_k$. Completeness and (knowledge) soundness follow by a similar analysis as in the proof of Theorem 7.7.

Acknowledgments

We thank Ignacio Cascudo for his help with the formulation of Theorem 3.6, Madhu Sudan for pointers on AG codes and Ohad Barta and Jens Groth for helpful discussions at early stages of this project. We thank Vinod Vaikuntanathan and Daniel Wichs for suggesting the optimization described in Remark 7.12. N. Bitansky was supported in part by the European Research Council (ERC) under the European Union's Horizon Europe research and innovation programme (grant agreement No. 101042417, acronym SPP). P. Harsha's research supported by the Department of Atomic Energy, Government of India (project no.: RTI4001) and partially supported by a Google India Research Award. Y. Ishai was supported by ERC grant NTSC (742754), BSF grant 2022370, ISF grant 2774/20, and ISF-NSFC grant 3127/23. R. Rothblum was supported by the European Union (ERC, FASTPROOF, 101041208). D. J. Wu was supported by NSF CNS-2140975, CNS-2318701, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

References

[ABF24] Gaspard Anthoine, David Balbás, and Dario Fiore. *Fully-succinct multi-key homomorphic signatures from standard assumptions*. In Leonid Reyzin and Douglas Stebila, eds., *Proc. 44th CRYPTO, Part III*, volume 14922 of *LNCS*, pages 317–351. Springer, 2024. ePrint.iacr:2024/895. 61

[ABH21] Per Austrin, Jonah Brown-Cohen, and Johan Håstad. *Optimal inapproximability with universal factor graphs*. In Dániel Marx, ed., *Proc. 32nd Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 434–453. 2021. eccc:2019/151. 9, 48

[ACY22] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. *Hardness of approximation for stochastic problems via interactive oracle proofs*. In Shachar Lovett, ed., *Proc. 37th Comput. Complexity Conf.*, volume 234 of *LIPIcs*, pages 24:1–24:16. Schloss Dagstuhl, 2022. ePrint.iacr:2022/168. 8

[ADI25] Gal Arnon, Jesko Dujmovic, and Yuval Ishai. *Designated-verifier SNARGs with one group element*. In Yael Tauman Kalai and Seny F. Kamara, eds., *Proc. 45th CRYPTO, Part VII*, volume 16006 of *LNCS*, pages 192–224. Springer, 2025. ePrint.iacr:2025/517. 7, 9, 58

[AL10] Yonatan Aumann and Yehuda Lindell. *Security against covert adversaries: Efficient protocols for realistic adversaries*. J. Cryptol., 23(2):281–343, 2010. (Preliminary version in *4th TCC*, 2007). ePrint.iacr:2007/060. 58

[ALMSS98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. *Proof verification and the hardness of approximation problems*. J. ACM, 45(3):501–555, 1998. (Preliminary version in *33rd FOCS*, 1992). eccc:1998/008. 1, 22

[App17] Benny Applebaum. *Exponentially-hard Gap-CSP and local PRG via local hardcore functions*. In Chris Umans, ed., *Proc. 58th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 836–847. 2017. eccc:2017/063. 5

[AS98] Sanjeev Arora and Shmuel Safra. *Probabilistic checking of proofs: A new characterization of NP*. J. ACM, 45(1):70–122, 1998. (Preliminary version in *33rd FOCS*, 1992). 1, 11

[BBCG+17] Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, and Madars Virza. *Computational integrity with a public random string from quasi-linear PCPs*. In Jean-Sébastien Coron and Jesper Buus Nielsen, eds., *Proc. 36th*

EUROCRYPT, Part III, volume 10212 of *LNCS*, pages 551–579. 2017. ePrint.iacr:2016/646. 1

[BBCGI19] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. *Zero-knowledge proofs on secret-shared data via fully linear PCPs*. In Alexandra Boldyreva and Daniele Micciancio, eds., *Proc. 39th CRYPTO, Part III*, volume 11694 of *LNCS*, pages 67–97. Springer, 2019. ePrint.iacr:2019/188. 2, 8, 10, 11, 15

[BBCGI21] ———. *Lightweight techniques for private heavy hitters*. In *Proc. IEEE Symp. Security and Privacy (S&P)*, pages 762–776. 2021. arXiv:2012.14884, ePrint.iacr:2021/017. 8

[BBGS14] Alp Bassa, Peter Beelen, Arnaldo Garcia, and Henning Stichtenoth. *An improvement of the Gilbert-Varshamov bound over nonprime fields*. IEEE Trans. Inform. Theory, 60(7):3859–3861, 2014. 17

[BC12] Nir Bitansky and Alessandro Chiesa. *Succinct arguments from multi-prover interactive proofs and their efficiency benefits*. In Reihaneh Safavi-Naini and Ran Canetti, eds., *Proc. 32nd CRYPTO*, volume 7417 of *LNCS*, pages 255–272. Springer, 2012. ePrint.iacr:2012/095. 61

[BCFL23] David Balbás, Dario Catalano, Dario Fiore, and Russell W. F. Lai. *Chainable functional commitments for unbounded-depth circuits*. In Guy N. Rothblum and Hoeteck Wee, eds., *Proc. 21st International Theory of Crypt. Conf. (TCC), Part III*, volume 14371 of *LNCS*, pages 363–393. Springer, 2023. 61

[BCGG$^+$14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. *Zerocash: Decentralized anonymous payments from bitcoin*. In Michael Backes, Adrian Perrig, and Helen Wang, eds., *Proc. IEEE Symp. Security and Privacy (S&P)*, pages 459–474. 2014. ePrint.iacr:2014/349. 1

[BCGT13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. *On the concrete efficiency of probabilistically-checkable proofs*. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, eds., *Proc. 45th ACM Symp. on Theory of Computing (STOC)*, pages 585–594. 2013. eccc:2012/045. 1

[BCIOP22] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. *Succinct non-interactive arguments via linear interactive proofs*. J. Cryptol., 35(3):15, 2022. (Preliminary version in *10th TCC*, 2013). ePrint.iacr:2012/718. 1, 2, 4, 6, 7, 8, 9, 13, 14, 32, 33, 35, 43, 45, 49, 50, 51, 61, 78, 79

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. *Interactive oracle proofs*. In Martin Hirt and Adam D. Smith, eds., *Proc. 14th International Theory of Crypt. Conf. (TCC), Part II*, volume 9986 of *LNCS*, pages 31–60. Springer, 2016. ePrint.iacr:2016/116. 1, 8

[BGHSV05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. *Short PCPs verifiable in polylogarithmic time*. In Luca Trevisan, ed., *Proc. 20th IEEE Conf. on Comput. Complexity*, pages 120–134. 2005. doi:10.1109/CCC.2005.27. 1

[BGHSV06] ———. *Robust PCPs of proximity, shorter PCPs and applications to coding*. SIAM J. Comput., 36(4):889–974, 2006. (Preliminary version in *36th STOC*, 2004). eccc:2004/021. 3, 11, 15, 22

[BHIRW24] Nir Bitansky, Prahladh Harsha, Yuval Ishai, Ron D. Rothblum, and David J. Wu. *Dot-product proofs and their applications*. In Santosh Vempala, ed., *Proc. 65th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 806–825. 2024. eccc:2024/114, ePrint.iacr:2024/1138. i, 10

[BIOW20] Ohad Barta, Yuval Ishai, Rafail Ostrovsky, and David J. Wu. *On succinct arguments and witness encryption from groups*. In Daniele Micciancio and Thomas Ristenpart, eds., *Proc.* 40*th CRYPTO, Part I*, volume 12170 of *LNCS*, pages 776–806. Springer, 2020. ePrint.iacr:2020/1319. 2, 6, 7, 9, 13, 14, 22, 32, 33, 35, 43, 44, 45, 49, 51, 58, 80, 81

[BISW18] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. *Quasi-optimal SNARGs via linear multi-prover interactive proofs*. In Jesper Buus Nielsen and Vincent Rijmen, eds., *Proc.* 37*th EUROCRYPT, Part III*, volume 10822 of *LNCS*, pages 222–255. Springer, 2018. ePrint.iacr:2018/133. 7

[BKKMS16] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. *Constant rate PCPs for circuit-sat with sublinear query complexity*. J. ACM, 63(4):32:1–32:57, 2016. (Preliminary version in *54th FOCS*, 2013). eccc:2013/085. 1, 8

[BS08] Eli Ben-Sasson and Madhu Sudan. *Short PCPs with polylog query complexity*. SIAM J. Comput., 38(2):551–607, 2008. (Preliminary version in *37th STOC*, 2005). eccc:2004/060. 1

[CB17] Henry Corrigan-Gibbs and Dan Boneh. *Prio: Private, robust, and scalable computation of aggregate statistics*. In Aditya Akella and Jon Howell, eds., *Proc. 14th NSDI*, pages 259–282. USENIX, 2017. arXiv:1703.06255. 8

[CFHK$^+$15] Craig Costello, Cédric Fournet, Jon Howell, Markulf Kohlweiss, Benjamin Kreuter, Michael Naehrig, Bryan Parno, and Samee Zahur. *Geppetto: Versatile verifiable computation*. In *Proc. IEEE Symp. Security and Privacy (S&P)*, pages 253–270. 2015. ePrint.iacr:2014/976. 7, 59

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. *Universally composable two-party and multi-party secure computation*. In *Proc.* 34*th ACM Symp. on Theory of Computing (STOC)*, pages 494–503. 2002. ePrint.iacr:2002/140. 7, 59

[CO99] Ran Canetti and Rafail Ostrovsky. *Secure computation with honest-looking parties: What if nobody is truly honest? (extended abstract)*. In Jeffrey Scott Vitter, Lawrence L. Larmore, and Frank Thomson Leighton, eds., *Proc.* 31*st ACM Symp. on Theory of Computing (STOC)*, pages 255–264. 1999. 58

[CP23] Leo de Castro and Chris Peikert. *Functional commitments for all functions, with transparent setup and from SIS*. In Carmit Hazay and Martijn Stam, eds., *Proc.* 42*nd EUROCRYPT, Part III*, volume 14006 of *LNCS*, pages 287–320. Springer, 2023. ePrint.iacr:2022/1368. 61

[CXY20] Ronald Cramer, Chaoping Xing, and Chen Yuan. *On the complexity of arithmetic secret sharing*. In Rafael Pass and Krzysztof Pietrzak, eds., *Proc.* 18*th International Theory of Crypt. Conf. (TCC), Part III*, volume 12552 of *LNCS*, pages 444–469. Springer, 2020. ePrint.iacr:2019/1137. 17

[DFGK14] George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. *Square span programs with applications to succinct NIZK arguments*. In Palash Sarkar and Tetsu Iwata, eds., *Proc. 20th ASIACRYPT (Part I)*, volume 8873 of *LNCS*, pages 532–550. Springer, 2014. ePrint.iacr:2014/718. 5, 11, 22, 36, 75

[Din07] Irit Dinur. *The PCP theorem by gap amplification*. J. ACM, 54(3):12, 2007. (Preliminary version in *38th STOC*, 2006). eccc:2005/046. 1

[Din16] ———. *Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover*, 2016. (manuscript). eccc:2016/128. 6, 17, 49

[DMS24] Michel Dellepere, Pratyush Mishra, and Alireza Shirzad. *Garuda and Pari: Faster and smaller SNARKs via equifficient polynomial commitments*, 2024. (manuscript). ePrint.iacr:2024/1245. 58

[DR06] Irit Dinur and Omer Reingold. *Assignment testers: Towards a combinatorial proof of the PCP Theorem*. SIAM J. Comput., 36:975–1024, 2006. (Preliminary version in *45th FOCS*, 2004). 3, 15

[ECZB21] Saba Eskandarian, Henry Corrigan-Gibbs, Matei Zaharia, and Dan Boneh. *Express: Lowering the cost of metadata-hiding communication with cryptographic privacy*. In Michael Bailey and Rachel Greenstadt, eds., *Proc. 30th USENIX Security Symposium*, pages 1775–1792. USENIX, 2021. arXiv:1911.09215. 8

[EG14] Alex Escala and Jens Groth. *Fine-tuning Groth-Sahai proofs*. In Hugo Krawczyk, ed., *Public-Key Cryptography (PKC)*, volume 8383 of *LNCS*, pages 630–649. Springer, 2014. ePrint.iacr:2013/662. 7, 59

[FGLSS96] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. *Interactive proofs and the hardness of approximating cliques*. J. ACM, 43(2):268–292, 1996. (Preliminary version in *32nd FOCS*, 1991). 1

[FJ12] Uriel Feige and Shlomo Jozeph. *Universal factor graphs*. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, eds., *Proc. 39th International Colloq. of Automata, Languages and Programming (ICALP), Part I*, volume 7391 of *LNCS*, pages 339–350. Springer, 2012. arXiv:1204.6484. 9, 48

[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, eds., *Proc. 32nd EUROCRYPT*, volume 7881 of *LNCS*, pages 626–645. Springer, 2013. ePrint.iacr:2012/215. 1, 11

[GLRSW24] Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, Yican Sun, and Kewen Wu. *Parameterized inapproximability hypothesis under exponential time hypothesis*. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, eds., *Proc. 56th ACM Symp. on Theory of Computing (STOC)*, pages 24–35. 2024. arXiv:2311.16587. 6

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008. 5

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. *Perfect non-interactive zero knowledge for NP*. In Serge Vaudenay, ed., *Proc. 25th EUROCRYPT*, volume 4004 of *LNCS*, pages 339–358. Springer, 2006. eccc:2005/097, ePrint.iacr:2005/290. 77

[Gro10] Jens Groth. *Short pairing-based non-interactive zero-knowledge arguments*. In Masayuki Abe, ed., *Proc. 16th ASIACRYPT*, volume 6477 of *LNCS*, pages 321–340. Springer, 2010. 1

[Gro16] ———. *On the size of pairing-based non-interactive arguments*. In Marc Fischlin and Jean-Sébastien Coron, eds., *Proc. 35th EUROCRYPT, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, 2016. 1, 58

[GS95] Arnaldo G. Garcia and Henning Stichtenoth. *Algebraic function fields over finite fields with many rational places*. IEEE Trans. Inform. Theory, 41(6):1548–1563, 1995. 17

[GS01] Venkatesan Guruswami and Madhu Sudan. *On representations of algebraic-geometry codes*. IEEE Trans. Inform. Theory, 47(4):1610–1613, 2001. (Preliminary version in *8th ESA*, 2000). 17

[GW11] Craig Gentry and Daniel Wichs. *Separating succinct non-interactive arguments from all falsifiable assumptions*. In Lance Fortnow and Salil P. Vadhan, eds., *Proc. 43rd ACM Symp. on Theory of Computing (STOC)*, pages 99–108. 2011. ePrint.iacr:2010/610. 61

[Hås01] Johan Håstad. *Some optimal inapproximability results*. J. ACM, 48(4):798–859, 2001. (Preliminary version in *29th STOC*, 1997). 2, 5, 9, 17, 48

[HKLT19] Prahladh Harsha, Subhash Khot, Euiwoong Lee, and Devanathan Thiruvenkatachari. *Improved hardness for 3LIN via linear label cover*. In Dimitris Achlioptas and László A. Vêgh, eds., *Proc. 22nd International Conf. on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, volume 137 of *LIPIcs*, pages 9:1–9:16. Schloss Dagstuhl, 2019. `eccc:2019/093`. 2, 6, 17

[IKNOS25] Yuval Ishai, Eyal Kushilevitz, Varun Narayanan, Rafail Ostrovsky, and Akash Shah. *Query-reusable proof systems*. In Serge Fehr and Pierre-Alain Fouque, eds., *Proc. 44th EUROCRYPT, Part IV*, volume 15604 of *LNCS*, pages 333–362. Springer, 2025. 1

[IKO07] Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. *Efficient arguments without short PCPs*. In Peter Bro Miltersen, ed., *Proc. 22nd IEEE Conf. on Comput. Complexity*, pages 278–291. 2007. 1, 22, 61

[IP01] Russell Impagliazzo and Ramamohan Paturi. *On the complexity of k-SAT*. J. Comput. Syst. Sci., 62(2):367–375, 2001. (Preliminary version in *14th CCC*, 1999). 17

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. *Which problems have strongly exponential complexity?* J. Comput. Syst. Sci., 63(4):512–530, 2001. (Preliminary version in *39th FOCS*, 1998). 17

[ISO19] ISO/IEC. *Information security – lightweight cryptography – part 2: Block ciphers*. ISO/IEC Standard 29192-2:2019, 2019. 57

[JJ22] Abhishek Jain and Zhengzhong Jin. *Indistinguishability obfuscation via mathematical proofs of equivalence*. In Jelani Nelson, ed., *Proc. 63rd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 1023–1034. 2022. `ePrint.iacr:2022/1430`. 7

[Kil92] Joe Kilian. *A note on efficient zero-knowledge proofs and arguments (extended abstract)*. In S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis, eds., *Proc. 24th ACM Symp. on Theory of Computing (STOC)*, pages 723–732. 1992. 1, 7, 59

[KPV14] Subhash Khot, Preyas Popat, and Nisheeth K. Vishnoi. *Almost polynomial factor hardness for closest vector problem with preprocessing*. SIAM J. Comput., 43(3):1184–1205, 2014. (Preliminary version in *44th STOC*, 2012). `arXiv:1109.2176`, `eccc:2011/119`. 2, 17

[KR08] Yael Tauman Kalai and Ran Raz. *Interactive PCP*. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, eds., *Proc. 35th International Colloq. of Automata, Languages and Programming (ICALP), Part II*, volume 5126 of *LNCS*, pages 536–547. Springer, 2008. `eccc:2007/031`. 1, 8

[Lip24] Helger Lipmaa. *Polymath: Groth16 is not the limit*. In Leonid Reyzin and Douglas Stebila, eds., *Proc. 44th CRYPTO, Part X*, volume 14929 of *LNCS*, pages 170–206. Springer, 2024. `ePrint.iacr:2024/916`. 58

[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, eds., *Proc. 43rd International Colloq. of Automata, Languages and Programming (ICALP)*, volume 55 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl, 2016. `ePrint.iacr:2016/766`. 61

[Mic00] Silvio Micali. *Computationally sound proofs*. SIAM J. Comput., 30(4):1253–1298, 2000. (Preliminary version in *35th FOCS*, 1994). 1

[MPV24] Surya Mathialagan, Spencer Peters, and Vinod Vaikuntanathan. *Adaptively sound zero-knowledge SNARKs for UP*. In Leonid Reyzin and Douglas Stebila, eds., *Proc. 44th CRYPTO, Part X*, volume 14929 of *LNCS*, pages 38–71. Springer, 2024. ePrint.iacr:2024/227. 7

[MR17] Pasin Manurangsi and Prasad Raghavendra. *A birthday repetition theorem and complexity of approximating dense CSPs*. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, eds., *Proc. 44th International Colloq. of Automata, Languages and Programming (ICALP)*, volume 80 of *LIPIcs*, pages 78:1–78:15. Schloss Dagstuhl, 2017. arXiv:1611.05530. 17

[Nec94] Vasilii Il'ich Nechaev. *К вопросу о сложности детерминированного алгоритма для дискретного логарифма (Russian) [Complexity of a determinate algorithm for the discrete logarithm]*. Matematicheskie Zametki, 55(2):91–101, 1994. (English translation in *Mathematical Notes*, 55(2):165–172, 1994). doi:10.1007/BF02113297. 7, 80

[NR22] Shafik Nassar and Ron D. Rothblum. *Succinct interactive oracle proofs: Applications and limitations*. In Yevgeniy Dodis and Thomas Shrimpton, eds., *Proc. 42nd CRYPTO, Part I*, volume 13507 of *LNCS*, pages 504–532. Springer, 2022. ePrint.iacr:2022/281. 8

[Pai99] Pascal Paillier. *Public-key cryptosystems based on composite degree residuosity classes*. In Jacques Stern, ed., *Proc. 18th EUROCRYPT*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999. 51

[Pet94] Erez Petrank. *The hardness of approximation: Gap location*. Comput. Complexity, 4:133–157, 1994. (Preliminary version in *2nd ISTCS*, 1993). 4

[PHGR16] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. *Pinocchio: nearly practical verifiable computation*. Commun. ACM, 59(2):103–112, 2016. (Preliminary version in *IEEE S&P*, 2013). ePrint.iacr:2013/279. 1

[Pol00] John M. Pollard. *Kangaroos, monopoly and discrete logarithms*. J. Cryptol., 13(4):437–447, 2000. 51

[Ran13] Hugues Randriambololona. *Asymptotically good binary linear codes with asymptotically good self-intersection spans*. IEEE Trans. Inform. Theory, 59(5):3038–3045, 2013. arXiv:1204.3057. 17

[Raz87] Alexander A. Razborov. *Нжние оценки размера схем ограниченной глубины в полном базисе, содержащем функцию логического сложения (Russian) [Lower bounds on the size of bounded depth circuits over a complete basis with logical addition]*. Mathematicheskie Zametki, 41(4):598–607, 1987. (English translation in *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987). doi:10.1007/BF01137685. 24

[Reg09] Oded Regev. *On lattices, learning with errors, random linear codes, and cryptography*. J. ACM, 56(6):34:1–34:40, 2009. (Preliminary version in *37th STOC*, 2005). 51

[RRR21] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. *Constant-round interactive proofs for delegating computation*. SIAM J. Comput., 50(3), 2021. (Preliminary version in *48th STOC*, 2016). eccc:2016/061. 1, 8

[RVW13] Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. *Interactive proofs of proximity: delegating computation in sublinear time*. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, eds., *Proc. 45th ACM Symp. on Theory of Computing (STOC)*, pages 793–802. 2013. 2

[Sho97] Victor Shoup. *Lower bounds for discrete logarithms and related problems*. In Walter Fumy, ed., *Proc. 16th EUROCRYPT*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997. 7, 80, 81

[Smo87] Roman Smolensky. *Algebraic methods in the theory of lower bounds for Boolean circuit complexity*. In Alfred V. Aho, ed., *Proc.* 19*th ACM Symp. on Theory of Computing (STOC)*, pages 77–82. 1987. 24

[SW21] Amit Sahai and Brent Waters. *How to use indistinguishability obfuscation: Deniable encryption, and more*. SIAM J. Comput., 50(3):857–908, 2021. (Preliminary version in *46th STOC*, 2014). ePrint.iacr:2013/454. 7

[Tha22] Justin Thaler. *Proofs, arguments, and zero-knowledge*. Found. Trends Priv. Secur., 4(2-4):117–660, 2022. doi:10.1561/3300000030. 18

[Vad12] Salil P. Vadhan. *Pseudorandomness*. Found. Trends Theor. Comput. Sci., 7(1-3):1–336, 2012. doi:10.1561/0400000010. 22

[VH97] Conny Voss and Tom Høholdt. *An explicit construction of a sequence of codes attaining the Tsfasman-Vladut-Zink bound: The first steps*. IEEE Trans. Inform. Theory, 43(1):128–135, 1997. 17

[WW23a] Hoeteck Wee and David J. Wu. *Lattice-based functional commitments: Fast verification and cryptanalysis*. In Jian Guo and Ron Steinfeld, eds., *Proc. 29th ASIACRYPT (Part V)*, volume 14442 of *LNCS*, pages 201–235. Springer, 2023. ePrint.iacr:2024/028. 61

[WW23b] ——. *Succinct vector, polynomial, and functional commitments from lattices*. In Carmit Hazay and Martijn Stam, eds., *Proc.* 42*nd EUROCRYPT, Part III*, volume 14006 of *LNCS*, pages 385–416. Springer, 2023. ePrint.iacr:2022/1515. 61

[WW24a] Brent Waters and David J. Wu. *Adaptively-sound succinct arguments for NP from indistinguishability obfuscation*. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, eds., *Proc.* 56*th ACM Symp. on Theory of Computing (STOC)*, pages 387–398. 2024. ePrint.iacr:2024/165. 7

[WW24b] Hoeteck Wee and David J. Wu. *Succinct functional commitments for circuits from k-lin*. In Marc Joye and Gregor Leander, eds., *Proc.* 43*rd EUROCRYPT, Part III*, volume 14652 of *LNCS*, pages 280–310. Springer, 2024. ePrint.iacr:2024/688. 61

[WZ24] Brent Waters and Mark Zhandry. *Adaptive security in SNARGs via iO and lossy functions*. In Leonid Reyzin and Douglas Stebila, eds., *Proc.* 44*th CRYPTO, Part X*, volume 14929 of *LNCS*, pages 72–104. Springer, 2024. ePrint.iacr:2024/254. 7

[Zim15] Joe Zimmerman. *How to obfuscate programs directly*. In Elisabeth Oswald and Marc Fischlin, eds., *Proc.* 34*th EUROCRYPT, Part II*, volume 9057 of *LNCS*, pages 439–467. Springer, 2015. ePrint.iacr:2014/776. 81

A Impossibility Results for DPPs with Perfect Completeness

In this section we show some impossibility results for DPPs with perfect completeness. The basic impossibility result says that such DPPs cannot have an accepting set of size 1. As a corollary, we get that DPPs over $\mathbb{F}_2$ (the two element field) cannot have perfect completeness.

We say that a language $\mathcal{L} \subseteq \mathbb{F}^n$ is *affine* if there exist parameters $k, m \in \mathbb{N}$, a matrix $\boldsymbol{A} \in \mathbb{F}^{k\times(n+m)}$ and vector $\boldsymbol{b} \in \mathbb{F}^k$ such that $\mathcal{L} = \left\{\boldsymbol{x} \in \mathbb{F}^n : \exists \boldsymbol{w} \in \mathbb{F}^m,\ \boldsymbol{A}\cdot(\boldsymbol{x}\|\boldsymbol{w}) = \boldsymbol{b}\right\}$.

Lemma A.1 (DPP with Accepting Set Size 1)**.** *Let $\mathbb{F}$ be a finite field. If $\mathcal{L}$ has a* DPP *over $\mathbb{F}$ with* perfect *completeness, an accepting set of size* 1 *and any soundness error $\delta < 1$, then $\mathcal{L}$ is affine.*

Proof. Since the accepting set is of size 1, all of the verifier's tests are of the form $\langle \boldsymbol{z}, \boldsymbol{x} \| \boldsymbol{\pi} \rangle = b$, with $\boldsymbol{z} \in \mathbb{F}^{n+m}$ and $b \in \mathbb{F}$. By perfect completeness, if $\boldsymbol{x} \in \mathcal{L}$, then all of these equations are satisfied, whereas if $\boldsymbol{x} \notin \mathcal{L}$, then at least one should not be satisfied. Thus, $\mathcal{L}$ is an affine language. □

In Proposition A.3 below, we give an example of a very simple language that is not affine. First however we show that in the case of $\mathbb{F}_2$, the accepting set is (without loss of generality) always of size 1 and therefore any DPP (for a non-affine language) must have imperfect completeness.

Lemma A.2 (DPP over $\mathbb{F}_2$ with Perfect Completeness is Affine)**.** *If $\mathcal{L}$ has a* DPP *over $\mathbb{F}_2$ with* perfect *completeness, and any soundness error $\delta < 1$, then $\mathcal{L}$ is affine.*

Proof. Over $\mathbb{F}_2$, the answer set A satisfies $A \subseteq \{0,1\}$. This means $A \in \{\varnothing, \{0\}, \{1\}, \{0,1\}\}$. Since the DPP has perfect completeness, and $\mathcal{L}$ is not empty, then it always holds that $A \neq \varnothing$. Similarly, any query in which $A = \{0,1\}$ can be removed without affecting completeness nor soundness. Thus, without loss of generality, the accepting set is always of size 1. The lemma now follows from Lemma A.1. □

Next, we give even an extremely simple language which is not affine: namely, the OR of two bits, and thus, cannot have a DPP with an accepting set of size 1.

Proposition A.3 (OR Language)**.** *The 2-bit OR language (i.e., $\{01, 10, 11\}$) is not affine over any field $\mathbb{F}$.*

Proof. Suppose that there exists a matrix $\boldsymbol{A}$ and a vector $\boldsymbol{b}$ such that for any $\boldsymbol{x} \in \{01, 10, 11\}$, there exists $\boldsymbol{w}_{\boldsymbol{x}}$ such that $\boldsymbol{A} \cdot (\boldsymbol{x} \| \boldsymbol{w}_{\boldsymbol{x}}) = b$. Consider the proof string $\boldsymbol{w} = \boldsymbol{w}_{01} + \boldsymbol{w}_{10} - \boldsymbol{w}_{11}$ for input $\boldsymbol{x} = 00$. Using the fact that $00 = 10 + 01 - 11$, we have that:

$$\begin{aligned}
\boldsymbol{A} \cdot (00 \| \boldsymbol{w}) &= \boldsymbol{A} \cdot \Big((01 \| \boldsymbol{w}_{01}) + (10 \| \boldsymbol{w}_{10}) - (11 \| \boldsymbol{w}_{11}) \Big) \\
&= \boldsymbol{A} \cdot (01 \| \boldsymbol{w}_{01}) + \boldsymbol{A} \cdot (10 \| \boldsymbol{w}_{10}) - \boldsymbol{A}(11 \| \boldsymbol{w}_{11}) \\
&= \boldsymbol{b} + \boldsymbol{b} - \boldsymbol{b} \\
&= \boldsymbol{b},
\end{aligned}$$

in contradiction to the fact that $\boldsymbol{x} = 00$ is not in the language. □

Finally, we show that over fields of characteristic 2, even an accepting set of size 2 does not suffice.

Proposition A.4 (Accepting Set Size of DPP for **UV** over Fields of Characteristic 2)**.** *Let $k, n \geq 2$ and $\mathbb{F} = \mathbb{F}_{2^k}$. The language* **UV** $\subseteq \mathbb{F}^n$ does not *have a* DPP *with perfect completeness, any soundness error $\delta < 1$, and accepting set A of size 2.*

Proof. Since $\mathbf{0} \in \mathbf{UV}$, we must have $0 \in A$. Suppose $A = \{0, c\}$ for some $c \in \mathbb{F} \setminus \{0\}$. Let π_1, π_2 be the DPP proofs for the unit vectors $\boldsymbol{e}_1, \boldsymbol{e}_2$, respectively. By perfect completeness, every possible query vector $\boldsymbol{q}$ must satisfy $\langle (\boldsymbol{e}_1 \| \pi_1), \boldsymbol{q} \rangle \in A$ and $\langle (\boldsymbol{e}_2 \| \pi_2), \boldsymbol{q} \rangle \in A$. Since $\mathbb{F}$ has characteristic 2, A is closed under addition. It follows that $\langle (\boldsymbol{e}_1 + \boldsymbol{e}_2 \| \pi_1 + \pi_2), \boldsymbol{q} \rangle \in A$, and so $\pi_1 + \pi_2$ is always accepted as a proof for $\boldsymbol{x} = \boldsymbol{e}_1 + \boldsymbol{e}_2 \notin \mathbf{UV}$. □

B 2-Query FLPCP for Boolean Circuits with Squaring Verification

Our abstract FLPCP construction from multiplication codes natively applies to arithmetic circuits and has 3 queries. In this section we construct a 2-query variant of this construction for Boolean circuits.

Our construction generalizes and simplifies a previous construction of a 2-query LPCP based on "square span programs" implicit in [DFGK14], and adapts it to the fully linear setting. First, in Appendix B.1 we construct a 2-query FLPCP that a given input is Boolean valued. Then, in Appendix B.2 we convert the latter to an FLPCP for Boolean circuit satisfiability.

B.1 2-Query FLPCP for Booleanity

Let **Bool** $= \{0,1\}^n$ (i.e., the "Booleanity" language). We construct a 2-query FLPCP for **Bool** over general fields.

Theorem B.1 (2-Query FLPCP for **Bool**). *Let $\mathbb{F}$ be a finite field and suppose that $E : \mathbb{F}^n \to \mathbb{F}^\ell$ is a multiplication code with respect to to the product code $E^\star : \mathbb{F}^{n^\star} \to \mathbb{F}^\ell$. Then,* Bool *has an* FLPCP *over $\mathbb{F}$ with 2 queries, proof length $n^\star$, soundness error $1 - \delta^\star + 1/|\mathbb{F}|$ and randomness complexity* $\log_2(\ell) + \log(|\mathbb{F}|)$*, where $\delta^\star$ is the minimal distance of $E^\star$. Furthermore, the verifier's accepting set is always $\{(\alpha, \alpha^2) : \alpha \in \mathbb{F}\}$.*

Furthermore, if E and E^* are systematic then the proof length is $n^* - n$.

The proof of the theorem mimics the proof of Theorems 4.5 and 4.6 but takes advantage of the specific problem to force two of the queries in these FLPCPs to be the same (thereby reducing the query complexity to 2).

Proof. Let $E \colon \mathbb{F}^n \to \mathbb{F}^\ell$ be a multiplication code with respect to the product code $E^\star \colon \mathbb{F}^{n^\star} \to \mathbb{F}^\ell$ and let $\delta^\star$ denote the distance of $E^\star$. Recall that both E and $E^\star$ are linear and systematic codes.

Let $\boldsymbol{x} \in \{0,1\}^n$. Since E is a multiplication code with respect to to $E^\star$, it holds that $E(\boldsymbol{x}) \star E(\boldsymbol{x}) \in E^\star$, and in particular there exists $\boldsymbol{w} \in \mathbb{F}^{n^\star}$ such that $E^\star(\boldsymbol{w}) = E(\boldsymbol{x}) \star E(\boldsymbol{x})$. The FLPCP proof string is simply $\boldsymbol{w}$. In case E^* is systematic, it suffice to include only the non-systematic part of $\boldsymbol{w}$.

Given linear access to the concatenation of the input $\boldsymbol{x}$ and the alleged proof string $\boldsymbol{w}$, the verifier samples a random index $i \in [\ell]$ and scalar $\lambda \in \mathbb{F}$ and checks that

$$E^\star(\boldsymbol{w})_i + \lambda \cdot E^\star(\boldsymbol{x} - \boldsymbol{w}_1 \| 0^{n^\star - n})_i = (E(\boldsymbol{x})_i)^2,$$

where $\boldsymbol{w}_1$ denotes the first n entries of $\boldsymbol{w} \in \mathbb{F}^{n^*}$. Note that each side of the equation consists of a single linear query to $(\boldsymbol{x} \| \boldsymbol{w})$.

Completeness. Suppose $\boldsymbol{x} \in \{0,1\}^n$ and let $\boldsymbol{w}$ be the proof string as described above.

By construction $E^\star(\boldsymbol{w}) = E(\boldsymbol{x}) \star E(\boldsymbol{x})$. Thus, for every $i \in [\ell]$, it holds that $E^\star(\boldsymbol{w})_i = (E(\boldsymbol{x})_i)^2$. Also, for $i \in [n]$, since E is systematic, it holds that $(\boldsymbol{w}_1)_i = (E^\star(\boldsymbol{w}))_i = ((E(\boldsymbol{x}))_i)^2 = (\boldsymbol{x}_i)^2 = \boldsymbol{x}_i$, where the last equality follows from the fact that $\boldsymbol{x}$ is Boolean valued. Thus, $\boldsymbol{w}_1 = \boldsymbol{x}$ and so:

$$E^\star(\boldsymbol{w})_i + \lambda \cdot E^\star(\boldsymbol{x} - \boldsymbol{w}_1 \| 0^{n^\star - n})_i = E^\star(\boldsymbol{w})_i = (E(\boldsymbol{x})_i)^2,$$

and so the verifier accepts.

Soundness. Let $\boldsymbol{x} \in \mathbb{F}^n$ and fix a proof string $\boldsymbol{w} \in \mathbb{F}^\ell$.

Suppose first that $\boldsymbol{w}_1 \neq \boldsymbol{x}$. In this case with all but $1-\delta^\star$ probability, it holds that $E^\star(\boldsymbol{w}_1-\boldsymbol{x})_i \neq 0$. When this is the case then $\lambda \cdot E^\star(\boldsymbol{x} - \boldsymbol{w}_1 \| 0^{n^\star - n})_i$ is uniformly distributed and the probability that the test passes is $1/|\mathbb{F}|$.

Thus, we may assume that $\boldsymbol{w}_1 = \boldsymbol{x}$. Suppose now that $E^\star(\boldsymbol{w}) \neq E(\boldsymbol{x}) \star E(\boldsymbol{x})$. Since E is a multiplication code with respect to $E^\star$, there exists $c^\star \in E^\star$ such that $c^\star = E(\boldsymbol{x}) \star E(\boldsymbol{x})$. Since $c^\star \neq E^\star(\boldsymbol{x})$, by the distance of $E^\star$, with probability at least $\delta^\star$ over $i \in [\ell]$, it holds that:

$$E^\star(\boldsymbol{w})_i \neq (c^\star)_i = \big(E(\boldsymbol{x}) \star E(\boldsymbol{x})\big)_i = (E(\boldsymbol{x})_i)^2,$$

and so the verifier's test fails with probability at least $\delta^\star$.

Thus, we may assume that $E^\star(\boldsymbol{w}) = E(\boldsymbol{x}) \star E(\boldsymbol{x})$. But this means that for every coordinate $i \in [n]$:

$$\boldsymbol{x}_i = \boldsymbol{w}_i = E^\star(\boldsymbol{w})_i = E(\boldsymbol{x})_i \cdot E(\boldsymbol{x})_i = (\boldsymbol{x}_i)^2,$$

which implies that $\boldsymbol{x} \in \{0,1\}^n$. □

Corollary B.2 (Reed-Solomon-Based FLPCP for **Bool**). *Let $\mathbb{F}$ be a finite field of size $|\mathbb{F}| > 2n+1$. Then, the language* **Bool** *has an* FLPCP *over $\mathbb{F}$ with 2 queries, proof length n, soundness error $(2n+1)/|\mathbb{F}|$ and randomness complexity $O(\log(|\mathbb{F}|))$. Furthermore, the verifier's accepting set is always* $\mathbf{Sq} = \{(\alpha, \alpha^2) : \alpha \in \mathbb{F}\}$.

B.2 From Booleanity to Boolean Circuits

The following proposition shows that testing whether a given input $\boldsymbol{x} \in \mathbb{F}^n$ belongs to $\{\boldsymbol{x} \in \{0,1\}^n : C(\boldsymbol{x}) = 1\}$ reduces to the Booleanity problem.

Given an input $x \in \{0,1\}^n$ to a Boolean circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size s, we say that $w \in \{0,1\}^s$ is a *full evaluation* if it contains the values of all gates in the evaluation of C on input x (using some predetermined fixed topological ordering of the gates).

Proposition B.3 (DPP for Boolean Circuit Satisfiability). *Let $\mathbb{F}$ be a field with characteristic greater than or equal to 5. Suppose that* **Bool** *has an* FLPCP *over $\mathbb{F}$ with query complexity $d = d(n)$, proof length $m = m(n)$, soundness error $\varepsilon = \varepsilon(n)$ and randomness complexity $r = r(n)$, for inputs of size n.*

Let $C\colon \{0,1\}^n \to \{0,1\}$ be a Boolean circuit of size s which consists of fan-in 2 NAND *gates. Then, the language*

$$\big\{(\boldsymbol{x}, \boldsymbol{w}) \in \{0,1\}^n \times \{0,1\}^{s-n} : C(\boldsymbol{x}) = 1 \text{ and } (\boldsymbol{x}\|\boldsymbol{w}) \text{ is the full evaluation for } C(\boldsymbol{x}) = 1\big\}$$

has an FLPCP *over $\mathbb{F}$ with query complexity $d(s)$, proof length $s + m(s)$, soundness error $\varepsilon(s)$, and randomness complexity $r(s)$.*

Proof. Given an input $(\boldsymbol{x}\|\boldsymbol{w})$, it suffices to check that:

1. $\boldsymbol{x} \in \{0,1\}^n$.
2. For every gate with input wires i, j and output wire k, it holds that $\boldsymbol{w}'_k = \mathsf{NAND}(\boldsymbol{w}'_i, \boldsymbol{w}'_j)$, where $\boldsymbol{w}'_i, \boldsymbol{w}'_j, \boldsymbol{w}'_k$ denote the corresponding variables in the extended witness.

We shall use the following useful fact (due to [GOS06, Lemma 1]), which can be verified by explicitly writing down the relevant truth tables.

Fact B.4 (NAND Gates)**.** *Let $a, b, c \in \{0,1\}$. Then, $c = \mathsf{NAND}(a,b)$ if and only if $a+b+2c-2 \in \{0,1\}$, where the arithmetic is over the integers.*

Fact B.4 holds over the integers, but given that the expression $a+b+2c-2$ is between -2 and 2 (for $a,b,c \in \{0,1\}$) it is also true over any field with characteristic greater than or equal to 5.

Thus, Item 2 above reduces to checking that for every gate with input wires i,j and output wire k, it holds that $w_i + w_j + 2w_k - 2 \in \{0,1\}$. Overall we get that checking that $(\boldsymbol{x}, \boldsymbol{w})$ is in the language reduces to checking that $\boldsymbol{M} \cdot (\boldsymbol{x} \| \boldsymbol{w}) \in \mathbf{Bool}$, for a suitable matrix $\boldsymbol{M} \in \mathbb{F}^{s \times s}$ arising from Fact B.4. The lemma then follows from Proposition 3.4. □

Using Proposition 3.3 we immediately derive the following corollary for Boolean circuit satisfiability.

Corollary B.5 (FLPCP for Boolean Circuit Satisfiability from FLPCP for **Bool**)**.** *Let $\mathbb{F}$ be a field with characteristic greater than or equal to 5. Suppose that* **Bool** *has an* FLPCP *over $\mathbb{F}$ with query complexity $d = d(n)$, proof length $m = m(n)$, soundness error $\varepsilon = \varepsilon(n)$ and randomness complexity $r = r(n)$, for inputs of size n.*

Let $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit of size s which consists of fan-in 2 NAND gates. Then, the language $\left\{\boldsymbol{x} \in \{0,1\}^n : \exists \boldsymbol{w} \in \{0,1\}^h : C(\boldsymbol{x}, \boldsymbol{w}) = 1\right\}$ has an FLPCP *over $\mathbb{F}$ with query complexity $d(s)$, proof length $s + m(s)$, soundness error $\varepsilon(s)$, and randomness complexity $r(s)$.*

Combining Corollary B.5 with Corollary B.2 we get that:

Corollary B.6 (FLPCP for Boolean Circuit Satisfiability)**.** *Let $\mathbb{F}$ be a field with characteristic greater than or equal to 5. Let $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit of size s which consists of fan-in 2 NAND gates. Then, the language $\left\{\boldsymbol{x} \in \{0,1\}^n : \exists \boldsymbol{w} \in \{0,1\}^h : C(\boldsymbol{x}, \boldsymbol{w}) = 1\right\}$ has an* FLPCP *over $\mathbb{F}$ with query complexity 2, proof length $2s$, soundness error $(2s+1)/|\mathbb{F}|$ and randomness complexity $O(\log(|\mathbb{F}|))$. Furthermore, the verifier's accepting set is always $\mathbf{Sq} = \{(\alpha, \alpha^2) : \alpha \in \mathbb{F}\}$.*

C Proof of Theorem 4.4 for Non-Squares

Let q be a prime power. Recall that our goal is to construct a DPP over $\mathbb{F}_q$ with soundness error $O(1/\sqrt{q})$.

Observing that q^2 is a square, the furthermore part of Theorem 3.6 gives a multiplication code $E\colon \mathbb{F}_{q^2}^k \to \mathbb{F}_{q^2}^{k/\rho_q}$ such that $\rho_q = 1/\mathrm{poly}(q)$ and the square code $E^\star$ has distance $1 - O(1/q)$. Combining this code with Theorem 4.6, we obtain an FLPCP for $\mathsf{dR1CS}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}$ over the field of order q^2 with 3 queries, proof length $k \cdot \mathrm{poly}(q)$, soundness error $1 - O(1/q)$ and randomness complexity $\log(k) + O(\log(q))$. The accepting set has cardinality q^4.

Each linear query over the field of order q^2, can be emulated by two linear queries over the field q. This follows from the fact that multiplication by a fixed element over the field $\mathbb{F}_{q^2}$ is a linear map over $\mathbb{F}_q$.

Thus, the above FLPCP can be viewed as a 6-query FLPCP over the field $\mathbb{F}_q$ with proof length $k \cdot \mathrm{poly}(q)$, soundness error $1 - O(1/q)$, randomness complexity $\log(k) + O(\log(q))$ and with an accepting set A of size q^4.

By Lemma 4.24, the language A has a DPP with proof length $\mathrm{poly}(q)$, soundness error $O(1/\sqrt{q})$ and randomness complexity $\mathrm{poly}(q)$.

We compose the 6-query FLPCP with the above DPP via Lemma 4.12. This results in a DPP with soundness error $O(1/\sqrt{q})$, proof length $k \cdot \mathrm{poly}(q) + 2^{\log(k)+O(\log(q)} \cdot \mathrm{poly}(q) = k \cdot \mathrm{poly}(q)$, and randomness complexity $\log(k) + \mathrm{poly}(q)$, as desired.

D SNARGs from Linear-Only Encryption

In this section, we recall the notion of a succinct non-interactive argument (SNARG) as well as the Bitansky, Chiesa, Ishai, Ostrovsky and Paneth [BCIOP22] compiler for constructing SNARGs in the preprocessing model from a linear interactive proof (LIP) together with a linear-only encryption scheme. For simplicity of exposition, we simply note here that any DPP is already an (input-oblivious) LIP. Correspondingly, we state the main theorem for a DPP in place of a LIP. This specialization suffices for all of the applications we consider in this work.

Definition D.1 (Succinct Non-Interactive Argument). *Let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of NP relations indexed by a security parameter $\lambda \in \mathbb{N}$ and let $\mathcal{L} = \{\mathcal{L}_\lambda\}_{\lambda\in\mathbb{N}}$ be the associated family of NP languages. A* preprocessing *succinct non-interactive argument (SNARG) for $\mathcal{R}$ is a tuple of three probabilistic polynomial-time algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ with the following properties:*

- $\mathcal{G}(1^\lambda) \to (\sigma, \tau)$*: On input the security parameter λ, the generator algorithm outputs a common reference string σ and a verification state τ.*
- $\mathcal{P}(\sigma, x, w) \to \pi$*: On input a common reference string σ, a statement x, and a witness w, the prover algorithm outputs a proof π.*
- $\mathcal{V}(\tau, x, \pi) \to \{0,1\}$*: On input the verification key τ, a statement x, and a proof π, the verification algorithm outputs a bit $b \in \{0,1\}$.*

The algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ should satisfy the following properties:

- *(Completeness:) for all security parameters $\lambda \in \mathbb{N}$ and every $(x, w) \in \mathcal{R}_\lambda$,*

$$\Pr\left[\mathcal{V}(\tau, x, \pi) = 1 : \begin{array}{c}(\sigma,\tau) \leftarrow \mathcal{G}(1^\lambda);\\ \pi \leftarrow \mathcal{P}(\sigma, x, w)\end{array}\right] \geq c.$$

  We refer to c as the completeness parameter. By default, we consider constructions with perfect completeness (i.e., $c = 1$).

- *(Soundness:) We consider two notions of soundness:*
  - *(Single-theorem adaptive soundness:) For every polynomial-size prover $\mathcal{P}^*$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all security parameters $\lambda \in \mathbb{N}$,*

$$\Pr\left[\mathcal{V}(\tau, x, \pi) = 1 \textit{ and } x \notin \mathcal{L}_\lambda : \begin{array}{c}(\sigma,\tau) \leftarrow \mathcal{G}(1^\lambda)\\ (x,\pi) \leftarrow \mathcal{P}^*(1^\lambda, \sigma)\end{array}\right] = \mathsf{negl}(\lambda).$$

  - *(Multi-theorem soundness:) For every polynomial-size prover $\mathcal{P}^*$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all security parameters $\lambda \in \mathbb{N}$,*

$$\Pr\left[\mathcal{V}(\tau, x, \pi) = 1 \textit{ and } x \notin \mathcal{L}_\lambda : \begin{array}{c}(\sigma,\tau) \leftarrow \mathcal{G}(1^\lambda)\\ (x,\pi) \leftarrow (\mathcal{P}^*)^{\mathcal{V}(\tau,\cdot,\cdot)}(1^\lambda, \sigma)\end{array}\right] = \mathsf{negl}(\lambda).$$

- *(Succinctness:) There exists a universal polynomial* $\mathrm{poly}(\cdot)$ *such that the running time of* $\mathcal{G}$ *is* $\mathrm{poly}(\lambda + |\mathcal{R}_\lambda|)$*, the running time of* $\mathcal{P}$ *is* $\mathrm{poly}(\lambda + |\mathcal{R}_\lambda|)$*, the running time of* $\mathcal{V}$ *is* $\mathrm{poly}(\lambda + |x| + \log |\mathcal{R}_\lambda|)$*, and the size of the proof is* $\mathrm{poly}(\lambda + \log s)$*.*

Remark D.2 (Public Verification vs. Designated Verifier)**.** *We say a SNARG is* publicly verifiable *if the verification state* τ *is simply the common reference string* σ *(i.e.,* $\tau = \sigma$*). Otherwise, we say the SNARG is* designated-verifier *(and knowledge of the verification state* τ *is needed to check proofs).*

Definition D.3 (Linear-Only Encryption [BCIOP22])**.** *Let* $\mathbb{F} = \{\mathbb{F}_\lambda\}_{\lambda \in \mathbb{N}}$ *be a field ensemble. A linear-only encryption scheme with plaintext space* $\mathbb{F}$ *is a tuple of efficient algorithms* $(\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add}, \mathsf{ImVer})$ *with the following properties:*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter* λ*, the key-generation algorithm outputs a secret key* sk *and a public key* pk*.*
- $\mathsf{Encrypt}(\mathsf{pk}, x) \to \mathsf{ct}$*: On input the public key* pk *and a message* $x \in \mathbb{F}$*, the encryption algorithm outputs a ciphertext* ct*.*
- $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}) \to x$*: On input the secret key* sk *and a ciphertext* ct*, the decryption algorithm outputs a message* $x \in \mathbb{F}$ *(or a special symbol* $\perp$ *to denote a decryption failure).*
- $\mathsf{Add}(\mathsf{pk}, \mathsf{ct}_1, \mathsf{ct}_2) \to \mathsf{ct}'$*: On input the public key* pk*, ciphertexts* $\mathsf{ct}_1, \mathsf{ct}_2$*, and scalars* $\alpha_1, \alpha_2 \in \mathbb{F}$*, the addition algorithm outputs an evaluated ciphertext* ct'*.*
- $\mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}) \to b$*: On input the secret key* sk *and a ciphertext* ct*, the image-verification algorithm outputs a bit* $b \in \{0,1\}$*.*

In addition, the above algorithms should satisfy the following properties:

- *(Correctness:) For every* $\lambda \in \mathbb{N}$ *and every message* $x \in \mathbb{F}_\lambda$*,*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}) = x \wedge \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}) = 1 : \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ \mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x) \end{array}\right] = 1.$$

  In addition, for all $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda)$*, and all ciphertexts* $\mathsf{ct}_1, \mathsf{ct}_2$ *where* $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_1) = x_1$ *and* $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_2) = x_2$ *for some* $x_1, x_2 \in \mathbb{F}_\lambda$*, we have that*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}') = x_1 + x_2 \wedge \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}') = 1 : \begin{array}{r} \mathsf{ct}' \leftarrow \mathsf{Add}(\mathsf{pk}, \mathsf{ct}_1, \mathsf{ct}_2); \\ x_1 \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_1); \\ x_2 \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}_2) \end{array}\right] = 1.$$

- *(Semantic security:) For an adversary* $\mathcal{A}$ *and a bit* $b \in \{0,1\}$*, we define the semantic security experiment as follows:*
  - *The challenger begins by sampling* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and gives* pk *to* $\mathcal{A}$*.*
  - *The adversary outputs two vectors* $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathbb{F}^\ell$*.*
  - *For each* $i \in \ell$*, the challenger computes* $\mathsf{ct}_i \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x_{b,i})$*. It gives* $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ *to* $\mathcal{A}$*.*

- *At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is also the output of the experiment.*

The encryption scheme is semantically secure if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$|\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| = \mathsf{negl}(\lambda).$$

- *(Linear-only:) For all polynomial-size adversaries $\mathcal{A}$, there exists a polynomial-size extractor $\mathcal{E}$ such that for all plaintext generators $\mathcal{M}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all security parameters λ,*

$$\Pr\left[\begin{array}{c} \exists i \in [k] : \mathsf{ImVer}(\mathsf{sk}, \mathsf{ct}'_i) = 1 \\ and \\ \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}'_i) \neq y_i \end{array} : \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda); \\ \boldsymbol{x} = (x_1, \ldots, x_m) \leftarrow \mathcal{M}(\mathsf{pk}); \\ \forall i \in [m] : \mathsf{ct}_i \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x_i); \\ (\mathsf{ct}'_1, \ldots, \mathsf{ct}'_k) \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{ct}_1, \ldots, \mathsf{ct}_m; r_\mathcal{A}); \\ (\boldsymbol{\Pi}, \boldsymbol{b}) \leftarrow \mathcal{E}(\mathsf{pk}, \mathsf{ct}_1, \ldots, \mathsf{ct}_m; r_\mathcal{A}); \\ \boldsymbol{y} = (y_1, \ldots, y_k) \leftarrow \boldsymbol{\Pi x} + \boldsymbol{b} \end{array}\right] = \mathsf{negl}(\lambda),$$

where $r_\mathcal{A}$ denotes the (uniform) randomness used by adversary $\mathcal{A}$.

E The Generic Group Model

As described in Section 7, we analyze the security of our interactive laconic argument in the generic group model [Nec94, Sho97]. In the generic group model, access to group elements are replaced by handles. The generic group oracle is responsible for maintaining a consistent mapping between handles and the group elements they represent. We follow the exact same syntax as used in [BIOW20, Appendix C]:

Definition E.1 (Generic Group Oracle)**.** *A generic group oracle is a stateful oracle $\mathcal{G}$ that responds to queries* $\mathsf{GGM.Setup}$, $\mathsf{GGM.Encode}$, $\mathsf{GGM.Add}$, $\mathsf{GGM.Test}$ *as follows:*

- *On a query $\mathsf{GGM.Setup}(1^\lambda)$, the generic group oracle samples two fresh nonces $\mathsf{pp}, \mathsf{sk} \leftarrow \{0,1\}^\lambda$ and a prime $p < 2^\lambda$. It outputs $(\mathsf{pp}, \mathsf{sk}, p)$. The oracle stores the values generated, initializes an empty table $\mathsf{T} \leftarrow \{\}$, and sets the internal state so subsequent invocations of* $\mathsf{GGM.Setup}$ *fail (with output $\perp$).*

- *On a query $\mathsf{GGM.Encode}(k, x)$ where $k \in \{0,1\}^\lambda$, $x \in \mathbb{F}_p$, the oracle checks that $k = \mathsf{sk}$ (returning $\perp$ if the check fails). The oracle then generates a fresh nonce $\xi \leftarrow \{0,1\}^\lambda$ and adds the entry $\xi \mapsto x$ to the table T, and replies with ξ.*

- *On a query $\mathsf{GGM.Add}(k, \xi_1, \xi_2)$ where $k, \xi_1, \xi_2 \in \{0,1\}^\lambda$, the oracle checks that $k = \mathsf{pp}$, that the handles ξ_1, ξ_2 are present in its internal table T, and are mapped to values $x_1, x_2 \in \mathbb{F}_p$, respectively (returning $\perp$ otherwise). If the checks pass, the oracle samples a fresh handle $\xi \leftarrow \{0,1\}^\lambda$ and adds the entry $\xi \mapsto (x_1 + x_2)$ to T, and replies with ξ. The addition oracle can be used to implement* scalar multiplication *by arbitrary $\mathbb{F}_p$ elements via repeated doubling.*

- *On a query* GGM.Test(k, ξ) *where* $k, x \in \{0,1\}^\lambda$, *the oracle checks that* $k = \mathsf{pp}$, *that the handle* ξ *is present in* T, *and that* ξ *maps to some value* $x \in \mathbb{F}_p$ *(returning* $\perp$ *otherwise). If the checks pass, the oracle returns "zero" if* $x = 0 \in \mathbb{F}_p$ *and "non-zero" otherwise.*

Remark E.2 (Unique Encodings)**.** *Many formulations, including [Sho97], model the generic group using a random injective function* $\sigma\colon \mathbb{F}_p \to \{0,1\}^\lambda$. *In this formulation, every value in* $\mathbb{F}_p$ *has a unique encoding, and there is no need for an explicit* GGM.Test *procedure (*GGM.Test *would just correspond to equality of bitstrings). Our formulation in Definition E.1 (taken verbatim from [BIOW20, Definition C.1]) samples a new* encoding *on every query and provides an explicit* GGM.Test *procedure for checking whether an element is an encoding of* 0 *(or equivalently, whether two elements are equal). We can implement a generic group model with unique encodings by using the* GGM.Test *procedure to test equality against the existing entries in the table* T *after each* GGM.Encode *and* GGM.Add *query, and returning the previously-computed handle if it is already present in the table. Otherwise, a new handle is sampled as usual. This transformation incurs a quadratic overhead in the number of queries. Thus, without loss of generality, we can assume fresh handles are output by* GGM.Encode *and* GGM.Add*, and equality-checking is handled through an explicit algorithm* GGM.Test*.*

Remark E.3 (Oracle Queries as Formal Polynomials [Zim15, Remark 2.11, adapted])**.** *Although the generic group oracle is defined formally in terms of "handles" (Definition E.1), it is oftentimes more conducive to regard each oracle query as referring to a formal query polynomial. The formal variables in this formal query polynomial are specified by the expressions supplied to the* GGM.Encode *oracle (as determined by the details of the construction), and the adversary can construct terms that refer to new polynomials by making queries to the group operation oracle* GGM.Add*. Rather than operating on a "handle," each valid* GGM.Test *query refers to a formal query polynomial, and the result of the query is "zero" if the polynomial evaluates to zero when its variables are instantiated with the joint distribution over their values in* $\mathbb{F}_p$ *as generated in the real security game.*