Missing Protocol PH23-PCS (Part 3)

This article adds support for Zero-knowledge to the PH23-KZG10 protocol.

1. How to Support ZK

To make the PH23-KZG10 protocol support ZK, we need to modify two parts of the protocol. First, we need to support Hiding in the KZG10 sub-protocol, which means that no information other than the evaluation will be leaked in any Evaluation proof. Second, we need to ensure that no information about the Witness vector $\vec{a}$ is leaked in the PH23 protocol.

First, we need a Perfect Hiding KZG10 protocol that can guarantee that no information other than the polynomial evaluation is leaked after each opening of the polynomial. The following is the KZG10 protocol from [KT23], with its main ideas derived from [PST13], [ZGKPP17], and [XZZPS19].

Hiding KZG10

$$SRS = ([1]_1, [\tau]_1, [\tau^2]_1, [\tau^3]_1, \ldots, [\tau^D]_1, {\color{red}[\gamma]_1}, [1]_2, [\tau]_2, {\color{red}[\gamma]_2}) \tag{1}$$

The commitment of a polynomial $f(X) \in \mathbb{F}[X]$ is defined as:

$$C_f = \mathsf{KZG.Commit}(f(X); {\color{blue}\rho_f}) = f_0 \cdot [1]_1 + f_1 \cdot [\tau]_1 + \cdots + f_d \cdot [\tau^d]_1 + {\color{blue}\rho_f} \cdot {\color{red}[\gamma]_1} \tag{2}$$

According to the properties of polynomial rings, $f(X)$ can be decomposed as:

$$f(X) = q(X) \cdot (X - z) + f(z) \tag{3}$$

The commitment of the quotient polynomial is calculated as follows, also requiring a Blinding Factor ${\color{blue}\rho_q}$ to protect the commitment of $q(X)$.

$$\begin{aligned} Q = \mathsf{KZG.Commit}(q(X); {\color{blue}\rho_q}) &= q_0 \cdot [1]_1 + q_1 \cdot [\tau]_1 + \cdots + q_d \cdot [\tau^{d-1}]_1 + {\color{blue}\rho_q} \cdot {\color{red}[\gamma]_1} \\ &= [q(\tau)]_1 + {\color{blue}\rho_q} \cdot {\color{red}[\gamma]_1} \end{aligned} \tag{4}$$

The Prover also needs to calculate an additional $\mathbb{G}_1$ element below to balance the verification formula:

$${\color{blue}E = \rho_f \cdot [1]_1 - \rho_q \cdot [\tau]_1 + (\rho_q \cdot z) \cdot [1]_1} \tag{5}$$

Then, the Evaluation proof consists of two $\mathbb{G}_1$ elements:

$$\pi = (Q, {\color{blue}E}) \tag{6}$$

Thus, the Verifier can verify using the following formula:

$$e\Big(C_f - f(z) \cdot [1]_1,\ [1]_2\Big) = e\Big(Q,\ [\tau]_2 - z \cdot [1]_2\Big) + {\color{blue}e\Big(E,\ {\color{red}[\gamma]_2}\Big)} \tag{7}$$

ZK for Sum Proof

In the process where the Prover uses the accumulation polynomial $z(X)$ to prove the sum value, information about the $\vec{z}$ vector, including information about the Witness $\vec{a}$, would also be leaked. Therefore, we need a ZK version of the sum proof protocol.

We have a multiplicative subgroup $H \subset \mathbb{F}$ of order N:

$$H = (1, \omega, \omega^2, \ldots, \omega^{N-1}) \tag{8}$$

We denote $\{L_i(X)\}_{i=0}^{N-1}$ as the Lagrange polynomials with respect to H, and $v_H(X) = X^N - 1$ is the vanishing polynomial on H.

Suppose we have a vector $\vec{a} = (a_0, a_1, \ldots, a_{N-1})$ with N elements, and we want to prove $\sum_i a_i = v$. The Prover has actually computed the commitment of $\vec{a}$, denoted as C_a.

$$C_a = \mathsf{KZG10.Commit}(a(X); \rho_a) = [a(\tau)]_1 + \rho_a \cdot [\gamma]_1 \tag{9}$$

Round 1

First, we need to determine how many times $z(X)$ will be opened, for example, $z(X)$ will be opened at ζ and $\omega^{-1} \cdot \zeta$. Then we introduce a random polynomial: $r(X)$,

$$r(X) = r_0 \cdot L_0(X) + r_1 \cdot L_1(X) + r_2 \cdot L_2(X) + r_3 \cdot L_3(X) \tag{10}$$

This polynomial contains four random factors. Why four? We'll see later.

The Prover then computes the commitment of $r(X)$ and introduces an additional Blinding Factor ρ_r:

$$C_r = \mathsf{KZG10.Commit}(r(X); \rho_r) = [r(\tau)]_1 + \rho_r \cdot [\gamma]_1 \tag{11}$$

The Prover computes a new sum $\sum_i r_i$:

$$v_r = r_0 + r_1 + r_2 + r_3 \tag{12}$$

The Prover sends C_r and v_r to the Verifier.

Round 2

The Verifier sends a random challenge $\beta \leftarrow_{\$} \mathbb{F}$ to the Prover.

The Prover constructs a new polynomial $a'(X)$ satisfying

$$a'(X) = a(X) + \beta \cdot r(X) \tag{13}$$

The Prover sends a mixed sum value v' to the Verifier:

$$v' = v_r + \beta \cdot v \tag{14}$$

At this point, the Prover and Verifier convert the sum proof target $\sum_i a_i = v$ into $\sum_i (a_i + \beta \cdot r_i) = v + \beta \cdot v_r$.

Round 3

The Verifier sends another random number $\alpha \leftarrow_{\$} \mathbb{F}$ to the Prover.

The Prover constructs constraint polynomials $h_0(X), h_1(X), h_2(X)$ satisfying

$$\begin{aligned} h_0(X) &= L_0(X) \cdot \big(z(X) - a(X)\big) \\ h_1(X) &= (X - 1) \cdot \big(z(X) - z(\omega^{-1} \cdot X) - a(X)\big) \\ h_2(X) &= L_{N-1}(X) \cdot \big(z(X) - v\big) \end{aligned} \tag{15}$$

The Prover constructs polynomial $h(X)$ satisfying

$$h(X) = h_0(X) + \alpha \cdot h_1(X) + \alpha^2 \cdot h_2(X) \tag{16}$$

The Prover computes the quotient polynomial $t(X)$ satisfying

$$h(X) = t(X) \cdot v_H(X) \tag{17}$$

The Prover computes the commitment of $z(X)$, C_z, and sends C_z

$$C_z = \mathsf{KZG10.Commit}(z(X); \rho_z) = [z(\tau)]_1 + \rho_z \cdot [\gamma]_1 \tag{18}$$

The Prover computes the commitment of $t(X)$, C_t, and sends C_t

$$C_t = \mathsf{KZG10.Commit}(t(X); \rho_t) = [t(\tau)]_1 + \rho_t \cdot [\gamma]_1 \tag{19}$$

Round 4

The Verifier sends a random evaluation point $\zeta \leftarrow_{\$} \mathbb{F}$

The Prover constructs quotient polynomials $q_a(X)$, $q_z(X)$, $q_t(X)$, and $q'_z(X)$ satisfying

$$q_a(X) = \frac{a'(X) - a'(\zeta)}{X - \zeta} \tag{20}$$

$$q_t(X) = \frac{t(X) - t(\zeta)}{X - \zeta} \tag{21}$$

$$q_z(X) = \frac{z(X) - z(\zeta)}{X - \zeta} \tag{22}$$

$$q'_z(X) = \frac{z(X) - z(\omega^{-1} \cdot \zeta)}{X - \omega^{-1} \cdot \zeta} \tag{23}$$

The Prover computes the commitments of the four quotient polynomials and introduces corresponding Blinding Factors $\rho_{q_a}, \rho_{q_z}, \rho_{q_t}, \rho_{q'_z}$

$$\begin{aligned}
Q_a &= \mathsf{KZG10.Commit}(q_a(X); \rho_{q_a}) = [q_a(\tau)]_1 + \rho_{q_a} \cdot [\gamma]_1 \\
Q_z &= \mathsf{KZG10.Commit}(q_z(X); \rho_{q_z}) = [q_z(\tau)]_1 + \rho_{q_z} \cdot [\gamma]_1 \\
Q_t &= \mathsf{KZG10.Commit}(q_t(X); \rho_{q_t}) = [q_t(\tau)]_1 + \rho_{q_t} \cdot [\gamma]_1 \\
Q'_z &= \mathsf{KZG10.Commit}(q'_z(X); \rho_{q'_z}) = [q'_z(\tau)]_1 + \rho_{q'_z} \cdot [\gamma]_1
\end{aligned} \tag{24}$$

The Prover also needs to construct four corresponding Blinding Factor commitments and send them to the Verifier:

$$\begin{aligned}
E_a &= (\rho_a + \beta \cdot \rho_r) \cdot [1]_1 - \rho_{q_a} \cdot [\tau]_1 + (\rho_{q_a} \cdot \zeta) \cdot [1]_1 \\
E_z &= \rho_z \cdot [1]_1 - \rho_{q_z} \cdot [\tau]_1 + (\rho_{q_z} \cdot \zeta) \cdot [1]_1 \\
E_t &= \rho_t \cdot [1]_1 - \rho_{q_t} \cdot [\tau]_1 + (\rho_{q_t} \cdot \zeta) \cdot [1]_1 \\
E'_z &= \rho_z \cdot [1]_1 - \rho_{q'_z} \cdot [\tau]_1 + (\rho_{q'_z} \cdot \omega^{-1} \cdot \zeta) \cdot [1]_1
\end{aligned} \tag{25}$$

Here we can see that during the proof process, the Prover needs to evaluate four polynomials, and the evaluations of these four polynomials would all leak information about $\vec{a}$. Therefore, the Prover adds a random polynomial $r(X)$ containing two additional random factors in Round 1. This way, all polynomial evaluations in the proof process are performed on $a'(X)$, rather than directly computing and evaluating $a(X)$.

Proof

$$\pi = (C_r, v_r, C_z, C_t, a'(\zeta), z(\zeta), t(\zeta), z(\omega^{-1}\cdot\zeta), Q_a, Q_z, Q_t, Q'_z, E_a, E_z, E_t, E'_z) \tag{26}$$

Verification

The Verifier first checks the following equation:

$$h(\zeta) = t(\zeta)\cdot v_H(\zeta) \tag{27}$$

where $v_H(\zeta)$ is computed by the Verifier, and $h(\zeta)$ is calculated using the following equation:

$$\begin{aligned} h(\zeta) = & L_0(\zeta)\cdot\big(z(\zeta)-a'(\zeta)\big) \\ & +\alpha\cdot(\zeta-1)\cdot\big(z(\zeta)-z(\omega^{-1}\cdot\zeta)-a'(\zeta)\big) \\ & +\alpha^2\cdot L_{N-1}(\zeta)\cdot\big(z(\zeta)-(v_r+\beta\cdot v)\big) \end{aligned} \tag{28}$$

Then the Verifier checks the correctness of $a'(\zeta), z(\zeta), t(\zeta), z(\omega^{-1}\cdot\zeta)$:

$$\begin{aligned} e\Big(C_{a'} - a'(\zeta)\cdot[1]_1,\ [1]_2\Big) &= e\Big(Q_a,\ [\tau]_2 - \zeta\cdot[1]_2\Big) + e\Big(E_a,\ [\gamma]_2\Big) \\ e\Big(C_z - z(\zeta)\cdot[1]_1,\ [1]_2\Big) &= e\Big(Q_z,\ [\tau]_2 - \zeta\cdot[1]_2\Big) + e\Big(E_z,\ [\gamma]_2\Big) \\ e\Big(C_t - t(\zeta)\cdot[1]_1,\ [1]_2\Big) &= e\Big(Q_t,\ [\tau]_2 - \zeta\cdot[1]_2\Big) + e\Big(E_t,\ [\gamma]_2\Big) \\ e\Big(C_z - (\omega^{-1}\cdot\zeta)\cdot[1]_1,\ [1]_2\Big) &= e\Big(Q'_z,\ [\tau]_2 - \omega^{-1}\cdot\zeta\cdot[1]_2\Big) + e\Big(E'_z,\ [\gamma]_2\Big) \end{aligned} \tag{29}$$

2. ZK-PH23-KZG10 Protocol (Optimized Version)

Below is the complete PH23-KZG10 protocol supporting Zero-knowledge.

Precomputation

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

$$v_H(X) = X^N - 1 \tag{30}$$

$$s_i(X) = \frac{v_H(X)}{v_{H_i}(X)} = \frac{X^N-1}{X^{2^i}-1} \tag{31}$$

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

$$\hat{w}_j = \prod_{l\neq j}\frac{1}{\omega^{2^j}-\omega^{2^l}} \tag{32}$$

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

Commit Computation Process

1. The Prover constructs a 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 $\{0,1\}^n$.

$$a(X) = a_0\cdot L_0(X) + a_1\cdot L_1(X) + a_2\cdot L_2(X) + \cdots + a_{N-1}\cdot L_{N-1}(X) \tag{33}$$

2. The Prover samples a random number $\rho_a \leftarrow_\$ \mathbb{F}$ to protect the commitment of $\vec{a}$.
3. The Prover computes the commitment of $\hat{f}(X)$, C_a, and sends C_a

$$C_a = a_0 \cdot A_0 + a_1 \cdot A_1 + a_2 \cdot A_2 + \cdots + a_{N-1} \cdot A_{N-1} + \rho_a \cdot [\gamma]_1 = [\hat{f}(\tau)]_1 + \rho_a \cdot [\gamma]_1 \quad (34)$$

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

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. $v = \tilde{f}(u_0, u_1, \ldots, u_{n-1})$: The computed value of the MLE polynomial $\tilde{f}$ at $\vec{X} = \vec{u}$.

Recall the constraint of the polynomial computation to be proven:

$$\tilde{f}(u_0, u_1, u_2, \ldots, u_{n-1}) = v \quad (35)$$

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

Round 1.

Prover:

1. Compute vector $\vec{c}$, where each element $c_i = \tilde{eq}(\mathsf{bits}(i), \vec{u})$
2. Construct polynomial $c(X)$, whose evaluation results on H are exactly $\vec{c}$.

$$c(X) = \sum_{i=0}^{N-1} c_i \cdot L_i(X) \quad (36)$$

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

$$C_c = \mathsf{KZG10.Commit}(\vec{c}) = [c(\tau)]_1 \quad (37)$$

4. Construct a Blinding polynomial $r(X) = r_0 \cdot L_0(X) + r_1 \cdot L_1(X)$, where $\{r_0, r_1\} \leftarrow_\$ \mathbb{F}^2$ are randomly sampled Blinding Factors.
5. Compute the commitment of $r(X)$, $C_r = [r(\tau)]_1$, and send C_r

$$C_r = \mathsf{KZG10.Commit}(r(X); \rho_r) = [r(\tau)]_1 + \rho_r \cdot [\gamma]_1 \quad (38)$$

6. Compute $v_r = \langle \vec{r}, \vec{c} \rangle$, and send v_r, where $\vec{r}$ is defined as:

$$\vec{r} \in \mathbb{F}^N = (r_0, r_1, 0, \cdots, 0) \quad (39)$$

Round 2.

Verifier: Send challenge numbers $\alpha, \beta \leftarrow_\$ \mathbb{F}_p^2$

Prover:

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

$$
\begin{aligned}
p_0(X) &= s_0(X) \cdot \Big(c(X) - (1-u_0)(1-u_1)\ldots(1-u_{n-1})\Big) \\
p_k(X) &= s_{k-1}(X) \cdot \Big(u_{n-k} \cdot c(X) - (1-u_{n-k}) \cdot c(\omega^{2^{n-k}} \cdot X)\Big), \quad k = 1 \ldots n
\end{aligned} \tag{40}
$$

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

$$
p(X) = p_0(X) + \alpha \cdot p_1(X) + \alpha^2 \cdot p_2(X) + \cdots + \alpha^n \cdot p_n(X) \tag{41}
$$

3. Construct $a'(X)$, and compute $\langle \vec{a}', \vec{c} \rangle = v'$

$$
a'(X) = a(X) + \beta \cdot r(X) \tag{42}
$$

4. Construct accumulation polynomial $z(X)$ satisfying

$$
\begin{aligned}
z(1) &= a'_0 \cdot c_0 \\
z(\omega_i) - z(\omega_{i-1}) &= a'(\omega_i) \cdot c(\omega_i), \quad i = 1, \ldots, N-1 \\
z(\omega^{N-1}) &= v'
\end{aligned} \tag{43}
$$

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

$$
\begin{aligned}
h_0(X) &= L_0(X) \cdot \big(z(X) - c_0 \cdot a'(X)\big) \\
h_1(X) &= (X-1) \cdot \big(z(X) - z(\omega^{-1} \cdot X) - a'(X) \cdot c(X)\big) \\
h_2(X) &= L_{N-1}(X) \cdot \big(z(X) - v'\big)
\end{aligned} \tag{44}
$$

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

$$
h(X) = p(X) + \alpha^{n+1} \cdot h_0(X) + \alpha^{n+2} \cdot h_1(X) + \alpha^{n+3} \cdot h_2(X) \tag{45}
$$

6. Compute the Quotient polynomial $t(X)$ satisfying

$$
h(X) = t(X) \cdot v_H(X) \tag{46}
$$

7. Sample $\rho_t, \rho_z \leftarrow_\$ \mathbb{F}_p^2$, compute $C_t = [t(\tau)]_1 + \rho_t \cdot [\gamma]_1$, $C_z = [z(\tau)]_1 + \rho_z \cdot [\gamma]_1$, and send C_t and C_z

$$
\begin{aligned}
C_t &= \mathsf{KZG10.Commit}(t(X); \rho_t) = [t(\tau)]_1 + \rho_t \cdot [\gamma]_1 \\
C_z &= \mathsf{KZG10.Commit}(z(X); \rho_z) = [z(\tau)]_1 + \rho_z \cdot [\gamma]_1
\end{aligned} \tag{47}
$$

Round 3.

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

Prover:

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

$$
s_0(\zeta), s_1(\zeta), \ldots, s_{n-1}(\zeta) \tag{48}
$$

Here the Prover can quickly compute $s_i(\zeta)$. From the formula of $s_i(X)$, we have

$$
\begin{aligned}
s_i(\zeta) &= \frac{\zeta^N - 1}{\zeta^{2^i} - 1} \\
&= \frac{(\zeta^N - 1)(\zeta^{2^i} + 1)}{(\zeta^{2^i} - 1)(\zeta^{2^i} + 1)} \\
&= \frac{\zeta^N - 1}{\zeta^{2^{i+1}} - 1} \cdot (\zeta^{2^i} + 1) \\
&= s_{i+1}(\zeta) \cdot (\zeta^{2^i} + 1)
\end{aligned} \tag{49}
$$

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

$$
s_{n-1}(\zeta) = \frac{\zeta^N - 1}{\zeta^{2^{n-1}} - 1} = \zeta^{2^{n-1}} + 1 \tag{50}
$$

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

2. Define the evaluation Domain D', which includes $n+1$ elements:

$$
D' = D\zeta = \{\zeta, \omega\zeta, \omega^2\zeta, \omega^4\zeta, \ldots, \omega^{2^{n-1}}\zeta\} \tag{51}
$$

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

$$
c(\zeta), c(\zeta \cdot \omega), c(\zeta \cdot \omega^2), c(\zeta \cdot \omega^4), \ldots, c(\zeta \cdot \omega^{2^{n-1}}) \tag{52}
$$

4. Compute and send $z(\omega^{-1} \cdot \zeta)$

5. Compute the Linearized Polynomial $l_\zeta(X)$

$$
\begin{aligned}
l_\zeta(X) = \Big(& s_0(\zeta) \cdot (c(\zeta) - c_0) \\
& + \alpha \cdot s_0(\zeta) \cdot (u_{n-1} \cdot c(\zeta) - (1 - u_{n-1}) \cdot c(\omega^{2^{n-1}} \cdot \zeta)) \\
& + \alpha^2 \cdot s_1(\zeta) \cdot (u_{n-2} \cdot c(\zeta) - (1 - u_{n-2}) \cdot c(\omega^{2^{n-2}} \cdot \zeta)) \\
& + \cdots \\
& + \alpha^{n-1} \cdot s_{n-2}(\zeta) \cdot (u_1 \cdot c(\zeta) - (1 - u_1) \cdot c(\omega^2 \cdot \zeta)) \\
& + \alpha^n \cdot s_{n-1}(\zeta) \cdot (u_0 \cdot c(\zeta) - (1 - u_0) \cdot c(\omega \cdot \zeta)) \\
& + \alpha^{n+1} \cdot (L_0(\zeta) \cdot \big(z(X) - c_0 \cdot a'(X)\big) \\
& + \alpha^{n+2} \cdot (\zeta - 1) \cdot \big(z(X) - z(\omega^{-1} \cdot \zeta) - c(\zeta) \cdot a'(X)\big) \\
& + \alpha^{n+3} \cdot L_{N-1}(\zeta) \cdot (z(X) - v') \\
& - v_H(\zeta) \cdot t(X) \Big)
\end{aligned} \tag{53}
$$

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

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

$$
\begin{gathered}
\alpha^{n+1} L_0(\zeta)(\rho_z - c_0 \cdot \rho_a) \\
+ \alpha^{n+2}(\zeta - 1)(\rho_z - c(\zeta) \cdot \rho_a) \\
+ \alpha^{n+3} L_{N-1}(\zeta) \cdot \rho_z \\
- v_H(\zeta) \cdot \rho_t
\end{gathered} \tag{54}
$$

$$\vec{c^*} = \Big(c(\omega\cdot\zeta), c(\omega^2\cdot\zeta), c(\omega^4\cdot\zeta), \ldots, c(\omega^{2^{n-1}}\cdot\zeta), c(\zeta)\Big) \tag{55}$$

The Prover can use the pre-computed Barycentric Weights $\{\hat{w}_i\}$ on D to quickly compute $c^*(X)$,

$$c^*(X) = \frac{c^*_0\cdot\frac{\hat{w}_0}{X-\omega\zeta} + c^*_1\cdot\frac{\hat{w}_1}{X-\omega^2\zeta} + \cdots + c^*_n\cdot\frac{\hat{w}_n}{X-\omega^{2^n}\zeta}}{\frac{\hat{w}_0}{X-\omega\zeta} + \frac{\hat{w}_1}{X-\omega^2\zeta} + \cdots + \frac{\hat{w}_n}{X-\omega^{2^n}\zeta}} \tag{56}$$

Here $\hat{w}_j$ are pre-computed values:

$$\hat{w}_j = \prod_{l\neq j}\frac{1}{\omega^{2^j} - \omega^{2^l}} \tag{57}$$

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

$$q_\zeta(X) = \frac{1}{X-\zeta}\cdot l_\zeta(X) \tag{58}$$

8. Compute the commitment of $q_\zeta(X)$, Q_ζ, and simultaneously sample a random number $\rho_q \leftarrow_\$ \mathbb{F}$ as the Blinding Factor for the commitment:

$$Q_\zeta = \mathsf{KZG10.Commit}(q_\zeta(X); \rho_q) = [q_\zeta(\tau)]_1 + \rho_q\cdot[\gamma]_1 \tag{59}$$

Error: Extra close brace or missing open brace

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

$$z_{D_\zeta}(X) = (X-\zeta\omega)\cdots(X-\zeta\omega^{2^{n-1}})(X-\zeta) \tag{60}$$

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

$$q_c(X) = \frac{(c(X) - c^*(X))}{(X-\zeta)(X-\omega\zeta)(X-\omega^2\zeta)\cdots(X-\omega^{2^{n-1}}\zeta)} \tag{61}$$

11. Compute the commitment of $q_c(X)$, Q_c and E_c. Since $c(X)$ doesn't contain any private information, there's no need to add a Blinding Factor:

$$Q_c = \mathsf{KZG10.Commit}(q_c(X)) = [q_c(\tau)]_1 \tag{62}$$

12. Construct Quotient polynomial $q_{\omega\zeta}(X)$ to prove the value of $z(X)$ at $\omega^{-1}\cdot\zeta$:

$$q_{\omega\zeta}(X) = \frac{z(X) - z(\omega^{-1}\cdot\zeta)}{X-\omega^{-1}\cdot\zeta} \tag{63}$$

13. Compute the commitment of $q_{\omega\zeta}(X)$, $Q_{\omega\zeta}$, and simultaneously sample a random number $\rho'_q \leftarrow_\$ \mathbb{F}$ as the Blinding Factor for the commitment:

$$Q_{\omega\zeta} = \mathsf{KZG10.Commit}(q_{\omega\zeta}(X); \rho'_q) = [q_{\omega\zeta}(\tau)]_1 + \rho'_q\cdot[\gamma]_1 \tag{64}$$

$$E_{\omega\zeta} = \rho_z\cdot[1]_1 - \rho'_q\cdot[\tau]_1 + (\omega^{-1}\cdot\zeta\cdot\rho'_q)\cdot[1]_1 \tag{65}$$

14. Send $\big(Q_c, Q_\zeta, E_\zeta, Q_{\omega\zeta}, E_{\omega\zeta}\big)$

Round 4.

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

$$q_\xi(X) = \frac{c(X) - c^*(\xi) - z_{D_\zeta}(\xi) \cdot q_c(X)}{X - \xi} \tag{66}$$

3. The Prover computes and sends the commitment of $q_\xi(X)$, Q_ξ

$$Q_\xi = \mathsf{KZG10.Commit}(q_\xi(X)) = [q_\xi(\tau)]_1 \tag{67}$$

Proof Representation

$9 \cdot \mathbb{G}_1$, $(n+1) \cdot \mathbb{F}$

$$\begin{aligned}\pi_{eval} = \big(&z(\omega^{-1} \cdot \zeta), c(\zeta),\ c(\omega \cdot \zeta), c(\omega^2 \cdot \zeta), c(\omega^4 \cdot \zeta), \ldots, c(\omega^{2^{n-1}} \cdot \zeta), \\ &C_c, C_t, C_z, Q_c, Q_\zeta, E_\zeta, Q_\xi, Q_{\omega\zeta}, E_{\omega\zeta}\big)\end{aligned} \tag{68}$$

Verification Process

1. The Verifier computes C'_a and v'

$$C'_a = C_a + \beta \cdot C_b \tag{69}$$

$$v' = v + \beta \cdot v_b \tag{70}$$

2. The Verifier computes $c^*(\xi)$ using pre-computed Barycentric Weights $\{\hat{w}_i\}$

$$c^*(\xi) = \frac{\sum_i c_i \frac{w_i}{\xi - x_i}}{\sum_i \frac{w_i}{\xi - x_i}} \tag{71}$$

3. The Verifier computes $v_H(\zeta), L_0(\zeta), L_{N-1}(\zeta)$

$$v_H(\zeta) = \zeta^N - 1 \tag{72}$$

$$L_0(\zeta) = \frac{1}{N} \cdot \frac{z_H(\zeta)}{\zeta - 1} \tag{73}$$

$$L_{N-1}(\zeta) = \frac{\omega^{N-1}}{N} \cdot \frac{z_H(\zeta)}{\zeta - \omega^{N-1}} \tag{74}$$

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

$$
\begin{aligned}
C_l = \Big(& (c(\zeta) - c_0) s_0(\zeta) \\
& + \alpha \cdot (u_{n-1} \cdot c(\zeta) - (1 - u_{n-1}) \cdot c(\omega^{2^{n-1}} \cdot \zeta)) \cdot s_0(\zeta) \\
& + \alpha^2 \cdot (u_{n-2} \cdot c(\zeta) - (1 - u_{n-2}) \cdot c(\omega^{2^{n-2}} \cdot \zeta)) \cdot s_1(\zeta) \\
& + \cdots \\
& + \alpha^{n-1} \cdot (u_1 \cdot c(\zeta) - (1 - u_1) \cdot c(\omega^2 \cdot \zeta)) \cdot s_{n-2}(\zeta) \\
& + \alpha^n \cdot (u_0 \cdot c(\zeta) - (1 - u_0) \cdot c(\omega \cdot \zeta)) \cdot s_{n-1}(\zeta) \\
& + \alpha^{n+1} \cdot L_0(\zeta) \cdot (C_z - c_0 \cdot C_a) \\
& + \alpha^{n+2} \cdot (\zeta - 1) \cdot \big(C_z - z(\omega^{-1} \cdot \zeta) - c(\zeta) \cdot C_a\big) \\
& + \alpha^{n+3} \cdot L_{N-1}(\zeta) \cdot (C_z - v') \\
& - v_H(\zeta) \cdot C_t \Big)
\end{aligned}
\tag{75}
$$

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

$$
\begin{aligned}
e(C_l + \zeta \cdot Q_\zeta, [1]_2) &\stackrel{?}{=} e(Q_\zeta, [\tau]_2) + e(E_\zeta, [\gamma]_2) \\
e(C - C^*(\xi) - z_{D_\zeta}(\xi) \cdot Q_c + \xi \cdot Q_\xi, [1]_2) &\stackrel{?}{=} e(Q_\xi, [\tau]_2) \\
e(Z + \zeta \cdot Q_{\omega\zeta} - z(\omega^{-1} \cdot \zeta) \cdot [1]_1, [1]_2) &\stackrel{?}{=} e(Q_{\omega\zeta}, [\tau]_2) + e(E_{\omega\zeta}, [\gamma]_2)
\end{aligned}
\tag{76}
$$

The merged verification only requires two Pairing operations:

$$
\begin{aligned}
P = & \Big(C_l + \zeta \cdot Q_\zeta\Big) \\
& + \eta \cdot \Big(C - C^* - z_{D_\zeta}(\xi) \cdot Q_c + \xi \cdot Q_\xi\Big) \\
& + \eta^2 \cdot \Big(C_z + \zeta \cdot Q_{\omega\zeta} - z(\omega^{-1} \cdot \zeta) \cdot [1]_1\Big)
\end{aligned}
\tag{77}
$$

$$
e\Big(P, [1]_2\Big) \stackrel{?}{=} e\Big(Q_\zeta + \eta \cdot Q_\xi + \eta^2 \cdot Q_{\omega\zeta}, [\tau]_2\Big) + e\Big(E_\zeta + \eta^2 \cdot E_{\omega\zeta}, [\gamma]_2\Big) \tag{78}
$$

3. Optimized Performance Analysis

Proof size: $9\ \mathbb{G}_1 + (n+1)\ \mathbb{F}$

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

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.
- [KZG10] Kate, Aniket, Gregory M. Zaverucha, and Ian Goldberg. "Constant-size commitments to polynomials and their applications." Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16. Springer Berlin Heidelberg, 2010.
- [KT23] Kohrita, Tohru, and Patrick Towa. "Zeromorph: Zero-knowledge multilinear-evaluation proofs from homomorphic univariate commitments." Cryptology ePrint Archive (2023). https://eprint.iacr.org/2023/917

- [PST13] Papamanthou, Charalampos, Elaine Shi, and Roberto Tamassia. "Signatures of correct computation." Theory of Cryptography Conference. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. https://eprint.iacr.org/2011/587
- [ZGKPP17] "A Zero-Knowledge Version of vSQL." Cryptology ePrint Archive (2023). https://eprint.iacr.org/2017/1146
- [XZZPS19] Tiancheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. "Libra: Succinct Zero-Knowledge Proofs with Optimal Prover Computation." https://eprint.iacr.org/2019/317
- [CHMMVW19] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas Ward. "Marlin: Preprocessing zkSNARKs with Universal and Updatable SRS." https://eprint.iacr.org/2019/1047