cq:* Cached quotients for fast lookups

Liam Eagen   Dario Fiore   Ariel Gabizon
IMDEA   Zeta Function Technologies

December 19, 2022

Abstract

We present a protocol for checking the values of a committed polynomial $\phi(X) \in \mathbb{F}_{<N}[X]$ over a multiplicative subgroup $\mathbb{H} \subset \mathbb{F}$ of size n are contained in a table $T \in \mathbb{F}^N$. 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, 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].

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. **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.

*Pronounced "seek you".

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 $\mathfrak{cq}$ that is quasi-linear in n and has no dependence on N after the preprocessing step.

1.1 Comparison of results

Table with relative proof size, prover ops, verifier ops
caulk caulk+ flookup baloo this work

1.2 Overview

-logarithmic derivative method
- For large table problem is computing A that agrees with $M/(t+\beta)$ on $\mathbb{V}$
- Need way to compute A

2 Preliminaries

2.1 Notation:

$\mathbb{H}$- small space $\mathbb{V}$- big space Lagrange bases for big and small space
AGM - real and ideal pairing checks, agm - real and ideal pairing KZG

2.2 log derivative method

Lemma from mvlookup

Lemma 2.1. *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

Theorem 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 after a preprocessing step of $O(N \cdot \log N)$ operations. Given input $f \in \mathbb{F}_{<n}[X]$ computes in $O(n \cdot \log n)$ $\mathbb{G}_2$ operations $\mathsf{cm} = [Q(x)]_2$ where $Q \in \mathbb{F}_{<N}[X]$ is such that*

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

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

Lemma 3.2. *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 $\left\{[x^i]_1\right\}_{i\in\{0,\ldots,N\}}$ 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.

Lemma 3.3. *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 $q_i := \left[{}^{d-N} \cdot Q_i(x)\right]_1$ 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.

4 Main protocol

Definition 4.1. *$\mathcal{R}$ is all pairs (cm, f) such that cm is a commitment to f and $f|_{\mathbb{H}} \subset T$.*
..bla problem is relation is defined only after srs is chosen

4.1 Definitions

Ad-hoc dfn of ks protocol for table lookup
Relations dependent on srs. Tuple gen,$\mathsf{IsInTable}_{\mathbb{H}}$

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

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

1. $\mathbf{P}$ computes poly $m \in \mathbb{F}_{<N}[X]$ such that $m_i =$ number of times $\mathfrak{t}_i$ appears in $f|_{\mathbb{H}}$
2. $\mathbf{P}$ sends $[m(x)]_1$.
3. $\mathbf{V}$ chooses and sends random $\beta \in \mathbb{F}$.
4. $\mathbf{P}$ computes $A \in \mathbb{F}_{<N}[X]$ such that for $i \in [N]$, $A_i = m_i/(\mathfrak{t}_i + \beta)$.

5. **P** sends $\mathsf{a} := [A(x)]_1$.
6. **P** computes $\mathsf{q_a} := [Q_A(x)]_2$ where $Q_A \in \mathbb{F}_{<N}[X]$ is such that

$$A(X)(T(X) + \beta) - m(X) = Q_A(X) \cdot Z_{\mathbb{V}}(X)$$

7. **P** computes $B \in \mathbb{F}_{<n}[X]$ such that for $i \in [n]$, $B_i = 1/(f_i + \beta)$.
8. **P** sends $\mathsf{q_b} := [B(x)]_1$.
9. **P** computes $Q_B(X)$ such that

$$B(X)(f(x) + \beta) - 1 = Q_B(X) \cdot Z_{\mathbb{H}}(X)$$

10. **P** computes and sends the value $a_0 := A(0)$.
11. **V** sets $b_0 := (N \cdot a_0)/n$.
12. **V** sends random $\alpha \in \mathbb{F}$.
13. **P** computes and sends $\mathsf{p} = [P(x)]_1$ where

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

14. **V** sends random $\gamma \in \mathbb{F}$.
15. **P** sends $a_\gamma := A(\gamma), b_\gamma := B(\gamma), Q_{b,\gamma} := Q_B(\gamma), f_\gamma := f(\gamma)$.
16. **P** sends KZG proofs to all these poly openings.
17. **V** checks that

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

18. As part of checking the correctness of $\mathsf{q_b}$, **V** computes $Z_{\mathbb{H}}(\gamma) = \gamma^n - 1$ and computes

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

19. As part of checking P is correct, **V** computes

$$P_\gamma := a_\gamma \cdot \gamma^{d-N} + \alpha b_\gamma \cdot \gamma^{d-n}$$

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

$$v := a_\gamma + \eta \cdot b_\gamma + \eta^2 \cdot f_\gamma + \eta^3 \cdot Q_{b,\gamma} + \eta^4 \cdot P_\gamma$$

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

$$h(X) := \frac{A(X) + \eta \cdot B(X) + \eta^2 \cdot f(X) + \eta^3 \cdot Q_B(X) + \eta^4 \cdot P(X) - v}{X - \gamma}$$

(c) **V** computes

$$\mathsf{c} := \mathsf{a} + \eta \cdot \mathsf{b} + \eta^2 \cdot \mathsf{f} + \eta^3 \cdot \mathsf{q_b} + \eta^4 \cdot \mathsf{p}$$

and checks that

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

21. To perform a batched KZG check for the correctness of the values a_0, b_0

(a) **V** sends random $\lambda \in \mathbb{F}$. **P** and **V** separately compute

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

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

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

(c) **V** computes

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

and checks that

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

Lemma 4.2. *The element q_A in Step 6 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: Look at the following events

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.

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

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

[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.