Notes on Virgo-PCS

Virgo 是一个基于 GKR 协议的 zkSNARK 证明系统，与 Libra 不同的是，Virgo 采用了不同的多项式承诺方案，或者论文中称之为 zkVPD -- Verifiable Polynomial Delegation。Virgo-zkVPD 基于源自 STARK 系统的 FRI （Fast Reed-Solomon IOP） 协议，因此是一个不需要可信预设置（Trusted Setup）的证明系统。其安全假设基于信息论与 Hash 函数的抗碰撞的安全假设。

本文介绍 Virgo-PCS 的协议原理和原论文有一些不同的地方。原论文中的 PCS 系统支持任意的多元多项式，而本文仅考虑 MLE 多项式。

1. 协议原理

对于任意的 MLE 多项式，$\tilde{f}(X_0, X_1, \cdots, X_{n-1})$，其可以表示为下面的系数形式（或 Monomial 形式）：

$$\tilde{f}(X_0, X_1, \cdots, X_{n-1}) = \sum_{i=0}^{2^n-1} c_i \cdot X_0^{i_0} X_1^{i_1} \cdots X_{n-1}^{i_{n-1}} \tag{1}$$

其中，$\vec{c} = (c_0, c_1, \cdots, c_{2^n-1})$ 是系数向量，bits$(i) = (i_0, i_1, \cdots, i_{n-1})$ 是 i 的二进制表示（Little Endian）。

那么如果我们考虑如何证明该多项式在一个给定点 $\vec{u} = (u_0, u_1, \cdots, u_{n-1})$ 的运算取值，即 $\tilde{f}(\vec{u}) = v$，那么我们只需要证明下面的「求和式」即可：

$$\sum_{i=0}^{2^n-1} c_i \cdot u_0^{i_0} u_1^{i_1} \cdots u_{n-1}^{i_{n-1}} = v \tag{2}$$

Virgo-PCS 的思路与 PH23-PCS 类似，采用一个 Univariate Sumcheck 协议来证明上面的求和式。即 Prover 先承诺 $\vec{c}$，然后通过下面的 Univariate Sumcheck 公式来证明：

$$c(X) \cdot m(X) = h(X) \cdot v_{\mathbb{H}}(X) + X \cdot g(X) + \frac{v}{N} \tag{3}$$

其中 $\mathbb{H}$ 是有限域 $\mathbb{F}_p$ 中的一个乘法子群，其大小为 $N = |\mathbb{H}| = 2^n$ ，而 $v_{\mathbb{H}} = \prod_{x\in\mathbb{H}}(X - x)$ 是 $\mathbb{H}$ 中的 vaishing polynomial。这里的 $m(X)$ 编码了 $\vec{m} = (m_0, m_1, \cdots, m_{2^n-1})$ 向量：

$$m_i = u_0^{i_0} u_1^{i_1} \cdots u_{n-1}^{i_{n-1}} \tag{4}$$

然后 Prover 计算 $h(X)$ 的 FRI 承诺，并发送给 Verifier。然后 Prover 和 Verifier 通过 FRI 协议来证明下面的 Low Degree 商多项式 $g(X)$ 的存在性：

$$g(X) = \frac{N \cdot c(X) \cdot m(X) - N \cdot h(X) \cdot v_{\mathbb{H}}(X) - v}{N \cdot X} \tag{5}$$

显然，只要我们能证明 $\deg(g) < N - 1$，那么我们就证明了 $\sum_{i=0}^{2^n-1} c_i \cdot m_i = v$。

这里解释下是如何将证明上面的「求和式」转换为证明 $\deg(g) < N - 1$ 。为了证明 $\sum_{i=0}^{2^n-1} c_i \cdot m_i = v$ ，首先可以将 c_i 与 m_i 用多项式 $c(X)$ 与 $m(X)$ 在 $\mathbb{H}$ 上进行编码，进而转换为证明

$$\sum_{x\in\mathbb{H}} c(x) \cdot m(x) = v \tag{6}$$

并且 $c(X) \cdot m(X)$ 的次数为 $N - 1 + N - 1 = 2N - 2$ 。对 $c(X) \cdot m(X)$ 进行分解可以得到

$$c(X)\cdot m(X)=g'(X)+h(X)\cdot v_{\mathbb{H}}(X) \tag{7}$$

其中 $\deg(h(X))<2N-1-N=N-1$ ，$\deg(g'(X))<N$ 。因此「求和式」证明可以转换为证明

$$\sum_{x\in\mathbb{H}}c(x)\cdot m(x)=\sum_{x\in\mathbb{H}}(g'(x)+h(x)\cdot v_{\mathbb{H}}(x))=\sum_{x\in\mathbb{H}}g'(x)=g'(0)\cdot N=v \tag{8}$$

上面倒数第二个等式是由下面这个引理得到的。

> **引理** ([BC99]) 令 $\mathbb{H}$ 为 $\mathbb{F}$ 的乘法陪集，$g(X)$ 是 $\mathbb{F}$ 上一个次数严格小于 $|\mathbb{H}|$ 的单变量多项式，那么 $\sum_{x\in\mathbb{H}}g(x)=g(0)\cdot|\mathbb{H}|$ 。

现在只需要证明 $\deg(g')<N$ 以及 $g'(0)=v/N$ 即可。这可以转换为证明多项式

$$\frac{g'(X)-g'(0)}{X-0}=\frac{g'(X)-v/N}{X-0}=\frac{N\cdot g'(X)-v}{N\cdot X} \tag{9}$$

的次数小于 $N-1$ ，上面的多项式即为

$$g(X)=\frac{N\cdot c(X)\cdot m(X)-N\cdot h(X)\cdot v_{\mathbb{H}}(X)-v}{N\cdot X} \tag{10}$$

至此证明「求和式」就转换为了证明 $\deg(g)<N-1$ 。

这个思路整体上没有问题，但是 Verifier 需要 $O(N)$ 的工作量，因为他要计算 $m(X)$ 在 κ 个抽样点的取值。而 $m(X)$ 必须是一个公开的由 $\vec{u}$ 计算得到的向量。

Virgo 论文认为计算 $m(X)$ 在一个点的取值可以利用一个 GKR 协议，将 Verifier 的计算代理给 Prover，同时通过 GKR 协议能够确保计算的正确性，这样 Verifier 只需要 $O(\log^2(N))$ 的工作量即可。

这个 GKR 电路分为下面四个部分：

1. 根据 $\vec{u}$ 计算 $\vec{m}$ 的值
2. 根据 $\vec{m}$ 通过 IFFT 算法计算得到 $m(X)$ 的系数向量
3. 根据 $m(X)$ 的系数向量计算 $m(X)$ 在 Extended Domain $\mathbb{L}$ 上的取值。
4. 根据 Verifier 提供的 FRI-Query 索引集合 Q，过滤出 $m(X)$ 在 $\{\mathbb{L}_i\}_{i\in Q}$ 上的取值。

由于这个 GKR 协议的所有计算都是基于公开值，并且电路的输入长度为 $n=\log(N)$，并且电路的深度为 $\log(N)$，电路的宽度为 $|\mathbb{L}|=N/\rho$，因此 Verifier 的计算量仅为 $O(\log^2(N))$ 即可完成验证。

2. 简化协议

协议参数

1. Domain $\mathbb{H}$ 为素数域 $\mathbb{F}_p$ 的乘法子群，大小为 $N=2^n$。
2. Extended Domain $\mathbb{L}\subset\mathbb{F}_p$ 为大小为 $|\mathbb{L}|=N/\rho$ 的乘法子群 Coset，其中 ρ 表示码率。

承诺计算

Prover 计算 $\tilde{f}(X_0,X_1,\cdots,X_{n-1})$ 的承诺值 $C_{\tilde{f}}$ 与一般的 FRI 协议类似，计算该 MLE 多项式所对应的 Unviariate 多项式 $c(X)$ 在 $\mathbb{L}$ 上的取值。

$$C_{\tilde{f}}=\mathsf{MerkleTree.Commit}(\vec{c}) \tag{11}$$

Evaluation 证明

公开输入

1. $C_{\tilde{f}}$
2. $\vec{u}$
3. $v = \tilde{f}(\vec{u})$

Round 1.

1. 计算 $\vec{m}$，并构造 $m(X)$，其 Evaluation 为 $\vec{m}$

$$m(X) = m_0 \cdot L_0(X) + m_1 \cdot L_1(X) + \cdots + m_{N-1} \cdot L_{N-1}(X) \tag{12}$$

2. 构造 $h(X)$，并计算其承诺 C_h ，其中 $h(X)$ 满足下面这样的等式

$$h(X) = \frac{c(X) \cdot m(X) - X \cdot g(X) - v/N}{v_{\mathbb{H}}(X)} \tag{13}$$

$$C_h = \mathsf{MerkleTree.Commit}(h|_{\mathbb{L}}) \tag{14}$$

Round 2.

Prover 和 Verifier 通过 FRI 协议证明 $g(X)$ 的存在性。在协议的 Query 阶段，Verifier 提供一个索引集合 Q，Prover 计算 $c(X)$，$h(X)$ 在 $\{x_i\}_{i\in Q}$ 上的取值：

$$\{(c(x_i), \pi_c(x_i))\} \leftarrow \mathsf{MerkleTree.Open}(i, c(X)|_{\mathbb{L}}), \quad \forall i \in Q \tag{15}$$

$$\{(h(x_i), \pi_h(x_i))\} \leftarrow \mathsf{MerkleTree.Open}(i, h(X)|_{\mathbb{L}}), \quad \forall i \in Q \tag{16}$$

这里所有的 x_i 都是 $\mathbb{L}$ 中的元素。

Round 3.

Verifier 验证 $\{c(x_i), h(x_i)\}_{i\in Q}$ 的正确性。

$$\mathsf{MerkleTree.Verify}(C_f, i, c(x_i), \pi_c(x_i)) \stackrel{?}{=} 1, \quad \forall i \in Q \tag{17}$$

$$\mathsf{MerkleTree.Verify}(C_h, i, h(x_i), \pi_h(x_i)) \stackrel{?}{=} 1, \quad \forall i \in Q \tag{18}$$

Round 4.

Prover 和 Verifier 运行 GKR 协议，计算 $m|_{\mathbb{L}}$ 的取值，并输出 $\{m|_{x_i}\}_{i\in Q}$ 的取值，这里 x_i 是乘法子群 $\mathbb{L}$ 中第 i 个元素。

Round 5.

Verifier 通过 $\{c(x_i), h(x_i), m(x_i)\}_{i\in Q}$ 来验证 FRI 协议中的每一步折叠的正确性。

3. 支持 Zero-Knowledge

为了支持 Zero-Knowledge 性质，Virgo 在两个地方引入了随机数：

1. 在 $c(X)$ 的承诺中加入了一个 Mask 多项式 $r(X)$
2. 在 Univariate Sumcheck 协议中，引入了一个随机的多项式 $s(X)$，在验证 $\sum_{x_i \in \mathbb{H}} c(x_i)m(x_i) = v$ 时，同时证明 $\sum_{x_i \in \mathbb{H}} s(x_i) = v'$。

承诺计算

Prover 抽样一个随机的多项式 $r(X)$，其 Degree 为 $\kappa - 1$，即包含有 κ 个随机数。

$$c^*(X) = c(X) + v_{\mathbb{H}}(X) \cdot r(X) \tag{19}$$

$$C_{\tilde{f}} = \mathsf{MerkleTree.Commit}(c^*(X)|_{\mathbb{L}}) \tag{20}$$

显然，$\deg(c^*(X)) = N + \kappa - 1$。

Evaluation 证明

公开输入

1. $C_{\tilde{f}}$
2. $\vec{u}$
3. $v = \tilde{f}(\vec{u})$

Witness

1. MLE 多项式 $\tilde{f}(X_0, X_1, \cdots, X_{n-1})$ 的系数向量 $\vec{c}$
2. 随机多项式 $r(X)$

Round 1.

1. Prover 计算 $\vec{m}$，并构造 $m(X)$，其 Evaluation 为 $\vec{m}$

$$m(X) = m_0 \cdot L_0(X) + m_1 \cdot L_1(X) + \cdots + m_{N-1} \cdot L_{N-1}(X) \tag{21}$$

2. Prover 抽样一个多项式 $s(X)$，其 Degree 为 $2N + \kappa - 1$，其在 $\mathbb{H}$ 上的求和为 v'

$$\sum_{a \in \mathbb{H}} s(a) = v' \tag{22}$$

3. Prover 计算 $s(X)$ 的承诺

$$C_s = \mathsf{MerkleTree.Commit}(s|_{\mathbb{L}}) \tag{23}$$

Round 2.

1. Verifier 提供一个随机数 α，用来聚合两个不同的 Sumcheck 协议的和。

$$v^* = v + \alpha \cdot v' \tag{24}$$

2. Prover 构造 $h(X)$，并计算其承诺 C_h ，$h(X)$ 满足

$$h(X) = \frac{c^*(X) \cdot m(X) + \alpha \cdot s(X) - X \cdot g(X) - v^*/N}{v_{\mathbb{H}}(X)} \tag{25}$$

$$C_h = \mathsf{MerkleTree.Commit}(h|_{\mathbb{L}}) \tag{26}$$

Round 3.

Prover 和 Verifier 通过 FRI 协议来证明 $g(X)$ 的 Degree 小于 $N-1$。这包括 $O(\log(N))$ 轮次的 Split-and-fold 折叠。

Round 4.

1. Verifier 抽样 κ 个随机索引，Q，并要求 Prover 提供 $c^*(X)$，$s(X)$ 与 $h(X)$ 在 $\{a_i\}_{i\in Q}$ 上的取值。这里 $a_i \in \mathbb{L}$，是 $\mathbb{L}$ 中的第 i 个元素。
2. Prover 发送 $c^*(X)$，$s(X)$ 与 $h(X)$ 在 $\{a_i\}_{i\in Q}$ 上的取值，并附加上 Merkle path。

$$\{(c^*(a_i), \pi_{c^*}(a_i))\} \leftarrow \mathsf{MerkleTree.Open}(i, c^*(X)|_{\mathbb{L}}), \quad \forall i \in Q \tag{27}$$

$$\{(s(a_i), \pi_s(a_i))\} \leftarrow \mathsf{MerkleTree.Open}(i, s(X)|_{\mathbb{L}}), \quad \forall i \in Q \tag{28}$$

$$\{(h(a_i), \pi_h(a_i))\} \leftarrow \mathsf{MerkleTree.Open}(i, h(X)|_{\mathbb{L}}), \quad \forall i \in Q \tag{29}$$

Round 5.

Prover 和 Verifier 运行 GKR 协议，计算 $m|_{\mathbb{L}}$ 的取值，并输出 $\{m|_{a_i}\}_{i\in Q}$ 的取值，这里 a_i 是乘法子群 $\mathbb{L}$ 中第 i 个元素。

Verification

1. Verifier 验证 $\{c^*(a_i), s(a_i), h(a_i)\}_{i\in Q}$ 的正确性。

$$\mathsf{MerkleTree.Verify}(C_f, i, c^*(a_i), \pi_{c^*}(a_i)) \stackrel{?}{=} 1, \quad \forall i \in Q \tag{30}$$

$$\mathsf{MerkleTree.Verify}(C_s, i, s(a_i), \pi_s(a_i)) \stackrel{?}{=} 1, \quad \forall i \in Q \tag{31}$$

$$\mathsf{MerkleTree.Verify}(C_h, i, h(a_i), \pi_h(a_i)) \stackrel{?}{=} 1, \quad \forall i \in Q \tag{32}$$

2. Verifier 通过 $\{c^*(a_i), s(a_i), h(a_i), m(a_i)\}_{i\in Q}$ 来验证 FRI 协议中的每一步折叠的正确性。

4. 总结

Virgo-PCS 是最早利用 MLE-to-Univariate 的思路来构造多项式承诺的协议。也是最早采用 Small Field，Mersenne-61 素数域来提高性能的协议。虽然 Virgo-PCS 协议要求 MLE 多项式以 Coefficient 形式给出，但是如果我们只考虑 MLE 多项式的承诺的话，我们可以无需将 MLE 多项式转换到 Coefficient （Monomial Basis）形式，而是直接利用 MLE 多项式的 Evaluation (Lagrange Basis) 形式进行证明。

References

1. [ZXZS19] Jiaheng Zhang, Tiancheng Xie, Yupeng Zhang, and Dawn Song. "Transparent Polynomial Delegation and Its Applications to Zero Knowledge Proof". In 2020 IEEE Symposium on Security and Privacy (SP), pp. 859-876. IEEE, 2020. https://eprint.iacr.org/2019/1482.
2. [PH23] Papini, Shahar, and Ulrich Haböck. "Improving logarithmic derivative lookups using GKR." Cryptology ePrint Archive (2023). https://eprint.iacr.org/2023/1284.

3. [BCRSVW19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. "Aurora: Transparent succinct arguments for R1CS." 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 38. Springer International Publishing, 2019. https://eprint.iacr.org/2018/828.
4. [BC99] Byott, Nigel P., and Robin J. Chapman. "Power sums over finite subspaces of a field." *Finite Fields and Their Applications* 5, no. 3 (1999): 254-265.