密码学 zk 系列

第 9 课：zkSnark-Plonk 证明系统

lynndell 博士

新火科技 密码学专家 lynndell2010@gmail.com

目录

密码学基础系列
1. **对称加密与哈希函数**
2. **公钥加密与数字签名**
3. **RSA、环签名、同态加密**
4. **承诺、零知识证明、BulletProof 范围证明、Diffie-Hellman 密钥协商**

门限签名系列

5. **Li17 两方签名与密钥刷新**
6. **GG18 门限签名**
7. **GG20 门限签名**

zk 系列

8. **Groth16 证明系统**
9. **Plonk 证明系统**
10. **UltraPlonk 证明系统**
11. **SHA256 查找表技术**
12. **Halo2 证明系统**
13. **zkSTARK 证明系统**

Plonk 证明系统 = 电路 + 多项式承诺

电路：任意算法多项式化。
多项式承诺：发送方证明多项式是正确的，验证方检测多项式是否正确。

电路：Plonk 标准门；UltraPlonk 对**电路门**进行优化（查找表与定制门）
多项式承诺：**深度讲解**。

1. 密码学承诺

Schwartz–Zippel 引理：令P为有限域$\mathbb{F}$上的多项式$P = F(x_1,\dots,x_n)$，其阶为d。令S为有限域$\mathbb{F}$的子集，从S中选择随机数$r_1,\dots,r_n$，则多项式等于零的概率可忽略，即

$$\Pr[P(r_1,\dots,r_n) = 0] \le \frac{d}{|S|}$$

在单变量情况下，等价于多项式的阶为d，则最多有d个根。

1.1 承诺概念

第 1 步：承诺：选择 x，计算 $y = f(x)$，发送函数值 y；

第 2 步：完全打开：发送原象 x；

第 3 步：校验： $y == f(x)$；

第 4 步：打开 n 个随机点；
第 5 步：校验 n 个随机点；

对函数是有一定要求：
- 函数求逆具有 **NP 困难**，需要暴力搜索，需要指数时间。
- 但是**校验简单**，降低 gas 费。
- 第 4/5 步骤代替第 2/3 步骤。根据 Schwartz-Zippel 引理，攻击者可作弊的概率可忽略。

普通的密码学承诺是第 1/2/3 步骤；
多项式承诺是第 1/4/5 步骤，从概率角度确保多项式是正确的，节约验证复杂度和通信复杂度。

1.2 哈希承诺

- **承诺**：广播哈希值 y
- **完全打开**：广播原象 x
- **校验**：验证一致性 y==hash(x)

1.3 Merkle 承诺与 Merkle 证明

- **承诺：**发送 Merkle root；
- **打开 1 个随机点：**发送叶子节点 x_i 和 path_i；
- **校验：**校验 $root == Merkle(x_i, path_i)$，且检查 root 在以太坊合约上。

问题：证明方需要证明其知道每个叶子节点的值 $x_i, i = 0, ..., 2^n$。

树高度为 100，一共有 2^{100} **个点。**

低效做法：完全打开

- **承诺：**发送 Merkle root
- **完全打开：**发送**所有**叶子节点 $x_i, i = 0, ..., 2^n$
- **校验：**校验 $root == Merkle(x_0, ..., x_{2^n})$，且检查 root 在以太坊合约上。

高效做法：检测 n 个点即可，没必要完全打开。

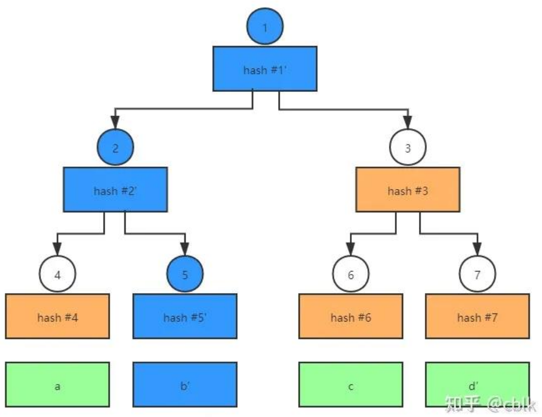
For i=0,i++,i<=100 {

- **承诺：**发送 Merkle root
- **打开 100 个随机点：**发送叶子节点 x_i 和 path_i
- **校验：**校验 $root == Merkle(x_i, path_i)$，且检查 root 在以太坊合约上。

}

发送数据长度和校验复杂度均非常低。

核心思想：从概率角度，不必完全打开每个叶子节点，打开 n 个点，n 次都正确，则伪造成功概率呈指数降低。验证方认可证明方知道所有叶子节点。



Merkle 证明

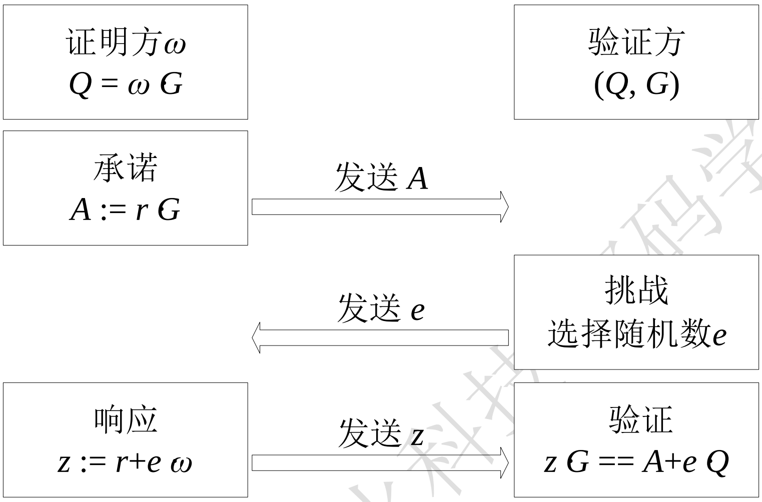
- 第 1 步：Merkle root1 存储在一层合约中；
- 第 2 步：证明方证明知道 **sk**，则**能花费该 token**。发送 Tx 和 path_i。
- 第 3 步：验证方根据 Tx 和 path_i 计算 root2，且该 root2 等于以太坊上的 root1。

1.4 Sigma 零知识证明中的承诺

Sigma 零知识证明：知道秘密 ω，且与公开输入 Q 满足离散对数关系 $Q = \omega \cdot G$ 。

1：（**承诺**）P 选择随机数 r，计算 $A = r \cdot G$，发送**承诺** A；
2：（**挑战**）V 发送随机数 e；
3：（**响应**）P 计算 $z = r + e \cdot \omega$， 发送 z；
4：（**验证**）V 校验 $z \cdot G == A + e \cdot Q$ 。

承诺使用随机数 r，但是没打开， 而是基于随机数 r 和秘密 ω 进行计算响应 z，用于隐藏秘密 ω。发送响应值 z，不会泄露秘密 ω.



1.5.Pedersen 承诺

初始化：椭圆曲线生成元为 G, H，标量域为 F_r，基域为 F_q 。

第 1 步：承诺：对金额 m 和随机数 r，计算 $P := m \cdot G + r \cdot H$，发送 P；
第 2 步：完全打开：发送 m 和 r;
第 3 步：校验： $P == m \cdot G + r \cdot H$；

同态性：
初始状态 Alice 的余额承诺是 0.

用户 1 承诺： $P_1 = m_1 \cdot G + r_1 \cdot H$，用户 1 告诉 Alice 支付了 m1 和随机数为 r1;

用户 2 承诺： $P_2 = m_2 \cdot G + r_2 \cdot H$，用户 2 告诉 Alice 支付了 m2 和随机数为 r2;

矿工更新 Alice 的**余额承诺：** $P_1 + P_2 = (m_1 + m_2) \cdot G + (r_1 + r_2) \cdot H$

Alice 知道 m1+m2 和随机数为 r1+r2，则 Alice 能花费该费用。
zk 证明知道 m1+m2 和随机数为 r1+r2，则能够花费该余额。
Tornado cash 原理。

后门【有毒废料】：

如果 Alice 知道 G,H 之间的离散对数 α **【有毒废料】**， $H=\alpha\cdot G$，则后果很严重。

真实情况：Alice 拥有小金额 m=1 和随机数 r，**资产承诺：** $P=m\cdot G+r\cdot H$ 。

Alice 能够计算 α^{-1}，选择一个大金额 m'=100，计算**随机数** $r'=r-(m'-m)\alpha^{-1}$

将 m',r' 作为完全打开，则矿工校验一致性 $P==m'\cdot G+r'\cdot H$，则成功花费大金额 m' 。

一致性原理如下：

$$\begin{aligned}&m'\cdot G+r'\cdot H\\&=m'\cdot G+\left(r-(m'-m)\alpha^{-1}\right)\cdot H\\&=m'\cdot G+r\cdot H-m'\alpha^{-1}\cdot H+m\alpha^{-1}\cdot H\\&=m\alpha^{-1}\cdot H+r\cdot H\\&=m\cdot G+r\cdot H\\&=P\end{aligned}$$

类似结论：KZG 承诺也是不能知道有毒废料 α 。如果知道，则后果很严重。

2.多项式承诺

- **第 1 步：承诺：选择 x，计算 $y=f(x)$，发送函数值 y；**
- *第 2 步：完全打开：发送 x；*
- *第 3 步：校验： $y==f(x)$；*
- **第 4 步：打开 n 个随机点；**
- **第 5 步：校验 n 个随机点；**

2.1 困难假设

（1）离散对数困难假设：椭圆曲线群 $\mathbb{G}^*$ 的生成元为 G 。有毒废料 $\alpha\in_R\mathbb{Z}_p^*$，已知 $G,\alpha G$，任意多项式时间攻击者 $\mathcal{A}_{DL}$ 能够计算出 α 的概率可忽略

$$\Pr[\mathcal{A}_{DL}(G,\alpha G)=\alpha]\le\varepsilon(\kappa)$$

κ κ为安全参数。$\mathcal{A}_{DL}$ **离散对数攻击者**的英文缩写。攻击者 Adversary。

（2）t 阶强 Diffie-Hellman 假设：已知**有毒废料** $\alpha\in_R\mathbb{Z}_p^*$，公开 $t+1$ 元组 $PK=\langle G,\alpha G,...,\alpha^t G\rangle\in\mathbb{G}^{t+1}$。对任意值 $c\in\mathbb{Z}_p\setminus\{-\alpha\}$，任意多项式时间攻击者 $\mathcal{A}_{t-SDH}$ 能够计算出 $(c,\frac{1}{\alpha+c}\cdot G)$ 的概率可忽略

$$\Pr[\mathcal{A}_{t-SDH}(G,\alpha G,...,\alpha^t G)=(c,\frac{1}{\alpha+c}\cdot G)]\le\varepsilon(\kappa)$$

（3）Q 阶离散对数假设：已知**有毒废料** $\alpha\in_R\mathbb{Z}_p^*$，公开群 $(\mathbb{G}_1,\mathbb{G}_2)$ 上的元组 $PK=\langle G_1,\alpha G_1,...,\alpha^Q G_1;G_2,\alpha G_2,...,\alpha^Q G_2\rangle$。任意多项式时间攻击者 $\mathcal{A}_{QDL}$ 能够计算出 $\alpha\in_R\mathbb{Z}_p^*$ 的概率可忽略

$$\Pr[\mathcal{A}_{QDL}\left(G_1,\alpha G_1,...,\alpha^Q G_1;G_2,\alpha G_2,...,\alpha^Q G_2\right)=\alpha]\le\varepsilon(\kappa)$$

2.2 多项式承诺定义

多项式承诺方案包括：Setup, Commit, **Open, VerifyPoly**, CreateWitness, VerifyEval.
使用第 4/5 步替换第 2/3 步。
打开 n 个点，证明方成功作弊的概率可忽略。

初始化：输入安全参数 κ κ，输出群 $\mathbb{G}^*$ 和用于承诺 t 阶多项式的**有毒废料**和公钥 (α,PK)。后续算法仅使用公钥 PK。**有毒废料** α 删除。

第 1 步：承诺：输入公钥 PK 和多项式 $\phi(x)$，输出承诺 C；（理解为公钥对多项式加密，或根据多项式的系数对公钥计算**离散对数点**）

第 2 步：完全打开：输出多项式 $\phi(x)$ 的系数；（数据量太大，不可取）

第 3 步：校验：验证承诺 C 与公钥 PK 多项式 $\phi(x)$ 满足一致性（验证复杂度高）；

第 4 步：打开 1 个随机点 i：输出 $(i,\phi(i),w_i)$，$w_i=\dfrac{\phi(x)-\phi(i)}{x-i}$ 是商多项式；

第 5 步：校验 1 个随机点：在承诺 C 中验证 $\phi(i)$ 确实在 i 处的值。

核心思想：以下 5 个描述等价
（1） 数据多项式与电路多项式满足运算关系，产生多项式；
（2） 多项式正确；
（3） 多项式的几个随机打开点正确；
（4） 商多项式存在；
（5） 验证方计算双线性映射成立；

2.3 多项式承诺性质

一致性：对于 $PK\leftarrow Setup(1^\kappa),C\leftarrow Commit(PK,\phi(x)),\phi(x)\in\mathbb{Z}_p[x]$

- **完全打开的一致性：**承诺 C 对应的打开信息 $\phi(x)$ 被成功验证

$$VerifyPoly(PK,C,\phi(x))=1$$

相同的输入，肯定是相同的输出。

- **随机打开点的一致性：**输出信息$(i,\phi(i),w_i)$被成功验证

$$VerifyEval(PK,C,i,\phi(i),w_i)=1$$

多项式绑定性（打开绑定性）：对于一个承诺C，任意多项式时间攻击者$\mathcal{A}$输出两个不同的多项式$\phi(x),\phi(x)'$，多项式验证成功**概率可忽略**

$$\Pr\left(\begin{array}{l} PK\leftarrow Setup(1^\kappa),(C,\phi(x),\phi'(x))\leftarrow\mathcal{A}(PK): \\ VerifyPoly(PK,C,\phi(x))=1\wedge \\ VerifyPoly(PK,C,\phi(x)')=1\wedge\phi(x)\neq\phi(x)' \end{array}\right)\leq\varepsilon(\kappa)$$

随机打开点绑定性（打开 100 个点绑定性）：对于一个承诺C，在位置i，任意多项式时间攻击者$\mathcal{A}$输出两个不同的多项式值$\phi(i),\phi(i)'$，求值验证成功概率可忽略

$$\Pr\left(\begin{array}{l} PK\leftarrow Setup(1^\kappa),(C,i,\phi(i),w_i,\phi'(i),w_i')\leftarrow\mathcal{A}(PK): \\ VerifyEval(PK,C,i,\phi(x),w_i)=1\wedge \\ VerifyEval(PK,C,i,\phi(x)',w_i')=1\wedge\phi(i)\neq\phi'(i) \end{array}\right)\leq\varepsilon(\kappa)$$

隐藏性（保密性）：对于多项式$\phi(x)$，已知$PK,C,(i,\phi(i_j),w_j),\ j\in[1,\deg(\phi)]$，且$VerifyEval(PK,C,i_j,\phi(i_j),w_j)=1$，任意多项式时间攻击者$\mathcal{A}$不能在其他索引$\hat{i}$处求多项式的值$\phi(\hat{i})$；

承诺约等于密文，攻击者不能猜到秘密。如果能够猜到秘密，则能作弊。

- 公钥加密（从 PK 的角度开始运算）→ 承诺
- 数字签名（从 sk 的角度开始运算）→ zk 秘密知识 witness 当作 sk

3.KZG 多项式承诺

系统初始化：椭圆曲线双线性群为$\mathcal{G}=(e,\mathbb{G},\mathbb{G}_T)$，**有毒废料**$\alpha\in_R\mathbb{Z}_p^*$，$t+1$元组$\left\langle G,\alpha\cdot G,...,\alpha^t\cdot G\right\rangle\in\mathbb{G}^{t+1}$，令输出为

$$PK=\left(\mathcal{G},G,\alpha\cdot G,...,\alpha^t\cdot G\right)$$

将 setup 分为 2 个集合：第一个集合为 PK，第二个集合为 VK。
这两个集合会**重叠**，不是互斥关系。

初始化专业术语：**common reference string CRS**；**struct reference string SRS**

3.1 情况 1：1 个多项式打开 1 个随机点

承诺：对于**一个多项式** $\phi(x)=\sum_{j=0}^{\deg(\phi)}\phi_j\cdot x^j$，计算多项式承诺

$$C=\phi(\alpha)\cdot G=\sum_{j=0}^{\deg(\phi)}\phi_j\cdot\alpha^jG$$

举例：

初始化：有毒废料 $\alpha=3$ **已删除**，公开信息为 $PK=\langle G,3G,3^2G,3^3G,3^4G,...,3^tG\rangle$

多项式： $\phi(x)=2+4x+6x^2+8x^3+9x^4$

多项式承诺计算过程： $C=\phi(3)\cdot G$

错误方法： 令 $x=3$，则 $\phi(3)=2+4\cdot3+6\cdot3^2+8\cdot3^3+9\cdot3^4$，然后倍点运算 $C=\phi(3)\cdot G$。因为 $\alpha=3$ 已删除，任何人不知道。

正确方法： 从 **PK** 角度计算

$$PK=\langle G,3G,3^2G,3^3G,3^4G,...,3^tG\rangle$$
$$\phi(x)=2+4x+6x^2+8x^3+9x^4$$
$$Dlog=\{2\cdot G,4\cdot(3G),6\cdot(3^2G),8\cdot(3^3G),9\cdot(3^4G)\}$$
$$\phi(3)\cdot G=2\cdot G+4\cdot(3G)+6\cdot(3^2G)+8\cdot(3^3G)+9\cdot(3^4G)$$
$$C=\phi(3)\cdot G$$

完全打开：输出多项式的系数

$$\phi(x)=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot x^j;i=1,...,t$$

验证：验证承诺 C 与公钥 PK、多项式 $\phi(x)$ 均满足一致性。基于多项式 $\phi(x)=\sum_{j=0}^{\deg(\phi)}\phi_j\cdot x^j$ 重新计算承诺

$$C'=\phi(\alpha)\cdot G=\sum_{j=0}^{\deg(\phi)}\phi_j\cdot\alpha^jG$$

如果 $C=C'$，则接受，否则拒绝。

注意：工程中不使用完全打开，而是使用打开一个随机点

打开一个随机点：计算商多项式

$$\varphi_i(x)=\frac{\phi(x)-\phi(i)}{x-i}$$

基于商多项式的系数和 PK，计算**商多项式承诺**

$$W_i = \varphi_i(\alpha)\cdot G$$

输出 $(i,\phi(i),W_i)$。

校验一个随机点： 如果以下等式成立，则接受，否则拒绝

$$e(C,G)=e(W_i,\alpha\cdot G-i\cdot G)\cdot e(G,G)^{\phi(i)}$$

KZG 承诺的 $VK=(G,\alpha G)$。

公式推导如下：

$$\begin{aligned}
&e(W_i,\alpha\cdot G-i\cdot G)\cdot e(G,G)^{\phi(i)}\\
&=e(\varphi_i(\alpha)\cdot G,(\alpha-i)\cdot G)\cdot e(G,G)^{\phi(i)}\\
&=e(\frac{\phi(\alpha)-\phi(i)}{\alpha-i}\cdot G,(\alpha-i)\cdot G)\cdot e(G,G)^{\phi(i)}\\
&=e(G,G)^{\phi(\alpha)-\phi(i)}\cdot e(G,G)^{\phi(i)}\\
&=e(G,G)^{\phi(\alpha)}\\
&=e(\phi(\alpha)\cdot G,G)\\
&=e(C,G)
\end{aligned}$$

反之，如果双线性映射验证成功，则在索引 i，多项式的值确实是 $\phi(i)$。

校验 n 个点，每次都正确，则每次都猜对概率为 $1-\frac{1}{k^n}$，概率可忽略。

关键结论 1：冲要条件：商多项式存在（除得尽）等价于双线性映射成立；
关键结论 2：冲要条件

- **商多项式存在（除得尽）等价于多项式的值 $(i,\phi(i))$ 正确；**
- **商多项式不存在（除不尽）等价于多项式的值 $(i,\phi(i))$ 错误；**

举例：除得尽

情况 1： 如果 $f(x)=ax+b$，则 $\frac{f(x)-f(c)}{x-c}=\frac{ax+b-ac-b}{x-c}=\frac{a(x-c)}{x-c}=a$ 就是求这条直线的斜率，斜率是恒定的，商多项式总是存在。

情况 2： 如果 $f(x)=ax^2+bx+c$，则

$$\frac{f(x)-f(d)}{x-d}=\frac{ax^2+bx+c-ad^2-bd-c}{x-d}=\frac{a(x^2-d^2)+b(x-d)}{x-d}=a(x+d)+b$$

二次多项式除以分母多项式，得到**一次商多项式**，**一次**商多项式总是存在。

情况 3： 如果 $f(x)=ax^3+bx^2+cx+d$，则

$$\frac{f(x)-f(e)}{x-e}=\frac{a(x^3-e^3)+b(x^2-e^2)+c(x-e)}{x-e}$$
$$x^3-e^3=(x-e)(x^2+ex+e^2)$$

三次多项式除以分母多项式，得到二次商多项式，二次商多项式总是存在。

情况 n：对于 n 阶多项式：

$$x^n-1=(x-1)(x^{n-1}+x^{n-2}+...+x+1)$$
$$x^n-a^n=(x-a)(x^{n-1}\cdot a^0+x^{n-2}\cdot a^1+...+x\cdot a^{n-2}+1\cdot a^{n-1})$$
$$x^{2n}-a^{2n}=(x^n-a^n)(x^n+a^n)$$

举例：二次多项式 f(x)=(x-1)(x-2)

情况 1：

f(x)=(x-1)(x-2)：f(1)=0 打开的函数值正确，多项式的求值正确

$$\frac{f(x)-f(1)}{x-1}=\frac{(x-1)(x-2)-0}{x-1}=x-2$$

商多项式存在，说明横坐标 x=1 对应的函数值 f(1)=0 是正确的。

f(x)=(x-1)(x-2)：f(1)=3 打开的函数值错误，多项式的求值错误

$$\frac{f(x)-f(1)}{x-1}=\frac{(x-1)(x-2)-3}{x-1}=\frac{x^2-3x-1}{x-1}$$

商多项式不存在，说明在 x=1 对应的函数值 f(1)=3 是错误的。

情况 2：

f(x)=(x-1)(x-2)：f(3)=2 打开的函数值正确，多项式的求值正确

$$\frac{f(x)-f(3)}{x-3}=\frac{(x-1)(x-2)-2}{x-3}=\frac{x^2-3x}{x-3}=x$$

商多项式存在，说明横坐标 x=3 对应的函数值 f(3)=2 是正确的。

f(x)=(x-1)(x-2)：f(3)=8 打开的函数值错误，多项式的求值错误

$$\frac{f(x)-f(3)}{x-3}=\frac{(x-1)(x-2)-8}{x-3}=\frac{x^2-3x-6}{x-3}$$

商多项式不存在，说明横坐标 x=3 对应的函数值 f(3)=8 是错误的。

情况 3：

f(x)=(x-1)(x-2)：f(11)=90 打开的函数值正确，多项式的求值正确

$$\frac{f(x)-f(11)}{x-11}=\frac{(x-1)(x-2)-90}{x-11}=\frac{x^2-3x-88}{x-11}=\frac{(x-11)(x+8)}{x-11}=x+8$$

商多项式存在，说明横坐标 x=11 对应的函数值 f(11)=90 是正确的。

f(x)=(x-1)(x-2)：f(11)=100 打开的函数值错误，多项式的求值错误

$$\frac{f(x)-f(11)}{x-11}=\frac{(x-1)(x-2)-100}{x-11}=\frac{x^2-3x-98}{x-11}$$

商多项式不存在，说明横坐标 x=11 对应的函数值 f(11)=100 是错误的。

关键结论 2：冲要条件

- **商多项式存在（除得尽）等价于多项式的值 $(i,\phi(i))$ 正确；**
- **商多项式不存在（除不尽）等价于多项式的值 $(i,\phi(i))$ 错误；**

核心思想：以下 5 个描述等价
（1） 数据多项式与电路多项式满足运算关系，产生多项式；
（2） 多项式正确；
（3） 多项式的几个随机打开点正确；
（4） 商多项式存在；
（5） 验证方计算双线性映射成立；

3.2 情况 2：t 个多项式打开 1 个随机点

承诺：对 **t 个多项式** $\phi_i(x)=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot x^j; i=1,....,t$，分别计算 ***t* 个多项式**承诺

$$C_i=\phi_i(\alpha)\cdot G_1=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot\alpha^j G_1$$
$$i=1,...,t$$

注释：计算出 t 个椭圆曲线离散对数点。

完全打开：输出 t 个多项式

$$\phi_i(x)=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot x^j; i=1,...,t$$

验证：验证 t 个承诺 C_i 与公钥 PK、t 个多项式 $\phi_i(x)$ 均满足一致性。基于多项式 $\phi_i(x)=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot x^j; i=1,....,t$ 分别计算 t 个承诺

$$C_i'=\phi_i(\alpha)G_1=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot\alpha^j G_1$$
$$i=1,...,t$$

如果 $C_i=C_i', i=1,....,t$，则接受，否则拒绝。

注意：工程中不使用完全打开，而是使用打开一个随机点。

打开一个随机点：基于上述多项式与承诺，基于 transcript 计算随机数 γ。计算**商多项式**

$$\varphi_z(x)=\sum_{i=1}^{t}\gamma^{i-1}\cdot\frac{\phi_i(x)-\phi_i(z)}{x-z}$$

计算**商多项式承诺**

$$W_z=\varphi_z(\alpha)\cdot G_1$$

输出 $(z,\phi_i(z),W_z),i=1,\dots,t$。

验证一个随机点：同样基于上述多项式与承诺，基于 transcript 计算随机数 γ。分别计算 t 个承诺 C_i 和函数值承诺的累加值

$$F=\sum_{i=1}^{t}\gamma^{i-1}\cdot C_i$$

$$V=\sum_{i=1}^{t}\gamma^{i-1}\phi_i(z)\cdot G_1$$

如果以下等式成立，则接受，否则拒绝

$$e(F-V,G_2)\cdot e(-W_z,\alpha G_2-zG_2)=1$$

公式推导过程如下：

$$\begin{aligned}
&e(F-V,G_2)\cdot e(-W,\alpha G_2-zG_2)\\
&=e\left(\sum_{i=1}^{t}\gamma^{i-1}\cdot\left(\phi_i(\alpha)-\phi_i(z)\right)\cdot G_1,G_2\right)\cdot e(-\varphi_z(\alpha)\cdot G_1,(\alpha-z)G_2)\\
&=e(G_1,G_2)^{\sum_{i=1}^{t}\gamma^{i-1}\cdot(\phi_i(\alpha)-\phi_i(z))}\cdot e(G_1,G_2)^{-\varphi_z(\alpha)(\alpha-z)}\\
&=e(G_1,G_2)^{\sum_{i=1}^{t}\gamma^{i-1}\cdot(\phi_i(\alpha)-\phi_i(z))}\cdot e(G_1,G_2)^{-\sum_{i=1}^{t}\gamma^{i-1}\cdot\frac{\phi_i(\alpha)-\phi(z)}{\alpha-z}(\alpha-z)}\\
&=1
\end{aligned}$$

反之，如果验证成功，则表明索引是 z，t 个多项式的值是 $\phi_i(z),i=1,\dots,t$。

核心思想：以下 5 个描述等价

（6） 数据多项式与电路多项式满足运算关系，产生多项式；

（7） 多项式正确；

（8） 多项式的几个随机打开点正确；

（9） 商多项式存在；

（10） 验证方计算双线性映射成立；

3.3 情况 3：t 个多项式打开多个随机点

承诺： 对t_1个多项式$\phi_i(x)=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot x^j;i=1,....,t_1$，计算$t_1$个多项式承诺

$$C_i=\phi_i(\alpha)\cdot G_1=\sum_{j=0}^{\deg(\phi_i)}\phi_{i,j}\cdot\alpha^j G_1$$
$$i=1,...,t_1$$

对t_2个多项式$\tilde{\phi}_i(x)=\sum_{j=0}^{\deg(\tilde{\phi}_i)}\tilde{\phi}_{i,j}\cdot x^j;i=1,....,t_2$，计算$t_2$个多项式承诺

$$\tilde{C}_i=\tilde{\phi}_i(\alpha)\cdot G_1=\sum_{j=0}^{\deg(\tilde{\phi}_i)}\tilde{\phi}_{i,j}\cdot\alpha^j G_1$$
$$i=1,...,t_2$$

完全打开承诺与验证省略

以下举例是打开 2 个随机点

打开 2 个随机点： 基于上述多项式与承诺，基于 transcript 计算 2 个随机数$\gamma,\tilde{\gamma}$。计算商多项式

$$\varphi_z(x)=\sum_{i=1}^{t_1}\gamma^{i-1}\cdot\frac{\phi_i(x)-\phi_i(z_1)}{x-z_1}$$
$$\tilde{\varphi}_z(x)=\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\cdot\frac{\tilde{\phi}_i(x)-\tilde{\phi}_i(z_2)}{x-z_2}$$

计算 2 个**商多项式承诺**

$$W_{z_1}=\varphi_{z_1}(\alpha)\cdot G_1$$
$$\tilde{W}_{z_2}=\tilde{\varphi}_{z_2}(\alpha)\cdot G_1$$

输出数据

$$(z_1,\phi_i(z_1),W_{z_1}),i=1,...,t_1;$$
$$(z_2,\tilde{\phi}_i(z_2),\tilde{W}_{z_2}),i=1,...,t_2$$

验证 2 个随机点： 同样基于上述多项式与承诺，基于 transcript 计算 2 个随机数$\gamma,\tilde{\gamma}$。**计算随机数**$r\in_R\mathbb{Z}_p^*$，分别计算t_1,t_2**个**承诺和函数值承诺的累加值

$$F_1=\sum_{i=1}^{t_1}\gamma^{i-1}\cdot C_i$$
$$F_2=\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\cdot\tilde{C}_i$$
$$V_1=\sum_{i=1}^{t_1}\gamma^{i-1}\phi_i(z_1)\cdot G_1$$
$$V_2=\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\tilde{\phi}_i(z_2)\cdot G_1$$
$$F=F_1-V_1+r\cdot(F_2-V_2)$$

如果以下等式成立，则接受，否则拒绝

$$e(F+z_1\cdot W_{z_1}+rz_2\cdot W_{z_2},G_2)\cdot e(-W_{z_1}-r\cdot W_{z_2},\alpha\cdot G_2)=1_{G_T}$$

公式推导过程如下：

$$\begin{aligned}
&e(F+z_1\cdot W_{z_1}+rz_2\cdot W_{z_2},G_2)\\
&=e\left(\left(\sum_{i=1}^{t_1}\gamma^{i-1}\cdot(\phi_i(\alpha)-\phi_i(z_1))+r\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\cdot\left(\tilde{\phi}_i(\alpha)-\tilde{\phi}_i(z_2)\right)+z_1\varphi_{z_1}(\alpha)+rz_2\tilde{\varphi}_{z_2}(\alpha)\right)\cdot G_1,G_2\right)\\
&=e(G_1,G_2)^{\left(\sum_{i=1}^{t_1}\gamma^{i-1}\cdot(\phi_i(\alpha)-\phi_i(z_1))+r\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\cdot\left(\tilde{\phi}_i(\alpha)-\tilde{\phi}_i(z_2)\right)+z_1\varphi_{z_1}(\alpha)+rz_2\tilde{\varphi}_{z_2}(\alpha)\right)}
\end{aligned}$$

$$\begin{aligned}
&e(-W_{z_1}-r\cdot W_{z_2},\alpha\cdot G_2)\\
&=e\left(-\varphi_{z_1}(\alpha)-r\tilde{\varphi}_{z_2}(\alpha)\cdot G_1,\alpha\cdot G_2\right)\\
&=e(G_1,G_2)^{-\left(\varphi_{z_1}(\alpha)+r\tilde{\varphi}_{z_2}(\alpha)\right)\cdot\alpha}
\end{aligned}$$

$$\begin{aligned}
&\sum_{i=1}^{t_1}\gamma^{i-1}\cdot(\phi_i(\alpha)-\phi_i(z_1))+r\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\cdot\left(\tilde{\phi}_i(\alpha)-\tilde{\phi}_i(z_2)\right)+z_1\varphi_{z_1}(\alpha)+rz_2\tilde{\varphi}_{z_2}(\alpha)-\left(\varphi_{z_1}(\alpha)+r\tilde{\varphi}_{z_2}(\alpha)\right)\cdot\alpha\\
&=\sum_{i=1}^{t_1}\gamma^{i-1}\cdot(\phi_i(\alpha)-\phi_i(z_1))+r\sum_{i=1}^{t_2}\tilde{\gamma}^{i-1}\cdot\left(\tilde{\phi}_i(\alpha)-\tilde{\phi}_i(z_2)\right)-(a-z_1)\varphi_{z_1}(\alpha)-r(a-z_2)\tilde{\varphi}_{z_2}(\alpha)\\
&=0
\end{aligned}$$

反之，如果验证成功，则表明在索引 z_1,z_2，t_1,t_2 个多项式的值分别是

$\phi_i(z_1),i=1,...,t_1;\tilde{\phi}_i(z_2),i=1,...,t_2$。

核心思想：以下 5 个描述等价

（1） 数据多项式与电路多项式满足运算关系，产生多项式；

（2） 多项式正确；

（3） 多项式的几个随机打开点正确；

（4） 商多项式存在；

（5） 验证方计算双线性映射成立；

KZG 承诺缺点：验证复杂度与多项式个数相关、打开点数相关。

多项式个数越多，验证越复杂；打开点数越多，验证越复杂。

Dan Boneh 承诺好一些：验证复杂度仅与多项式个数相关，与随机打开点数无关。

4.Dan Boneh 承诺（Shplonk）

论文：Efficient polynomial commitment schemes for multiple points and polynomials

完全打开承诺与验证过程与 KZG 承诺相同，**省略。**

Dan Boneh 承诺**多点批量验证**的计算复杂度低于 KZG 承诺，节约 gas 费。

高阶多项式 f 在集合 $S \subset \mathbb{F}$ 上函数值**等于**低阶多项式 $r \in \mathbb{F}_{<|S|}[X]$ 在 $z \in S$ 点上的函数值 $r(z)=f(z)$，即两个阶不同的多项式，函数值有交集。

关键结论 1：条件①等价于条件②

条件①：对于 $|S|$ 个打开点 $z \in S$，两个多项式的值相等 $r(z)=f(z)$；

条件②目标多项式 $Z_S(X)=\prod_{z\in S}(X-z)$ 是多项式 $f(X)-r(X)$ 的因子。

条件① $z \in S$，$r(z)=f(z)$ **等价于条件②** $Q(x):=\dfrac{f(X)-r(X)}{Z_S(X)}$ 是多项式，称为**商多项式；分母称为目标多项式，消失多项式 Vanish Polynomial**

换个角度：令 $g(X)=f(X)-r(X)$，多项式 $g(X)$ 等于零的解为 $z \in S$，则有整除关系 $\dfrac{g(X)-g(z)}{Z_S(X)}$，即商多项式存在 $Q'(x)=\dfrac{g(X)-g(z)}{Z_S(X)}$。

结论（正面）：子集 $S \subset T \subset \mathbb{F}$，多项式 $g \in \mathbb{F}_{<|S|}[X]$，存在以下**充要条件**：

$$Z_S(X)\,|\,g(X) \Leftrightarrow Z_{T\setminus S}(X)\cdot Z_S(X)\,|\,Z_{T\setminus S}(X)\cdot g(X) \Leftrightarrow Z_T(X)\,|\,Z_{T\setminus S}(X)\cdot g(X)$$

条件②：$Z_S(X)\,|\,g(X)$

条件③：$Z_T(X)\,|\,Z_{T\setminus S}(X)\cdot g(X)$

关键结论 2：条件②等价于条件③，是冲要条件

Dan Boneh 承诺使用条件②③。因此，如果②或③成立，则条件①成立。

关键结论 2（反面）：k 个多项式 $F_1,\ldots,F_k \in \mathbb{F}_{<n}[X]$。$Z \in \mathbb{F}_{<n}[X]$ 在域 $\mathbb{F}$ 上能够分解为不同的线性因子。如果 $i \in [k]$，Z 不整除 F_i ，$Z \nmid F_i$，则除去可忽略概率 $k/|\mathbb{F}|$，Z 不整除 G，$Z \nmid G$，其中 $G=\sum_{j=1}^{k}\gamma^{j-1}F_j$ 。

$$Z \nmid F_i \Leftrightarrow Z \nmid \sum_{j=1}^{k}\gamma^{j-1}F_j$$

总结：以下 3 个条件等价

条件① $z \in S$，$r(z) = f(z)$，对于多个打开点，两个多项式的值相等（**验证复杂度与打开点数相关、与多项式个数相关**）

条件② $Z_S(X) | g(X)$，多项式 $Z_S(X) = \prod_{z \in S}(X - z)$ 是 $g(X) = f(X) - r(X)$ 的因子，（验证复杂度**仅与多项式个数相关**）

条件③ $Z_T(X) | Z_{T \backslash S}(X) \cdot g(X)$（验证复杂度**仅与多项式个数相关**，**进一步降低**）

4.1 情况 1：t 个多项式打开 s 个随机点，使用条件②

系统初始化： 双线性群为 $\mathcal{G} = (e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$，有毒废料 $\alpha \in_R \mathbb{Z}_p^*$，$d + t$ 元组 $\left\langle G_1, \alpha G_1, ..., \alpha^{d-1} G_1; G_2, \alpha G_2, ..., \alpha^t G_2 \right\rangle \in (\mathbb{G}_1^d, \mathbb{G}_2^t)$，令输出为

$$PK = \left(\mathcal{G}, G_1, \alpha G_1, ..., \alpha^{d-1} G_1; G_2, \alpha G_2, ..., \alpha^t G_2\right)$$

多项式承诺： 对 t 个多项式 $f_i(x) = \sum_{j=0}^{\deg(f_i)} f_{i,j} \cdot X^j; i = 1, ..., t$，分别计算 t 个多项式承诺

$$C_i = f_i(\alpha) \cdot G_1 = \sum_{j=0}^{\deg(f_i)} f_{i,j} \cdot \alpha^j G_1$$
$$i = 1, ..., t$$

打开 S 个随机点： 基于上述多项式与承诺，基于 transcript 计算随机数 γ。计算商多项式

注意使用条件②： $Z_S(X) | g(X)$

$$h(X) = \sum_{i \in [k]} \gamma^{i-1} \frac{f_i(X) - r_i(X)}{Z_{S_i(X)}}$$

计算并发送 1 个商多项式承诺

$$W = h(\alpha) \cdot G_1$$

验证 S 个随机点： 对于 $i \in [k]$， 如果以下等式成立，则接受，否则拒绝

$$\prod_{i \in [k]} e\left(\gamma^{i-1} \cdot (C_i - r_i(\alpha) \cdot G_1), Z_{T \backslash S_i}(\alpha) \cdot G_2\right) = e\left(W, Z_T(\alpha) \cdot G_2\right)$$

验证复杂度： $(t-1) \cdot \mathbb{G}_1, (t^2 + t) \cdot \mathbb{G}_2, (k+1)\hat{e}$，进行了 k+1 个**双线性映射**

公式推导过程如下：

$$\prod_{i\in[k]} e\left(\gamma^{i-1}\cdot(C_i - r_i(\alpha)\cdot G_1), Z_{T\setminus S_i}(\alpha)\cdot G_2\right)$$

$$=\prod_{i\in[k]} e\left(G_1,G_2\right)^{\gamma^{i-1}\cdot(f_i(\alpha)-r_i(\alpha))\cdot Z_{T\setminus S_i}(\alpha)}$$

$$=e\left(G_1,G_2\right)^{\sum_{i\in[k]}\gamma^{i-1}\cdot(f_i(\alpha)-r_i(\alpha))\cdot Z_{T\setminus S_i}(\alpha)}$$

$$e\left(W, Z_T(\alpha)\cdot G_2\right)$$

$$=e\left(\sum\nolimits_{i\in[k]}\gamma^{i-1}\frac{f_i(\alpha)-r_i(\alpha)}{Z_{S_i(\alpha)}}\cdot G_1, Z_T(\alpha)\cdot G_2\right)$$

$$=e\left(\sum\nolimits_{i\in[k]}\gamma^{i-1}\frac{f_i(\alpha)-r_i(\alpha)}{Z_{S_i(\alpha)}}\cdot G_1, Z_T(\alpha)\cdot G_2\right)$$

$$=e\left(G_1,G_2\right)^{Z_T(\alpha)\cdot\sum_{i\in[k]}\gamma^{i-1}\frac{f_i(\alpha)-r_i(\alpha)}{Z_{S_i(\alpha)}}}$$

反之，如果双线性验证成功，则表明对于索引 $z\in S$**，** $r(z)=f(z)$**。**

承诺了所有值，随机打开点的函数值是正确的，则所有值都是正确的。

优点：验证复杂度仅与多项式个数相关，与随机打开点数无关。

4.2 情况 2：t 个多项式打开 s 个随机点，使用条件③

优势与缺点：增加证明长度，降低验证方的计算复杂度。

系统初始化： 双线性群为 $\mathcal{G}=(e,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T)$，随机数 $\alpha\in_R\mathbb{Z}_p^*$， $d+2$ 元组 $\left\langle G_1,\alpha G_1,...,\alpha^{d-1}G_1;G_2,\alpha G_2\right\rangle\in(\mathbb{G}_1^d,\mathbb{G}_2^2)$，令输出为

$$PK=\left(\mathcal{G},G_1,\alpha G_1,...,\alpha^{d-1}G_1;G_2,\alpha G_2\right)$$

承诺： 对 t 个多项式 $f_i(x)=\sum_{j=0}^{\deg(f_i)}f_{i,j}\cdot X^j; i=1,....,t$，分别计算 t 个多项式承诺

$$C_i=f_i(\alpha)\cdot G_1=\sum\nolimits_{j=0}^{\deg(f_i)}f_{i,j}\cdot\alpha^j G_1$$

$$i=1,...,t$$

打开 S 个随机点： 基于上述多项式与承诺，基于 transcript 计算随机数 γ。计算商多项式

注意使用条件③： $Z_T(X)\,|\,Z_{T\setminus S}(X)\cdot g(X)$

$$h(X)=\frac{\sum_{i\in[k]}\gamma^{i-1}\cdot Z_{T\setminus S_i}(X)\cdot\left(f_i(X)-r_i(X)\right)}{Z_T(X)}$$

计算并发送 1 个商多项式承诺 W_1

$$W_1=h(\alpha)\cdot G_1$$

基于上述多项式承诺计算随机数 z。

区别点：计算一个辅助多项式

$$f_z(X)=\sum_{i\in[k]}\gamma^{i-1}\cdot Z_{T\setminus S_i}(z)\cdot\left(f_i(X)-r_i(z)\right)$$
$$L(X)=f_z(X)-Z_T(z)\cdot h(X)$$

由于 $L(z)=f_z(z)-Z_T(z)\cdot h(z)=0$，则 $(X-z)\,|\,L(X)$。

区别点：计算并发送 1 个辅助多项式的商多项式承诺 W_2

$$W_2=\frac{L(\alpha)}{\alpha-z}\cdot G_1$$

额外发送**一个辅助线性多项式的承诺 W_2。**

一共发送 W_1 和 W_2，数据长度增加。

验证 S 个随机点：基于 transcript 计算随机数 γ。如果以下等式成立，则接受，否则拒绝

$$e\left(\sum_{i\in[k]}\gamma^{i-1}\cdot Z_{T\setminus S_i}(z)\cdot(C_i-r_i(z)\cdot G_1)-Z_T(z)\cdot W_1,G_2\right)=e\left(W_2,(\alpha-z)\cdot G_2\right)$$

仅 2 个双线性映射，**验证复杂度降低。**

优点：验证复杂度仅与多项式个数相关，与随机打开点数无关。

4.3 情况 3：t 个多项式打开 s 个随机点，使用条件③

与 4.2 相同的步骤省略，仅描述不同点。

验证：基于 transcript 计算随机数 γ。如果以下等式成立，则接受，否则拒绝

4.2 节验证等式如下：

$$e\left(\sum_{i\in[k]}\gamma^{i-1}\cdot Z_{T\setminus S_i}(z)\cdot(C_i-r_i(z)\cdot G_1)-Z_T(z)\cdot W_1,G_2\right)=e\left(W_2,(\alpha-z)\cdot G_2\right)$$

本节验证等式如下：

$$e\left(F+z\cdot W',G_2\right)=e\left(W',\alpha\cdot G_2\right)$$

其中，$F=-Z_T(z)\cdot W+\sum_{i\in[k]}\gamma^{i-1}Z_{T\setminus S_i}(z)\cdot C_i-\left(\sum_{i\in[k]}\gamma^{i-1}\cdot Z_{T\setminus S_i}(z)\cdot r_i(z)\right)\cdot G_1$

验证复杂度 k+3 个标量乘（倍点运算）；if k=1，则一共 4 个标量乘。

优势与缺点：证明长度增加，降低验证方的计算复杂度，降低验证 gas 费。

优点：验证复杂度仅与多项式个数相关，与随机打开点数无关。

4.4 情况 4：t 个多项式打开 s 个随机点，使用条件③

仅描述不同点。

随机打开点承诺：与 4.3 节相同点省略。

计算并发送商多项式承诺 W'

4.2 与 4.3 节 $W'=\dfrac{L(\alpha)}{\alpha-z}\cdot G_1$

本节 $W'=\dfrac{L(\alpha)}{Z_{T/S_1}(z)\cdot(\alpha-z)}\cdot G_1$

验证：基于 transcript 计算随机数 γ。如果以下等式成立，则接受，否则拒绝

$$e\left(F+z\cdot W',G_2\right)=e\left(W',\alpha\cdot G_2\right)$$

本节与 4.3 节的区别：

$$F=\frac{-Z_T(z)}{Z_{T/S_1}(z)}\cdot W+\sum_{i\in[k]}\gamma^{i-1}\cdot\frac{Z_{T\setminus S_i}(z)}{Z_{T/S_1}(z)}\cdot C_i-\sum_{i\in[k]}\gamma^{i-1}\cdot\frac{Z_{T\setminus S_i}(z)}{Z_{T/S_1}(z)}\cdot r_i(z)\cdot G_1$$

If k=1, then $F=\dfrac{-Z_T(z)}{Z_{T/S_1}(z)}\cdot W+C-r_1(z)\cdot G_1$

当 k=1 时，仅有 3 个倍点运算，分别为 $zW',\dfrac{-Z_T(z)}{Z_{T/S_1}(z)}\cdot W,r_1(z)\cdot G_1$，计算复杂度降到最低。

归一化优化技术：

$$a,b,c,d \in \mathbb{F}$$

$$ab = cd \Leftrightarrow 1\cdot b = (a^{-1}c)d \Leftrightarrow 1\cdot b = c(da^{-1})$$

$$e(a\cdot G_1, b\cdot G_2) = e(c\cdot G_1, d\cdot G_2)$$

$$e(G_1, b\cdot G_2) = e(ca^{-1}\cdot G_1, d\cdot G_2)$$

$$e(G_1, b\cdot G_2) = e(c\cdot G_1, da^{-1}\cdot G_2)$$

存在问题：Plonk 证明系统是多个多项式多点打开，即 $k>1$。
解决方案：
Fflonk 将多个多项式单点打开等价转化为 1 个多项式多点打开，即 $k=1$。

核心思想：

- 4 个二进制与 1 个 16 进制互相等价表达；
- 1 个 100 维的向量 $\vec{a}$ 与 10 个 10 维向量 $\vec{b}_1,...,\vec{b}_{10}$ 的互相等价表达；
- **n 个相对简单的多项式 f1,...,fn 与 1 个相对复杂的多项式 g 能够等价表达。**

5.Fflonk 多个多项式的组合

令 D 为某个域。

- 向量 $D^{(1)}$ 的每个元素均在域 D 中；
- 向量的向量记为 $D^{(2)}$，每个元素是 $D^{(1)}$；
- 向量的向量的向量记为 $D^{(3)}$，每个元素是 $D^{(2)}$；

元素个数增加速度： $n,n^2,n^4,n^8,...,n^{2^k},k=0,1,2,3,...$

- 有限域向量 $\mathbb{F}^{(1)}$，其元素为 $\bar{S} \in \mathbb{F}^{(1)}$；
- 向量的有限域向量 $\mathbb{F}^{(2)}$，其元素为 $\bar{\bar{S}} \in \mathbb{F}^{(2)}$；
- 向量的向量的有限域向量 $\mathbb{F}^{(3)}$，其元素为 $\bar{\bar{\bar{S}}} \in \mathbb{F}^{(3)}$；

元素个数增加速度： $n,n^2,n^4,n^8,...,n^{2^k},k=0,1,2,3,...$

- t 维向量 $\bar{f} \in D^{t,(1)}$， $\bar{f}$ 的每个元素为 $f_i, 0\le i<t$。
- 向量的向量 $\bar{\bar{f}} \in D^{t,(2)}$， $\bar{\bar{f}}$ 的每个元素为 $\bar{f}_i, 0\le i<|\bar{\bar{f}}|$。

元素个数增加速度： $n,n^2,n^4,n^8,...,n^{2^k},k=0,1,2,3,...$

$\mathbb{F}_{<d}[X]$指多项式$\mathbb{F}[X]$中的元素的阶小于 d。

多项式上的操作

对于向量多项式 $\bar{f} \in \mathbb{F}(X)^t$，且 $x \in \mathbb{F}$。使用 $\bar{f}(x)$ 表达 $\mathbb{F}^{t,(1)}$ 中的向量

$$\bar{f}(x) := \left(f_0(x), \dots, f_{t-1}(x) \right)$$

对于 $\bar{f} \in \mathbb{F}[X]^t$ 和点向量 $\bar{Z} \in \mathbb{F}^l$，使用 $\bar{f}(\bar{Z})$ 表达 $(\mathbb{F}^t)^l$ 中的向量

$$\bar{f}(\bar{Z}) := \left(\bar{f}(Z_j) \right)_{0 \le j < l}$$

对于多项式向量的向量 $\bar{\bar{f}} \in \mathbb{F}[X]^{(2)}$，点向量的向量 $\bar{\bar{Z}} \in \mathbb{F}^{(2)}$，具有 $\left|\bar{\bar{f}}\right| = \left|\bar{\bar{Z}}\right|$，使用 $\bar{\bar{f}}(\bar{\bar{Z}})$ 表达 $\mathbb{F}^{(3)}$ 中的向量

$$\bar{\bar{f}}(\bar{\bar{Z}}) := \left(\bar{f}_i(\bar{Z}_i) \right), 0 < i < \left|\bar{\bar{f}}\right|$$

核心思想：

n 个相对简单的多项式 f1,…,fn 与 1 个相对复杂的多项式 g 能够等价表达。

多项式的组合与分解

定义： t 个多项式的组合 $combine_t(\bar{f}) : \mathbb{F}[X]^t \to \mathbb{F}[X]$

具体映射：t 个多项式 $f_i(X), i = 1, \dots, t$ **组合为一个多项式**

$$g(X) := \sum_{i<t} f_i(X^t) \cdot X^i$$

定义： 一个多项式的分解 $decompose_t(g) : \mathbb{F}[X] \to \mathbb{F}[X]^t$

具体映射： 一个多项式分解为 t 个多项式

$$g(X) := \sum_{i<t} f_i(X^t) \cdot X^i$$

因此，t 个多项式组合为一个多项式，然后再分解为 t 个多项式

$$decompose_t\left(combine_t(\bar{f})\right) == \bar{f}$$

令 $p := |\mathbb{F}|$，对于正整数 $t \mid (p-1)$。

令 $\omega_t \in \mathbb{F}$ 为 **t 次单位根**，即对任意 $i < t$，存在 $\omega_t^t = 1, \omega_t^i \ne 1$。

对于 $z, x \in \mathbb{F}$，且 $z^t = x, z^i \ne x, i < t$，则称 z 为最小整数表达。

定义向量 $roots_t(x) := (z \cdot \omega_t^0, \dots, z \cdot \omega_t^{t-1})$。

对于向量多项式

对于向量 $v \in \mathbb{F}^t$ **和** $x \in \mathbb{F}$**，定义** $v(x) := \sum_{i<t} v_i x^i$ 。

对于向量 $v, S \in \mathbb{F}^{(1)}$**，定义** $v(X) := \left(v(x)\right)_{x \in S}$ 。

对任意 $x \in \mathbb{F}, \bar{S} \in \mathbb{F}^t, \bar{f} \in \mathbb{F}[X]^t$，定义 $\bar{Z} := root_t(x), g := combine_t(\bar{f}), \bar{S}' := \bar{S}(\bar{Z})$

定理：以下 2 个条件等价：

- **t 个多项式** $\bar{f}(X)$ **在 1 个点** x **处求值；**
- **1 个多项式** $g(X)$ **在 t 个点** $\bar{Z}$ **处求值；**

$$\bar{f}(x) = \bar{S} \Leftrightarrow g(\bar{Z}) = \bar{S}'$$

证明：对任意 $z \in \bar{Z}$，

$$g(z) = \sum_{i<t} f_i(z^t) \cdot z^i = \sum_{i<t} f_i(x) \cdot z^i = \bar{f}(x) \cdot (z)$$

所以 $g(\bar{Z}) = \bar{f}(x) \cdot (\bar{Z})$ 。

以下 4 个**等价转换：冲要条件**

1. **右边：** $g(\bar{Z}) = \bar{S}'$
2. $\bar{f}(x) \cdot (\bar{Z}) = \bar{S}'$，且 $\bar{S}' = \bar{S}(\bar{Z})$
3. $\bar{f}(x) \cdot (\bar{Z}) = \bar{S}(\bar{Z})$
4. **左边：** $\bar{f}(x) = \bar{S}$

Fflonk 核心结论：将 n 个多项式 1 点打开等价转化为 1 个多项式 n 点打开

$$g(X) = \sum_{i<t} f_i(X^t) \cdot X^i$$

定理一般化 1：n 个多项式 m 个点打开等价转化为 1 个多项式 n*m 点打开。

举例 1：3 个多项式 f1,f2,f3，在 3 个点打开 x1,x2,x3

第 1 轮

1. 多项式 f1,f2,f3 在 x1 打开等价转化为多项式 g1 在 y1,y2,y3 点打开
2. 多项式 f1,f2,f3 在 x2 打开等价转化为多项式 g1 在 y4,y5,y6 点打开
3. 多项式 f1,f2,f3 在 x3 打开等价转化为多项式 g1 在 y7,y8,y9 点打开

第 2 轮

1 个多项式 g1 在 y1,…,y9 打开，使用 4.4 节的方案仅有 **3 个倍点运算**，验证复杂度很低。

定理一般化 2：

- **n1 个多项式 m1 个点打开；**
- **n2 个多项式 m2 个点打开；**
- **n3 个多项式 m3 个点打开；**

等价转化为 1 个多项式(n1+n2+n3)*(m1+m2+m3)点打开。

举例 2： 更接近 Plonk

多项式 f1,f2 在 2 个点打开 x1,x2

多项式 f3 在 2 个点打开 x2,x3

第 1 轮

1. 多项式 f1,f2,f3(添加)在 x1 打开等价转化为多项式 g1 在 y1,y2,y3 点打开
2. 多项式 f1,f2,f3 在 x2 打开等价转化为多项式 g1 在 y4,y5,y6 点打开
3. 多项式 f1(添加),f2(添加),f3 在 x3 打开等价转化为多项式 g1 在 y7,y8,y9 点打开

第 2 轮

1 个多项式 g1 在 y1,…,y9 打开，使用 4.4 节的方案，仅有 3 个倍点运算，验证复杂度很低。

Plonk 证明系统 = 电路 + 多项式承诺

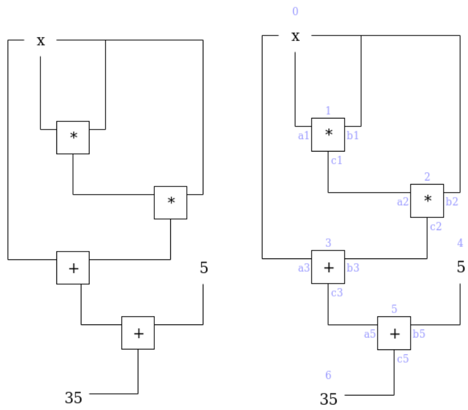
电路化：将任意算法等价转换为多项式。

6.Plonk 证明系统

6.1 电路化：门约束和线约束

任意一个多项式时间的算法 $P(x)=x^3+x+5=35$，求解 x。（$x=3$）

第一步：使用电路对算法 $P(x)$ 进行如图 1 所示表达。如图 2 所示，给元器件和导线添加对应的标号 index。



这个电路包含 2 类：**元器件**（**运算单元**：乘法门和加法门）、**导线**。

- Groth16 基于 R1CS 电路约束构造多项式：a*b=c

(a_1+…+a_1000)(b_1+….b_200)=(c_1+…+c_10)

- Plonk 基于门约束与线约束构造多项式。

R1CS 表达能力很强，Plonk 门表达能力很弱。
对于一个复杂的算法，工程经验：R1CS 电路门为 n 个，则 Plonk 门通常约 8n 至 12n 个。
元器件（运算单元）和导线对数据信号 x 有约束作用。
运算单元（乘法门和加法门）对信号的计算约束作用称为**门约束**，如

$$a \cdot b = c$$
$$a + b = c$$

导线对数据信号的相等约束作用称为**线约束**，如 $c_1 = a_2, c_2 = b_3, c_3 = a_5$。

6.1.1 门约束

创建一个通用方程

$$Q_{L_i} \cdot a_i + Q_{R_i} \cdot b_i + Q_{O_i} \cdot c_i + Q_{M_i} \cdot a_i \cdot b_i + Q_{C_i} = 0$$

方程组中的系数 $Q_{L_i}, Q_{R_i}, Q_{O_i}, Q_{M_i}, Q_{C_i}$ 取特定的值，能够表达特定的加法门、乘法门等。

举例 1：如果需要表达**加法门**，则令：

$$i = 0, Q_{L_0} = Q_{R_0} = 1, Q_{O_0} = -1, Q_{M_0} = Q_{C_0} = 0$$

则通用方程化简为

$$a_0 + b_0 = c_0$$

通用方程成功表达出第 0 个门为**加法门**。
举例 2：如果需要表达**乘法门**，则令：

$$i = 1, Q_{L_1} = Q_{R_1} = Q_{C_1} = 0, Q_{M_1} = 1, Q_{O_1} = -1$$

则通用方程化简为

$$a_1 \cdot b_1 = c_1$$

通用方程成功表达出第 1 个门为**乘法门**。
举例 3：如果需要表达**常量**，则令

$$i = 2, Q_{L_2} = 1, Q_{R_2} = Q_{M_2} = Q_{O_2} = 0, Q_{C_2} = -x$$

则通用方程化简为

$$a_2 = x$$

通用方程成功表达出第 2 个左输入信号 a_2 为常量 x。

因此，**通用方程**能够表达加法门、乘法门、常量门、加法门与乘法门的耦合等。

存在缺点：上述通用方程涉及到大量的离散值。
例如，如果需要 x 个加法门、y 个乘法门、z 个常量门，则有 $x+y+z$ 个**通用方程**，

$Q_{L_i}, Q_{R_i}, Q_{O_i}, Q_{M_i}, Q_{C_i}, a_i, b_i, c_i$ 分别有 $x+y+z$ 个。

解决方案：多项式值表达等价转换为多项式系数表达。
将上述大量的值记为多项式的值，索引 index 为变量 x。有 $x+y+z$ 个值，则对应有 $x+y+z$ 个变量 $[0,...,x+y+z-1]$。使用快速傅里叶变换 FFT，基于多项式的值表达计算多项式的系数表达，**则将大量的值封装到单个多项式中。**

举例：以下 **3 个线性方程（A）**等价于一个多项式方程

$$\begin{aligned} 2x_1 - x_2 + 3x_3 - 8 = 0 \\ x_1 + 4x_2 - 5x_3 - 5 = 0 \\ 8x_1 - x_2 - x_3 + 2 = 0 \end{aligned}$$

横坐标索引 index $x = (0,1,2)$，**纵坐标**分别为：

$$Q_L = (2,1,8)、Q_R = (-1,4,-1)、Q_O = (3,-5,-1)、Q_C = (-8,-5,2)$$

因此，可以使用 FFT 将多项式的值表达计算多项式的系数表达

$$\begin{aligned} Q_L(x) = q_1 + q_2 \cdot x + q_3 \cdot x^2 \\ Q_R(x) = r_1 + r_2 \cdot x + r_3 \cdot x^2 \\ Q_O(x) = o_1 + o_2 \cdot x + o_3 \cdot x^2 \\ Q_C(x) = c_1 + c_2 \cdot x + c_3 \cdot x^2 \end{aligned}$$

可以得到**一个多项式方程**

$$\begin{aligned} &Q_L(x) \cdot x_1 + Q_R(x) \cdot x_2 + Q_O(x) \cdot x_3 + Q_C = 0 \\ &x = (0,1,2) \end{aligned}$$

- 当 $x = 0$ 时，多项式方程表达第 1 个线性方程；
- 当 $x = 1$ 时，多项式方程表达出第 2 个线性方程；
- 当 $x = 2$ 时，多项式方程表达出第 3 个线性方程。

在上面建立的表示约束方程中，x_1, x_2, x_3 变量在每个方程中是相同的。

在 3 个方程中的 x_1, x_2, x_3 可以不同，令 $x_1 = (a_1, a_2, a_3), x_2 = (b_1, b_2, b_3), x_3 = (c_1, c_2, c_3)$，则 **3 个线性方程（B）**变为

$$\begin{aligned} 2a_1 - b_1 + 3c_1 - 8 = 0 \\ a_2 + 4b_2 - 5c_2 - 5 = 0 \\ 8a_3 - b_3 - c_3 + 2 = 0 \end{aligned}$$

横坐标 $x = (0,1,2)$，纵坐标分别为 $(a_1, a_2, a_3), (b_1, b_2, b_3), (c_1, c_2, c_3)$ 则构成三个多项式 $a(x), b(x), c(x)$ 的**值表达**转换为**系数表达后**，可以得到**一般化的多项式约束系统**

$$Q_L(x)\cdot a(x)+Q_R(x)\cdot b(x)+Q_O(x)\cdot c(x)+Q_M(x)\cdot a(x)\cdot b(x)+Q_C(x)=0$$
$$x=(0,1,2)$$

- 电路多项式 $Q_L(x),Q_R(x),Q_O(x),Q_M(x),Q_C(x)$；
- 数据多项式 $a(x),b(x),c(x)$。

因此，完成了**线性运算系统**到**多项式约束系统**的转换。

另一方面，基于横坐标 $x=(0,1,2)$，能够构造出一个公开的**目标多项式** $Z(x)$

$$Z(x)=(x-0)(x-1)(x-2)$$

对于线性方程组

$$\begin{aligned}2x_1-x_2+3x_3-8=0\\x_1+4x_2-5x_3-5=0\\8x_1-x_2-x_3+2=0\end{aligned}$$

1.当 $x=0,x_1=1,x_2=6,x_3=4$，是多项式方程 $Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C=0$ 的解，也是方程 $2x_1-x_2+3x_3-8=0$ 的解。

2.当 $x=1,x_1=1,x_2=6,x_3=4$，是多项式方程 $Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C=0$ 的解，也是方程 $x_1+4x_2-5x_3-5=0$ 的解。

3.当 $x=2,x_1=1,x_2=6,x_3=4$，是多项式方程 $Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C=0$ 的解，也是方程 $8x_1-x_2-x_3+2=0$ 的解。

因此，**多项式约束系统** $Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C(x)=0$ 的解为

$$\begin{aligned}(0,1,6,4)\\(1,1,6,4)\\(2,1,6,4)\end{aligned}$$

基于 $(0,1,2)$ 构造出**目标多项式** $Z(x)$，则以下两个多项式方程等价

$$Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C(x)=0$$
$$\Updownarrow$$
$$Z(x)\cdot H(x)=0$$

理解为：在某个特定域中，门约束成立 $Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C(x)=0$**，则**

所有坐标的函数值等于零 $Z(x)\cdot H(x)=0$。

对多项式约束系统

$$Q_L(x)\cdot x_1+Q_R(x)\cdot x_2+Q_O(x)\cdot x_3+Q_C(x)=Z(x)\cdot H(x)$$

令 $x_1=(a_1,a_2,a_3),x_2=(b_1,b_2,b_3),x_3=(c_1,c_2,c_3)$，得到一般化的多项式约束系统

$$Q_L(x)\cdot a(x)+Q_R(x)\cdot b(x)+Q_O(x)\cdot c(x)+Q_M(x)\cdot a(x)\cdot b(x)+Q_C(x)=Z(x)\cdot H(x)$$

电路多项式 $Q_L(x),Q_R(x),Q_O(x),Q_M(x),Q_C(x)$

秘密数据多项式 $a(x),b(x),c(x)$

秘密数据 $a(x),b(x),c(x)$ 满足电路 $Q_L(x),Q_R(x),Q_O(x),Q_M(x),Q_C(x)$ 约束。

使用多个 **Plonk** 门 $Q_{L_i}\cdot a_i+Q_{R_i}\cdot b_i+Q_{O_i}\cdot c_i+Q_{M_i}\cdot a_i\cdot b_i+Q_{C_i}=0$ 表达所有的运算关系 f。

6.1.2 大数范围内线约束

在电路系统中，上述多项式方程实现了信号在电路门之间的运算约束。
使用**坐标对累加器**表达信号在同一条导线之间的处处相等。
在同一条导线上信号处处相等，则两个相等的信号在运算上是可以进行置换的，所以**线约束**也称为**置换约束**。

核心概念：坐标对累加器

给定多项式 $P(x)$ 的值表达，横坐标 $x=(0,1,2,3)$，纵坐标 $y=(-2,1,0,1)$。注意 $y_1=y_3=1$。

使用 FFT，基于横坐标值 x 计算多项式 X **系数表达**

$$X(x)=x$$

使用 FFT，基于纵坐标值 y 计算多项式 Y **系数表达**

$$Y(x)=x^3-5x^2+7x-2$$

定义：坐标对累加器 $P(x)$ **递归表达**

$$\begin{aligned}&P(n+1)=P(n)\cdot\big(u+X(n)+v\cdot Y(n)\big)\\&P(0)=1\\&n=0,1,2,\ldots\end{aligned}$$

其中，u,v 是两个随机数常量。
因此，基于**递归表达**能够计算出累加器的**通用表达**

$$
\begin{aligned}
P(n) &= P(n-1)\cdot\left(u+X(n-1)+v\cdot Y(n-1)\right) \\
&= P(n-2)\cdot\left(u+X(n-2)+v\cdot Y(n-2)\right)\cdot\left(u+X(n-1)+v\cdot Y(n-1)\right) \\
&= \dots \\
&= P(0)\prod_{i=0}^{n-1}\left(u+X(i)+v\cdot Y(i)\right) \\
&= \prod_{i=0}^{n-1}\left(u+X(i)+v\cdot Y(i)\right)
\end{aligned}
$$

对于映射 $\begin{pmatrix} X(i) \\ \downarrow \\ Y(i) \end{pmatrix} = \begin{pmatrix} x_0,x_1,x_2,x_3 \\ \downarrow \\ y_0,y_1,y_2,y_3 \end{pmatrix} = \begin{pmatrix} 0,1,2,3 \\ \downarrow \\ -2,1,0,1 \end{pmatrix}$，对应的坐标累加值为

$$P(4) = \left(u+x_0+v\cdot y_0\right)\cdot\left(u+x_1+v\cdot y_1\right)\cdot\left(u+x_2+v\cdot y_2\right)\cdot\left(u+x_3+v\cdot y_3\right)$$

对 $P(4)$ 进行**变换（乘法交换律）**，得到 $\tilde{P}(4)$

$$\tilde{P}(4) = \left(u+x_0+v\cdot y_0\right)\cdot\left(u+x_3+v\cdot y_3\right)\cdot\left(u+x_2+v\cdot y_2\right)\cdot\left(u+x_1+v\cdot y_1\right)$$

由于 $y_1 = y_3$， 对 $\tilde{P}(4)$ 进行**替换**，得到 $P(4)'$

$$P(4)' = \left(u+x_0+v\cdot y_0\right)\cdot\left(u+x_3+v\cdot y_1\right)\cdot\left(u+x_2+v\cdot y_2\right)\cdot\left(u+x_1+v\cdot y_3\right)$$

上述过程中

$$P(4) = \tilde{P}(4) = P(4)'$$

$P(4)$ 对应的映射为 $\begin{pmatrix} X(i) \\ \downarrow \\ Y(i) \end{pmatrix} = \begin{pmatrix} x_0,x_1,x_2,x_3 \\ \downarrow \\ y_0,y_1,y_2,y_3 \end{pmatrix} = \begin{pmatrix} 0,1,2,3 \\ \downarrow \\ -2,1,0,1 \end{pmatrix}$

$P(4)'$ 对应的映射为 $\begin{pmatrix} X(i) \\ \downarrow \\ Y(i) \end{pmatrix} = \begin{pmatrix} x_0,x_3,x_2,x_1 \\ \downarrow \\ y_0,y_1,y_2,y_3 \end{pmatrix} = \begin{pmatrix} 0,3,2,1 \\ \downarrow \\ -2,1,0,1 \end{pmatrix}$

因此，如果 $y_1 = y_3$，则改变**横坐标索引 index**，而**坐标对累加器的函数值不变** $P(4) = P(4)'$。

因此，得出以下关键结论 1。

关键结论 1：改变横坐标的索引 $x_i \rightleftarrows x_j$，如果对应纵坐标相等 $y_i = y_j$，则对应的坐标对累加器的函数值不变 $P(n) = P(n)'$，其中 $n = \max\{i, j\}+1$。

关键结论 2：反之，改变横坐标的索引 $x_i \rightleftarrows x_j$，如果坐标对累加器的函数值

相等 $P(n)=P(n)'$，则两个纵坐标 $y_i=y_j$ 相等，确保线约束成立。

冲要条件

证明：上述举例中改变横坐标的**索引** $x_1 \rightleftarrows x_3$，且 $P(4)=P(4)'$，则

$$(u+x_0+v\cdot y_0)\cdot(u+x_1+v\cdot y_1)\cdot(u+x_2+v\cdot y_2)\cdot(u+x_3+v\cdot y_3)$$
$$=$$
$$(u+x_0+v\cdot y_0)\cdot(u+x_3+v\cdot y_1)\cdot(u+x_2+v\cdot y_2)\cdot(u+x_1+v\cdot y_3)$$

由于 u,v 为随机数，根据 **Schwartz-Zippel** 引理和乘积一致性定理得出，除去可忽略概率，上述等式中的各项对应相等。仅有以下 2 中情况，

情况 1：

$$u+x_1+v\cdot y_1=u+x_3+v\cdot y_1$$
$$u+x_3+v\cdot y_3=u+x_1+v\cdot y_3$$

情况 2：

$$u+x_1+v\cdot y_1=u+x_1+v\cdot y_3$$
$$u+x_3+v\cdot y_3=u+x_3+v\cdot y_1$$

情况 1 公式化简后**不成立**。

情况 2 公式化简后得到 $y_1=y_3$，因此**关键结论 2** 成立。

举例：

$$X(i)=(x_0,x_1,x_2,x_3)=(0,1,2,3)$$
$$Y(i)=(y_0,y_1,y_2,y_3)=(-2,1,0,1)$$
$$P(4)=-240$$

$$X(i)'=(x_0,x_3,x_2,x_1)=(0,3,2,1)$$
$$Y(i)'=(y_0,y_1,y_2,y_3)=(-2,1,0,1)$$
$$P(4)'=-240$$

$$P(4)=p(4)'$$

因此，如果 $P(n)=P(n)'$，则验证方认可 $y_i=y_j, n=\max\{i,j\}+1$。因此，通过验证坐标值累加器的相等性，完成两个函数值相等性验证，这就是电路中两个信号相等，即两个信号在同一条导线上。因此，通过坐标值累加器多项式 $P(n)$ 表达出**线约束【置换约束】**。

证明方：需要证明 $y_1=y_3$，则生成**坐标值累加器** $P(4)=p(4)'$；

验证方：验证 $P(4)=p(4)'$，则验证成功。

将 1 条导线上的线约束扩展为 3 条导线上的线约束，则

三条导线 a,b,c 的横坐标分别为 $X_a=(0,1,2,3), X_b=(4,5,6,7), X_c=(8,9,10,11)$，

对应的纵坐标为 $a=(a_0,a_1,a_2,a_3), b=(b_0,b_1,b_2,b_3),, c=(c_0,c_1,c_2,c_3)$。

证明方：证明 $a_2=b_3$，则生成坐标值累加器 $P_a(3),P_a(4)',P_b(3),P_b(4)'$；

验证方：验证 $P_a(3)\cdot P_b(4)=P_a(3)'\cdot P_b(4)'$。

核心思想：1 个 100 维的向量 $\vec{a}$ 与 10 个 10 维向量 $\vec{b}_1,...,\vec{b}_{10}$ 的互相等价表达；

（一维）需要证明：a(1) = a(3)
第一个累加器计算：(0，a(0))，(1，a(1))，(2，a(2))，(3，a(3))，(4，a(4))
第二个累加器计算：(0，a(0))，(3，a(1))，(2，a(2))，(1，a(3))，(4，a(4))
累加器计算结果相同 $P_a(4)=P_a(4)'$，可以推导出 a(1) = a(3)。

电路中总共有**三组连线（三维）**：a/b/c。为表示不同连线之间的置换约束，三组连线采用统一的编号（**一维表达三维**）。
门数为 n，a 的连线编号从 0～n-1，b 的连线编号从 n～2n-1，c 的连线编号从 2n～3n-1，则 Xa(x) = x, Xb(x) = n+x, Xc(x) = 2n+x。
举例：n=5，为证明 a(2) = b(4)
第一个累加器 Xa(x)=0,1,2,3,4; Xb(x)=5,6,7,8,9; Xc(x)=10,11,12,13,14,
第二个累加器 X'a(x) =0,1,9,3,4; X'b(x) = 5,6,7,8,2; X'c(x) = 10,11,12,13,14,
两个累加器计算结果相同 $P(10)=P(10)'$，可以推导出 a(2) = b(4)

$$P_a(5)\cdot P_b(5)=P_a(5)'\cdot P_b'(5)$$

一般化为：

$$P_a(n)\cdot P_b(n)\cdot P_c(n)=P_a(n)'\cdot P_b'(n)\cdot P_c'(n)$$

在实际应用时，涉及计算复杂度较高的**多项式值表达等价转化为多项式系数表达**。为提高变换速度，使用**快速傅里叶变换 FFT**。**横坐标 x** 不使用数字索引 $0,1,2,...n-1$，而应该使用 n 次单位根 $\omega^n=1$ 对应 FFT 或**模 p 原根对应 NTT**。

对于 **n 次单位根**，有以下重要等式

$$Z(x)=(x-\omega^0)(x-\omega^1)(x-\omega^2)...(x-\omega^{n-1})=x^n-1=0$$

坐标值累加器对应修改如下

$$P(n)=\prod_{i=0}^{n-1}\left(u+X(i)+v\cdot Y(i)\right)$$
$$\Downarrow$$
$$P(\omega^n)=\prod_{i=0}^{n-1}\left(u+X(\omega^i)+v\cdot Y(\omega^i)\right)$$

6.1.3 约束汇总

综上，信号在电路中的门约束和线约束如下：

Z(x)·H(x)=0 理解为"在某个特定域中的所有坐标的函数值等于零"。

（1） 门约束：

$$Q_L(x)\cdot a(x)+Q_R(x)\cdot b(x)+Q_O(x)\cdot c(x)+Q_M(x)\cdot a(x)\cdot b(x)+Q_C(x)=Z(x)\cdot H(x)$$

（2） 线约束：门和门之间的连线正确

确定累加的计算在 X 编号变化前后的计算正确。

$$\begin{aligned}
P_a(\omega x)-P_a(x)(u+x+va(x))&=Z(x)H_1(x)\\
P_{a'}(\omega x)-P_{a'}(x)\left(u+\sigma_a(x)+va(x)\right)&=Z(x)H_2(x)\\
P_b(\omega x)-P_b(x)\left(u+g\cdot x+vb(x)\right)&=Z(x)H_3(x)\\
P_{b'}(\omega x)-P_{b'}(x)\left(u+\sigma_b\cdot x+vb(x)\right)&=Z(x)H_4(x)\\
P_c(\omega x)-P_c(x)\left(u+g^2\cdot x+vc(x)\right)&=Z(x)H_5(x)\\
P_{c'}(\omega x)-P_{c'}(x)\left(u+\sigma_c\cdot x+vc(x)\right)&=Z(x)H_6(x)\\
x&=\omega^0,\omega^1,\ldots,\omega^{n-1}
\end{aligned}$$

$P_a(\omega x)-P_a(x)(u+x+va(x))=Z(x)H_1(x)$，理解为某一行数据 a 管脚与下一行数据 a 管脚信号相等，即有一条导线连接。

起始约束和结束约束：

$$P_a(1)=P_b(1)=P_c(1)=P_{a'}(1)=P_{b'}(1)=P_{c'}(1)=1$$
$$P_a(\omega^n)P_b(\omega^n)P_c(\omega^n)=P_{a'}(\omega^n)P_{b'}(\omega^n)P_{c'}(\omega^n)$$

6.2 Plonk 证明系统

6.2.1 门约束

横坐标的取值范围： $X\in\omega^0,\ldots,\omega^{n-1}$

三端口数据 $a(X),b(X),c(X)$ 多项式的门约束系统：

$$Q_L(X)\cdot a(X)+Q_R(X)\cdot b(X)+Q_O(X)\cdot c(X)+Q_M(X)\cdot a(X)\cdot b(X)+Q_C(X)=Z(x)\cdot H(x)$$

6.2.2 乘法群上的线约束

阶为n的**群**$\mathbb{G}$，**生成元为**G。$\{L_i\}_{i\in[n]}$**是群**$\mathbb{G}$上的**拉格朗日基**

$$L_i(j\cdot G)=1, i=j$$
$$L_i(j\cdot G)=0, i\neq j$$

对于 $f,g\in\mathbb{F}_{<n}[X]$，置换$\sigma:[n]\to[n]$，$i\cdot G\in\mathbb{G}, i=1,...,n$。

则记为$g=\sigma(f)$。

$$index=i\cdot G$$
$$index'=\sigma(i)\cdot G$$
$$g(i\cdot G)=f(\sigma(i)\cdot G)$$

多项式预处理：（双方都知道导线多项式的值）在域$\mathbb{F}_{<n}[X]$上的**多项式**

$S_{ID}(g^i)=i, S_\sigma(g^i)=\sigma(i), i\in[n]$；

输入多项式：（证明方知道保密数据多项式） $f,g\in\mathbb{F}_{<n}[X]$；

验证方：选择并发送随机数β,γ；

证明方：多项式$\begin{matrix} f'(i\cdot G)=f(i\cdot G)+\beta\cdot i+\gamma \\ g'(i\cdot G)=g(i\cdot G)+\beta\cdot\sigma(i)+\gamma \end{matrix}$，计算

$$Z(i\cdot G)=\prod_{1\le j\le i}\frac{f'(i\cdot G)}{g'(i\cdot G)}, i=2,...,n$$

其中，$Z(G)=1$。发送**坐标对累加器**的值Z。

验证方：对于**所有**$a\in\mathbb{G}$

$$L_1(a)\big(Z(a)-1\big)=0$$
$$Z(a)f'(a)=g'(a)Z(a\cdot G)$$

公式推导过程如下：

如果$a=1\cdot G$，则$L_1(1\cdot G)=1$，则

$$L_1(G)\big(Z(G)-1\big)=0\Rightarrow Z(G)=1$$

如果$a=i\cdot G, i=2,...,n$，则

$$L_1(i\cdot G)=0$$

所以$L_1(a)\big(Z(a)-1\big)=0$成立。

群 $\mathbb{G}$ 的阶为 n，则 $G=(n+1)\cdot G$，所以

$$
\begin{aligned}
1=Z(G)=Z((n+1)\cdot G)=Z(n\cdot G)\frac{f'(n\cdot G)}{g'(n\cdot G)} \\
=Z((n-1)\cdot G)\frac{f'((n-1)\cdot G)}{g'((n-1)\cdot G)}\frac{f'(n\cdot G)}{g'(n\cdot G)} \\
=\ldots \\
=Z(G)\prod_{i=1}^{n}\frac{f'(i\cdot G)}{g'(i\cdot G)} \\
=\prod_{i=1}^{n}\frac{f'(i\cdot G)}{g'(i\cdot G)} \\
=\prod_{i=1}^{i}\frac{f(i\cdot G)+\beta\cdot i+\gamma}{g(i\cdot G)+\beta\cdot\sigma(i)+\gamma}
\end{aligned}
$$

所以坐标对累加器的值相等：

$$
\begin{aligned}
P=\prod_{i=1}^{i}f(i\cdot G)+\beta\cdot i+\gamma \\
P'=\prod_{i=1}^{i}g(i\cdot G)+\beta\cdot\sigma(i)+\gamma \\
P=P'
\end{aligned}
$$

根据 **Schwartz-Zippel** 引理和乘积一致性定理，除去可忽略概率，上述等式中的**各项对应相等**。因此，在横坐标 index 为 i 的函数值 $g(i\cdot G)$ 等于横坐标 index 为 $\sigma(i)$ 的函数值 $f(\sigma(i)\cdot G)$

$$g(i\cdot G)=f(\sigma(i)\cdot G)$$

所以满足置换关系 $g=\sigma(f)$，实现线约束。

6.2.3 Plonk 核心协议

符号表达： G_1,G_2 分别是群 $\mathbb{G}_1,\mathbb{G}_2$ 的生成元，χ χ

$$
\begin{aligned}
[g]_1=[g(\chi)]_1=[g(X)]\cdot G_1\in\mathbb{G}_1, \\
[g]_2=[g(\chi)]_2=[g(X)]\cdot G_2\in\mathbb{G}_2
\end{aligned}
$$

系统初始化：Plonk 门数为 n，基于有毒废料生成 KZG 的(PK1, VK1)

$$(PK1,VK1)=(\chi\cdot[1]_1,\ldots,\chi^{n+5}\cdot[1]_1)$$

门多项式：

$$(q_{M_i}, q_{A_i}, q_{B_i}, q_{C_i}, q_{Const_i})_{i=1}^{n}, \sigma(X)$$

$$q_M(X) = \sum_{i=1}^{n} q_{M_i} L_i(X)$$

$$q_A(X) = \sum_{i=1}^{n} q_{A_i} L_i(X)$$

$$q_B(X) = \sum_{i=1}^{n} q_{B_i} L_i(X)$$

$$q_C(X) = \sum_{i=1}^{n} q_{C_i} L_i(X)$$

$$q_{Const}(X) = \sum_{i=1}^{n} q_{Const_i} L_i(X)$$

线多项式：

$$S_{\sigma_1}(X) = \sum_{i=1}^{n} \sigma(i) L_i(X)$$

$$S_{\sigma_2}(X) = \sum_{i=1}^{n} \sigma(n+i) L_i(X)$$

$$S_{\sigma_3}(X) = \sum_{i=1}^{n} \sigma(2n+i) L_i(X)$$

底层算法确定了运算门和线，就是 Plonk 的(PK2, VK2)。

对 Plonk，如果需要修改电路，仅修改 VK2，不需要修改 VK1。
对 Groth16，如果要修改电路，则 VK 修改起来很麻烦，计算复杂度很高。

Public input: $l, (w_i)_{i \in [l]}$

Prover input: $(w_i)_{i \in [3n]}$，**witness**

Round 1:【电路门管脚数据多项式承诺】每个电路门的输入三个信号多项式

$$a(X) = (b_1 X + b_2) Z_H(X) + \sum_{i=1}^{n} w_i L_i(X)$$

$$b(X) = (b_3 X + b_4) Z_H(X) + \sum_{i=1}^{n} w_{n+i} L_i(X)$$

$$c(X) = (b_5 X + b_6) Z_H(X) + \sum_{i=1}^{n} w_{2n+i} L_i(X)$$

输出**管脚数据多项式承诺**$[a]_1 = [a(\chi)]_1, [b]_1 = [b(\chi)]_1, [c]_1 = [c(\chi)]_1$

注意：
Rollup 功能，可以去掉随机项，减少计算复杂度，**仅实现数据压缩、计算压缩的功能，缺少零知识。**
zkRollup 功能，则不能去掉随机项。
Zcash 要实现零知识，则不能去掉随机项。

Round 2:【线数据多项式承诺】基于承诺计算随机数 $\beta, \gamma \in \mathbb{F}_p$，计算置换多项式

$$z(X)=(b_7X^2+b_8X+b_9)Z_H(X)$$
$$L_1(X)+$$
$$\sum_{i=1}^{n-1}L_{i+1}(X)\prod_{j=1}^{i}\frac{w_j+\beta\omega^{j-1}+\gamma}{w_j+\sigma(j)\beta+\gamma}\frac{w_{n+j}+\beta k_1\omega^{j-1}+\gamma}{w_{n+j}+\sigma(n+j)\beta+\gamma}\frac{w_{2n+j}+\beta k_2\omega^{j-1}+\gamma}{w_{2n+j}+\sigma(2n+j)\beta+\gamma}$$

输出**线数据多项式承诺** $[z]_1=[z(\chi)]_1$

添加零知识的**其他方法**，在**置换集合**中添加**随机数函数值**。

Round 3:【保密数据满足门约束与线约束】基于上述承诺计算随机数 $\alpha\in\mathbb{F}_p$。

商多项式存在，则确保上述门约束与线约束成立

$t(X)=$

$$\left(q_A(X)\cdot a(X)+q_B(X)\cdot b(X)+q_C(X)\cdot c(X)+q_M(X)\cdot a(X)\cdot b(X)+q_{Const}(X)\right)\frac{1}{Z_H(X)}$$

$$+\Big(\big(a(X)+\beta X+\gamma\big)\big(b(X)+\beta k_1X+\gamma\big)\big(c(X)+\beta k_2X+\gamma\big)z(X)\Big)\frac{\alpha}{Z_H(X)}$$

$$-\Big(\big(a(X)+\beta S_{\sigma_1}(X)+\gamma\big)\big(b(X)+\beta k_1S_{\sigma_2}(X)+\gamma\big)\big(c(X)+\beta k_2S_{\sigma_3}(X)+\gamma\big)z(\omega X)\Big)\frac{\alpha}{Z_H(X)}$$

$$+\big(z(X)-1\big)L_1(X)\frac{\alpha^2}{Z_H(X)}$$

将 $t(X)$ 分解为 3 个多项式

$$t(X)=t_{lo}(X)+X^n\cdot t_{mid}(X)+X^{2n}\cdot t_{hi}(X)$$

其中

$$[t_{lo}]_1=[t_{lo}(\chi)]_1+b_{10}X^n,[t_{mid}]_1=[t_{mid}(\chi)]_1-b_{10}+b_{11}X^n,[t_{hi}]_1=[t_{hi}(\chi)]_1-b_{11}。$$

红色部分是随机项，缺少这部分，则无法实现零知识。

输出**商多项式的承诺** $([t_{lo}]_1,[t_{mid}]_1,[t_{hi}]_1)$。

这个商多项式确保门约束和线约束正确。

Round 4: **【多项式随机打开点】**基于上述承诺计算随机数 $\Im\in\mathbb{F}_p$。计算**多项式的值**

$$\bar{a}=a(\Im),\bar{b}=b(\Im),\bar{c}=c(\Im),$$
$$\bar{s}_{\sigma_1}=S_{\sigma_1}(\Im),\bar{s}_{\sigma_2}=S_{\sigma_2}(\Im),$$
$$\bar{z}_\omega=z(\omega\Im)$$

计算一个辅助的**线性多项式**

$$
\begin{aligned}
r(X) = & \left(\bar{a}\bar{b}q_M(X) + \bar{a}q_A(X) + \bar{b}q_B(X) + \bar{c}q_C(X) + q_{Const}(X)\right) \\
& +\alpha\left(\left(\bar{a}+\beta\Im+\gamma\right)\left(\bar{b}+\beta k_1\Im+\gamma\right)\left(\bar{c}+\beta k_2\Im+\gamma\right)z(X)\right) \\
& -\left(\left(\bar{a}+\beta\bar{s}_{\sigma_1}+\gamma\right)\left(\bar{b}+\beta k_1\bar{s}_{\sigma_2}+\gamma\right)\left(\bar{c}+\beta\cdot S_{\sigma_3}(X)+\gamma\right)\right)\bar{z}_{\omega} \\
& +\alpha^2\left(z(X)-1\right)L_1(\Im) \\
& -Z_H(\Im)\cdot\left(t_{lo}(X)+\Im^n t_{mid}(X)+\Im^{2n}t_{hi}(X)\right)
\end{aligned}
$$

计算**函数值** $\bar{r} = r(\Im)$。

输出**函数值** $\bar{a},\bar{b},\bar{c},\bar{s}_{\sigma_0},\bar{s}_{\sigma_1},\bar{z}_{\omega}$。

Round 5: **【商多项式】**基于上述承诺计算随机数 $v \in \mathbb{F}_p$。

计算 KZG 承诺中的商多项式，确保上述所有多项式正确

$$
W_{\Im}(X) = \frac{1}{X-\Im}\left(\begin{aligned}
& t_{lo}(X)+\Im^n\cdot t_{mid}(X)+\Im^{2n}\cdot t_{hi}(X)-\bar{t} \\
& +v\left(r(X)-\bar{r}\right) \\
& +v^2\left(a(X)-\bar{a}\right) \\
& +v^3\left(b(X)-\bar{b}\right) \\
& +v^4\left(c(X)-\bar{c}\right) \\
& +v^5\left(S_{\sigma_1}(X)-\bar{s}_{\sigma_1}\right) \\
& +v^6\left(S_{\sigma_2}(X)-\bar{s}_{\sigma_2}\right)
\end{aligned}\right)
$$

$$
W_{\omega\Im}(X) = \frac{z(X)-\bar{z}_{\omega}}{X-\omega\Im}
$$

计算并输出商多项式承诺 $[W_{\Im}]_1 = [W_{\Im}(\chi)]_1, [W_{\omega\Im}]_1 = [W_{\omega\Im}(\chi)]_1$。

这个**商多项式存在**，则确保多项式的随机打开点正确，**包括：门多项式、线多项式、门约束与线性约束多项式、目标多项式均正确。**

最终发送数据

$$
\pi_{SNARK} = \left(\begin{aligned}
& [a]_1,[b]_1,[c]_1,[z]_1,[t_{lo}]_1,[t_{mid}]_1,[t_{hi}]_1,[W_{\Im}]_1,[W_{\omega\Im}]_1 \\
& \bar{a},\bar{b},\bar{c},\bar{s}_{\sigma_1},\bar{s}_{\sigma_2},\bar{z}_{\omega}
\end{aligned}\right)
$$

验证密钥 VK 预先存储到以太坊一层合约

算法是公开的，算法原理=电路原理=Plonk 的 VK2

$$[q_M]_1=[q_M(\chi)]_1,[q_A]_1=[q_A(\chi)]_1,[q_B]_1=[q_B(\chi)]_1,[q_C]_1=[q_C(\chi)]_1,$$
$$[S_{\sigma_1}]_1=[S_{\sigma_1}(\chi)]_1,[S_{\sigma_2}]_1=[S_{\sigma_2}(\chi)]_1,[S_{\sigma_3}]_1=[S_{\sigma_3}(\chi)]_1,[\chi]_1$$

- $[\chi]_1$ 是 KZG 承诺 Pairing 所需的 VK1；
- 其余是 Plonk 的电路（门与线）VK2；

验证算法

群元素**合法性**检查$[a]_1,[b]_1,[c]_1,[z]_1,[t_{lo}]_1,[t_{mid}]_1,[t_{hi}]_1,[W_{\Im}]_1,[W_{\omega\Im}]_1\in\mathbb{G}_1^9$

函数值**范围检测**$\bar{a},\bar{b},\bar{c},\bar{s}_{\sigma_1},\bar{s}_{\sigma_2},\bar{z}_{\omega}\in\mathbb{F}_p^6$ **标量域**

L 个**公共输入**范围检测$(w_i)_{i\in[l]}\in\mathbb{F}_p^l$ **标量域**

基于多项式承诺计算随机数$\beta,\gamma,\alpha,\Im,v,u\in\mathbb{F}_p$

多项式求值$Z_H(\Im)=\Im^n-1$

拉格朗日基求值$L_1(\Im)=\dfrac{\omega(\Im^n-1)}{n(\Im-\omega)}$

公共输入多项式求值$PI(\Im)=\sum_{i\in l}w_iL_i(\Im)$

计算一个辅助的线性多项式

$$r_0:=PI(\Im)-L_1(\Im)\alpha^2-\alpha\left(\bar{a}+\beta\bar{s}_{\sigma_1}+\gamma\right)\left(\bar{b}+\beta k_1\bar{s}_{\sigma_2}+\gamma\right)\left(\bar{c}+\gamma\right)\bar{z}_{\omega}$$
$$r'(X):=r(X)-r_0$$

基于多项式的值，计算多项式**函数值的承诺**

$$[D]_1=v\cdot[r]_1+u\cdot[z]_1$$

$$\begin{aligned}[D]_1=&\left(\bar{a}\bar{b}[q_M]_1+\bar{a}[q_A]_1+\bar{b}[q_B]_1+\bar{c}[q_C]_1+[q_{Const}]_1\right)\\&+\left(\left(\bar{a}+\beta\Im+\gamma\right)\left(\bar{b}+\beta k_1\Im+\gamma\right)\left(\bar{c}+\beta k_2\Im+\gamma\right)\alpha v+L_1(\Im)\alpha^2+u\right)\cdot[z]_1\\&-\left(\left(\bar{a}+\beta\bar{s}_{\sigma_1}+\gamma\right)\left(\bar{b}+\beta k_1\bar{s}_{\sigma_2}+\gamma\right)\alpha\beta\bar{z}_{\omega}\right)[S_{\sigma_3}]_1\\&-Z_H(\Im)\left([t_{lo}]_1+\Im^n\cdot[t_{mid}]_1+\Im^{2n}\cdot[t_{lo}]_1\right)\end{aligned}$$

基于多项式函数值，计算多项式函数值的**承诺**的线性组合

$$[F]_1=[D]_1+v[a]_1+v^2[b]_1+v^3[c]_1+v^4[S_{\sigma_1}]_1+v^5[S_{\sigma_2}]_1$$

计算函数值的承诺

$$[E]_1=\left[-r_0+v\bar{a}+v^2\bar{b}+v^3\bar{c}+v^4\bar{s}_{\sigma_1}+v^5\bar{s}_{\sigma_2}+u\bar{z}_{\omega}\right]_1$$

双线性映射验证

$$e\left([W_{\mathfrak{Z}}]_1+u\cdot[W_{\omega\mathfrak{Z}}]_1,[\chi]_2\right)==e\left(\mathfrak{I}\cdot[W_{\mathfrak{Z}}]_1+u\mathfrak{I}\omega\cdot[W_{\omega\mathfrak{Z}}]_1+[F]_1-[E]_1,[1]_2\right)$$

对比：

Groth16：

缺点：CRS 非常复杂（有毒废料+电路承诺，电路固定）

优点：证明简单、验证简单

Plonk：

优点：KZG 的 CRS 非常简单（有毒废料），不包括电路。

缺点：证明复杂很多（电路承诺，电路自由），验证复杂。

Plonk 的优化：

8. To save a verifier scalar multiplication, we split r into its constant and non-constant terms. Compute r's constant term:

$$r_0 := \mathsf{PI}(\mathfrak{z}) - \mathsf{L}_1(\mathfrak{z})\alpha^2 - \alpha(\bar{a}+\beta\bar{\mathsf{s}}_{\sigma1}+\gamma)(\bar{b}+\beta\bar{\mathsf{s}}_{\sigma2}+\gamma)(\bar{c}+\gamma)\bar{z}_\omega,$$

and let $r'(X) := r(X) - r_0$.

9. Compute first part of batched polynomial commitment $[D]_1 := [r']_1 + u\cdot[z]_1$: 2个倍点运算

$$[D]_1 := \begin{array}{l} \bar{a}\bar{b}\cdot[q_M]_1 + \bar{a}\cdot[q_L]_1 + \bar{b}\cdot[q_R]_1 + \bar{c}\cdot[q_O]_1 + [q_C]_1 \quad \text{数据多项式的值与门运算} \\ +\left((\bar{a}+\beta\mathfrak{z}+\gamma)(\bar{b}+\beta k_1\mathfrak{z}+\gamma)(\bar{c}+\beta k_2\mathfrak{z}+\gamma)\alpha + \mathsf{L}_1(\mathfrak{z})\alpha^2+u\right)\cdot[z]_1 \\ -(\bar{a}+\beta\bar{\mathsf{s}}_{\sigma1}+\gamma)(\bar{b}+\beta\bar{\mathsf{s}}_{\sigma2}+\gamma)\alpha\beta\bar{z}_\omega\cdot[s_{\sigma3}]_1 \quad \text{线多项式的值与线运算} \\ -Z_H(\mathfrak{z})([t_{lo}]_1+\mathfrak{z}^n\cdot[t_{mid}]_1+\mathfrak{z}^{2n}\cdot[t_{hi}]_1) \quad \text{门约束与线约束的组合值} \end{array}$$

10. Compute full batched polynomial commitment $[F]_1$:

$$[F]_1 := [D]_1 + \underbrace{v\cdot[a]_1 + v^2\cdot[b]_1 + v^3\cdot[c]_1 + v^4\cdot[s_{\sigma1}]_1 + v^5\cdot[s_{\sigma2}]_1}_{\text{数据多项式与线多项式的承诺}}$$

11. Compute group-encoded batch evaluation $[E]_1$:

$$[E]_1 := \begin{pmatrix} \overset{\text{辅助值}r_0}{-r_0} + v\bar{a} + v^2\bar{b} + v^3\bar{c} \\ +v^4\bar{\mathsf{s}}_{\sigma1} + v^5\bar{\mathsf{s}}_{\sigma2} + u\bar{z}_\omega \end{pmatrix}\cdot[1]_1$$

多项式随机打开点

12. Batch validate all evaluations:

$$e([W_\mathfrak{z}]_1 + u\cdot[W_{\mathfrak{z}\omega}]_1, [x]_2) \stackrel{?}{=} e(\mathfrak{z}\cdot[W_\mathfrak{z}]_1 + u\mathfrak{z}\omega\cdot[W_{\mathfrak{z}\omega}]_1 + [F]_1 - [E]_1, [1]_2)$$

情况 1：KZG 承诺：

- 多个多项式、多点打开：绿色下划线是群 G1 上的 18 个倍点运算。

情况 2：Dan 承诺：

- 与打开点数无关，所以标黄部分合并；
- 左边减少 1 个倍点运算，由证明方提供 $e\left(W',\alpha\cdot G_2\right)$
- 右边减少 1 个，所以一共是 **16 个倍点**运算 $e\left(F+z\cdot W',G_2\right)$

情况 3：Fflonk + Dan 承诺：

- **Fflonk 组合：** n 个多项式 1 个打开点组合为 1 个多项式 n 个打开点；

- **Dan 承诺** 4.4 节的验证等式 $e\left(W_2, x\cdot G_2\right)=e\left(F+z\cdot W_2, G_2\right)$ 仅 3 个倍点运算。第 9 步有 2 个倍点运算，一共 **5 个倍点**运算。

7.聚合证明系统

方案设计:

任意算法 Algorithm_i，对应电路 Circuit_i 和 VK_i；
基于 Plonk **验证算法**对应电路 Circuit0 和 VK0;
对应的验证密钥为 VK0 存储到以太坊一层；

证明方：输入多组 $(Proof_i, VK_i)_{i=1,...,n}$ 到验证电路 Circuit0，生成 $Proof\,0$；

验证方：校验 $Valid\,/\,Invalid \leftarrow Verify_{miller-loop}(Proof\,0, VK0)$，等价于完成对多组 $(Proof_i, VK_i)_{i=1,...,n}$ 的校验。

关键点 1：Plonk 验证算法包含一个双线性映射

在**电路**上表达双线性映射原理，涉及 **millier-loop**。**循环次数**取决于计算出来的随机数，而不是固定值。因此，循环发生变化，导致电路也发生变化，导致 VK' 发生变化。以太坊一层矿工无法使用变化的 VK'。
必须要使用一个固定的验证密钥，对应一个固定的验证算法电路，否则容易作弊。

解决方案：验证电路只验证**固定部分**，不验证双线性映射。把多个双线性映射进行线性组合，耦合到 $Proof\,0$ 中，使得以太坊一层的矿工在进行 1 个双线性映射验证时，则**等价于**把多个的双线性映射的线性组合也验证了。

举例：有 n 个证明需要电路校验，发送数据为 $proof_1, ..., proof_n$

$$\begin{aligned}&proof_1=\{a_1\cdot G_1, b_1\cdot G_2, c_1\cdot G_1, d_1\cdot G_2\}\\&...\\&proof_n=\{a_n\cdot G_1, b_n\cdot G_2, c_n\cdot G_1, d_n\cdot G_2\}\end{aligned}$$

电路校验如下

$$\begin{aligned}&e(a_1\cdot G_1, b_1\cdot G_2)=e(c_1\cdot G_1, d_1\cdot G_2)\\&...\\&e(a_n\cdot G_1, b_n\cdot G_2)=e(c_n\cdot G_1, d_n\cdot G_2)\end{aligned}$$

有 1 个最终 $Proof\,0$ 需要以太坊矿工校验

$$e(x\cdot G_1, y\cdot G_2)=e(j\cdot G_1, k\cdot G_2)$$

修改：计算 2 个随机数 $v_1, \tilde{v}_1$， 如下**线性组合**

$$
\begin{aligned}
P_1 &= v_1^0 a_1 \cdot G_1 + \ldots + v_1^n a_n \cdot G_1 + v_1^{n+1} x \cdot G_1 \\
P_2 &= \tilde{v}_1^0 b_1 \cdot G_2 + \ldots + \tilde{v}_1^n b_3 \cdot G_2 + \tilde{v}_1^{n+1} y \cdot G_1 \\
P_3 &= v_1^0 c_1 \cdot G_1 + \ldots + v_1^n c_3 \cdot G_1 + v_1^{n+1} j \cdot G_1 \\
P_4 &= \tilde{v}_1^0 d_1 \cdot G_2 + \ldots + \tilde{v}_1^n d_3 \cdot G_2 + \tilde{v}_1^{n+1} k \cdot G_1
\end{aligned}
$$

发送数据为 $Proof\,0 = (P_1, P_2, P_3, P_4)$。

以太坊矿工进行最终的**双线性映射**

$$e(P_1, P_2) = e(P_3, P_4)$$

优势：验证电路不需要双线性映射（**miller-loop**），验证复杂度很低，且验证密钥固定。

关键点 2：取值范围

Bn256 曲线标量域小于基域。
电路上的随机数取值范围是标量域；
椭圆曲线点的取值范围是基域。

解决方案：基域和标量域之间的进制转换

人类十进制与计算机二进制互相表达。
使用 2 个标量域表达 1 个基域。

8.任意 for 循环的电路约束

背景：任意 for 循环，每次计算复杂度都不一样，则对应的电路 Circuit_1 每次都发生变化，则对应的 VK_1 每次都发生变化不一样。但是，验证 VK_1 需要提前存到以太坊合约中，矿工才能够验证 Proof 的正确性。
问题：不断变化的 VK_1 无法提前存储到以太坊一层合约中。

解决方案：聚合证明
步骤 1：任意算法（包括 for 循环）的验证电路 Circuit_1 生成 proof_1，对应一个 **VK_1**。
步骤 2：将 proof_1 和 **VK_1** 作为数据，输入到**验证电路 Circuit_2** 中。其中，该验证电路 Circuit_2 那仅表达 plonk 的验证算法的**固定部分**，则对应的 VK_2 是固定值。因此，将固定值 VK_2 存储到以太坊一层合约中，实现对 proof_2 进行正确性验证。

反之，如果矿工验证成功，则 proof_2 正确，则 proof_1 正确，则任意算法（包括 for 循环）运算正确。

Zcash 递归零知识证明

使用特殊的递归曲线：标量域 = 基域

用向量内积承诺替换 KZG 或 Dan 承诺，去掉双线性映射，实现递归零知识证明。

lynndell 博士 新火科技 密码学专家 lynndell2010@gmail.com