**DDH-based Private Intersection-Sum Protocol概述**

**1. 协议目的**

该协议让两个参与方私下计算两个集合的交集权重和，也就是说：

* 一方拥有一个标识符集合；
* 另一方拥有带权重的标识符集合；

双方希望只知道交集中的权重总和，但不能泄露各自集合的具体内容和权重。

**2. 核心技术点**

* 使用一种特殊的数学群结构，参与方将自己的元素映射到群里，并通过各自的私钥进行指数运算，使得只有双方都“加密”过的元素能匹配上，保护集合隐私。
* 使用哈希函数将原始标识符转换为群元素，避免直接暴露原始数据。
* 通过同态加密技术对权重进行加密，支持在密文状态下进行加法运算，实现权重的安全累加。
* 协议过程中的数据均采用乱序发送，避免泄露元素顺序信息。
* 对最终密文进行随机刷新，防止重放攻击和关联分析。

**3. 协议流程简述**

**初始化阶段**

* 双方各自生成随机的私钥，互不公开。
* 拥有带权集合的一方生成同态加密的公私钥对，将公钥发送给另一方。

**第一步（发送第一批加密元素）**

* 第一方将自己的集合中的每个元素，通过哈希函数映射到群元素，再使用自己的私钥进行指数运算。
* 将计算结果乱序发送给第二方。

**第二步（第二方处理并发送）**

* 第二方对收到的元素再用自己的私钥做一次指数运算，得到双方都“加密”过的群元素集合。
* 同时，对自己集合中的元素做哈希映射及私钥指数运算，得到自己的加密元素。
* 对每个元素对应的权重使用同态加密加密。
* 将双方都“加密”过的集合和加密权重对乱序发送给第一方。

**第三步（第一方计算交集并生成加密和）**

* 第一方对收到的第二方加密元素，再使用自己的私钥指数运算，得到双重加密元素。
* 利用双重加密元素判断哪些元素属于交集。
* 对交集对应的同态加密权重密文累加，得到交集权重的加密总和。
* 对结果密文进行随机刷新以增强安全性。
* 将刷新后的密文发送给第二方。

**最终输出**

* 第二方使用私钥解密得到交集权重和，完成协议目的。

**4. 设计思路及安全保障**

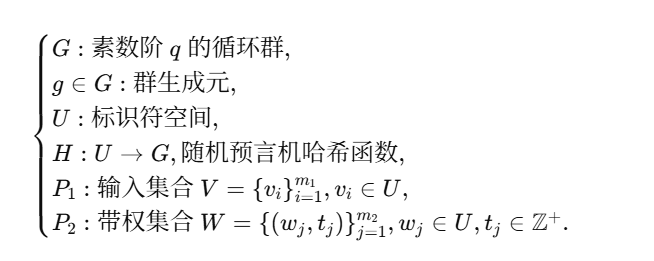
* **隐私保护：** 私钥指数运算和哈希映射双重保护输入集合，单方无法推断对方具体元素。
* **权重保密：** 同态加密确保权重在计算过程全程被加密保护，不被泄露。
* **交集确认方式：** 双重指数运算使得只有双方元素相同的标识符才能匹配，保证交集正确性。
* **防信息泄露：** 数据乱序发送避免顺序关联攻击，随机刷新防止重放及统计攻击。
* **安全假设依赖：** 协议安全性基于离散对数难题和同态加密的安全性。

**5. 注意事项**

* 所用的群和哈希函数需满足严格的密码学安全标准。
* 同态加密参数应选择足够大以抵抗当前的密码分析攻击。
* 双方必须保证私钥和随机数生成的安全性。
* 协议交互轮数固定，网络延迟和通信稳定性需保证。
* 性能方面，指数运算和加密操作计算量较大，实际应用时需要考虑优化。
* 需要防范潜在的重放和侧信道攻击，合理设计刷新机制和协议实现。

DDH-based Private Intersection-Sum Protocol实现

参数与符号：



**协议步骤：**

1. **初始化（Setup）：**

文本, 信件

AI 生成的内容可能不正确。

1. Round 1 (P1):

文本, 信件

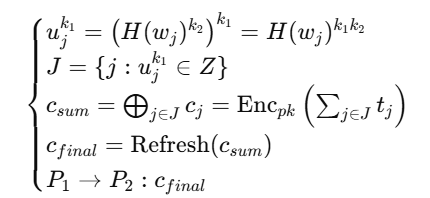
AI 生成的内容可能不正确。

1. Round 2 (P2):

文本, 信件

AI 生成的内容可能不正确。

1. Round 1 (P1):



5.输出

文本, 信件

AI 生成的内容可能不正确。

**DDH-based Private Intersection-Sum Protocol 的实现**

**1. 群结构和基础操作**

**GroupG 类模拟协议中的数学群，主要负责：**

* **通过 hash\_to\_group 方法，把输入的字符串（比如邮箱地址）用 SHA-256 哈希映射为群内的一个元素（一个整数）。**
* **使用 exp 方法完成群里的“幂运算”，即对群元素进行指数运算，保护输入隐私。**
* **random\_exp 方法生成随机私钥指数，代表协议中双方的秘密私钥。**

**2. 第一方（Party1）代码实现**

* **初始化时，Party1 生成私钥 k1 和保存自己的输入集合 V。**
* **Round 1中，round1() 函数完成：**
  + **对集合中每个元素用群的哈希函数映射。**
  + **对哈希值使用私钥 k1 做指数幂运算，得到加密后的元素。**
  + **打乱顺序后返回这些加密元素给第二方。**
* **Round 3中，round3() 函数实现：**
  + **接收第二方发来的元素和加密权重对列表。**
  + **对第二方的元素再次用私钥 k1 做指数运算，得到双重加密结果。**
  + **检查双重加密结果是否在之前的集合 Z 中，确定交集元素。**
  + **对交集元素对应的权重密文用同态加法累加。**
  + **使用 randomize() 函数对累加后的密文随机化，防止重放攻击。**
  + **返回加密后的交集权重和给第二方。**

**3. 第二方（Party2）代码实现**

* **初始化时，Party2 生成私钥 k2，并生成 Paillier 同态加密密钥对 (pk, sk)，保存带权集合 W。**
* **Round 2中，round2() 函数完成：**
  + **接收第一方发送的加密元素列表。**
  + **对每个元素用自己的私钥 k2 再做一次指数运算，得到双方都加密过的元素集合 Z，并乱序。**
  + **对自己带权集合中每个元素做哈希映射和 k2 次幂，得到加密元素。**
  + **对对应权重用 Paillier 公钥加密。**
  + **将加密元素和对应密文权重组成对 (H(w\_j)^k2, Enc(t\_j))，乱序发送给第一方。**
* **输出阶段中，使用私钥 sk 解密第一方返回的加密交集权重和，得到结果。**

**代码运行结果：**

