

[CDP11] 组会论文笔记

这是前三周小组会上的一篇笔记。

阅读：

- Ronald Cramer, Ivan Damgard and Valerio Pastro. On the **Amortized** Complexity of Zero Knowledge Protocols for Multiplicative Relations

贡献：

- 给出了信息论承诺方案和线性秘密分享方案
- 给出了乘法协议（over field），通信复杂度相比于[CDD99]降低了 l 倍
- 给出了乘法协议（over integer），相比于[DO97,DF02]，不需要任何假设，并且通信复杂度增加较少。
- 将乘法协议扩展到电路中，给出了验证电路的协议，并给出了利用MPC-in the head进一步降低了通信复杂度的方法。

关键技术：

- 预处理的信息论承诺方案：预处理阶段要使得承诺方和接收方带有**未知随机数**，且二者的未知随机数**相互关联**
- 乘法协议/验证电路协议：将许多相似的陈述的证明合并到一个协议中，使得每个证明的**摊销复杂度**变小
- 验证电路协议：利用MPC-in the head，将对一些随机数的同态承诺换为对MPC各方的View的承诺（不需要满足同态性），进而降低通信复杂度

1 工具一：信息论承诺

利用添加**预处理**过程，使得信息论安全、简单高效的**同态**承诺方案成为可能。

1.1 有限域上

K 是一个很小的域， L 是其扩域，下面是对 $v_1, \dots, v_\ell \in K$ 的承诺：

- 预处理阶段：可信第三方发送随机的 $u_1, \dots, u_\ell \leftarrow K$ 以及对应的 $\{m_i := a \cdot u_i + b_i\}_{i \in [\ell]}$ 给承诺者，发送对应的 $a, b_i \leftarrow L$ 给接收者
 - 可以认为发送者持有 u_i 信息及其 $\text{MAC}_{sk_i}(u_i)$ ，而接收者持有 $sk_i = (a, b_i)$
- 承诺阶段：发送者发送 $u_i - v_i$ 作为承诺；接收者计算 $\text{MAC}_{sk_i}(u_i - v_i)$
- 打开阶段：发送者发送 $v_i, \text{MAC}_{sk_i}(u_i)$ 作为打开；接收者检验是否有 $a \cdot v_i + \text{MAC}_{sk_i}(u_i - v_i) = \text{MAC}_{sk_i}(u_i)$

此带有处理的承诺具有统计 binding 和统计 hiding 性质：

- 统计 binding：承诺发送者在打开阶段无法打开两个 $v_i \neq v'_i$ ，否则发送者可以计算出 (a, b_i) ，事实上 (a, b_i) 的信息是信息论掩盖的（发送者只收到预处理的 $u_i, \text{MAC}_{sk_i}(u_i)$ ）
- 统计 hiding：接收者无法判断发送者承诺的是消息 v_i 还是 v'_i ，这也是一样的原因，接收者只知道 $sk_i = (a, b_i)$ ，而没有关于 u_i 的信息

具体地，我们可以计算出打破binding的概率：

- $1/|L|$

同时也容易注意到上述性质是具有同态性的：

- 预处理部分： $u_i + u_j, \text{MAC}_{a, b_i + b_j}(u_i + u_j) := \text{MAC}_{sk_i}(u_i) + \text{MAC}_{sk_j}(u_j)$
- 发送者信息部分： $v_i + v_j$
- 接收者密钥部分： $sk_{ij} := (a, b_i + b_j) = sk_i \oplus sk_j$

- 承诺部分: $(u_i - v_i) + (u_j - v_j)$
- 检验部分: $a \cdot (v_i + v_j) + \text{MAC}_{sk_{ij}}(u_i - v_i + u_j - v_j) = \text{MAC}_{sk_i}(u_i) + \text{MAC}_{sk_j}(u_j)$

1.2 整数上

要承诺 k 比特的数 m_1, \dots, m_ℓ :

- 预处理:
 - 发送给承诺者 $u_i \leftarrow [-2^{k+\kappa}, \dots, 2^{k+\kappa}]$ 和对应的 $\text{MAC}_{a,b_i}(u_i)$
 - 发送给接收者 $a \leftarrow \text{Prime}([-2^\kappa, \dots, 2^\kappa])$ 和 $b_i \leftarrow [-2^{k+3\kappa}, \dots, 2^{k+3\kappa}]$
- 其他的都是类似的

此带有处理的承诺具有统计 binding 和统计 hiding 性质:

- 统计 binding: 能否计算 $a \cdot \Delta_1 = \Delta_2$, 仅仅知道 $u_i, \text{MAC}_{a,b_i}(u_i)$
 - 这里也可以看出为什么 a, b_i 的范围要如此取, b_i 需要完全 smudge $a \cdot u_i$
 - 需要计算 condition on $u_i, \text{MAC}_{a,b_i}(u_i)$ 下, (a, b_i) 有多少种选择可能: $\Theta(2^\kappa/\kappa)$
- 统计 hiding: 使用 smudging lemma

同时也容易注意到上述性质是具有同态性的, 但是由于是在 \mathbb{Z} 上的, 无法支持无限次操作。

2 工具二: 线性秘密分享

下面给出支持打包 (packed) 线性秘密分享方案的线性代数描述, 可以对照 **packed shamir sharing** 进行理解: (LSSS 是 SS 的一种特殊情况, SS 的完整描述需要对 **控制结构** 的刻画, 例如可以利用布尔电路; 而 LSSS 我们可以很方便地使用)

- 下面的描述可以用 packed-Shamir Sharing 作为参考物, 一部分 index 上放 secrets (例如可以放在最前面的 index, 这部分不会被分发出去!), 另外的 index 上放冗余的秘密编码信息 (这部分分发出去!)
- C 是秘密分享的编码空间, $C \subset K^m$, 假定其维度为 n
- I 表示选定的 index 集合 $I = [m]$
- $\mathbf{x} = (x_i)_{i \in I} \in K^m$
- **索引限制映射** restriction: $\pi_A : \mathbf{x} \mapsto (x_i)_{i \in A}$
- **【Shares 的分布】** 关于 C , 称 S 提供了 S -uniformity (刻画常有的 shares 的分布性质) 当且仅当 $\pi_S(C) = K^{|S|}$
 - 若把 C 表示为矩阵, 由某组基底表出, 那么这些基底的索引限制映射在 $K^{|S|}$ 上满秩
- **【Recover 算法的存在】** 关于 C , 称 A 索引确定了 S 索引 (处理 SS 方案的冗余, 例如 SS 方案当中的恢复 Recover 算法), 有: $\exists f \forall \mathbf{c} \in C, s.t. (f \circ \pi_A)(\mathbf{c}) = \pi_S(\mathbf{c})$
 - 用矩阵可以表示为 $\exists \mathbf{F} \forall \mathbf{x} \in \mathbb{F}_{|K|}^n : \mathbf{FAC} \cdot \mathbf{x} = \mathbf{SC} \cdot \mathbf{x}$
 - 即有 $(\mathbf{FA} - \mathbf{S}) \cdot \mathbf{C} = 0$, 其含义是可以找到投影算子 \mathbf{F} , 使得任意向量 $\mathbf{x} \in \mathbb{F}_{|K|}^m$ 有在 \mathbf{FA} 和 \mathbf{S} 二者上的投影在 \mathbf{C}^\perp 的同一个陪集上
- **【Privacy 的保证】** 关于 C , 称 A 索引和 S 索引独立 (刻画两部分的 shares 不相互影响, 即 privacy 性质), 有: $\pi_{A,S}(\mathbf{c}) = (\pi_A(\mathbf{c}), \pi_S(\mathbf{c}))$ 可以跑遍 $\pi_A(C) \times \pi_S(C)$
 - 直觉是由于 $\pi_{A,S}$ 映射是 regular 的, 对于任意一个投影 $\pi_A(\mathbf{c})$ 都不会影响 $\pi_S(\mathbf{c})$ 上的值
 - 矩阵角度: $(\mathbf{AC})^\perp$ 包含 \mathbf{SC} 组成的向量空间, 即存在 $\mathbf{F}, s.t. \mathbf{F} \cdot (\mathbf{AC})^\perp = \mathbf{SC}$
- **【编码算法的特殊“转移”性质】** 关于 C , 称 g 编码函数为 S -generator, 当且仅当:
 - 编码算法: $g : K^{|S|+e} \rightarrow C$ 为满射, 即 $K^{|S|+e}$ 大于等于 C 的维度, 刻画 LSSS 的编码算法 (此处 e 是为了放置随机数隐藏秘密s)
 - S 拥有 S -uniformity (被下面的“编码转移特点”蕴涵? 有了这个条件也能设计 g 满足“编码转移特点”)

- 编码转移特点: $\pi_{[|S|]} = \pi_S \circ g$, 此处可以看作秘密的编码 g 的特点, $K^{|S|+e}$ 中前 $|S|$ 的秘密会迁移到编码后 C 的索引集合 S 上
- g 编码函数的存在性:
 - 关于 C, S 提供 S -uniformity
 - $|S| + e \geq \dim(C)$

基于上述的线性代数描述的术语, 我们下面定义 LSSS 方案:

- $S \subset I, S^* = I \setminus S$; S^* 上存放分发的分片, 被称为**玩家索引集**, 而 S 上放上秘密和不会分发出去的分片, 被称为**秘密索引集**, $\pi_j(C)$ 被称为第 j 个玩家的 share 份额 (若 $j \in S^*$)
- (C, S) -LSSS 是指: 关于 C , 有 S -uniformity, 以及 S^* 决定 S (即 $|S^*| \geq \dim(C)$)
 - 我们关心的 (C, S) -LSSS generator 是针对 C 的 S -generator
 - 产生 shares 的过程: 选择 $K^{|S|+e}$ 的前 $|S|$ 位置作为**秘密** \mathbf{s} , 再随机填充剩下的 e 个位置作为噪声 (防止能直接通过 C 的少数个位置的值就可以直接学习到 \mathbf{s}), 得到 $\rho_{\mathbf{s}}$, 得到 $\mathbf{c} = g(\rho_{\mathbf{s}}) \in C$, 分配 $\pi_{S^*}(\mathbf{c})$ 给诸位玩家
- A -privacy: $A \subset S^*$ 且关于 C , A 索引上的份额和 S 索引上的份额相互独立, 即 A 索引集为一个 unqualified set
 - 若对任意 $|A| = t$ 的 A 都有 A -privacy, 则称为 t -privacy
- A -reconstruction: A 决定 S , 即 A 索引集为一个 qualified set
 - 若对任意 $|A| = r$ 的 A 都有 A -reconstruction, 则称为 r -privacy
- 容易验证, 对于 LSSS 而言, 都具有上述的 threshold 性质

Schur-Product Transform:

$$ST(C) := \left\{ \forall m \in \mathbb{N} : \sum_{i=1}^m a_i \cdot (\mathbf{x}_i * \mathbf{y}_i) \mid \mathbf{x}_i, \mathbf{y}_i \in C, a_i \in \mathbb{N} \right\}$$

显然经过该乘积变换后, 维数不减; 此时 $|S^*| \leq \dim(C)$ 可能并不一定成立, 如果仍然成立, 我们可有找到 \hat{r} -product reconstruction

Sweeping vectors: 如果我们想要找到 C 上 S 索引集上取值为 $(x_1, \dots, x_{|S|})$ 且 unqualified set A 中对应的份额都与分享秘密 0 的份额一样

- 找到 $\pi_{A,S}(\mathbf{c}_{A,j}) = (\mathbf{0}, e_j)$ 及其在 g 下的原像 $\mathbf{w}_{A,j}$
- 选择 ρ_0 的前 $|S|$ 位为 0

$$\rho_0 + \sum_{j=1}^{|S|} x_j \cdot \mathbf{w}_{A,j}$$

3 主协议: 批量乘法验证

主协议本质的想法, 是利用承诺的**同态性**, 将验证 $\mathbf{x} * \mathbf{y} = \mathbf{z}$ 转化为验证 $\text{Encode}(\mathbf{x}, \mathbf{r}_x) * \text{Encode}(\mathbf{y}, \mathbf{r}_y) = \text{Encode}(\mathbf{z}, \mathbf{r}_z)$, 由于有“编码平移性”, 所以可以保证 soundness:

- 若没有 $\mathbf{x} * \mathbf{y} = \mathbf{z}$, 则对于任意的 $\mathbf{r}_x, \mathbf{r}_y$ 都找不到 \mathbf{r}_z , 使得上述 Encode 的式子成立, 那么恶意 prover 无法蒙骗过关, 因为 Encode 本身的纠错特性
- 另外一方面, 我们也可以注意到 $\mathbf{r}_{x,y,z}$ 的加入可以方便地实现 ZK
- Note: 事实上, 整个 zkSNARK 中的 PIOP 构造就是关于纠错码的故事; 例如上述的编码如果为 RS code, 是不是 $[\mathbf{x}]$ 就是最简单的多项式承诺, 即把特定的 evaluation representation 承诺起来?

在上述的指导思想下, 主协议的设计是简单的:

- NP Relation = $\{([\mathbf{x}], [\mathbf{y}], [\mathbf{z}]; \mathbf{x}, \mathbf{y}, \mathbf{z}) \mid [\mathbf{x}], [\mathbf{y}], [\mathbf{z}] \text{ open to } \mathbf{x}, \mathbf{y}, \mathbf{z} \wedge \mathbf{x} * \mathbf{y} = \mathbf{z}\}$
- pp 当中包含 M, \hat{M}

- Step 1: Prover 发送 $[\mathbf{r}_x], [\mathbf{r}_y], [\mathbf{r}_z]$
- Step 2: Verifier 选择 Encode 后随机的索引集 $O \subset S^*$ 打开 (注意为了保证 ZK, 此处 O 需要为 unqualified set, 这和 CDS94 的思路一致)
- Step 3: Prover 打开, Verifier 验证:
 - 打开是否正确
 - 对应的打开是否满足乘法关系

Protocol Verify Multiplication

1. The prover chooses two vectors $\mathbf{r}_x, \mathbf{r}_y \in K^e$, and sets $\rho_x = (\mathbf{x}, \mathbf{r}_x)$, $\rho_y = (\mathbf{y}, \mathbf{r}_y)$. Define $\mathbf{c}_x = M\rho_x$, $\mathbf{c}_y = M\rho_y$. Now, the prover computes $\hat{\rho}_z \in K^{l+\hat{e}}$ such that $\hat{\rho}_z$ is consistent with secret \mathbf{z} and such that $\widehat{M}\hat{\rho}_z = \mathbf{c}_x * \mathbf{c}_y$.
Note that this is possible by solving a system of linear equations, exactly because $\mathbf{x} * \mathbf{y} = \mathbf{z}$. We then write $\hat{\rho}_z = (\mathbf{z}, \hat{\mathbf{r}}_z)$ for some $\hat{\mathbf{r}}_z \in K^{\hat{e}}$. Set $\hat{\mathbf{c}}_z = \widehat{M}\hat{\rho}_z$.
2. The prover sends vectors of commitments $[\mathbf{r}_x], [\mathbf{r}_y], [\hat{\mathbf{r}}_z]$ to the verifier. Together with the commitments to \mathbf{x}, \mathbf{y} and \mathbf{z} , the verifier now holds vectors of commitments $[\rho_x], [\rho_y], [\hat{\rho}_z]$.
3. The verifier chooses t uniform indices $O \subset S^*$ and sends them to the prover.
4. Let \mathbf{m}_i be the i 'th row of M and $\hat{\mathbf{m}}_i$ the i 'th row of \widehat{M} . For each $i \in O$, using the homomorphic property of the commitments, both prover and verifier compute commitments

$$[(\mathbf{c}_x)_i] = [\rho_x]^{\mathbf{m}_i}, \quad [(\mathbf{c}_y)_i] = [\rho_y]^{\mathbf{m}_i}, \quad [(\hat{\mathbf{c}}_z)_i] = [\hat{\rho}_z]^{\hat{\mathbf{m}}_i}.$$

The prover opens these commitments to the verifier.

5. The verifier accepts if and only if the opened values satisfy $(\mathbf{c}_x)_i \cdot (\mathbf{c}_y)_i = (\hat{\mathbf{c}}_z)_i$ for all $i \in O$.

小问题: \hat{M} 从算法的角度来说, 应该如何选择?

我们假定 g 的像空间的一组基底为 $\mathbf{m}_1, \dots, \mathbf{m}_k$

$$\left(\sum_{i=1}^k a_i \cdot \mathbf{w}_i \right) * \left(\sum_{i=1}^k b_i \cdot \mathbf{w}_i \right)$$

从而我们只需要选择 $(\mathbf{w}_1, \dots, \mathbf{w}_k) \otimes (\mathbf{w}_1, \dots, \mathbf{w}_k)$ 的一组基即可

主定理: 若使用上述的信息论承诺, 则上述算法是完美零知识的, 且如果存在 $i, x_i y_i \neq z_i$, 能通过的上述协议的概率最多为:

$$((\hat{r} - 1)/d)^t + 1/|L|$$

ZK 分析:

- 由于 perfect hiding 性质的存在, 我们可以将 $[\mathbf{x}, \mathbf{r}_x]$ 解释为 $[\mathbf{r}_1]$, 类似地解释其他的, 然后保证乘积关系
- 利用 LSSS 的 Privacy 性质可以证明对应的分布完全相同

Soundness 分析 (此处的分析比起常用的 soundness 性质要弱! 此处直接假定了 $[\mathbf{x}], [\mathbf{y}], [\mathbf{z}]$ 直接对应到 $\mathbf{x}, \mathbf{y}, \mathbf{z}$):

- 根据纠错码的性质, 我们知道最多有 $\hat{r} - 1$ 个相同, 能在 $((\hat{r} - 1)/d)^t$ 概率内逃过检查
- 或者能够采用新的打开方式 (打破 binding)
- 对上述使用 Union Bound

关于 Soundness 的讨论, 由于本文的 NP Relation 带有预处理, 可能需要定义为如下:

$$\{(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3; \mathbf{x}, \mathbf{y}, \mathbf{z}) \mid R_{pp_c}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3; \mathbf{x}, \mathbf{y}, \mathbf{z}) = 1\}$$

- 这里 statement 是 $(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3; \mathbf{x}, \mathbf{y}, \mathbf{z})$
- $(pp_c, pp_r) \leftarrow \text{Setup}(1^\kappa)$ 是隐私的只有 prover (committer)、verifier (receiver) 拿到的预处理参数
- R_{pp_c} 算法即为利用 pp_c 进行打开并检验 $\mathbf{x} * \mathbf{y} = \mathbf{z}$, 注意到这里是完美 binding

则此处 soundness 定义和一般的 ZKP 不同，考虑一种带有预处理的 soundness：

$$\Pr[(pp_c, pp_r) \leftarrow \text{Setup}(1^\kappa), \text{view}_V \leftarrow \langle \mathcal{A}(pp_c), \mathcal{V}(1^\kappa, pp_r) \rangle : R_{pp_c}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{x}, \mathbf{y}, \mathbf{z}) = 0 \wedge \text{Verify}_{pp_r}(\text{view}_V) = 1] \leq \text{negl}(1^\kappa)$$

接下来，再讨论一点和协议相关的问题：

- O 有两种表示方法：一种是直接表示 O 中的数字，第二种是使用 bitmap
- LSSS 的参数设计：通信复杂度显然是 $\Theta(\kappa_c(e + \hat{e}) + \kappa_o t + d)$

- 假定我们想要达到 2^{-u} 的安全性，同时能保证

$$e, \hat{e} = O(u), \hat{r} = O(l + u), t = \Theta(u), d = O(l + u), (\hat{r} - 1)/d = 1/2$$

- 从而可以实现均摊通信复杂性 $O(\frac{u}{t}(\kappa_c + \kappa_o))$ ，再注意到之前的信息论承诺方案有 $\kappa_c = O(1), \kappa_o = O(u)$ ，那么，只要 $l > u^2$ ，则均摊复杂性可以达到 $O(1)$
- 一个具体的例子：考虑一个 packed shamir sharing，参数选择如下
 - 求值集合 $\{q_1, \dots, q_l, p_1, \dots, p_t, \dots, p_d\}$
 - $2(t + l - 1) < d$
 - 从而为 l -multi-secret K -linear SS of length d , with t -privacy, $(2t + 2l - 1)$ -reconstruction
 - 具体参数选择： $u = l, t = l, d = 8l, |K| > 9l, e = 2l, \hat{e} = 4l - 1$ ，此处唯一无法实现的是 $\kappa_c = O(1)$ ，即我们需要思考如何在常数大小的域上进行选择
 - 但是如果我们的感兴趣的命题是在小域 $K_0 \subset K$ 上的，而我秘密分享方案也是在 K 上的， $[K : K_0] = \mu$ ，那可以利用添加位置以坐标进行模拟（可以验证仍然可以进行对应的矩阵计算）
- 更优的参数选择：
 - 此处需要 Algebraic Function Field 理论，可以达到上述期望的参数
- 应用到电路验证问题上：直接对线路进行承诺，然后利用此具有 amortization 的协议

4 协议扩展

本质的想法：

- 将电路中输入和输出之间满足关系进行批量验证。对输入和输出进行编码，编码后原来输入上的一些微小改变在编码后都被放大了，验证者对编码后的值进行随机挑战，若输入和输出不满足特定的关系，则验证者可以以很大的概率捕捉到证明者的谎言。

协议简单描述：

- 与批量乘法协议基本一样，只是将原来要验证的关系 $\mathbf{x} * \mathbf{y} = \mathbf{z}$ 拓展到更一般的关系 $D(\mathbf{x}_1, \dots, \mathbf{x}_v)$

Protocol Verify Circuit

1. The prover chooses v vectors $\mathbf{r}_1, \dots, \mathbf{r}_v \in K^e$, and sets $\rho_j = (\mathbf{x}_j, \mathbf{r}_j)$ for $j = 1, \dots, v$. Define $\mathbf{c}_j = M\rho_j$. Now, the prover computes $\tilde{\rho}_{\mathbf{z}} \in K^{l+\tilde{e}}$ such that $\tilde{\rho}_{\mathbf{z}}$ is consistent with secret \mathbf{z} and such that $\tilde{M}\tilde{\rho}_{\mathbf{z}} = D(\mathbf{x}_1, \dots, \mathbf{x}_v)$. Note that this is possible by solving a system of linear equations, because $D(\mathbf{x}_1, \dots, \mathbf{x}_v) = \mathbf{z}$. We then write $\tilde{\rho}_{\mathbf{z}} = (\mathbf{z}, \tilde{\mathbf{r}}_{\mathbf{z}})$ for some $\tilde{\mathbf{r}}_{\mathbf{z}} \in K^{\tilde{e}}$. Set $\tilde{\mathbf{c}}_{\mathbf{z}} = \tilde{M}\tilde{\rho}_{\mathbf{z}}$.
2. The prover sends vectors of commitments $[\mathbf{r}_j]$, $j = 1, \dots, v$ and $[\tilde{\mathbf{r}}_{\mathbf{z}}]$ to the verifier. Together with the commitments to \mathbf{x}_j and \mathbf{z} , the verifier now holds vectors of commitments $[\rho_j]$, $j = 1, \dots, v$, and $[\tilde{\rho}_{\mathbf{z}}]$.
3. The verifier chooses t uniform indices $O \subset S^*$ and sends them to the prover.
4. Let \mathbf{m}_i be the i 'th row of M and let $\tilde{\mathbf{m}}_i$ be the i 'th row of \tilde{M} . For each $i \in O$, using the homomorphic property of the commitments, both prover and verifier compute commitments

$$[(\mathbf{c}_j)_i] = [\rho_j]^{\mathbf{m}_i}, \text{ for } j = 1, \dots, v, \quad [(\tilde{\mathbf{c}}_{\mathbf{z}})_i] = [\tilde{\rho}_{\mathbf{z}}]^{\tilde{\mathbf{m}}_i}.$$

The prover opens these commitments to the verifier.

5. The verifier accepts if and only if the opened values satisfy $D((\mathbf{c}_1)_i, \dots, (\mathbf{c}_v)_i) = (\tilde{\mathbf{c}}_{\mathbf{z}})_i$ for all $i \in O$.

Using a similar proof as for theorem 1, one easily shows

开销:

- 若电路的乘法深度为 δ , 电路的输入个数为 v , 若保证错误概率为 2^{-l} , 则摊销复杂度为 $O(2^\delta \kappa + v\kappa + \delta \log l)$

两个变体:

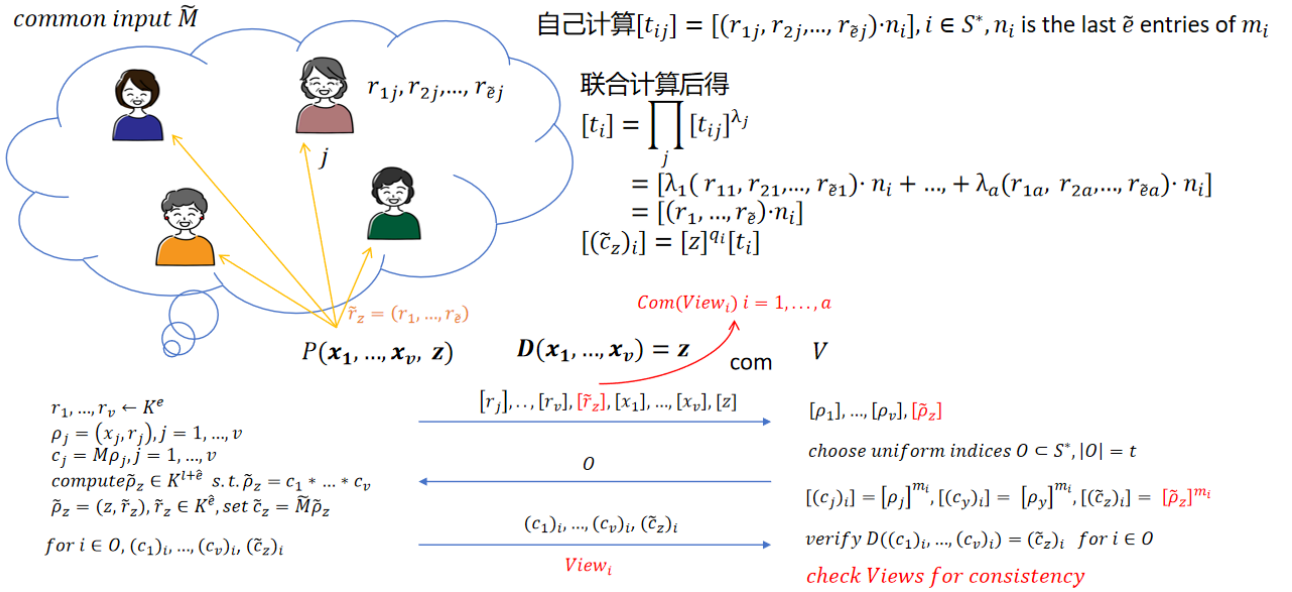
- 乘法门: 对电路 D 中的每个乘法门 T , 利用 Verify Multiplication protocol 进行验证
 - 对乘法门 T 的输入 \mathbf{x}_T 、 \mathbf{y}_T 和输出 \mathbf{z}_T 进行承诺, 利用 Verify Multiplication protocol 进行验证 $\mathbf{x}_T * \mathbf{y}_T = \mathbf{z}_T$
 - 由于承诺具有同态性, 在电路 D 中的线性操作可以由验证者自己验证。

开销: $O(\mu\kappa + v\kappa)$, 其中 μ 是电路中乘法门的数量。

参数选择: 当 $\delta = O(\log v)$ 或者 $\delta = O(v)$, 且 $\mu > 2^\delta$ 时, 使用 Verify Circuit protocol 开销更低

- 结合 MPC-in-the-Head:
 - **Purpose:** 在 Verify Circuit Protocol 中, 由于 $\tilde{e} = 2^\delta n + 1 - l$, 导致 $\tilde{\mathbf{r}}_{\mathbf{z}} = (r_1, \dots, r_{\tilde{e}})$ 长度比较长, 因此利用 MPC-in the head 的方法, 证明者不直接发送 $[\tilde{\mathbf{r}}_{\mathbf{z}}]$, 而发送需要的 $[(\tilde{\mathbf{c}}_{\mathbf{z}})_i]$

Secrets	Players	For player j :
u	j	$[t_{ij}] = [(r_{1j}, r_{2j}, \dots, r_{\tilde{e}j}) \cdot n_i], i \in S^*, n_i \text{ is the last } \tilde{e} \text{ entries of } m_i$
$r_1 \xrightarrow{f(x) = r_1 + b_1x + \dots + b_{a-1}x^{a-1}}$	$1 \quad r_{11}, r_{21}, \dots, r_{\tilde{e}1}$	$[t_i] = \prod_j [t_{ij}]^{\lambda_j}$
\vdots	\vdots	$= [\lambda_1(r_{11}, r_{21}, \dots, r_{\tilde{e}1}) \cdot n_i + \dots + \lambda_a(r_{1a}, r_{2a}, \dots, r_{\tilde{e}a}) \cdot n_i]$
\vdots	\vdots	$= [(r_1, \dots, r_{\tilde{e}}) \cdot n_i]$
$r_{\tilde{e}} \xrightarrow{h(x) = r_{\tilde{e}} + c_1x + \dots + c_{a-1}x^{a-1}}$	$a \quad r_{1a}, r_{2a}, \dots, r_{\tilde{e}a}$	$[(\tilde{\mathbf{c}}_{\mathbf{z}})_i] = [z]^{a_i} [t_i]$
$r_1 = \lambda_1 r_{11} + \lambda_2 r_{12} + \dots + \lambda_a r_{1a}$	$f(x) = \sum_{i=1}^{t+l} f(x_i) \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$	
\vdots		
\vdots		
$r_{\tilde{e}} = \lambda_1 r_{\tilde{e}1} + \lambda_2 r_{\tilde{e}2} + \dots + \lambda_a r_{\tilde{e}a}$	$\lambda_i = \prod_{j \neq i} \frac{0 - j}{i - j}$	



- 开销：使用MPC-in the head后，验证电路的协议中 $O(\tilde{e}\kappa_c)$ 部分的开销变为 $O((at + a)\kappa_c)$

5 整数上的补正

本质的想法：

- 将秘密 s_1, \dots, s_l 和随机数 r_1, \dots, r_t 看作在固定集合上 $K, |K| = l + t$ 上的求值，从而确定一个次数小于等于 $l + t - 1$ 的多项式，利用多项式在 $\mathbb{Z} \setminus K$ 中的求值得到秘密份额（可以看作 s_1, \dots, s_l 编码后得到的值），验证者选择的挑战 $|O| \leq T$ ，不会泄露关于秘密的任何信息，而且由于LSS的重构性质，保证了若份额之间满足关系 $Encode(\mathbf{x}, \mathbf{r}_x) * Encode(\mathbf{y}, \mathbf{r}_y) = Encode(\mathbf{z}, \mathbf{r}_z)$ ，那么秘密之间以很大的概率满足 $\mathbf{x} * \mathbf{y} = \mathbf{z}$ 。

协议的简单描述：

- The prover chooses $\mathbf{a}_x, \mathbf{a}_y \in \mathbb{Z}^t$ and uses Lagrange interpolation (over the rationals) to generate two polynomials g_x, g_y , having degree $t + l$, such that

$$g_x(-i) = x_i, \quad g_x(-l - j) = (\mathbf{a}_x)_j, \quad g_y(-i) = y_i, \quad g_y(-l - j) = (\mathbf{a}_y)_j,$$

for $i = 1, \dots, l$ and $j = 1, \dots, t$. The prover now sets $\hat{g}_z = g_x \cdot g_y$, $f_x = \Delta g_x$, $f_y = \Delta g_y$ and $\hat{f}_z = \Delta^2 \hat{g}_z$. As explained above, f_x and f_y are polynomials with integral coefficients and have degree at most $t + l$. Notice that \hat{f}_z is also a polynomial with integral coefficients, but has degree at most $2(t + l)$.

- The prover sends commitments $[\mathbf{f}_x]$, $[\mathbf{f}_y]$ and $[\hat{\mathbf{f}}_z]$.
- The verifier checks that $[\mathbf{x}]$, $[\mathbf{y}]$ and $[\mathbf{z}]$ are consistent with f_x, f_y and \hat{f}_z : namely for all $i = 1, \dots, l$ it computes

$$[\mathbf{f}_x]^{\text{ev}(-i)} [\Delta x_i]^{-1}, \quad [\mathbf{f}_y]^{\text{ev}(-i)} [\Delta y_i]^{-1}, \quad [\hat{\mathbf{f}}_z]^{\text{ev}(-i)} [\Delta^2 z_i]^{-1},$$

and asks the prover to open these commitments to zero. If any of these openings do not agree with the commitments, the verifier quits.

- The prover defines the vector $x_C = ([\mathbf{x}], [\mathbf{y}], [\mathbf{z}], [\mathbf{f}_x], [\mathbf{f}_y], [\hat{\mathbf{f}}_z])$ containing committed values. We think of x_C as a vector of instances for the protocol P_C . The prover computes a vector a_C as the first message for the protocol P_C with instance x_C . the prover sends a_C to the verifier.
- The verifier chooses t uniform indices $O \subset \{1, \dots, d\}$. Similarly as above, the verifier computes

$$[\mathbf{f}_x]^{\text{ev}(i)} = [(\mathbf{b}_x)_i], \quad [\mathbf{f}_y]^{\text{ev}(i)} = [(\mathbf{b}_y)_i], \quad [\hat{\mathbf{f}}_z]^{\text{ev}(i)} = [(\hat{\mathbf{b}}_z)_i],$$

for $i \in O$. The verifier generates a vector e_C as a challenge on (x_C, a_C) according to P_C . The verifier sends e_C together with the index set O to the prover.

- The prover computes the vector z_C as a reply for (x_C, a_C, e_C) according to P_C . The prover sends z_C together with the openings of $[(\mathbf{b}_x)_i]$, $[(\mathbf{b}_y)_i]$ and $[(\hat{\mathbf{b}}_z)_i]$ for $i \in O$.
- The verifier accepts if and only if (x_C, a_C, e_C, z_C) is an accepted conversation for P_C and the opened values satisfy $(\mathbf{b}_x)_i \cdot (\mathbf{b}_y)_i = (\hat{\mathbf{b}}_z)_i$ for $i \in O$.

