缺失的协议 PH23-PCS（三）

本文给 PH23-KZG10 协议加上对 Zero-knowledge 的支持。

1. 如何支持 ZK

为了让 PH23-KZG10 协议支持 ZK，我们需要修改两个部分的协议，一是要在 KZG10 子协议中支持 Hiding，即在任何一次 Evaluation 证明中，都不会泄漏除了求值之外的信息；二是确保在 PH23 协议中，不会泄漏 Witness 向量，即 $\vec{a}$ 的信息。

首先我们需要一个 Perfect Hiding KZG10 协议，它可以保证多项式在每一次打开后，都不会泄漏多项式除多项式求值之外的其它信息。下面是 [KT23] 中的的 KZG10 协议，其主要思想来源于 [PST13]，[ZGKPP17]，与 [XZZPS19]。

Hiding KZG10

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

一个多项式 $f(X) \in \mathbb{F}[X]$ 的承诺定义为：

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

根据多项式环的性质，$f(X)$ 可以分解为：

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

那么商多项式的承诺计算如下，同样需要一个 Blinding Factor ${\color{blue}\rho_q}$ 来保护 $q(X)$ 的承诺。

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

同时 Prover 还要计算下面一个额外的 $\mathbb{G}_1$ 元素，用来配平验证公式：

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

那么 Evaluation 证明由两个 $\mathbb{G}_1$ 元素组成：

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

于是，Verifier 可以通过下面的公式来验证：

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

求和证明的 ZK

在 Prover 采用累加多项式 $z(X)$ 来证明求和值的过程中，也会泄漏 $\vec{z}$ 向量的信息，其中也包括了 Witness $\vec{a}$ 的信息。因此，我们需要一个 ZK 版本的求和证明协议。

我们有一个阶为 N 的乘法子群 $H \subset \mathbb{F}$：

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

我们记 $\{L_i(X)\}_{i=0}^{N-1}$ 关于 H 的 Lagrange 多项式，$v_H(X) = X^N - 1$ 是 H 上的消失多项式。

假设有一个 N 个元素的向量 $\vec{a} = (a_0, a_1, \ldots, a_{N-1})$，我们希望证明 $\sum_i a_i = v$。Prover 实现计算了 $\vec{a}$ 的承诺，记为 C_a。

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

Round 1

首先，我们要确定在 $z(X)$ 会被打开几次，比如，$z(X)$ 会被在 ζ 和 $\omega^{-1} \cdot \zeta$ 两处打开。那么我们引入一个随机的多项式：$r(X)$，

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

这个多项式包含四个随机因子。为什么是四个？我们后面会看到。

Prover 然后计算 $r(X)$ 的承诺，并引入一个额外的 Blinding Factor ρ_r：

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

Prover 计算一个新的求和 $\sum_i r_i$:

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

Prover 发送 C_r 与 v_r 给 Verifier。

Round 2

Verifier 发送一个随机挑战数 $\beta \leftarrow_\$ \mathbb{F}$ 给 Prover。

Prover 构造一个新的多项式 $a'(X)$，满足

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

Prover 发送给 Verifier一个混合的求和值 v':

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

这时，Prover 和 Verifier 把求和证明的目标 $\sum_i a_i = v$ 转换成 $\sum_i (a_i + \beta \cdot r_i) = v + \beta \cdot v_r$。

Round 3

Verifier 再发送一个随机数 $\alpha \leftarrow_\$ \mathbb{F}$ 给 Prover。

Prover 构造约束多项式 $h_0(X), h_1(X), h_2(X)$，满足

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

Prover 构造多项式 $h(X)$，满足

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

Prover 计算商多项式 $t(X)$，满足

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

Prover 计算 $z(X)$ 的承诺 C_z，并发送 C_z

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

Prover 计算 $t(X)$ 的承诺 C_t，并发送 C_t

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

Round 4

Verifier 发送随机求值点 $\zeta \leftarrow_{\$} \mathbb{F}$

Prover 构造 商多项式 $q_a(X)$, $q_z(X)$, $q_t(X)$ 与 $q'_z(X)$, 满足

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

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

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

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

Prover 计算四个商多项式的承诺，并引入相应的 Blinding Factor $\rho_{q_a},\rho_{q_z},\rho_{q_t},\rho_{q'_z}$

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

Prover 还要构造四个相应的 Blinding Factor 的承诺，并发送给 Verifier：

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

这里可以看到，在证明过程中，Prover 需要在四个多项式上进行求值，并且这四个多项式的求值都会泄漏 $\vec{a}$ 的信息，因此 Prover 在 Round 1 增加一个包含两个额外随机因子的随机多项式 $r(X)$。这样证明过程中的多项式求值都在 $a'(X)$ 上进行，而非直接对 $a(X)$ 运算求值。

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

Verifier 首先验证下面的等式：

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

其中 $v_H(\zeta)$ 由 Verifier 计算，$h(\zeta)$ 由下面的等式计算：

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

然后 Verifier 验证 $a'(\zeta), z(\zeta), t(\zeta), z(\omega^{-1}\cdot\zeta)$ 正确性：

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

2. ZK-PH23-KZG10 协议（优化版）

下面是完整的支持 Zero-knowledge 的 PH23-KZG10 协议。

Precomputation

1. 预计算 $s_0(X),\ldots,s_{n-1}(X)$ 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. 预计算 $D=(1,\omega,\omega^2,\ldots,\omega^{2^{n-1}})$ 上的 Bary-Centric Weights $\{\hat{w}_i\}$。这个可以加速

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

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

Commit 计算过程

1. Prover 构造一元多项式 $a(X)$，使其 Evaluation form 等于 $\vec{a}=(a_0,a_1,\ldots,a_{N-1})$，其中 $a_i=\tilde{f}(\mathsf{bits}(i))$, 为 $\tilde{f}$ 在 Boolean Hypercube $\{0,1\}^n$ 上的取值。

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

2. Prover 抽样一个随机数 $\rho_a\leftarrow_\$\mathbb{F}$，用来保护 $\vec{a}$ 的承诺。
3. Prover 计算 $\hat{f}(X)$ 的承诺 C_a，并发送 C_a

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

其中 $A_0=[L_0(\tau)]_1, A_1=[L_1(\tau)]_1, A_2=[L_2(\tau)]_1,\ldots,A_{N-1}=[L_{2^{n-1}}(\tau)]_1$ ，在预计算过程中已经得到。

Evaluation 证明协议

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. $\vec{u} = (u_0, u_1, \ldots, u_{n-1})$: 求值点
3. $v = \tilde{f}(u_0, u_1, \ldots, u_{n-1})$: MLE 多项式 $\tilde{f}$ 在 $\vec{X} = \vec{u}$ 处的运算值。

回忆下证明的多项式运算的约束：

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

这里 $\vec{u} = (u_0, u_1, u_2, \ldots, u_{n-1})$ 是一个公开的挑战点。

Round 1.

Prover:

1. 计算向量 $\vec{c}$，其中每个元素 $c_i = \tilde{eq}(\mathsf{bits}(i), \vec{u})$
2. 构造多项式 $c(X)$，其在 H 上的运算结果恰好是 $\vec{c}$ 。

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

3. 计算 $c(X)$ 的承诺 $C_c = [c(\tau)]_1$，并发送 C_c

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

4. 构造一个 Blinding 多项式 $r(X) = r_0 \cdot L_0(X) + r_1 \cdot L_1(X)$，其中 $\{r_0, r_1\} \leftarrow_{\$} \mathbb{F}^2$ 是随机抽样的 Blinding Factor。
5. 计算 $r(X)$ 的承诺 $C_r = [r(\tau)]_1$，并发送 C_r

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

6. 计算 $v_r = \langle \vec{r}, \vec{c} \rangle$，并发送 v_r，其中 $\vec{r}$ 定义如下：

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

Round 2.

Verifier: 发送挑战数 $\alpha, \beta \leftarrow_{\$} \mathbb{F}_p^2$

Prover:

1. 构造关于 $\vec{c}$ 的约束多项式 $p_0(X), \ldots, p_n(X)$

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

2. 把 $\{p_i(X)\}$ 聚合为一个多项式 $p(X)$

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

3. 构造 $a'(X)$ ，并计算 $\langle \vec{a}', \vec{c} \rangle = v'$

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

4. 构造累加多项式 $z(X)$，满足

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

4. 构造约束多项式 $h_0(X), h_1(X), h_2(X)$，满足

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

5. 把 $p(X)$ 和 $h_0(X), h_1(X), h_2(X)$ 聚合为一个多项式 $h(X)$，满足

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

6. 计算 Quotient 多项式 $t(X)$，满足

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

7. 抽样 $\rho_t, \rho_z \leftarrow_\$ \mathbb{F}_p^2$，计算 $C_t = [t(\tau)]_1 + \rho_t \cdot [\gamma]_1$， $C_z = [z(\tau)]_1 + \rho_z \cdot [\gamma]_1$，并发送 C_t 和 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: 发送随机求值点 $\zeta \leftarrow_\$ \mathbb{F}$

Prover:

1. 计算 $s_i(X)$ 在 ζ 处的取值：

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

这里 Prover 可以快速计算 $s_i(\zeta)$ ，由 $s_i(X)$ 的公式得

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

因此 $s_i(\zeta)$ 的值可以通过 $s_{i+1}(\zeta)$ 计算得到，而

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

因此可以得到一个 $O(n)$ 的算法来计算 $s_i(\zeta)$ ，并且这里不含除法运算。计算过程是：
$s_{n-1}(\zeta) \to s_{n-2}(\zeta) \to \cdots \to s_0(\zeta)$ 。

2. 定义求值 Domain D'，包含 $n+1$ 个元素：

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

3. 计算并发送 $c(X)$ 在 D' 上的取值

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

4. 计算并发送 $z(\omega^{-1}\cdot\zeta)$

5. 计算 Linearized Polynomial $l_\zeta(X)$

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

显然，$r_\zeta(\zeta)=0$，因此这个运算值不需要发给 Verifier，并且 $[r_\zeta(\tau)]_1$ 可以由 Verifier 自行构造。

6. 构造多项式 $c^*(X)$，它是下面向量在 $D\zeta$ 上的插值多项式

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

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

Prover 可以利用事先预计算的 D 上的Bary-Centric Weights $\{\hat{w}_i\}$ 来快速计算 $c^*(X)$，

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

这里 $\hat{w}_j$ 为预计算的值：

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

7. 因为 $l_\zeta(\zeta)=0$，所以存在 Quotient 多项式 $q_\zeta(X)$ 满足

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

8. 计算 $q_\zeta(X)$ 的承诺 Q_ζ，并同时抽样一个随机数 $\rho_q \leftarrow_\$ \mathbb{F}$ 作为承诺的 Blinding Factor：

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

Error: Extra close brace or missing open brace

9. 构造 $D\zeta$ 上的消失多项式 $z_{D_\zeta}(X)$

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

10. 构造 Quotient 多项式 $q_c(X)$:

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

11. 计算 $q_c(X)$ 的承诺 Q_c 与 E_c，由于 $c(X)$ 中不含有任何私有信息，所以不需要添加 Blinding Factor:

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

12. 构造 Quotient 多项式 $q_{\omega\zeta}(X)$，用来证明 $z(X)$ 在 $\omega^{-1}\cdot\zeta$ 处的取值：

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

13. 计算 $q_{\omega\zeta}(X)$ 的承诺 $Q_{\omega\zeta}$，并同时抽样一个随机数 $\rho'_q \leftarrow_\$ \mathbb{F}$ 作为承诺的 Blinding Factor：

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

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

14. 发送 $\left(Q_c, Q_\zeta, E_\zeta, Q_{\omega\zeta}, E_{\omega\zeta}\right)$

Round 4.

1. Verifier 发送第二个随机挑战点 $\xi \leftarrow_\$ \mathbb{F}$
2. Prover 构造第三个 Quotient 多项式 $q_\xi(X)$

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

3. Prover 计算并发送 $q_\xi(X)$ 的承诺 Q_ξ

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

证明表示

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

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

验证过程

1. Verifier 计算 C'_a 与 v'

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

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

2. Verifier 计算 $c^*(\xi)$ 使用预计算的 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. Verifier 计算 $v_H(\zeta), L_0(\zeta), L_{N-1}(\zeta)$

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

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

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

4. Verifier 计算 $s_0(\zeta), \ldots, s_{n-1}(\zeta)$，其计算方法可以采用前文提到的递推方式进行计算。
5. Verifier 计算线性化多项式的承诺 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. Verifier 产生随机数 η 来合并下面的 Pairing 验证：

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

合并后的验证只需要两个 Pairing 运算：

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

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

3. 优化性能分析

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