Missing Protocol PH23-PCS (Part 2)

This article provides the complete optimized protocol for PH23-KZG10.

1. Protocol Framework and Optimization

First, let's review the simple process of the Evaluation Argument in the PH23+KZG10 protocol, and then we'll look at areas for optimization.

P: Send commitment C_c of $c(X)$ V: Send random number α to aggregate constraint equations for multiple polynomials P: Calculate the set of public polynomials $\{s_i(X)\}$ P: Calculate the aggregated constraint polynomial $h(X)$

$$h(X) = G(c(X), s_0(X), s_1(X), \ldots, s_{n-1}(X), z(X), z(\omega^{-1}X), X) \tag{1}$$

P: Calculate commitment C_t of quotient polynomial $t(X)$, commitment C_z of $z(X)$

$$t(X) = \frac{h(X)}{v_H(X)} \tag{2}$$

V: Send random evaluation point ζ

P: Calculate $c(\zeta \cdot \omega), c(\zeta \cdot \omega^2), c(\zeta \cdot \omega^4), \ldots, c(\zeta \cdot \omega^{2^{n-1}})$, $c(\zeta)$, and $z(\zeta)$, $z(\omega^{-1} \cdot \zeta)$, $t(\zeta)$, $a(\zeta)$; Send KZG10 Evaluation Arguments for the above polynomial evaluations

V: Verify all KZG10 Evaluation Arguments, then verify the following equation:

$$h(\zeta) \stackrel{?}{=} t(\zeta) \cdot v_H(\zeta) \tag{3}$$

Optimization of Multi-point Evaluation Proof for $c^*(X)$

In the proof, the Prover needs to prove the Evaluation of polynomial $c(X)$ at $n+1$ points, namely

$$c(\omega \cdot \zeta), c(\omega^2 \cdot \zeta), c(\omega^4 \cdot \zeta), \ldots, c(\omega^{2^{n-1}} \cdot \zeta), c(\zeta) \tag{4}$$

Using the technique from the [BDFG20] paper, if a polynomial $f(X)$ has Evaluation $\vec{v} = (v_0, v_1, \ldots, v_{m-1})$ at m points $D = (z_0, z_1, \ldots, z_{m-1})$, define $f^*(X)$ as the interpolation polynomial of $\vec{v}$ on D, i.e., $\deg(f^*(X)) = m - 1$, and $f^*(z_i) = f(z_i), \forall i \in [0, m)$

$$v_D(X) = \prod_{i=0}^{m-1}(X - z_i) \tag{5}$$

Then $f(X)$ satisfies the following equation:

$$f(X) - f^*(X) = q(X) \cdot (X - z_0)(X - z_1) \cdots (X - z_{m-1}) \tag{6}$$

This equation is easy to verify because when $X = z_i$, the left side of the equation equals zero, so $f(X) - f^*(X)$ can be divided by $(X - z_i)$. For all $i = 0, 1, \ldots, m-1$, $f(X) - f^*(X)$ can be divided by $v_D(X)$,

$$v_D(X) = \prod_{i=0}^{m-1}(X - z_i) \tag{7}$$

In this way, the Prover only needs to prove to the Verifier that there exists $q(X)$ such that $f(X) - f^*(X) = q(X) \cdot v_D(X)$, then the Evaluation of $f(X)$ on D equals $\vec{v}$. This equation can be verified by the Verifier providing a random challenge point $X = \xi$, where $v_D(\xi)$ and $f^*(\xi)$ can be calculated by the Verifier, and $f(\xi)$ and $q(\xi)$ can be proven through KZG10's Evaluation Argument.

Optimization of $c^*(X)$ Polynomial Calculation

The Prover can construct polynomial $c^*(X)$, which is the interpolation polynomial of the following vector on ζD. The advantage of doing this is to allow the Prover to prove the Evaluation of $c(X)$ at multiple different points at once, denoted as $\vec{c^*}$:

$$c(\omega \cdot \zeta), c(\omega^2 \cdot \zeta), c(\omega^4 \cdot \zeta), \ldots, c(\omega^{2^{n-1}} \cdot \zeta), c(\zeta) \tag{8}$$

We introduce D satisfying $|D| = n + 1$, defined as

$$D = \left(\omega,\ \omega^2,\ \omega^4,\ \ldots,\ \omega^{2^{n-1}}, \omega^{2^n} = 1\right) \tag{9}$$

Then the Evaluation Domain of $c^*(X)$ can be expressed as ζD,

$$D' = D\zeta = \left(\omega \cdot \zeta,\ \omega^2 \cdot \zeta,\ \omega^4 \cdot \zeta,\ \ldots,\ \omega^{2^{n-1}} \cdot \zeta,\ \zeta\right) \tag{10}$$

Its Vanishing polynomial $v_{D'}(X)$ is defined as follows:

$$v_{D'}(X) = (X - \omega\zeta)(X - \omega^2\zeta)(X - \omega^4\zeta)\cdots(X - \omega^{2^n}\zeta) \tag{11}$$

The Lagrange polynomial on D' can be defined as follows:

$$L_j^{D'}(X) = \hat{d}_j \cdot \frac{v_{D'}(X)}{X - \omega^{2^j}\zeta}, \qquad j = 0, 1, \ldots, n \tag{12}$$

where $\hat{d}_j$ are the Bary-Centric Weights on D', defined as

$$\hat{d}_j = \prod_{l \neq j} \frac{1}{\zeta \cdot \omega^{2^j} - \zeta \cdot \omega^{2^l}} = \frac{1}{\zeta^n} \cdot \prod_{l \neq j} \frac{1}{\omega^{2^j} - \omega^{2^l}} = \frac{1}{\zeta^n} \cdot \hat{w}_j \tag{13}$$

Here $\hat{w}_j$ are the Bary-Centric Weights on D, and their definition is only related to D, independent of ζ. Therefore, we can precompute $\hat{w}_j$ and then use $\hat{w}_j$ to calculate $c^*(X)$:

$$c^*(X) = c_0^* \cdot L_0^{D'}(X) + c_1^* \cdot L_1^{D'}(X) + \cdots + c_n^* \cdot L_n^{D'}(X) \tag{14}$$

The above equation can be further optimized by dividing the right side by a constant polynomial $g(X) = 1$

$$g(X) = 1 \cdot L_0^{D'}(X) + 1 \cdot L_1^{D'}(X) + \cdots + 1 \cdot L_n^{D'}(X) \tag{15}$$

We can get:

$$c^*(X) = \frac{c^*(X)}{g(X)} = \frac{c_0^* \cdot L_0^{D'}(X) + c_1^* \cdot L_1^{D'}(X) + \cdots + c_n^* \cdot L_n^{D'}(X)}{g(X)} \tag{16}$$

Expanding $g(X)$ and $L_i^{D'}(X)$, we can get:

$$c^*(X) = \frac{c_0^* \cdot \hat{d}_0 \cdot \frac{z_{D'}(X)}{X-\omega\zeta} + c_1^* \cdot \hat{d}_1 \cdot \frac{z_{D'}(X)}{X-\omega^2\zeta} + \cdots + c_n^* \cdot \hat{d}_n \cdot \frac{z_{D'}(X)}{X-\omega^{2^n}\zeta}}{1 \cdot \hat{d}_0 \cdot \frac{z_{D'}(X)}{X-\omega\zeta} + 1 \cdot \hat{d}_1 \cdot \frac{z_{D'}(X)}{X-\omega^2\zeta} + \cdots + 1 \cdot \hat{d}_n \cdot \frac{z_{D'}(X)}{X-\omega^{2^n}\zeta}} \tag{17}$$

Canceling out $z_{D'}(X)$ in both numerator and denominator, we can get

$$c^*(X) = \frac{c_0^* \cdot \hat{d}_0 \cdot \frac{1}{X-\omega\zeta} + c_1^* \cdot \hat{d}_1 \cdot \frac{1}{X-\omega^2\zeta} + \cdots + c_n^* \cdot \hat{d}_n \cdot \frac{1}{X-\omega^{2^n}\zeta}}{1 \cdot \hat{d}_0 \cdot \frac{1}{X-\omega\zeta} + 1 \cdot \hat{d}_1 \cdot \frac{1}{X-\omega^2\zeta} + \cdots + 1 \cdot \hat{d}_n \cdot \frac{1}{X-\omega^{2^n}\zeta}} \tag{18}$$

Expanding the definition of $\hat{d}_i$ and canceling out $\frac{1}{\zeta^n}$ in both numerator and denominator, we can get

$$c^*(X) = \frac{c_0^* \cdot \frac{\hat{w}_0}{X-\omega\zeta} + c_1^* \cdot \frac{\hat{w}_1}{X-\omega^2\zeta} + \cdots + c_n^* \cdot \frac{\hat{w}_n}{X-\omega^{2^n}\zeta}}{\frac{\hat{w}_0}{X-\omega\zeta} + \frac{\hat{w}_1}{X-\omega^2\zeta} + \cdots + \frac{\hat{w}_n}{X-\omega^{2^n}\zeta}} \tag{19}$$

The Prover can use the precomputed Bary-Centric Weights $\{\hat{w}_i\}$ on D to quickly calculate $c^*(X)$, if n is fixed. Nevertheless, the computational complexity of $c^*(X)$ is still $O(n\log^2(n))$. However, considering that $n = \log(N)$, the computational complexity of $c^*(X)$ is logarithmic.

$$c^*(X) = \sum_{j=0}^{n-1} \frac{\hat{w}_j}{\zeta^n} \cdot \frac{z_{D_\zeta}(X)}{X - \zeta \cdot \omega^{2^j}} \tag{20}$$

The definition of the precomputed $\hat{w}_j$ is

$$\hat{w}_j = \prod_{l \neq j} \frac{1}{\omega^{2^j} - \omega^{2^l}} \tag{21}$$

Moreover, the Verifier needs to calculate the value of $c^*(X)$ at a certain challenge point, such as $X = \xi$. The Verifier can use the above equation to calculate $c^*(\xi)$ based on $\vec{c}^*$ provided by the Prover with a time complexity of $O(\log N)$.

2. PH23+KZG10 Protocol (Optimized Version)

For the KZG10 protocol, because its Commitment has additive homomorphism.

Precomputation

1. Precompute $s_0(X), \ldots, s_{n-1}(X)$ and $v_H(X)$

$$v_H(X) = X^N - 1 \tag{22}$$

$$s_i(X) = \frac{v_H(X)}{v_{H_i}(X)} = \frac{X^N - 1}{X^{2^i} - 1} \tag{23}$$

2. Precompute Bary-Centric Weights $\{\hat{w}_i\}$ on $D = (1, \omega, \omega^2, \ldots, \omega^{2^{n-1}})$. This can accelerate

$$\hat{w}_j = \prod_{l \neq j} \frac{1}{\omega^{2^j} - \omega^{2^l}} \tag{24}$$

3. Precompute KZG10 SRS of Lagrange Basis
$A_0 = [L_0(\tau)]_1, A_1 = [L_1(\tau)]_1, A_2 = [L_2(\tau)]_1, \ldots, A_{N-1} = [L_{2^{n-1}}(\tau)]_1$

Common inputs

1. $C_a = [\hat{f}(\tau)]_1$: the (uni-variate) commitment of $\tilde{f}(X_0, X_1, \ldots, X_{n-1})$
2. $\vec{u} = (u_0, u_1, \ldots, u_{n-1})$: evaluation point
3. $v = \tilde{f}(u_0, u_1, \ldots, u_{n-1})$: computation value of MLE polynomial $\tilde{f}$ at $\vec{X} = \vec{u}$

Commit Calculation Process

1. Prover constructs univariate polynomial $a(X)$ such that its Evaluation form equals $\vec{a} = (a_0, a_1, \ldots, a_{N-1})$, where $a_i = \tilde{f}(\mathsf{bits}(i))$, which is the value of $\tilde{f}$ on the Boolean Hypercube $\{0,1\}^n$.

$$a(X) = a_0 \cdot L_0(X) + a_1 \cdot L_1(X) + a_2 \cdot L_2(X) + \cdots + a_{N-1} \cdot L_{N-1}(X) \tag{25}$$

2. Prover calculates commitment C_a of $\hat{f}(X)$ and sends C_a

$$C_a = a_0 \cdot A_0 + a_1 \cdot A_1 + a_2 \cdot A_2 + \cdots + a_{N-1} \cdot A_{N-1} = [\hat{f}(\tau)]_1 \tag{26}$$

where $A_0 = [L_0(\tau)]_1, A_1 = [L_1(\tau)]_1, A_2 = [L_2(\tau)]_1, \ldots, A_{N-1} = [L_{2^{n-1}}(\tau)]_1$ have been obtained in the precomputation process.

Evaluation Proof Protocol

Recall the constraint of polynomial computation to be proved:

$$\tilde{f}(u_0, u_1, u_2, \ldots, u_{n-1}) = v \tag{27}$$

Here $\vec{u} = (u_0, u_1, u_2, \ldots, u_{n-1})$ is a public challenge point.

Round 1.

Prover:

1. Calculate vector $\vec{c}$, where each element $c_i = \tilde{eq}(\mathsf{bits}(i), \vec{u})$
2. Construct polynomial $c(X)$, whose computation result on H is exactly $\vec{c}$.

$$c(X) = \sum_{i=0}^{N-1} c_i \cdot L_i(X) \tag{28}$$

3. Calculate commitment $C_c = [c(\tau)]_1$ of $c(X)$ and send C_c

$$C_c = \mathsf{KZG10.Commit}(\vec{c}) = [c(\tau)]_1 \tag{29}$$

Round 2.

Verifier: Send challenge number $\alpha \leftarrow_{\$} \mathbb{F}_p$

Prover:

1. Construct constraint polynomials $p_0(X), \ldots, p_n(X)$ about $\vec{c}$

$$
\begin{aligned}
p_0(X) &= s_0(X) \cdot \Big(c(X) - (1-u_0)(1-u_1)\ldots(1-u_{n-1})\Big) \\
p_k(X) &= s_{k-1}(X) \cdot \Big(u_{n-k} \cdot c(X) - (1-u_{n-k}) \cdot c(\omega^{2^{n-k}} \cdot X)\Big), \quad k = 1 \ldots n
\end{aligned} \tag{30}
$$

2. Aggregate $\{p_i(X)\}$ into one polynomial $p(X)$

$$
p(X) = p_0(X) + \alpha \cdot p_1(X) + \alpha^2 \cdot p_2(X) + \cdots + \alpha^n \cdot p_n(X) \tag{31}
$$

3. Construct accumulation polynomial $z(X)$, satisfying

$$
\begin{aligned}
z(1) &= a_0 \cdot c_0 \\
z(\omega_i) - z(\omega_{i-1}) &= a(\omega_i) \cdot c(\omega_i), \quad i = 1, \ldots, N-1 \\
z(\omega^{N-1}) &= v
\end{aligned} \tag{32}
$$

4. Construct constraint polynomials $h_0(X), h_1(X), h_2(X)$, satisfying

$$
\begin{aligned}
h_0(X) &= L_0(X) \cdot \big(z(X) - c_0 \cdot a(X)\big) \\
h_1(X) &= (X-1) \cdot \big(z(X) - z(\omega^{-1} \cdot X) - a(X) \cdot c(X)\big) \\
h_2(X) &= L_{N-1}(X) \cdot \big(z(X) - v\big)
\end{aligned} \tag{33}
$$

5. Aggregate $p(X)$ and $h_0(X), h_1(X), h_2(X)$ into one polynomial $h(X)$, satisfying

$$
h(X) = p(X) + \alpha^{n+1} \cdot h_0(X) + \alpha^{n+2} \cdot h_1(X) + \alpha^{n+3} \cdot h_2(X) \tag{34}
$$

6. Calculate Quotient polynomial $t(X)$, satisfying

$$
h(X) = t(X) \cdot v_H(X) \tag{35}
$$

7. Calculate $C_t = [t(\tau)]_1, C_z = [z(\tau)]_1$, and send C_t and C_z

$$
\begin{aligned}
C_t &= \mathsf{KZG10.Commit}(t(X)) = [t(\tau)]_1 \\
C_z &= \mathsf{KZG10.Commit}(z(X)) = [z(\tau)]_1
\end{aligned} \tag{36}
$$

Round 3.

Verifier: Send random evaluation point $\zeta \leftarrow_{\$} \mathbb{F}_p$

Prover:

1. Calculate the values of $s_i(X)$ at ζ:

$$
s_0(\zeta), s_1(\zeta), \ldots, s_{n-1}(\zeta) \tag{37}
$$

Here the Prover can efficiently calculate $s_i(\zeta)$. From the formula of $s_i(X)$, we get

$$
\begin{aligned}
s_i(\zeta) &= \frac{\zeta^N - 1}{\zeta^{2^i} - 1} \\
&= \frac{(\zeta^N - 1)(\zeta^{2^i} + 1)}{(\zeta^{2^i} - 1)(\zeta^{2^i} + 1)} \\
&= \frac{\zeta^N - 1}{\zeta^{2^{i+1}} - 1} \cdot (\zeta^{2^i} + 1) \\
&= s_{i+1}(\zeta) \cdot (\zeta^{2^i} + 1)
\end{aligned} \tag{38}
$$

Therefore, the value of $s_i(\zeta)$ can be calculated from $s_{i+1}(\zeta)$, and

$$s_{n-1}(\zeta) = \frac{\zeta^N - 1}{\zeta^{2^{n-1}} - 1} = \zeta^{2^{n-1}} + 1 \tag{39}$$

Thus, we can get an $O(n)$ algorithm to calculate $s_i(\zeta)$, and it doesn't contain division operations. The calculation process is: $s_{n-1}(\zeta) \to s_{n-2}(\zeta) \to \cdots \to s_0(\zeta)$.

2. Define evaluation Domain D', containing $n+1$ elements:

$$D' = D\zeta = \{\zeta, \omega\zeta, \omega^2\zeta, \omega^4\zeta, \ldots, \omega^{2^{n-1}}\zeta\} \tag{40}$$

3. Calculate and send the values of $c(X)$ on D'

$$c(\zeta), c(\zeta\cdot\omega), c(\zeta\cdot\omega^2), c(\zeta\cdot\omega^4), \ldots, c(\zeta\cdot\omega^{2^{n-1}}) \tag{41}$$

4. Calculate and send $z(\omega^{-1}\cdot\zeta)$
5. Calculate Linearized Polynomial $l_\zeta(X)$

$$\begin{aligned}
l_\zeta(X) = \Big(& s_0(\zeta)\cdot(c(\zeta) - c_0) \\
& + \alpha\cdot s_0(\zeta)\cdot(u_{n-1}\cdot c(\zeta) - (1-u_{n-1})\cdot c(\omega^{2^{n-1}}\cdot\zeta)) \\
& + \alpha^2\cdot s_1(\zeta)\cdot(u_{n-2}\cdot c(\zeta) - (1-u_{n-2})\cdot c(\omega^{2^{n-2}}\cdot\zeta)) \\
& + \cdots \\
& + \alpha^{n-1}\cdot s_{n-2}(\zeta)\cdot(u_1\cdot c(\zeta) - (1-u_1)\cdot c(\omega^2\cdot\zeta)) \\
& + \alpha^{n}\cdot s_{n-1}(\zeta)\cdot(u_0\cdot c(\zeta) - (1-u_0)\cdot c(\omega\cdot\zeta)) \\
& + \alpha^{n+1}\cdot(L_0(\zeta)\cdot\big(z(X) - c_0\cdot a(X)) \\
& + \alpha^{n+2}\cdot(\zeta-1)\cdot\big(z(X) - z(\omega^{-1}\cdot\zeta) - c(\zeta)\cdot a(X)) \\
& + \alpha^{n+3}\cdot L_{N-1}(\zeta)\cdot(z(X) - v) \\
& - v_H(\zeta)\cdot t(X)\Big)
\end{aligned} \tag{42}$$

Obviously, $l_\zeta(\zeta) = 0$, so this computation value doesn't need to be sent to the Verifier, and $[l_\zeta(\tau)]_1$ can be constructed by the Verifier themselves.

6. Construct polynomial $c^*(X)$, which is the interpolation polynomial of the following vector on $D\zeta$

$$\vec{c^*} = \Big(c(\omega\cdot\zeta), c(\omega^2\cdot\zeta), c(\omega^4\cdot\zeta), \ldots, c(\omega^{2^{n-1}}\cdot\zeta), c(\zeta)\Big) \tag{43}$$

The Prover can use the precomputed Bary-Centric Weights $\{\hat{w}_i\}$ on D to quickly calculate $c^*(X)$,

$$c^*(X) = \frac{c_0^*\cdot\frac{\hat{w}_0}{X-\omega\zeta} + c_1^*\cdot\frac{\hat{w}_1}{X-\omega^2\zeta} + \cdots + c_n^*\cdot\frac{\hat{w}_n}{X-\omega^{2^n}\zeta}}{\frac{\hat{w}_0}{X-\omega\zeta} + \frac{\hat{w}_1}{X-\omega^2\zeta} + \cdots + \frac{\hat{w}_n}{X-\omega^{2^n}\zeta}} \tag{44}$$

Here $\hat{w}_j$ are precomputed values:

$$\hat{w}_j = \prod_{l\neq j}\frac{1}{\omega^{2^j} - \omega^{2^l}} \tag{45}$$

7. Because $l_\zeta(\zeta) = 0$, there exists a Quotient polynomial $q_\zeta(X)$ satisfying

$$q_\zeta(X) = \frac{1}{X-\zeta} \cdot l_\zeta(X) \tag{46}$$

8. Construct vanishing polynomial $z_{D_\zeta}(X)$ on $D\zeta$

$$z_{D_\zeta}(X) = (X - \zeta\omega)\cdots(X - \zeta\omega^{2^{n-1}})(X - \zeta) \tag{47}$$

9. Construct Quotient polynomial $q_c(X)$:

$$q_c(X) = \frac{(c(X) - c^*(X))}{(X-\zeta)(X-\omega\zeta)(X-\omega^2\zeta)\cdots(X-\omega^{2^{n-1}}\zeta)} \tag{48}$$

10. Construct Quotient polynomial $q_{\omega\zeta}(X)$

$$q_{\omega\zeta}(X) = \frac{z(X) - z(\omega^{-1}\cdot\zeta)}{X - \omega^{-1}\cdot\zeta} \tag{49}$$

11. Send $\big(Q_c = [q_c(\tau)]_1, Q_\zeta = [q_\zeta(\tau)]_1, Q_{\omega\zeta} = [q_{\omega\zeta}(\tau)]_1, \big)$

Round 4.

1. Verifier sends the second random challenge point $\xi \leftarrow_\$ \mathbb{F}_p$
2. Prover constructs the third Quotient polynomial $q_\xi(X)$

$$q_\xi(X) = \frac{c(X) - c^*(\xi) - z_{D_\zeta}(\xi)\cdot q_c(X)}{X - \xi} \tag{50}$$

3. Prover calculates and sends Q_ξ

$$Q_\xi = \mathsf{KZG10.Commit}(q_\xi(X)) = [q_\xi(\tau)]_1 \tag{51}$$

Proof

$7\cdot\mathbb{G}_1$, $(n+1)\cdot\mathbb{F}$

$$\begin{aligned}\pi_{eval} = \big(&z(\omega^{-1}\cdot\zeta), c(\zeta),\ c(\omega\cdot\zeta), c(\omega^2\cdot\zeta), c(\omega^4\cdot\zeta), \ldots, c(\omega^{2^{n-1}}\cdot\zeta),\\ &C_c, C_t, C_z, Q_c, Q_\zeta, Q_\xi, Q_{\omega\zeta}\big)\end{aligned} \tag{52}$$

Verification Process

1. Verifier calculates $c^*(\xi)$ using precomputed Barycentric Weights $\{\hat{w}_i\}$

$$c^*(\xi) = \frac{\sum_i c_i \frac{w_i}{\xi - x_i}}{\sum_i \frac{w_i}{\xi - x_i}} \tag{53}$$

2. Verifier calculates $v_H(\zeta), L_0(\zeta), L_{N-1}(\zeta)$

$$v_H(\zeta) = \zeta^N - 1 \tag{54}$$

$$L_0(\zeta) = \frac{1}{N}\cdot\frac{z_H(\zeta)}{\zeta - 1} \tag{55}$$

$$L_{N-1}(\zeta) = \frac{\omega^{N-1}}{N} \cdot \frac{z_H(\zeta)}{\zeta - \omega^{N-1}} \tag{56}$$

3. Verifier calculates $s_0(\zeta), \ldots, s_{n-1}(\zeta)$, which can be calculated using the recursive method mentioned earlier.
4. Verifier calculates the commitment of the linearized polynomial C_l

$$\begin{aligned}
C_l = \Big(& (c(\zeta) - c_0)s_0(\zeta) \\
& + \alpha \cdot (u_{n-1} \cdot c(\zeta) - (1 - u_{n-1}) \cdot c(\omega^{2^{n-1}} \cdot \zeta)) \cdot s_0(\zeta) \\
& + \alpha^2 \cdot (u_{n-2} \cdot c(\zeta) - (1 - u_{n-2}) \cdot c(\omega^{2^{n-2}} \cdot \zeta)) \cdot s_1(\zeta) \\
& + \cdots \\
& + \alpha^{n-1} \cdot (u_1 \cdot c(\zeta) - (1 - u_1) \cdot c(\omega^2 \cdot \zeta)) \cdot s_{n-2}(\zeta) \\
& + \alpha^n \cdot (u_0 \cdot c(\zeta) - (1 - u_0) \cdot c(\omega \cdot \zeta)) \cdot s_{n-1}(\zeta) \\
& + \alpha^{n+1} \cdot L_0(\zeta) \cdot (C_z - c_0 \cdot C_a) \\
& + \alpha^{n+2} \cdot (\zeta - 1) \cdot \big(C_z - z(\omega^{-1} \cdot \zeta) - c(\zeta) \cdot C_a) \\
& + \alpha^{n+3} \cdot L_{N-1}(\zeta) \cdot (C_z - v) \\
& - v_H(\zeta) \cdot C_t \Big)
\end{aligned} \tag{57}$$

5. Verifier generates a random number η to merge the following Pairing verifications:

$$\begin{aligned}
e(C_l + \zeta \cdot Q_\zeta, [1]_2) &\stackrel{?}{=} e(Q_\zeta, [\tau]_2) \\
e(C - C^*(\xi) - z_{D_\zeta}(\xi) \cdot Q_c + \xi \cdot Q_\xi, [1]_2) &\stackrel{?}{=} e(Q_\xi, [\tau]_2) \\
e(C_z + \zeta \cdot Q_{\omega\zeta} - z(\omega^{-1} \cdot \zeta) \cdot [1]_1, [1]_2) &\stackrel{?}{=} e(Q_{\omega\zeta}, [\tau]_2)
\end{aligned} \tag{58}$$

After merging, the verification only needs two Pairing operations.

$$\begin{aligned}
P = & \Big(C_l + \zeta \cdot Q_\zeta\Big) \\
& + \eta \cdot \Big(C - C^* - z_{D_\zeta}(\xi) \cdot Q_c + \xi \cdot Q_\xi\Big) \\
& + \eta^2 \cdot \Big(C_z + \zeta \cdot Q_{\omega\zeta} - z(\omega^{-1} \cdot \zeta) \cdot [1]_1\Big)
\end{aligned} \tag{59}$$

$$e\Big(P, [1]_2\Big) \stackrel{?}{=} e\Big(Q_\zeta + \eta \cdot Q_\xi + \eta^2 \cdot Q_{\omega\zeta}, [\tau]_2\Big) \tag{60}$$

3. Optimized Performance Analysis

Proof size: $7\ \mathbb{G}_1 + (n+1)\ \mathbb{F}$

Prover's cost

- Commit phase: $O(N \log N)\ \mathbb{F}$ + $\mathbb{G}_1$
- Evaluation phase: $O(N \log N)\ \mathbb{F}$ + $7\ \mathbb{G}_1$

Verifier's cost: $4\ \mathbb{F} + O(n)\ \mathbb{F} + 3\ \mathbb{G}_1 + 2\ P$

References

- [BDFG20] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. "Efficient polynomial commitment schemes for multiple points and polynomials". Cryptology {ePrint} Archive, Paper 2020/081. https://eprint.iacr.org/2020/081.