cq:* Cached quotients for fast lookups

Liam Eagen Dario Fiore Ariel Gabizon
IMDEA Zeta Function Technologies

December 23, 2022

Abstract

We present a protocol for checking the values of a committed polynomial $f(X) \in \mathbb{F}_{<N}[X]$ over a multiplicative subgroup $\mathbb{H} \subset \mathbb{F}$ of size n are contained in a table $\mathsf{t} \in \mathbb{F}^N$. After an $O(N \log N)$ preprocessing step, the prover algorithm runs in time $O(n \log n)$. Thus, we continue to improve upon the recent breakthrough sequence of results[ZBK+22, PK22, ?, ?] starting from Caulk [ZBK+22], which achieve sublinear complexity in the table size N. The two most recent works in this seqence [?, ?] achieved prover complexity $O(n \cdot \log^2 n)$.

Moreover, cq has the following attractive features.

1. As in [ZBK+22, PK22, ?] our construction relies on homomorphic table commitments, which makes them amenable to vector lookups in the manner described in Section 4 of [GW20].
2. As opposed to [ZBK+22, PK22, ?, ?] the cq verifier doesn't involve pairings with prover defined $\mathbb{G}_2$ points, which makes recursive aggregation of proofs more convenient.
3. The construction can be altered to a version we call cq* that loses the mentioned aggregatability, increase preprocessing time to $O(n \cdot N)$, and in return reduce prover complexity to a *linear* number of field and group operations!

1 Introduction

The *lookup problem* is fundamental to the efficiency of modern zk-SNARKs. Somewhat informally, it asks for a protocol to prove the values of a committed polynomial $\phi(X) \in \mathbb{F}_{<n}[X]$ are contained in a table T of size N of predefined legal values. When the table T corresponds to an operation without an efficient low-degree arithmetization in $\mathbb{F}$, such a protocol produces significant savings in proof construction time for programs containing the operation. Building on previous work of [BCG+18], plookup [GW20] was the first to explicitly describe a solution to this problem in the polynomial-IOP context.

*Pronounced "seek you".

plookup described a protocol with prover complexity quasilinear in both n and N. This left the intriguing question of whether the dependence on N could be made *sublinear* after performing a preprocessing step for the table T. Caulk [ZBK+22] answered this question in the affirmative by leveraging bi-linear pairings, achieving a run time of $O(n^2 + n \log N)$. Caulk+ [PK22] improved this to $O(n^2)$ getting rid of the dependence on table size completely.

However, the quadratic dependence on n of these works makes them impractical for a circuit with many lookup gates. We resolve this issue by giving a protocol called cq that is quasi-linear in n and has no dependence on N after the preprocessing step.

1.1 Comparison of results

Table 1: Scheme comparison. n = witness size, N = Table size, "Aggregatable"= All prover defined pairing arguments in $\mathbb{G}_1$

Scheme	Preprocessing	Proof size	Prover Work	Verifier Work	Homomorphic?	Aggregatable?
Caulk [ZBK+22]	O($N \log N$) $\mathbb{F}$,$\mathbb{G}_1$	14 $\mathbb{G}_1$, 1 $\mathbb{G}_2$, 4 $\mathbb{F}$	$3n + m - \ell$ $\mathbb{G}_1$ exp, n $\mathbb{G}_2$ exp	2 $\mathbb{G}_1$, 1 $\mathbb{G}_2$	✓	✗
Caulk+ [PK22]	$O(N \log N)$ $\mathbb{F}$,$\mathbb{G}_1$	7 $\mathbb{G}_1$, 1 $\mathbb{G}_2$, 2 $\mathbb{F}$	$18n$ $\mathbb{G}_1$ exp	4 $\mathbb{G}_1$, 2 $\mathbb{F}$	✓	✗
flookup [?]	$O(N \log^2 N)$ $\mathbb{F}$,$\mathbb{G}_1$	6 $\mathbb{G}_1$, 1 $\mathbb{G}_2$, 4 $\mathbb{F}$	$273n$ $\mathbb{G}_1$ exp	20 $\mathbb{G}_1$, 16 $\mathbb{F}$	✗	✗
ba*loo* [?]	$O(N \log N)$ $\mathbb{F}$,$\mathbb{G}_1$	12 $\mathbb{G}_1$, 1 $\mathbb{G}_2$, 4 $\mathbb{F}$	$8n$ $\mathbb{G}_1$ exp	6 $\mathbb{G}_1$, 4 $\mathbb{F}$	✓	✗
cq	$O(N \log N)$ $\mathbb{F}$,$\mathbb{G}_1$	8 $\mathbb{G}_1$, 4 $\mathbb{F}$	$11n + 11a$ $\mathbb{G}_1$ exp , $\approx 54(n+a)\log(n+a)$ $\mathbb{F}$ mul	7 $\mathbb{G}_1$, 6 $\mathbb{F}$	✓	✓
cq*	$O(N \log N + N \cdot n \log n)$ $\mathbb{F}$,$\mathbb{G}_1$ $O(n \log n)$ $\mathbb{G}_2$	6 $\mathbb{G}_1$, 1 $\mathbb{G}_2$, 1 $\mathbb{F}$	$9n + 9a$ $\mathbb{G}_1$ exp , $\approx 54(n+a)\log(n+a)$ $\mathbb{F}$ mul	9 $\mathbb{G}_1$, 6 $\mathbb{F}$	✓	✗

Table with relative proof size, prover ops, verifier ops proof-size caulk caulk+ flookup baloo 12 $\mathbb{G}_1$, 1 $\mathbb{G}_2$, 4 $\mathbb{F}$ this work 6 G1, 1 G2

1.2 Technical Overview

The innovation of Caulk While [ZBK+22, PK22, ?, ?] use preprocessing and pairings to extract a subtable of witness size;

Our approach here we use preprocessing and pairings more directly to run an existing lookup protocol - mvlookup, in time independent from table size -logarithmic derivative method Let's review this protocol: It relies on the following lemma from [?] that says that $f|_{\mathbb{H}} \in \mathfrak{t}$ if and only if for some $m \in \mathbb{F}^N$

$$\sum_{i \in [N]} \frac{m_i}{X + t_i} = \sum_{i \in [n]} \frac{1}{X + f_i}$$

Roughly, the protocol of [?] checks this identity on a random β, by sending polynomials A and B that agree on $\mathbb{V}$ with the rational function values of the LHS and RHS respectively. Given commitments to A, B we can check the equality holds via various sumcheck techniques, e.g. that described in [BCR+19]. The RHS is not a problem because it is a sum of size n. Interpolating A, and computing its commitment is actually not a problem either, because the number of non-zero values is at most n. So if we precompute the commitments to the Lagrange base of $\mathbb{V}$ we're fine.

The main challenge, and innovation, is to convince the verifier $\mathbf{V}$ that A is correctly formed.

This protocol is amenable, because polynomials involved have sparsity depending on witness - For large table problem is computing A that agrees with $m/(\mathsf{t}+\beta)$ on $\mathbb{V}$

- Need way to compute A

2 Preliminaries

2.1 Terminology and Conventions

We assume our field $\mathbb{F}$ is of prime order. We denote by $\mathbb{F}_{<d}[X]$ the set of univariate polynomials over $\mathbb{F}$ of degree smaller than d. We assume all algorithms described receive as an implicit parameter the security parameter λ.

Whenever we use the term *efficient*, we mean an algorithm running in time $\mathsf{poly}(\lambda)$. Furthermore, we assume an *object generator* $\mathcal{O}$ that is run with input λ before all protocols, and returns all fields and groups used. Specifically, in our protocol $\mathcal{O}(\lambda) = (\mathbb{F}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2, g_t)$ where

- $\mathbb{F}$ is a prime field of super-polynomial size $r = \lambda^{\omega(1)}$.
- $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$ are all groups of size r, and e is an efficiently computable non-degenerate pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$.
- g_1, g_2 are uniformly chosen generators such that $e(g_1, g_2) = g_t$.

We usually let the λ parameter be implicit, i.e. write $\mathbb{F}$ instead of $\mathbb{F}(\lambda)$. We write $\mathbb{G}_1$ and $\mathbb{G}_2$ additively. We use the notations $[x]_1 := x \cdot g_1$ and $[x]_2 := x \cdot g_2$.

We often denote by $[n]$ the integers $\{1, \ldots, n\}$. We use the acronym e.w.p for "except with probability"; i.e. e.w.p γ means with probability *at least* $1 - \gamma$.

universal SRS-based public-coin protocols We describe public-coin (meaning the verifier messages are uniformly chosen) interactive protocols between a prover and verifier; when deriving results for non-interactive protocols, we implicitly assume we can get a proof length equal to the total communication of the prover, using the Fiat-Shamir transform/a random oracle. Using this reduction between interactive and non-interactive protocols, we can refer to the "proof length" of an interactive protocol.

We allow our protocols to have access to a structured reference string (SRS) that can be derived in deterministic $\mathsf{poly}(\lambda)$-time from an "SRS of monomials" of the form $\left\{[x^i]_1\right\}_{a \le i \le b}$, $\left\{[x^i]_2\right\}_{c \le i \le d}$, for uniform $x \in \mathbb{F}$, and some integers a, b, c, d with absolute value bounded by $\mathsf{poly}(\lambda)$. It then follows from Bowe et al. [BGM17] that the required SRS can be derived in a universal and updatable setup requiring only one honest participant; in the sense that an adversary controlling all but one of the participants in the setup does not gain more than a $\mathsf{negl}(\lambda)$ advantage in its probability of producing a proof of any statement.

For notational simplicity, we sometimes use the SRS srs as an implicit parameter in protocols, and do not explicitly write it.

Aurora lemma

Lemma 2.1. *Let $H \subset \mathbb{F}$ be a multiplicative subgroup of size t. For $f \in \mathbb{F}_{<t}[X]$, we have*

$$\sum_{a \in H} f(a) = n \cdot a(0)$$

2.2 Idealized verifier checks for algebraic adversaries

We introduce some terminology from [GWC19] to capture analysis in the Algebraic Group Model of Fuchsbauer, Kiltz and Loss[FKL18].

First we say our srs *has degree* Q if all elements of $\mathsf{srs_i}$ are of the form $[f(x)]_i$ for $f \in \mathbb{F}_{<Q}[X]$ and uniform $x \in \mathbb{F}$. In the following discussion let us assume we are executing a protocol with a degree Q SRS, and denote by $f_{i,j}$ the corresponding polynomial for the j'th element of $\mathsf{srs_i}$.

Denote by a, b the vectors of $\mathbb{F}$-elements whose encodings in $\mathbb{G}_1, \mathbb{G}_2$ an algebraic adversary $\mathcal{A}$ outputs during a protocol execution; e.g., the j'th $\mathbb{G}_1$ element output by $\mathcal{A}$ is $[a_j]_1$.

By a "real pairing check" we mean a check of the form

$$(a \cdot T_1) \cdot (T_2 \cdot b) = 0$$

for some matrices T_1, T_2 over $\mathbb{F}$. Note that such a check can indeed be done efficiently given the encoded elements and the pairing function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$.

Given such a "real pairing check", and the adversary $\mathcal{A}$ and protocol execution during which the elements were output, define the corresponding "ideal check" as follows. Since $\mathcal{A}$ is algebraic when he outputs $[a_j]_i$ he also outputs a vector v such that, from linearity, $a_j = \sum v_\ell f_{i,\ell}(x) = R_{i,j}(x)$ for $R_{i,j}(X) := \sum v_\ell f_{i,\ell}(X)$. Denote, for $i \in \{1,2\}$ the vector of polynomials $R_i = (R_{i,j})_j$. The corresponding ideal check, checks as a polynomial identity whether

$$(R_1 \cdot T_1) \cdot (T_2 \cdot R_2) \equiv 0$$

The following lemma is inspired by [FKL18]'s analysis of [Gro16], and tells us that for soundness analysis against algebraic adversaries it suffices to look at ideal checks. Before stating the lemma we define the Q-DLOG assumption similarly to [FKL18].

Definition 2.2. *Fix integer Q. The Q-*DLOG assumption for $(\mathbb{G}_1, \mathbb{G}_2)$ *states that given*

$$[1]_1, [x]_1, \ldots, \left[x^Q\right]_1, [1]_2, [x]_2, \ldots, \left[x^Q\right]_2$$

for uniformly chosen $x \in \mathbb{F}$, the probability of an efficient $\mathcal{A}$ outputting x is $\mathsf{negl}(\lambda)$.

Lemma 2.3. *Assume the Q-DLOG for $(\mathbb{G}_1, \mathbb{G}_2)$. Given an algebraic adversary $\mathcal{A}$ participating in a protocol with a degree Q SRS, the probability of any real pairing check passing is larger by at most an additive $\mathsf{negl}(\lambda)$ factor than the probability the corresponding ideal check holds.*

AGM - real and ideal pairing checks, agm - real and ideal pairing KZG

2.3 log derivative method

Lemma from mvlookup

Lemma 2.4. *Given $f \in \mathbb{F}^n$, and $t \in \mathbb{F}^N$, we have $f \subset t$ as sets if and only if for some $m \in \mathbb{F}^N$ the following identity of rational functions holds*

$$\sum_{i\in[n]} \frac{1}{X + f_i} = \sum_{i\in[N]} \frac{m_i}{X + t_i}.$$

3 Cached quotients

Notation: In this section and the next we use the following conventions. $\mathbb{V} \subset \mathbb{F}$ denotes a mutliplicative subgroup of order N which is a power of two. We denote by $\mathbf{g}$ a generator of $\mathbb{V}$. Hence, $\mathbb{V} = \left\{\mathbf{g}, \mathbf{g}^2, \ldots, \mathbf{g}^N = 1\right\}$. Given $P \in \mathbb{F}[X]$ and integer $i \in [N]$, we denote $P_i := P(\mathbf{g}^i)$. For $i \in [N]$, we denote by $L_i \in \mathbb{F}_{<N}[X]$ the i'th Lagrange polynomial of $\mathbb{V}$. Thus, $(L_i)_i = 1$ and $(L_i)_j = 0$ for $i \neq j \in [N]$.

For a polynomial $A(X) \in \mathbb{F}_{<N}[X]$, we say it is *$n$-sparse* if $A_i \neq 0$ for at most n values $i \in [N]$. The *sparse representation* of such A consists of the (at most) n pairs (i, A_i) such that $A_i \neq 0$. We denote $\mathrm{supp}(A) := \{i \in [N] | A_i \neq 0\}$.

The main result of this section is a method to compute a commitment to a quotient polynomial - derived from a product with a preprocessed polynomial; in a number of operations depending only on the sparsity of the other polynomial in the product.

The result crucially relies on the following lemma derived from a result of Feist and Khovratovich[FK].

Lemma 3.1. *Fix $T \in \mathbb{F}_{<N}[X]$, and a subgroup $\mathbb{V} \subset \mathbb{F}$ of size N. There is an algorithm that given the $\mathbb{G}_1$ elements $\left\{\left[x^i\right]_1\right\}_{i\in\{0,\ldots,N\}}$ computes for $i \in [N]$, the elements $q_i := [Q_i(x)]_1$ where $Q_i(X) \in \mathbb{F}[X]$ is such that*

$$L_i(X) \cdot T(X) = T_i \cdot L_i(X) + Z_{\mathbb{V}}(X) \cdot Q_i(X)$$

in $O(N \cdot \log N)$ $\mathbb{G}_1$ operations.

Proof. Recall the definition of the Lagrange polynomial

$$L_i(X) = \frac{Z_{\mathbb{V}}(X)}{Z'_{\mathbb{V}}(\mathbf{g}^i)(X - \mathbf{g}^i)}.$$

Substituting this definition, we can write the quotient $Q_i(X)$ as

$$Q_i(X) = \frac{T(X) - T_i}{Z'_{\mathbb{V}}(\mathbf{g}^i)(X - \mathbf{g}^i)} = Z'_{\mathbb{V}}(\omega^i)^{-1} K_i(X),$$

for $K_i(X) := \frac{T(X)-T_i}{X-\mathbf{g}^i}$. Note that the values $\{[K_i(X)]_1\}_{i\in[N]}$ are exactly the KZG opening proofs of $T(X)$ at the elements of $\mathbb{V}$. Thus, the algorithm of Feist and Khovratovich [FK, Tom] can be used to compute commitments to all the proofs $[K_i(X)]_1$ in $O(N \log N)$ $\mathbb{G}_1$-operations. This works by writing the vector of $[K_i(X)]_1$ as a the product of a matrix with the vector of $\left[X^i\right]_1$. This matrix is a DFT matrix times a Toeplitz matrix, both of which have algorithms for evaluating matrix vector products in $O(N \log N)$ operations. Thus, all the KZG proofs can be computed in $O(N \log N)$ field operations and operations in $\mathbb{G}_1$.

Finally, the algorithm just needs to scale each $[K_i(X)]_1$ by $Z'_{\mathbb{V}}(\omega^i)$ to compute $[Q_i(X)]_1$. Conveniently, these values admit a very simple description when $Z_{\mathbb{V}}(X) = X^N - 1$ is a group of roots of unity.

$$Z'_{\mathbb{V}}(X)^{-1} = (NX^{N-1})^{-1} \equiv X/N \bmod Z_{\mathbb{V}}(X)$$

In total, the prover computes the coefficients of $T(X)$ in $O(N \log N)$ field operations, computes the KZG proofs for $T(\omega^i) = t_i$ in $O(N \log N)$ group operations, and then scales these proofs by ω^i/n in $O(N)$ group operations. In total, this takes $O(N \log N)$ field and group operations in $\mathbb{G}_1$. $\square$

We're now ready to state the main theorem of this section.

Theorem 3.2. *Fix integer parameters $0 \le n \le N$ such that n, N are powers of two. Fix $T \in \mathbb{F}_{<N}[X]$, and a subgroup $\mathbb{V} \subset \mathbb{F}$ of size N. Let $\mathsf{srs} = \left\{\left[x^i\right]_1\right\}_{i\in[0,\ldots,N]}$ for some $x \in \mathbb{F}$. There is an algorithm $\mathscr{A}$ that after a preprocessing step of $O(N \log N)$ $\mathbb{F}$- and $\mathbb{G}_1$-operations starting with srs does the following.*

Given input $A(X) \in \mathbb{F}_{<N}[X]$ that is n-sparse and given in sparse representation, $\mathscr{A}$ computes in $O(n)$ $\mathbb{F}$-operations and n $\mathbb{G}_1$-operations the element $\mathsf{cm} = [Q(x)]_1$ where $Q \in \mathbb{F}_{<N}[X]$ is such that

$$A(X) \cdot T(X) = Q(X) \cdot Z_{\mathbb{V}}(X) + R(X),$$

for $R(X) \in \mathbb{F}_{<N}[X]$.

Proof. The preprocessing step consists of computing the quotient commitments $[Q_i(X)]_1$ in $O(N \log N)$ operations, as described in Lemma 3.1. As stated in the lemma, for each $i \in [N]$ we have

$$L_i(X) \cdot T(X) = T_i \cdot L_i(X) + Z_{\mathbb{V}}(X) \cdot Q_i(X).$$

By assumption, the polynomial $A(X)$ can be written as a linear combination of at most n summands in the Lagrange basis of $\mathbb{V}$.

$$A(X) = \sum_{i\in\mathrm{supp}(A)} A_i \cdot L_i(X)$$

Substituting this into the product with $T(X)$, and substituting each of the products $L_i(X)T(X)$ with the appropriate cached quotient $Q_i(X)$ we find

$$A(X)T(X) = \sum_{i\in\mathrm{supp}(A)} A_i \cdot L_i(X)T(X) = \sum_{i\in\mathrm{supp}(A)} A_i \cdot T_i L_i(X) + A_i \cdot Z_{\mathbb{V}}(X)Q_i(X)$$

$$= \sum_{i\in\mathrm{supp}(A)} A_i \cdot T_i L_i(X) + Z_{\mathbb{V}}(X) \cdot \sum_{i\in\mathrm{supp}(A)} A_i \cdot Q_i(X).$$

Observing that the terms of the first sum are all of degree smaller than N, we get that

$$Q(X) = \sum_{i\in\mathrm{supp}(A)} A_i \cdot Q_i(X)$$
$$R(X) = \sum_{i\in\mathrm{supp}(A)} A_i T_i \cdot L_i(X)$$

Hence, commitments to both the quotient $Q(X)$ and remainder $R(X)$ can be computed in at most n group operations as

$$[Q(X)]_1 = \sum_{i\in\mathrm{supp}(A)} A_i \cdot [Q_i(X)]_1$$
$$[R(X)]_1 = \sum_{i\in\mathrm{supp}(A)} A_i T_i \cdot [L_i(X)]_1$$

$\square$

4 $\mathfrak{cq}$ - our main protocol

Definition 4.1. $\mathsf{gen}(\mathsf{t}, N, \mathsf{srs}_0 = ([x^i]_1, [x^i]_2)_{i\in[0,\ldots,d]})$:

1. *Compute for* $i \in [N]$:
   (a) $q_i = [Q_i(x)]_1$ *such that*
   $$L_i(X) \cdot T(X) = T_i \cdot L_i(X) + Z_{\mathbb{V}}(X) \cdot Q_i(X).$$
   (b) $[L_i(x)]_1, [x^{d-N} L_i(x)]_1, [x^{d-n}]_1$
   (c) $\left[\frac{L_i(x) - L_i(0)}{x}\right]_1$.

Tuple gen, $\mathsf{IsInTable}_{\mathbb{H}}$

- $\mathsf{gen}(\mathsf{t}, N, \mathsf{srs}_0) \to \mathsf{srs}$
- $\mathsf{IsInTable}_{\mathbb{H}}$ *an interactive public coin protocol between* $\mathbf{P}$ *and* $\mathbf{V}$ *where* $\mathbf{P}$ *has input* $f \in \mathbb{F}_{<n}[X]$, $\mathbf{V}$ *has* $[f(x)]_1$. *Both have* t *and* srs. *such that*

– *Completeness:If* $f|_{\mathbb{H}} \subset \mathsf{t}$ *then* $\mathbf{V}$ *outputs* acc *with probability one.*
– *Knowledge soundness in the algebraic group model: For any* $\mathsf{t} \in \mathbb{F}^n$*, the probability of any algebraic* $\mathcal{A}$ *to win the following game is* $\mathsf{negl}(\lambda)$
  1. *We generate for uniform* $x \in \mathbb{F}$, $\mathsf{srs0} = (\left[x^i\right]_1, \left[x^i\right]_2)_{i\in[0,\ldots,d}$.
  2. $\mathcal{A}$ *chooses* $\mathsf{t} \in \mathbb{F}^N$.
  3. *We compute* $\mathsf{srs} = \mathsf{gen}(\mathsf{t}, N, \mathsf{srs0})$.
  4. $\mathcal{A}$ *sends a message* cm *and values* $f_1, \ldots, f_n$ *such that* $\mathsf{cm} = \sum_{i\in[n]} f_i \cdot [L_i(x)]_1$.
  5. $\mathcal{A}$ *and* $\mathbf{V}$ *engage in the protocol* $\mathsf{IsInTable}_{\mathbb{H}}(\mathsf{t}, \mathsf{cm})$ *with* $\mathcal{A}$ *taking the role of* $\mathbf{P}$.
  6. $\mathcal{A}$ *wins if*
     * $\mathbf{V}$ *outputs* acc, *and*
     * $f|_{\mathbb{H}} \not\subset \mathsf{t}$.

Main protocol: Preprocessed inputs: $[Z_{\mathbb{V}}(x)]_2$, $[T(x)]_2$ Input (cm, f).

Round 1: Committing to the multiplicites vector

1. $\mathbf{P}$ computes poly $m \in \mathbb{F}_{<N}[X]$ such that m_i = number of times t_i appears in $f|_{\mathbb{H}}$
2. $\mathbf{P}$ sends $\mathsf{m} := [m(x)]_1$.

Round 2: Interpolating the rational identity at a random β; checking correctness of A's values + degree checks for A, B using pairings

1. $\mathbf{V}$ chooses and sends random $\alpha, \beta \in \mathbb{F}$.
2. $\mathbf{P}$ computes $A \in \mathbb{F}_{<N}[X]$ such that for $i \in [N]$, $A_i = m_i/(\mathsf{t}_i + \beta)$.
3. $\mathbf{P}$ sends $\mathsf{a} := [A(x)]_1$.
4. $\mathbf{P}$ computes $\mathsf{q_a} := [Q_A(x)]_2$ where $Q_A \in \mathbb{F}_{<N}[X]$ is such that
$$A(X)(T(X) + \beta) - m(X) = Q_A(X) \cdot Z_{\mathbb{V}}(X)$$
5. $\mathbf{P}$ computes $B \in \mathbb{F}_{<n}[X]$ such that for $i \in [n]$, $B_i = 1/(f_i + \beta)$.
6. $\mathbf{P}$ sends $\mathsf{q_b} := [B(x)]_1$.
7. $\mathbf{P}$ computes $Q_B(X)$ such that
$$B(X)(f(x) + \beta) - 1 = Q_B(X) \cdot Z_{\mathbb{H}}(X)$$

8. **P** computes and sends the value $a_0 := A(0)$.
9. **V** sets $b_0 := (N \cdot a_0)/n$.
10. **P** computes and sends $\mathsf{p} = [P(x)]_1$ where

$$P(X) := A(X) \cdot X^{d-N} + \alpha \cdot B(X) \cdot X^{d-n}.$$

11. **V** checks that A encodes the correct values:

$$e(\mathsf{a}, [T(x)]_2 + [\beta]_2) = e(\mathsf{q_a}, [Z_{\mathbb{V}}(x)]_2) \cdot e(\mathsf{m}, [1]_2)$$

12. **V** checks that A, B have the appropriate degrees:

$$e\left(\mathsf{a}, \left[x^{d-N}\right]_2\right) \cdot e\left(\alpha \cdot \mathsf{b}, \left[x^{d-n}\right]_2\right) = e(\mathsf{p}, [1]_2).$$

Round 3: Checking correctness of B at random $\gamma \in \mathbb{F}$

1. **V** sends random $\gamma, \eta, \zeta \in \mathbb{F}$.
2. **P** sends $b_\gamma := B(\gamma), f_\gamma := f(\gamma)$.
3. As part of checking the correctness of B, **V** computes $Z_{\mathbb{H}}(\gamma) = \gamma^n - 1$ and

$$Q_{b,\gamma} := \frac{b_\gamma \cdot (f_\gamma + \beta) - 1}{Z_{\mathbb{H}}(\gamma)}.$$

4. To perform a batched KZG check for the correctness of the values $a_\gamma, b_\gamma, f_\gamma$
   (a) **V** sends random $\eta \in \mathbb{F}$. **P** and **V** separately compute

$$v := b_\gamma + \eta \cdot f_\gamma + \eta^2 \cdot Q_{b,\gamma}.$$

   (b) **P** computes $\pi_\gamma := [h(x)]_1$ for

$$h(X) := \frac{B(X) + \eta \cdot f(X) + \eta^2 \cdot Q_B(X) - v}{X - \gamma}$$

   (c) **V** computes

$$\mathsf{c} := \mathsf{b} + \eta \cdot \mathsf{f} + \eta^2 \cdot \mathsf{q_b}$$

   and checks that

$$e(\mathsf{c} - [v]_1 + \gamma \cdot \pi_\gamma, [1]_2) = e(\pi_\gamma, [x]_2)$$

5. To perform a batched KZG check for the correctness of the values a_0, b_0
   (a) **P** and **V** separately compute

$$u := a_0 + \zeta \cdot b_0.$$

(b) **P** computes and sends $\pi_0 := [h_0(x)]_1$ for

$$h_0(X) := \frac{A(X) + \zeta \cdot B(X)}{X}$$

(c) **V** computes

$$\mathsf{c}_0 := \mathsf{a} + \zeta \mathsf{b}$$

and checks that

$$e(\mathsf{c}_0 - [u]_1, [1]_2) = e(\pi_0, [x]_2)$$

Stats: verifier pairings:5 - pair a with random combination of T and $\left[x^{d-N}\right]_2$, pair $\mathsf{q_a}$ with $Z_{\mathbb{V}}$. pair b with $[d-n]_2$ for degree check. Proof size - 8 $\mathbb{G}_1$- a,b,p,m,$\mathsf{q_a}$,$\mathsf{q_b}$ π_γ,π_0 4 $\mathbb{F}$- b_γ,$Q_{b,\gamma}$,f_γ,a_0

Note that we can split p to two proofs and that reduces a verifier pairing

Lemma 4.2. *The element q_A in Step 4 can be computed in $n \log n$ $\mathbb{G}_2$-operations and $O(n \log n)$ $\mathbb{F}$-operations*

Lemma 4.3. *The elements π_0, π_γ can be computed in $2 \cdot n \log n$ $\mathbb{G}_1$-operations and $O(n \log n)$ $\mathbb{F}$-operations*

Knowledge soundness proof: Let $\mathcal{A}$ be an efficient algebraic adversary participating in the Knowledge Soundness game from Definition 4.1. We show its probability of winning the game is $\mathsf{negl}(\lambda)$. Let $f \in \mathbb{F}_{<d}[X]$ be the polynomial sent by $\mathcal{A}$ in the first step of the game such that $\mathsf{cm} = [f(x)]_1$. As $\mathcal{A}$ is algebraic, when sending the commitments m,a,b,p,$\mathsf{q_a}$,$\mathsf{q_b}$,π_γ,π_0 during protocol execution it also sends polynomials $m(X), A(X), B(X), P(X), Q_A(X), Q_B(X), h(X), h_0(X) \in \mathbb{F}_{<d}[X]$ such that the former are their corresponding commitments. Let E be the event **V** outputs acc. Note that the event that $\mathcal{A}$ wins the game is contained in E. E implies all pairing checks passed. Let $A \subset E$ be the event that one of the corresponding ideal pairing checks didn't pass. According to Lemma 2.3, $\Pr(A = \mathsf{negl}(\lambda))$. Given A didn't occur, we have

- From step 11
$$A(X)(T(X) + \beta) - M(X) = Q_A(X) \cdot Z_{\mathbb{V}}(X)$$
Which means that for all $i \in [N]$,
$$A_i = \frac{M_i}{T_i + \beta)}$$

- From step 12
$$X^{d-N} A(X) + \alpha \cdot X^{d-n} B(X) = P(X),$$
which implies e.w.p. $1/|\mathbb{F}|$ over $\alpha \in \mathbb{F}$, that $\deg(A) < N$ and $\deg(B) < n$.

- From the checks of steps 4c and 2c, e.w.p. $n/|\mathbb{F}|$ over $\eta, \zeta \in \mathbb{F}$ (see e.g. Section 3 of [GWC19] for an expalantion of the correctness of batched KZG [KZG10]). $b_\gamma = B(\gamma), Q_{b,\gamma} = Q_B(\gamma), f_\gamma = f(\gamma), a_0 = A(0), b_0 = B(0)$.
- Which implies by how $Q_{b,\gamma}$ is set in step 3 that e.w.p. $(2n)/|\mathbb{F}|$ over γ
$$B(X) \cdot (f(X) + \beta) = 1 + Q_B(X) Z_{\mathbb{H}}(X),$$
which implies for all $i \in [n]$ that $B(\omega^i) = \frac{1}{f(\omega^i)+\beta}$.
- We know have using Lemma 2.1 that
$$N \cdot a_0 = \sum_{i \in [N]} A_i = \sum_{i \in [N]} \frac{m_i}{T_i + \beta)}$$
$$n \cdot b_0 = \sum_{i \in [n]} B(\omega^i) = \sum_{i \in [n]} \frac{1}{f(\omega^i) + \beta}$$
Thus e.w.p. $(n \cdot N)/|\mathbb{F}|$ over $\beta \in \mathbb{F}$, we have that
$$\sum_{i \in [N]} \frac{m_i}{T_i + X)} = \sum_{i \in [n]} \frac{1}{f(\omega^i) + X},$$
which implies $f|_{\mathbb{H}} \in \mathsf{t}$.

In summary, we have shown the event that $\mathbf{V}$ outputs acc while $f|_{\mathbb{H}} \not\subset \mathsf{t}$ is contained in a constant number of events with probability $\mathsf{negl}(\lambda)$; and so $\mathfrak{cq}$ satisfies the knowledge soundness property.

5 $\mathfrak{cq}^*$

The only point where $\mathbf{P}$ does $O(n \log n)$ operations in $\mathfrak{cq}$ is using FFT's to compute $Q_B(X)$. Similarly to what we did with $Q_A(X)$ we could try to only compute the commitment $[Q_B(x)]_1$ $\mathsf{gen}(\mathsf{t}, \mathsf{srs})$

Main protocol: Preprocessed inputs: $[Z_{\mathbb{V}}(x)]_2$, $[T(x)]_2$ Input (cm, f).

Round 1: Committing to the multiplicites vector, and f in $\mathbb{G}_2$

1. $\mathbf{P}$ computes poly $m \in \mathbb{F}_{<N}[X]$ such that m_i = number of times t_i appears in $f|_{\mathbb{H}}$
2. $\mathbf{P}$ sends $\mathsf{m} := [m(x)]_1$.
3. $\mathbf{P}$ sends $\mathsf{f}_2 := [f(x)]_2$
4. $\mathbf{V}$ checks correctness of f_2 via
$$e(\mathsf{f}, [1]_2) = e([1]_1, \mathsf{f}_2).$$

Round 2: Interpolating the rational identity at a random β; checking correctness of degrees and values of A, B using pairings

1. **V** chooses and sends random $\alpha, \beta \in \mathbb{F}$.
2. **P** computes $A \in \mathbb{F}_{<N}[X]$ such that for $i \in [N]$, $A_i = m_i/(\mathsf{t}_i + \beta)$.
3. **P** sends $\mathsf{a} := [A(x)]_1$.
4. **P** computes $\mathsf{q_a} := [Q_A(x)]_2$ where $Q_A \in \mathbb{F}_{<N}[X]$ is such that
$$A(X)(T(X) + \beta) - m(X) = Q_A(X) \cdot Z_{\mathbb{V}}(X)$$
5. **P** computes $B \in \mathbb{F}_{<n}[X]$ such that for $i \in [n]$, $B_i = 1/(f_i + \beta)$.
6. **P** sends $\mathsf{q_b} := [B(x)]_1$.
7. **P** computes $\mathsf{q_b} := [Q_B(X)]_1$ such that
$$B(X)(f(x) + \beta) - 1 = Q_B(X) \cdot Z_{\mathbb{H}}(X)$$
8. **P** computes and sends the value $a_0 := A(0)$.
9. **V** sets $b_0 := (N \cdot a_0)/n$.
10. **P** computes and sends $\mathsf{p} = [P(x)]_1$ where
$$P(X) := A(X) \cdot X^{d-N} + \alpha \cdot B(X) \cdot X^{d-n}.$$
11. **V** checks that A encodes the correct values:
$$e(\mathsf{a}, [T(x)]_2 + [\beta]_2) = e(\mathsf{q_a}, [Z_{\mathbb{V}}(x)]_2) \cdot e(\mathsf{m}, [1]_2)$$
12. **V** checks that B encodes the correct values:
$$e(\mathsf{b}, [\mathsf{f}]_2 + [\beta]_2) = e(\mathsf{q_b}, [Z_{\mathbb{H}}(x)]_2) \cdot e([1]_1, [1]_2)$$
13. **V** checks that A, B have the appropriate degrees:
$$e\left(\mathsf{a}, \left[x^{d-N}\right]_2\right) \cdot e\left(\alpha \cdot \mathsf{b}, \left[x^{d-n}\right]_2\right) = e(\mathsf{p}, [1]_2).$$

Round 3: Checking correctness of $A(0), B(0)$

1. $\mathbf{V}$ sends random $\zeta \in \mathbb{F}$.
2. To perform a batched KZG check for the correctness of the values a_0, b_0
   (a) $\mathbf{P}$ and $\mathbf{V}$ separately compute
   $$u := a_0 + \zeta \cdot b_0.$$
   (b) $\mathbf{P}$ computes and sends $\pi_0 := [h_0(x)]_1$ for
   $$h_0(X) := \frac{A(X) + \zeta \cdot B(X)}{X}$$
   (c) $\mathbf{V}$ computes
   $$\mathsf{c}_0 := \mathsf{a} + \zeta\mathsf{b}$$
   and checks that
   $$e(\mathsf{c}_0 - [u]_1, [1]_2) = e(\pi_0, [x]_2)$$

Stats: verifier pairings:5 - pair a with random combination of T and $\left[x^{d-N}\right]_2$, pair q_a with $Z_{\mathbb{V}}$. pair b with $[d-n]_2$ for degree check. Proof size - 6 $\mathbb{G}_1$- a,b,p,m,q_a,q_b,π_0 1 $\mathbb{G}_2$- f_2 1 $\mathbb{F}$- a_0

References

[BCG+18] J. Bootle, A. Cerulli, J. Groth, S. K. Jakobsen, and M. Maller. Arya: Nearly linear-time zero-knowledge proofs for correct program execution. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*, volume 11272 of *Lecture Notes in Computer Science*, pages 595–626. Springer, 2018.

[BCR+19] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward. Aurora: Transparent succinct arguments for R1CS. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, pages 103–128, 2019.

[BGM17] S. Bowe, A. Gabizon, and I. Miers. Scalable multi-party computation for zk-snark parameters in the random beacon model. Cryptology ePrint Archive, Report 2017/1050, 2017. `https://eprint.iacr.org/2017/1050`.

[FK] D. Feist and D. Khovratovich. Fast amortized kate proofs.

[FKL18] G. Fuchsbauer, E. Kiltz, and J. Loss. The algebraic group model and its applications. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, pages 33–62, 2018.

[Gro16] J. Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 305–326, 2016.

[GW20] A. Gabizon and Z. J. Williamson. plookup: A simplified polynomial protocol for lookup tables. *IACR Cryptol. ePrint Arch.*, page 315, 2020.

[GWC19] A. Gabizon, Z. J. Williamson, and O. Ciobotaru. PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptology ePrint Archive*, 2019:953, 2019.

[KZG10] A. Kate, G. M. Zaverucha, and I. Goldberg. Constant-size commitments to polynomials and their applications. pages 177–194, 2010.

[PK22] J. Posen and A. A. Kattis. Caulk+: Table-independent lookup arguments. 2022.

[Tom] A. Tomescu. Feist-khovratovich technique for computing kzg proofs fast.

[ZBK+22] A. Zapico, V. Buterin, D. Khovratovich, M. Maller, A. Nitulescu, and M. Simkin. Caulk: Lookup arguments in sublinear time. *IACR Cryptol. ePrint Arch.*, page 621, 2022.