软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2023,34(7):3343−3353 [doi: 10.13328/j.cnki.jos.006529]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

集合交集元素之和的保密计算*

李顺东[1], 张凯鑫[1], 杨 晨[1], 汪榆淋[2]

[1](陕西师范大学 计算机科学学院, 陕西 西安 710119)

[2](陕西师范大学 数学与统计学院, 陕西 西安 710119)

通信作者: 李顺东, E-mail: shundong@snnu.edu.cn

摘 要: 安全多方计算是国际密码学的研究热点之一, 保密计算集合交集元素之和问题是安全多方计算比较新的问题之一. 该问题在工商业、医疗健康等领域具有重要的理论意义和实用价值. 现有解决方案是在有全集情况下设计的, 在计算过程中会泄露交集的势且存在一定的误判. 在半诚实模型下基于 Paillier 同态加密算法设计了 3 个协议, 协议 1 计算共有标识符的数量 (即用户标识符交集的势) 以及与这些用户相关联的整数值之和, 协议 2 和协议 3 是在不泄露交集势的情况下计算交集元素关联值之和. 整个计算过程不泄露关于协议双方私人输入的任何更多信息. 所提协议是在无全集情况下设计的, 采用模拟范例证明了所设计协议的安全性, 用实验验证协议的高效性.

关键词: 密码学; 安全多方计算; 交集和; 同态加密; 随机置换

中图法分类号: TP309

中文引用格式: 李顺东, 张凯鑫, 杨晨, 汪榆淋. 集合交集元素之和的保密计算. 软件学报, 2023, 34(7): 3343–3353. http://www.jos.org.cn/1000-9825/6529.htm

英文引用格式: Li SD, Zhang KX, Yang C, Wang YL. Secure Intersection-sum Computation. Ruan Jian Xue Bao/Journal of Software, 2023, 34(7): 3343–3353 (in Chinese). http://www.jos.org.cn/1000-9825/6529.htm

Secure Intersection-sum Computation

LI Shun-Dong[1], ZHANG Kai-Xin[1], YANG Chen[1], WANG Yu-Lin[2]

[1](School of Computer Science, Shaanxi Normal University, Xi'an 710119, China)

[2](School of Mathematics and Statistics, Shaanxi Normal University, Xi'an 710119 China)

Abstract: Secure multi-party computation is one of the hot issues in international cryptographic community. The secure computation of intersection-sum is a new problem of secure multi-party computation. The problem has important theoretical significance and practical value in the fields of industry, commerce, and healthcare. The existing solutions are designed under the condition that the private sets are subsets of a universal set and the intersection cardinality will be leaked and there are some false probabilities. This study, based on the Paillier cryptosystem, designs three protocols for the intersection-sum problem. These protocols are secure in the semi-honest model. Protocol 1 privately computes the number of common identifiers (i.e., user identifier intersection cardinality) and the sum of the integer values associated with these users, Protocol 2 and Protocol 3 privately compute the sum of the associated integer values of intersection elements without leaking the intersection cardinality. The whole computation process does not reveal any more information about their private inputs except for the intersection-sum. The protocols do not restrict that the private sets are subsets of a universal set, and they can be applied in more scenarios. It is proved, by using the simulation paradigm, that these protocols are secure in the semi-honest model. The efficiency of the protocols is also tested by experiments.

Key words: cryptography; secure multi-party computation; intersection-sum; homomorphic encryption; random permutation

1 引 言

大数据时代, 海量数据的交叉计算可以为科研、医疗、金融等领域提供更好的支持. 许多企业或组织出于信

* 基金项目: 国家自然科学基金 (61272435)
收稿时间: 2021-06-07; 修改时间: 2021-07-25, 2021-10-18, 2021-11-02; 采用时间: 2021-11-11; jos 在线出版时间: 2022-11-30
CNKI 网络首发时间: 2022-12-01

息安全的考虑, 内部数据是不对外开放的, 形成一个个数据孤岛, 数据的价值无法体现或变现. 安全多方计算 (secure multiparty computation, SMC) 可以很好地解决这一难题, 能够在保证各方数据安全的同时, 得到预期的计算结果. SMC 最早由图灵奖获得者姚期智教授[1]在 1982 年提出, 姚教授以著名的百万富翁问题来说明安全两方计算. 百万富翁问题指的是, 在没有可信第三方的前提下, 两个百万富翁如何在不泄露自己的真实财产状况的前提下来比较谁更富有. 后经 Ben-Or 和 Goldwassers 等人[2]的发展, 成为现代密码学中非常活跃的研究领域, 即安全多方计算. 安全多方计算是指多个持有隐私数据的参与者, 共同执行一个协议 (如求最大值), 并获得计算结果, 在计算过程中所有参与者的隐私数据都不会泄露.

保密的集合计算问题是保密科学计算中的一个重要问题. 现有的集合计算问题大致可分为: 保密判定元素与集合的关系[3,4]、保密计算集合交集[5−10]、保密计算集合并集[11−13]、保密计算集合交集势[14−18]、多重集的保密计算与应用[19]、集合包含关系[20,21]、集合交集势与交集元素关联值之和[22−24]等问题. 本文研究集合交集元素关联值之和的保密计算问题, 该问题描述为: 一方拥有集合 $V=\{v_i\}_{i=1}^{n}$, 另一方拥有集合 $W=\{(w_j,t_j)\}_{j=1}^{m}$, 其中 $\{v_i\}$, $\{w_j\}$ 是标识符集合, t_j 是标识符 w_j 的整数关联值. 双方想要保密计算标识符交集元素关联值 t_j 之和, 我们将标识符交集元素关联值之和的计算问题称之为集合交集元素之和的保密计算问题. 该问题在工商业、医疗健康等领域有重要的应用价值. 文献 [24] 首次提出了集合交集元素之和的问题, 该问题可以看作是集合交集的扩展. 文献 [6,8,25−28] 对于集合交集问题的研究目前已经有了很多成果, 但计算集合交集元素关联值之和问题只有文献 [22−24], 其解决方案是借助现有交集协议和洗牌技术来计算集合交集势与交集元素关联值之和. 文献 [22] 的解决方案是在有全集情况下基于 Diffie-Hellman 困难假设来计算集合交集的势和交集元素关联值之和. 文献 [23] 解决方案是在无全集情况下基于随机不经意传输和加密的 Bloom Filter 设计的, 前者计算过程中会泄露交集的势, 而随机不经意传输和加密的 Bloom Filter 存在一定的误判. 文献 [24] 是将文献 [23] 中在半诚实模型下基于随机不经意传输设计的协议设计转化为恶意模型下的协议. 据我们了解, 本文第一次提出在无全集情况下不泄露交集的势来保密计算集合交集元素关联值之和的问题, 该问题在工商业领域的广告转化率[22]问题上有重要的应用价值.

当用户在某网站看到一个公司的在线广告, 并在该公司线上商店购买商品时, 就会产生广告转换率. 谷歌拥有浏览广告的人群列表, 广告客户拥有实际购买广告商品的人群列表. 谷歌和广告客户必须分享各自的列表, 来计算交集的大小和交集元素关联值之和. 广告客户想知道他的收入有多少可归功于在线广告. 然而, 这些归因统计所需数据分为两部分: Google 知道哪些用户浏览过某个广告, 广告客户知道哪些用户购买了产品以及他们花了多少钱. 双方可能不愿意或无法公开基础数据, 但仍希望计算有多少用户同时看到广告并购买了商品, 以及这些用户共花了多少钱. 他们希望这样做, 同时确保在输入数据集中除了这些关联值和之外, 不显示任何关于用户的信息. 广告转化率用数学语言表示为: 一方持有标识符集合 $V=\{v_i\}_{i=1}^{n}$, 对应浏览过广告的用户, 另一方持有集合是由标识符 w_j 和整数值 t_j 组成的元组所构成的集合 $W=\{(w_j,t_j)\}_{j=1}^{m}$, 分别对应的是购买了相关商品的用户和他们花了多少钱, 双方想要保密计算标识符集合交集的势及交集元素关联值之和. 谷歌利用了一种基于隐私保护交集的协议. 谷歌所使用的协议在 Ion 等人[22]的文章中有描述, 但他们的协议是在全集情况下并且有一定的误判, 在不存在全集的情况下如何保密计算集合交集元素关联值之和是一个新的挑战. 本文提出的协议可以实现无全集下精确地计算, 更大程度上满足了 Google 的要求. 本文贡献如下.

(1) 本文首次提出在不泄露交集势的情况下保密计算无全集情况下集合交集元素关联值之和.

(2) 协议 1 基于 Paillier 公钥加密方案的双秘钥技术和保密随机置换技术计算集合交集的势与交集元素关联值之和的问题, 本文提出的双秘钥加密技术可以用于解决许多安全多方计算问题.

(3) 协议 2 是在协议 1 的基础上通过添加假元素的方法来计算集合交集元素关联值之和的问题, 计算过程中不会泄露交集的势.

(4) 协议 3 是对协议 2 的改进, 通过采用哈希分桶的方式, 巧妙地避开两两元素比较, 降低了计算复杂度.

本文第 2 节介绍了协议所用到的预备知识. 第 3–5 节介绍了集合交集的势与交集元素关联值之和的保密计算协议以及各个协议的安全性、正确性证明. 第 6 节介绍了研究问题的扩展与应用. 第 7 节给出了本文协议的实验分析. 第 8 节给出了本文的总结.

2 预备知识

2.1 安全性定义

假设 Alice 和 Bob 分别拥有保密数据 x, y, 他们想保密计算概率多项式函数 $f(x,y)=(f_1(x,y),f_2(x,y))$ 使得 Alice 得到 $f_1(x,y)$, Bob 得到 $f_2(x,y)$ 而不泄露 x, y. 假设存在他们都绝对信任的第三者 TTP, 他们就可以分别将自己的保密数据 x, y 发送给 TTP. TTP 计算 $f(x,y)$, 并将 $f_1(x,y)$ 发送给 Alice, $f_2(x,y)$ 发送给 Bob. 这种借助可信第三者的协议称为理想协议. 理想协议除了揭示 $f(x,y)$ 之外 (这就是协议的目的), 没有泄露关于 x, y 的任何信息, 因此被认为是最安全的密码学协议. 如果没有他们绝对信任的 TTP, 要完成此项任务他们需要一个双方计算协议 π.

● 半诚实模型. 半诚实参与者是指在协议执行过程中严格按照协议要求履行协议, 但他们可能会因为好奇而将协议执行过程中获得的信息记录下来, 在执行完协议后试图根据记录的信息推算出其他参与者的输入信息, 因此半诚实参与者又称为诚实但好奇的参与者. 如果协议是为半诚实参与者设计的协议, 我们称协议为半诚实模型下的协议. 本文协议假设所有参与者都是半诚实的.

假设 Alice 在执行双方计算协议的过程中得到的消息序列为 $view_1^{\pi}(x,y)=(x,r^1,m_1^1,\ldots,m_i^1)$, 其中 r^1 表示 Alice 选的随机数, m_i^1 表示 Alice 收到的第 i 个信息; $f_1(x,y)$ 为 Alice 执行协议得到的输出结果. 执行协议后 Alice 能够用于推算其他参与者的输入的信息只有 $view_1^{\pi}(x,y)$. 在执行协议时 Bob 得到的信息序列和输出结果也可类似定义.

定义 1. 半诚实模型下的安全性[29]. 假设协议 π 是计算概率多项式时间函数 f 的双方计算协议, 如果能够利用 x, y 和 $f(x,y)$ 构造概率多项式时间算法 S_1 与 S_2 (也称这样的多项式时间算法为模拟器), 使得:

$$\{S_1(x,f_1(x,y))\}_{x,y}\overset{c}{\equiv}\{view_1^{\pi}(x,y)\}_{x,y},\{S_2(y,f_2(x,y))\}_{x,y}\overset{c}{\equiv}\{view_2^{\pi}(x,y)\}_{x,y} \tag{1}$$

其中, $\overset{c}{\equiv}$ 表示计算不可区分, 则协议 π 保密地计算函数 $f(x,y)$.

● 模拟范例[29]. 根据定义 1, 要证明一个两方计算协议在半诚实模型下是安全的, 就必须构造使公式 (1) 成立的模拟器 S_1 和 S_2. 利用构造模拟器证明密码协议安全性的方法被称为模拟范例. 模拟范例是密码学中广泛使用的一种证明密码协议安全性的方法, 广泛用于证明比特承诺、不经意传输协议、零知识证明协议和安全多方计算协议的安全性, 文献 [30] 给出了具体的模拟器构造实例.

模拟范例的实质是说因为理想协议是安全的, 如果参与者从实际安全多方计算协议 π 得到的信息不比从理想协议得到的信息更多 (说明从理想协议得不到的信息也无法从实际协议 π 得到), 则实际的安全多方计算协议 π 是安全的, 因为模拟器的 $f_1(x,y)$ 可以看作是从可信的第三者得到的结果. 有了 $x, f_1(x,y)$, 模拟器 S_1 就可以模拟协议的执行过程得到一个消息序列, 这个消息序列和执行实际协议 π 得到的消息序列是计算不可区分的. 参与者可以从实际协议的 $view_1^{\pi}(x,y)$ 中得到任何信息, 都可以从模拟过程中得到, 即可以从理想协议中得到; 其逆否命题则表明从理想协议得不到的信息也不能从实际协议 π 得到, 因而协议 π 是安全的.

2.2 平衡哈希[4]

假设有 n 个元素, 随机哈希函数 H 和 B 个哈希桶. Azar 等人[31]的平衡分配哈希方法可以将 n 个元素映射到 B 个桶中, 使每个桶中最多包含 k 个元素. 当 n 个元素映射完, 某个桶中元素不足 k 个时, 需要添加假元素使得每个桶中元素均为 k 个.

2.3 Paillier 密码系统

Paillier 密码系统是一种具有加法同态性的公钥密码系统, 方案是语义安全的, 具体描述如下[32].

● 密钥生成. 给定安全参数 k, 生成两个 k 比特的大素数 p, q, 计算 $N=pq$, $\lambda=lcm(p-1,q-1)$. 定义函数 $L(x)=\dfrac{x-1}{N}$, 随机选择一个生成元 $g\in Z_N^*$, 使得 $gcd(L(g^{\lambda}\bmod N^2),N)=1$. 则系统的公钥为 (g,N), 私钥为 λ. Paillier 的明文空间和密文空间分别为 Z_N 和 $Z_{N^2}^*$. 下文中, 加密算法和解密算法分别记为 E 和 D.

● 加密过程. 要加密明文 $m\in Z_N$, 选取随机数 $r\in Z_N$, 计算密文: $c=g^m r^N \bmod N^2$.

● 解密过程. 对密文 c, 计算: $m=\dfrac{L(c^{\lambda}\bmod N^2)}{L(g^{\lambda}\bmod N^2)}\bmod N$.

● 加法同态性.

$$E(m_1)\times E(m_2)=g^{m_1}r_1^N g^{m_2}r_{r_2}^N \bmod N^2=g^{m_1+m_2}(r_1r_2)^N \bmod N^2=E(m_1+m_2).$$

进一步还可得到 Paillier 加密方案的下述乘法性质:

$$E(m_1)^{m_2} \bmod N^2=E(m_1m_2).$$

Paillier 加密方案是语义安全的, 这意味一个明文可以加密成多个不同的密文形式, 并且加密所得的所有密文是计算不可区分的.

3 无全集下求集合交集的势与交集元素关联值之和

● 问题描述. Alice 拥有集合 $V=\{v_i\}_{i=1}^n$, Bob 拥有集合 $W=\{(w_j,t_j)\}_{j=1}^m$. 我们称$\{v_i\}$, $\{w_j\}$ 为标识符集合, t_j 是标识符 w_j 的关联值. 双方希望知道他们共有标识符的个数即标识符集合交集的势, 以及交集元素关联值之和, 而不泄露关于他们输入的其他任何信息.

● 计算原理. 将集合 V 中的每一个标识符 v_i $(1\leqslant i\leqslant n)$ 和集合 W 中的每一个标识符 w_j $(1\leqslant j\leqslant m)$ 进行相减, 得到 $s_{ij}=v_i-w_j$, 将 s_{ij} 和 t_j 组成一个元组 $c_{ij}=(s_{ij},t_j)$. 共需要计算 nm 次, 将元组 c_{ij} 构成集合 $C=\{c_{11},\dots,c_{1m},\dots,c_{n1},\dots,c_{nm}\}=\{(s_{11},t_1),\dots,(s_{1m},t_m),\dots,(s_{n1},t_1),\dots,(s_{nm},t_m)\}$. 如果元组 c_{ij} 中 s_{ij} 等于 0, 即 $v_i=w_j$, 说明 v_i 是标识符交集元素, 将使 $v_i=w_j$ 的 w_j 的关联值 t_j 相加, 即为所求集合交集元素关联值之和 S, 集合 C 中 s_{ij} 等于 0 的个数为交集的势 T.

例. Alice 拥有集合 $V=\{v_i\}_{i=1}^3=\{a,b,k\}$, Bob 拥有集合 $W=\{(w_j,t_j)\}_{j=1}^3=\{(a,3),(k,5),(c,8)\}$, v_i 与 w_j 为标识符, t_j 为关联值. 双方想求标识符交集的势和交集元素关联值之和.

Bob 将所有元素进行 $s_{ij}=v_i-w_j$ 运算得:

$$\begin{cases}c_{11}=(s_{11},t_1)=(a-a,3),c_{12}=(s_{12},t_2)=(a-k,5),c_{13}=(s_{13},t_3)=(a-c,8),\\ c_{21}=(s_{21},t_1)=(b-a,3),c_{22}=(s_{22},t_2)=(b-k,5),c_{23}=(s_{23},t_3)=(b-c,8),\\ c_{31}=(s_{31},t_1)=(k-a,3),c_{32}=(s_{32},t_2)=(k-k,5),c_{33}=(s_{33},t_3)=(k-c,8).\end{cases}$$

根据上述计算可知, $s_{11}=0$、$s_{32}=0$, 所以交集的势 T 为 2, 交集元素关联值之和为 $S=t_1+t_2=3+5=8$. 直接这样计算没有保密性可言, 在密文上完成上述计算过程就可以实现保密计算. 在保密计算时, 我们会将用户标识符先转化为对应的 ASCII 码来进行计算.

3.1 具体协议

协议 1. 保密计算无全集下集合交集的势与交集元素关联值之和.

输入: Alice、Bob 各自的私有集合 $V=\{v_i\}_{i=1}^n$, $W=\{(w_j,t_j\}_{j=1}^m$, $t_j<\dfrac{N}{m}$.

输出: $|\{v_i\}_{i=1}^n\cap\{w_j\}_{j=1}^m|$, $S=\sum\limits_{i:w_i\in V}t_i$.

(1) (G,D,E) 是 Paillier 同态加密方案, τ 是设定的安全参数, Alice、Bob 运行 $G(\tau)$ 分别生成公私钥对 (pk_1,sk_1), (pk_2,sk_2), pk_1、pk_2 是公开的.

(2) Alice 用自己的公钥 pk_1 加密集合 V 得到 $[V]=E_{pk_1}(V)=\{[v_i]\}_{i=1}^n$ 并发送给 Bob.

(3) Bob 用 Alice 的公钥 pk_1 加密集合 W 中的标识符 w_j, 用自己的公钥 pk_2 加密关联值 t_j 得到 $[W]=(E_{pk_1}(w_j),E_{pk_2}(t_j))_{j=1}^m=([w_j],[t_j])_{j=1}^m$.

(4) 对于每一个元素 v_i、w_j, Bob 选择随机数 r_{ij}, 根据同态性计算: $[s_{ij}]=([v_i]\times[N-w_j])^{r_{ij}}=[v_i-w_j]^{r_{ij}}=[r_{ij}(v_i-w_j)](1\leqslant i\leqslant n,1\leqslant j\leqslant m)$, Bob 得到元组 $[c_{ij}]=([s_{ij}],[t_j])$, 将每次计算结果 $[c_{ij}]$ 组成集合 $C=\{[c_{11}],\dots,[c_{1m}],\dots,[c_{n1}],\dots,[c_{nm}]\}$, 将 C 随机置换后发送给 Alice (因为 C 为集合, 置换后仍然记为 C).

(5) Alice 用私钥 sk_1 解密集合 C 中所有的 $[s_{ij}]$, 若解密结果为 0, 则将对应的 $[t_{ij}]$ 相乘, 即 $[S]=\prod\limits_{s'_{ij}=0}[t_j]$, 解密结果中 0 的个数即为标识符交集的势 T, Alice 将 $[S]$ 发送给 Bob.

(6) Bob 用私钥 sk_2 解密 $[S]$ 得到 S, 即为集合 V、W 交集元素关联值之和.

3.2 协议的正确性

定理 1. 协议 1 能正确地计算无全集下集合交集的势与交集元素关联值之和.

证明: Alice 加密集合 V 中的元素得到密文 $[V]=\{[v_i]\}_{i=1}^{n}$, Bob 加密集合 W 得到密文 $E(W)=\{([w_j],[t_j])\}_{j=1}^{m}$. 如果标识符集合有交集, 则 $[V]$ 和 $[W]$ 中各自至少有一个元素使得 $[v_i]$ 与 $[w_j]$ 相等, 即 $[s_{ij}]=[v_i-w_j]=0$. 集合 C 中 s_{ij} 等于 0 的个数为标识符交集的势, 协议 1 假设 $t_j<\dfrac{N}{m}$, 则 $\sum_{j=1}^{m}t_j$ 不超过 N, 将交集元素对应 $[t_j]$ 相乘, 解密出来即为标识符交集元素关联值之和.

3.3 协议的安全性

定理 2. 保密计算无全集下集合交集的势与交集元素关联值之和的协议 1 是安全的.

证明: 在半诚实模型下, 通过构造模拟器 S_1 和 S_2 使公式 (1) 成立来证明本定理. 在协议 1 中:

$$view_1^{\pi}(V,W)=\{V,r_1,C,f_1(V,W)\},view_2^{\pi}(V,W)=\{W,r_2,[V],f_2(V,W)\},$$

其中, V、W 是 Alice 和 Bob 的输入, r_1 是协议 1 中 Alice 加密时所选择的随机数集合, C 是 Bob 将 $[v_i]$ 与 $[w_j]$ 循环计算的结果置换后发送给 Alice 的集合, r_2 是协议 1 中 Bob 加密时选择的随机数和计算时选择的随机数组成的集合, $[V]$ 是 Alice 发送给 Bob 的密文信息. $f_1(V,W)=f_2(V,W)=(|\{v_i\}_{i=1}^{n}\cap\{w_j\}_{j=1}^{m}|,S)$ 分别是 Alice、Bob 的计算结果.

首先构造模拟器 S_1 来模拟 $view_1^{\pi}(V,W)$. S_1 模拟过程如下.

(1) 接收输入 $(V,f_1(V,W))$, 根据 $f_1(V,W)$ 的值, 选择集合 $W'=\{(w'_j,t'_j)\}_{j=1}^{m}$, 使得 $f_1(V,W')=f_1(V,W)$. S_1 运行 Paillier 密码系统的 $G(\tau)$ 分别生成 pk_1,sk_1 和 pk_2,sk_2 [30].

(2) 对 $j\in[1,m]$, S_1 选 m 个随机数, 记为 r'_1, 则 $r_1\overset{c}{\equiv}r'_1$. 用随机数和 pk_1 加密 $w'_1,\ldots,w'_m$ 得到 $[W]=\{[w'_j]\}_{j=1}^{m}$.

(3) 对于 $i=1,\ldots,n$, $j=1,\ldots,m$, S_1 选择 mn 个随机数 r'_{ij} 并利用同态性计算 $s'_{ij}=[r'_{ij}(v_i-w'_j)]$, 用 pk_2 加密 $\{t'_j\}_{j=1}^{m}$ 得到 $\{[t'_j]\}_{j=1}^{m}$, 根据协议将 $\{[r_{ij}(v_i-w'_j)]\}_1^{mn}$ 和 $\{[t'_j]\}_{j=1}^{m}$ 组合成: $C'=[c'_{11},\ldots,c'_{1m},c'_{n1},\ldots,c'_{nm}]$, 其中 $c'_{ij}=([s'_{ij}],[t'_j])$. 对 C' 进行随机置换. C 和 C' 中的 $\{[t_j]\}_{j=1}^{m}$ 和 $\{[t'_j]\}_{j=1}^{m}$ 是利用 pk_2 加密的, 因为 Paillier 密码系统是语义安全的, 所以 $\{[t_j]\}_{j=1}^{m}\overset{c}{\equiv}\{[t'_j]\}_{j=1}^{m}$.

(4) S_1 用私钥 sk_1 解密 C' 中的 s'_{ij}, 计算 $s'_{ij}=0$ 的个数就得到 $|\{v_i\}_{i=1}^{n}\cap\{w'_j\}_{j=1}^{m}|$, 因为经过随机置换, 所以无法判断哪个 w'_j 在交集中, 哪个不在交集中, 又因为 $|\{v_i\}_{i=1}^{n}\cap\{w_j\}_{j=1}^{m}|=|\{v_i\}_{i=1}^{n}\cap\{w'_j\}_{j=1}^{m}|$, 即使解密之后 s_{ij} 和 s'_{ij} 也是计算不可区分的. 令:

$$\{S_1(V,f_1(V,W))\}=\{V,r'_1,C',f_1(V,W')\},$$

则有:

$$\{S_1(V,f_1(V,W))\}_{V,W}\overset{c}{\equiv}\{view_1^{\pi}(V,W)\}_{V,W}.$$

类似地, S_2 的模拟过程构造如下.

(1) 接收输入 $(W,f_2(V,W))$, 根据 $f_2(V,W)$ 的值, 选择集合 $V'=\{v'_i\}_{i=1}^{n}$, 使得 $f_2(V',W)=f_2(V,W)$.

(2) S_2 运行 Paillier 密码系统的 $G(\tau)$ 生成公钥 pk_1,sk_1 和 pk_2,sk_2, S_2 选择 n 个随机数, 记为 r'_2, 则 $r_2\overset{c}{\equiv}r'_2$. 用这些随机数和 pk_1 加密集合 V' 得到 $[V']=\{[v'_i]\}_{i=1}^{n}$.

(3) 对每个 $i\in[1,n]$, $j\in[1,m]$, 选择 mn 个随机数 r'_{ij}, 计算如下:

$[s'_{ij}]=([v'_i]\times[N-w_j])^{r'_{ij}}=[v'_i-w_j]^{r'_{ij}}=[r'_{ij}(v'_i-w_j)]$, $C'=[c'_{11},\ldots,c'_{1m},c'_{n1},\ldots,c'_{nm}]$, 其中 $[c'_{ij}]=([s'_{ij}],[t_j])$,

对 C' 进行随机置换.

(4) S_2 用私钥 sk_1 解密 C' 中的 s'_{ij}, $s'_{ij}=0$ 的个数即为 $|\{v'_i\}_{i=1}^{n}\cap\{w_j\}_{j=1}^{m}|$. 如果 s'_{ij} 等于 0, 则将对应的 $[t_j]$ 相乘得到 $[S']$. 解密 $[S']$ 得到 S'. 在协议执行中, $view_2^{\pi}(V,W)=\{W,r_2,[V],f_2(V,W)\}$, 令:

$$S_2(W,f_2(V,W))=\{W,r'_2,[V'],f_2(V',W)\}.$$

由于 $[V']$ 是 Alice 加密的, 根据 Paillier 加密算法的语义安全性, Paillier 加密算法加密的密文都是计算不可区分的, 因此 $[V]\overset{c}{\equiv}[V']$, 因为 $f_2(V,W)=f_2(V',W)$, 故有:

$$\{S_2(W,f_2(V,W))\}_{V,W}\overset{c}{\equiv}\{view_2^{\pi}(V,W)\}_{V,W}.$$

因此, 协议 1 能够保密地计算无全集下集合交集的势与交集元素关联值之和.

4 不泄露交集势的情况

● 问题描述. Alice 拥有集合 $V=\{v_i\}_{i=1}^n$, Bob 拥有集合 $W=\{(w_j,t_j)\}_{j=1}^m$. 我们称 $\{v_i\}$, $\{w_j\}$ 为标识符集合, t_j 是标识符 w_j 的关联值, 限制集合 $\{v_i\}_{i=1}^n$, $\{w_j\}_{j=1}^m$ 为普通集. 双方希望知道 $\{v_i\}\cap\{w_j\}$ 元素关联值之和, 而不泄露交集的势及其他任何信息.

4.1 协议 2

● 计算原理. 协议 1 是计算标识符交集的势和交集元素关联值之和, 我们现在要在不泄露交集势的情况下求共有标识符交集元素关联值之和, 方法和协议 1 一样, 不同的是为了防止泄露交集势, 我们需要给集合 W 中随机添加一些假元素, 再进行计算. Bob 拥有的真实数据记为 $W=\{(w_j,t_j)\}_{j=1}^m$, 添加假元素的标识符从标识符分布中随机选择, 并记为 $T=\{(w_j,0)\}_{j=m+1}^{m+h}(h\leqslant m)$, 真假元素组成集合 $W'=\{(w_j,t_j)\}_{j=1}^{m+h}$, 限制集合 $\{v_i\}_{i=1}^n$, $\{w_j\}_{j=1}^m$ 为普通集, 添加假元素之后的集合 $\{w_j\}_{j=1}^{m+h}$ 仍为普通集. 集合 T 与 V 即使有交集, 但其对应的 t_j 为 0, 对集合 V 与 W 的交集元素和的计算结果没影响. 因此, 计算集合 $V=\{v_i\}_{i=1}^n$ 和集合 $W=\{(w_j,t_j)\}_{j=1}^m$ 中交集元素关联值之和的问题转换为求集合 $V=\{v_i\}_{i=1}^n$ 和集合 $W'=\{(w_j,t_j)\}_{j=1}^{m+h}$ 中交集元素关联值之和的问题.

例: Alice 拥有集合 $V=\{v_i\}_{i=1}^3=\{a,b,k\}$, Bob 拥有集合 $W=\{(w_j,t_j)\}_{j=1}^3=\{(a,2),(k,4),(c,6)\}$, v_i 与 w_j 为标识符, t_j 为关联值. 双方想求交集元素关联值之和.

(1) Bob 随机添 3 个假元素 $\{(a,0),(k,0),(c,0)\}$ 得到集合 $W'=\{(w_j,t_j)\}_{j=1}^6=\{(a,2),(k,4),(c,6),(a,0),(k,0),(c,0)\}$.

(2) Bob 将所有元素进行 $s_{ij}=v_i-w_j$ 运算得:

$$\begin{cases}c_{11}=(s_{11},t_1)=(a-a,2),\ c_{12}=(s_{12},t_2)=(a-k,4),\ c_{13}=(s_{13},t_3)=(a-c,6),\\ c_{21}=(s_{21},t_1)=(b-a,2),\ c_{22}=(s_{22},t_2)=(b-k,4),\ c_{23}=(s_{23},t_3)=(b-c,6),\\ c_{31}=(s_{31},t_1)=(k-a,2),\ c_{32}=(s_{32},t_2)=(k-k,4),\ c_{33}=(s_{33},t_3)=(k-c,6),\\ c_{14}=(s_{14},t_4)=(a-a,0),\ c_{15}=(s_{15},t_5)=(a-k,0),\ c_{16}=(s_{16},t_6)=(a-c,0),\\ c_{24}=(s_{24},t_4)=(b-a,0),\ c_{25}=(s_{25},t_5)=(b-k,0),\ c_{26}=(s_{26},t_6)=(b-c,0),\\ c_{34}=(s_{34},t_4)=(k-a,0),\ c_{35}=(s_{35},t_5)=(k-k,0),\ c_{36}=(s_{36},t_6)=(k-c,0).\end{cases}$$

根据上述计算可知, $s_{11}=0$、$s_{14}=0$、$s_{32}=0$、$s_{35}=0$, 0 的个数为 4, 是集合 V 与 W' 的交集势 T, 没有泄露集合 V 与 W 的交集势 2, 交集元素关联值之和为 $S=t_1+t_4+t_2+t_5=2+0+4+0=6$, S 也是集合 V 与 W 的交集元素关联值之和. 在保密计算时, 我们会将用户标识符先转化为对应的 ASCII 码来进行计算, 具体实现如下.

4.2 具体协议

协议 2. 保密计算集合交集元素关联值之和而不泄露交集的势.

输入: Alice、Bob 各自的私有集合 $V=\{v_i\}_{i=1}^n$, $W=\{(w_j,t_j)\}_{j=1}^m$, $t_j<\dfrac{N}{m}$.

输出: $S=\sum_{i:w_i\in V}t_i$.

(1) (G,D,E) 是 Paillier 同态加密方案, τ 是设定的安全参数, Alice、Bob 运行 $G(\tau)$ 分别生成公私钥对 (pk_1,sk_1), (pk_2,sk_2), pk_1、pk_2 是公开的.

(2) Alice 用自己的公钥 pk_1 加密集合 V 得到 $[V]=E_{pk_1}(V)=\{[v_i]\}_{i=1}^n$ 并发送给 Bob.

(3) Bob 随机添 h 个假元素 $T=\{(w_j,0)\}_{j=m+1}^{m+h}$, 组成集合 $W'=\{(w_j,t_j)\}_{j=1}^{m+h}$, 用 Alice 的公钥 pk_1 加密集合 W' 中的标识符 w_j, 用自己的公钥 pk_2 加密关联值 t_j 得到 $[W']=(E_{pk_1}(w_j),E_{pk_2}(t_j))_{j=1}^{m+h}=([w_j],[t_j])_{j=1}^{m+h}$.

(4) 对于每一个元素 v_i、w_j, Bob 选择随机数 r_{ij}, 根据同态性计算: $[s_{ij}]=([v_i]\times[N-w_j])^{r_{ij}}=[v_i-w_j]^{r_{ij}}=[r_{ij}(v_i-w_j)](1\leqslant i\leqslant n,1\leqslant j\leqslant m+h)$, Bob 得到元组 $[c_{ij}]=([s_{ij}],[t_j])$. Bob 需要进行 $n(m+h)$ 次计算, 将每次计算结果 $[c_{ij}]$ 组成集合 $C=\{[c_{11}],\ldots,[c_{1m+h}],\ldots,[c_{n1}],\ldots,[c_{nm+h}]\}$, 将 C 随机置换后并发送给 Alice.

(5) Alice 用私钥 sk_1 解密集合 C 中所有的 $[s_{ij}]$, 若解密结果为 0, 则将对应的 $[t_j]$ 相乘, 即为: $[S]=\prod_{s'_{ij}=0}[t_j]$. 将

$[S]$发送给 Bob.

(6) Bob 用私钥 sk_2 解密 $[S]$ 得到 S, 即为集合 V、W 交集元素关联值之和.

4.3 协议的正确性

定理 3. 协议 2 能正确地计算无全集下集合交集元素关联值之和而不泄露交集势.

证明: Alice 加密集合 V 中的元素得到密文 $[V]=\{[v_i]\}_{i=1}^{n}$, Bob 随机添加 h 个假元素, $T=\{(w_j,0)\}_{j=m+1}^{m+h}$ 和集合 W 组成集合 W' 并加密得 $E(W')=\{([w_j],[t_j])\}_{j=1}^{m+h}$. 如果标识符集合有交集, 则 $[V]$ 中一个元素最多和 $[W']$ 中的一个元素相等, 即 $[s_{ij}]=[v_i-w_j]$. Alice 知道哪些 $w_j\in V$, 但不知道这些属于 V 的 w_j 对应的 t_j 是否等于 0, 即不知道 w_j 是真元素还是假元素, 因此不知道交集的势. 因为 t_j 为 0, 且 $t_j<\frac{N}{m}$, 将交集元素关联值 $[t_j]$ 相乘, 解密出来即为集合 V 与集合 W 中交集元素关联值之和.

4.4 协议的安全性

定理 4. 保密计算无全集下集合交集元素关联值之和而不泄露交集势的协议 2 是安全的.

证明: 协议 2 的安全性完全类似于协议 1 的安全性, 可以用同样方法证明, 这里省略证明过程.

5 协议 2 的改进方案

协议 2 计算过程中需要将集合 V 中的每一个元素 v_i 和集合 W' 中所有的 w_j 进行比较, 共需要比较 $n(m+h)$ 次. 为了降低计算复杂性, 我们设计了协议 3, 先将标识符元素哈希分桶, 这样就只需要将 w_j 与其对应哈希桶里的 v_i 进行比较即可, 相当于把无全集集合转为有全集集合进行计算, 这样大大降低了计算复杂性.

● 计算原理. 采用平衡哈希的方式进行计算. Alice 和 Bob 商定一个随机哈希函数 H, Bob 确定 B 个哈希桶, 通过平衡哈希计算使得每个桶中元素个数为 k, 且没有相同元素. 当哈希桶个数 B 越大时, k 越小; B 越小时, k 越大. 当 k 越小时, 协议的性能越好 (计算复杂性越低). 为了提高协议的安全性, 通过向桶中添加假元素的方法防止泄露交集的势. 当桶内元素较少时, 可以添加适当个数的假元素; 当桶内元素较多时, 可以添加少量假元素, 因此本文在可行范围内将 k 值取得尽可能小.

Alice 将所有元素 v_i 进行哈希分桶并加密发送给 Bob, Bob 将所有元素 w_j 哈希分桶, 当某个桶中元素个数 $l<k$ 时, 添加 $p(p+l=k)$ 个假元素 $\{w_j,0\}_{j=l+1}^{k}$ 使每个桶中的元素个数为 k. 假元素中 w_j 从标识符所在分布中进行选择, 关联值均取 0. 限制集合 $\{v_i\}_{i=1}^{n}$, $\{w_j\}_{j=1}^{m}$ 为普通集, 添加假元素之后的集合 $\{w_j\}_{j=1}^{Bk}$ 仍为普通集. Bob 将 w_j 进行哈希找对应桶中元素相减得到 $s_{ji}=w_j-v_i$, 将 (s_{ji},t_j) 发送给 Alice. 若 s_{ji} 为 0, 将 t_j 相加, 即为集合交集元素关联值之和.

5.1 具体协议

协议 3. 保密计算集合交集元素关联值之和而不泄露交集的势.

输入: Alice、Bob 各自的私有集合 $V=\{v_i\}_{i=1}^{n}$, $W=\{(w_j,t_j)\}_{j=1}^{m}$, $t_j<\frac{N}{m}$.

输出: $S=\sum_{i:w_i\in V}t_i$.

(1) (G,D,E) 是 Paillier 同态加密方案, τ 是设定的安全参数, Alice、Bob 运行 $G(\tau)$ 分别生成公私钥对 (pk_1,sk_1), (pk_2,sk_2), pk_1、pk_2 是公开的, 双方商定一个随机哈希函数 H.

(2) Alice 将每个元素 v_i 哈希分桶后, 用公钥 pk_1 加密集合 V 得到 $[V]=E_{pk_1}(V)=\{[v_i]\}_{i=1}^{n}$ 并发送给 Bob.

(3) Bob 确定哈希桶的个数 B 和桶内元素个数 k. 将每一个元素 w_j 哈希分桶, 当每个哈希桶内元素个数 $l<k$ 时, 向该哈希桶中添加假元素 $\{w_j,0\}_{j=l+1}^{k}$ 使得每个桶内元素个数为 k, 用 pk_1 加密 w_j 记为 $[w_j]$, 用 pk_2 加密 t_j 记为 $[t_j]$. Bob 选择随机数 r_{ji} 将每个元素 w_j 与对应桶中 Alice 的元素进行相减, 计算得到结果 $[p_{ji}]=([w_j][N-v_i])^{r_{ji}}=[r_{ji}(w_j-v_i)]$, 将所有的计算结果 $([p_{ji}],[t_j])$ 组成集合 R, 发送给 Alice.

(4) Alice 用私钥 sk_1 对集合 R 中的 $[p_{ji}]$ 进行解密, 如果解密结果为 0, 则将 w_j 的关联值 $[t_j]$ 相乘得到 $[S]=\prod_{j:w_j\in V}[t_j]$, 并发送给 Bob.

(5) Bob 用私钥 sk_2 解密 $[S]$ 得到 S, 即为集合 V 、 W 交集元素关联值之和.

5.2 协议的正确性

定理 5. 协议 3 能正确地计算无全集下集合交集元素关联值之和而不泄露交集势.

证明: Alice 和 Bob 使用同一个哈希函数, Bob 将所有元素哈希分桶并添加假元素, Alice 也进行哈希分桶. 如果 v_i 与 w_j 相等, 则 v_i 与 w_j 相等必定哈希到同一个桶中, 则 $\left[p_{ji}\right]=\left[w_j-v_i\right]=[0]$, 将 $[t_j]$ 相乘, 即使 Bob 添加的假元素与 $\{v_i\}$ 有交集, 但其关联值为 0, 且 $t_j<\dfrac{N}{m}$, 不会影响集合 V 、 W 标识符交集元素关联值之和.

5.3 协议的安全性

定理 6. 保密计算无全集下集合交集元素关联值之和而不泄露交集势的协议 3 是安全的.

证明: 协议 3 的安全性完全类似于协议 1 的安全性, 因此省略证明.

6 扩展与应用

(1) 集合交集势及其变体

本文所设计的协议 1 会泄露交集的势, 协议 2 和协议 3 不会泄露交集的势. 交集的势是否公开在不同的应用场景中, 需求不同. 例如, 在广告转换率场景下, 双方想知道有多少人看了广告并购买了商品以及这些用户所花钱的均值或关联值和的其他一些统计计算时, 交集势的公开是有必要的. 当客户想比较不同的广告商对产品的宣传力度时, 则需要比较两个集合交集关联值之和, 这种情况就不应泄露交集的势.

在某些场景下, 一方可以拥有多个不同类型的关联值, 可以算所有关联值之和或者各个类型对应关联值之和, 也可以计算关联值之和的方差、均值等各种统计问题.

(2) 应用

当客户想知道自己在不同地域投放广告的效果时, 可以通过比较不同地域广告转换率中的集合交集元素之和来进行比较. 客户想在 Google 投放广告, 当用户搜索客户提供的这类产品或服务时, Google 就会向他们展示这类产品或服务的广告. 该广告可能会展示在 Google 搜索、Google 地图以及与 Google 合作的众多网站上. 只有当广告取得效果时才需要付费. 例如: 有用户点击广告致电客户时、访问客户的网站时, 或查询前往客户实体店的实际路线时. 针对每一种广告效果设置不同的价格, 在广告投放一个月之后, 计算客户应该向 Google 支付多少广告费用. Google 网站有一个用户 id 集合, 客户有一个用户 id 和该用户所访问不同网站时, 客户应付费用组成的集合, 计算两个 id 集合交集元素对应费用之和, 即为客户应付的广告费用.

7 效率分析

7.1 计算复杂性和通信复杂性

本文采用模指数运算次数来衡量计算复杂性, 一般 Paillier 加密算法加密一次需要 2 次模指数运算, 解密一次需要 1 次模指数运算. 但当 $g=kN+1$ 时, Paillier 加密算法加密一次需要 1 次模指数运算, 本文考虑 Paillier 加密算法加密一次需要 2 次模指数运算的情况.

● 计算复杂性. 文献 [22] 是在有全集情况下计算集合交集的势与关联值之和的问题, 文献 [23] 是在无全集情况下计算集合交集的势与关联值之和的问题, 且计算结果存在一定的误判, 本文协议 1 是在无全集情况下计算集合交集的势与关联值之和, 比文献 [22] 的协议适用范围更广, 而且能够实现精确的计算. 文献 [24] 在恶意模型下设计的, 该协议所对应的半诚实模型下的协议为文献 [23] 中基于随机不经意传输的解决方案, 因此本文协议 1 不与上述文献方案比较. 本文协议 2、协议 3 提出在无全集情况下保密计算集合交集元素之和, 本文协议 2、协议 3 进行比较即可, 不与其他文献作比较.

本文协议 1 中, Alice 加密 n 次, 解密 nm 次, Bob 加密 $2m$ 次, 解密 1 次, 即模指数运算 $m(4+n)+2n+1$ 次 (n 为集合 V 的大小, m 为集合 W 中元组的个数). 协议 2 是在协议 1 的基础上添加 h 个假元素后进行相同操作, 因此模

指数运算 $(m+h)(4+n)+2n+1$ 次. 协议 3 采用哈希分桶技术, 本文考虑 B 个桶的情况, Alice 加密 n 次, 解密 1 次, Bob 加密 $2Bk$ 次, 解密 nk 次 (k 为每个桶中的元素个数), 即模指数运算为 $2(n+2Bk)+nk+1$.

● 通信复杂性. 我们采用轮数来衡量通信复杂性, 协议 1, 2, 3 均需要 3 轮通信, 具体分析见表 1.

表 1 判断集合交集元素之和的计算复杂性和通信复杂性

比较项	本文协议1	协议2	协议3
计算复杂性	$m(4+n)+2n+1$	$(m+h)(4+n)+2n+1$	$2(n+2Bk)+nk+1$
通信复杂性	3	3	3

7.2 实验数据分析

● 实验测试环境. Windows 10 64 位操作系统, 处理器是 Intel(R) Core(TM) i5-6600CPU@3.31 GHz, 内存是 8.00 GB, 在 MyEclipse 6.6 用 Java 语言运行实现.

● 实验方法. 本实验以集合 $V=\{v_i\}_{i=1}^{n}$ 和集合 $W=\{(w_j,t_j)\}_{j=1}^{m}$ 为例, 设定集合 V 的大小为 $n=500$, 集合 W 的大小 m 依次取 1, 2, …, 20, 针对每一个 m 均进行 100 次模拟实验测试, 统计协议执行时间的平均值 (忽略协议中的预处理时间). 实验所选取的 Paillier 模数为 1 024 比特. 图 1 为本文协议 1 集合交集势与关联值之和执行时间随集合 W 个数增长的变化规律. 图 2 为本文协议 2、协议 3 集合交集关联值和执行时间随集合 W 个数增长的变化规律.

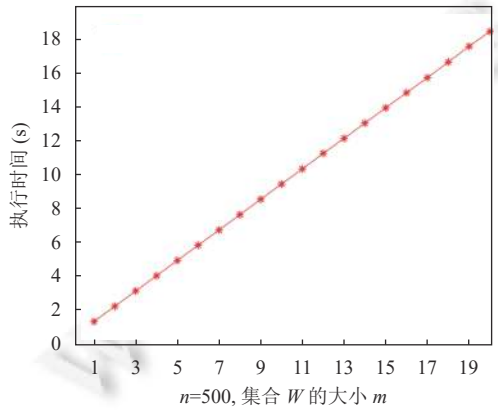

图 1 当 $n=500$ 集合交集势与交集元素关联值之和执行时间随集合 W 个数增长的变化规律

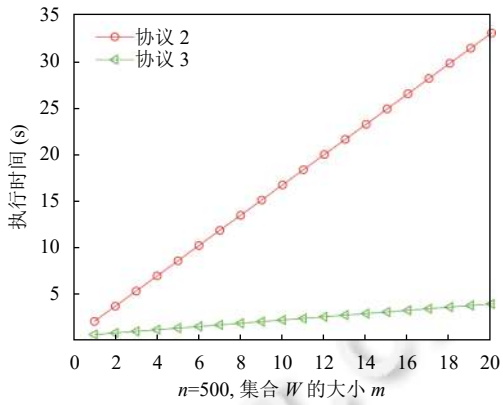

图 2 当 $n=500$ 集合交集元素关联值之和执行时间随集合 W 个数增长的变化规律 (不泄露交集的势)

图 2 实验结果表明, 在不泄露交集势计算集合交集元素关联值之和时, 基于哈希分桶技术的协议 3 比协议 2 计算复杂度明显降低.

8 结 论

保密计算无全集下集合交集元素关联值之和的问题是安全多方计算比较新的问题之一, 在工商业、医疗健康等领域有重要的应用价值. 本文基于 Paillier 同态加密算法设计了 3 个协议, 协议 1 可以计算得到交集的势与对应关联值之和, 协议 2 和协议 3 在不泄露交集势的情况下计算集合交集元素关联值之和. 针对泄不泄露交集的势这个问题, 不同的应用场景可能要求不同, 因此本文设计的 3 个协议适用范围比较广, 具有实际的应用价值. 最后本文通过理论数据分析和实验数据表明, 我们的方案可以高效地计算集合交集元素关联值之和的问题. 本文所设计的协议都是在半诚实模型下进行的, 后续我们将研究恶意模型下保密计算无全集下集合交集元素关联值之和的问题.

References:

[1] Yao AC. Protocols for secure computations. In: Proc. of the 23rd Annual Symp. on Foundations of Computer Science. Chicago: IEEE,

1982. 160–164. [doi: 10.1109/SFCS.1982.38]

[2] Ben-Or M, Goldwasser S, Wigfderson A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proc. of the 20th Annual ACM Symp. on Theory of Computing. Chicago: Association for Computing Machinery, 1988. 1–10. [doi: 10.1145/62212.62213]

[3] Li SD, Wang DS, Dai YQ. Symmetric cryptographic protocols for extended millionaires' problem. Science in China Series F: Information Sciences, 2009, 52(6): 974–982. [doi: 10.1007/s11432-009-0109-6]

[4] Freedman MJ, Nissim K, Pinkas B. Efficient private matching and set intersection. In: Cachin C, Camenisch JL, eds. Advances in Cryptology (EUROCRYPT 2004). Lecture Notes in Computer Science, vol. 3027. Interlaken: Springer, 2004. 1–19. [doi: 10.1007/978-3-540-24676-3_1]

[5] Resenede ACD, de Freitas Aranha D. Faster unbalanced Private Set Intersection in the semi-honest setting. Journal of Cryptographic Engineering, 2021, 11: 21–38. [doi: 10.1007/s13389-020-00242-7]

[6] Falk BH, Noble D, Ostrovsky R. Private set intersection with linear communication from general assumptions. In: Proc. of the 18th ACM Workshop on Privacy in the Electronic Society. London: Association for Computing Machinery, 2019. 14–25. [doi: 10.1145/3338498.3358645]

[7] Le PH, Ranellucci S, Gordon SD. Two-party private set intersection with an untrusted third party. In: Proc. of the 2019 ACM SIGSAC Conf. on Computer and Communications Security. London: Association for Computing Machinery, 2019. 2403–2420. [doi: 10.1145/3319535.3345661]

[8] Ciampi M, Orlandi C. Combining private set-intersection with secure two-party computation. In: Catalano D, de Prisco R, eds. Security and Cryptography for Networks (SCN 2018). Lecture Notes in Computer Science, vol. 11035. Amalfi: Springer, 2018. 464–482. [doi: 10.1007/978-3-319-98113-0_25]

[9] Wang ZS, Banawan K, Ulukus S. Multi-party private set intersection: An information-theoretic approach. IEEE Journal on Selected Areas in Information Theory, 2021, 2(1): 366–379. [doi: 10.1109/JSAIT.2021.3057597]

[10] Debnath SK, Sakurai K, Dey K, Kundu N. Secure outsourced private set intersection with linear complexity. In: Proc. of the 2021 IEEE Conf. on Dependable and Secure Computing (DSC). Aizuwakamatsu: IEEE, 2021. 1–8. [doi: 10.1109/DSC49826.2021.9346230]

[11] Seo JH, Cheon JH, Katz J. Constant-round multi-party private set union using reversed laurent series. In: Fischlin M, Buchmann J, Manulis M, eds. Public Key Cryptography (PKC 2012). Lecture Notes in Computer Science, vol. 7293. Darmstadt: Springer, 2012. 398–412. [doi: 10.1007/978-3-642-30057-8_24]

[12] Blanton M, Aguiar E. Private and oblivious set and multiset operations. Int'l Journal of Information Security, 2016, 15(5): 493–518. [doi: 10.1007/s10207-015-0301-1]

[13] Chun JY, Hong D, Jeong IR, Lee DH. Privacy-preserving disjunctive normal form operations on distributed sets. Information Sciences, 2013, 231: 113–122. [doi: 10.1016/j.ins.2011.07.003]

[14] Debnath SK, Stănică P, Kundu N, Choudhury T. Secure and efficient multiparty private set intersection cardinality. Advances in Mathematics of Communications, 2021, 15(2): 365–386. [doi: 10.3934/amc.2020071]

[15] Branco P, Döttling N, Pu SH. Multiparty cardinality testing for threshold private intersection. In: Garay JA, ed. Public-key Cryptography (PKC 2021). Lecture Notes in Computer Science, vol. 12711. Springer, 2021. 32–60. [doi: 10.1007/978-3-030-75248-4_2]

[16] Debnath SK, Dutta R. Secure and efficient private set intersection cardinality using bloom filter. In: Lopez J, Mitchell C, eds. Information Security (ISC 2015). Lecture Notes in Computer Science, vol. 9290. Trondheim: Springer, 2015. 209–226. [doi: 10.1007/978-3-319-23318-5_12]

[17] Egert R, Fischlin M, Gend D, Jacob S, Senker M, Tillmanns J. Privately computing set-union and set-intersection cardinality via Bloom filters. In: Foo E, Stebila D, eds. Information Security and Privacy (ACISP 2015). Lecture Notes in Computer Science, vol. 9144. Brisbane: Springer, 2015. 413–430. [doi: 10.1007/978-3-319-19962-7_24]

[18] Dou JW, Liu XH, Zhou SF, Li SD. Efficient secure multiparty set operations protocols and their application. Chinese Journal of Computers, 2018, 41(8): 1844–1860 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2018.01844]

[19] Dou JW, Chen MY. Secure multiset operations and their applications. Acta Electronica Sinica, 2020, 48(1): 204–208 (in Chinese with English abstract). [doi: 10.3969/j.issn.0372-2112.2020.01.025]

[20] Chen ZH, Li SD, Huang Q, Ding Y, Liu YR. Secure computation of two set-relationships with the unencrypted method. Ruan Jian Xue Bao/Journal of Software, 2018, 29(2): 473–482 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5262.htm [doi: 10.13328/j.cnki.jos.005262]

[21] Zhou SF, Li SD, Dou JW, Geng YL, Liu X. Efficient secure multiparty subset computation. Security and Communication Networks, 2017, 2017: 9717580. [doi: 10.1155/2017/9717580]

[22] Ion M, Kreuter B, Nergiz E, Patel S, Saxena S, Seth K, Shanahan D, Yung M. Private intersection-sum protocol with applications to attributing aggregate ad conversions. IACR Cryptology ePrint Archive, 2017, 2017: 738–751.

[23] Ion M, Kreuter B, Nergiz AE, Patel S, Saxena S, Seth K, Raykova M, Shanahan D, Yung M. On deploying secure computing: Private intersection-sum-with-cardinality. In: Proc. of the 2020 IEEE European Symp. on Security and Privacy (EuroS&P). Genoa: IEEE, 2020. 370–389. [doi: 10.1109/EuroSP48549.2020.00031]

[24] Miao P, Patel S, Raykova M, Seth K, Yung M. Two-sided malicious security for private intersection-sum with cardinality. In: Micciancio D, Ristenpart T, eds. Advances in Cryptology (CRYPTO 2020). Lecture Notes in Computer Science, vol. 12172. Santa Barbara: Springer, 2020. 3–33. [doi: 10.1007/978-3-030-56877-1_1]

[25] Kolesnikov V, Kumaresan R, Rosulek M, Trieu K. Efficient batched oblivious PRF with applications to private set intersection. In: Proc. of the 2016 ACM SIGSAC Conf. on Computer and Communications Security. Vienna: Association for Computing Machinery, 2016. 818–829. [doi: 10.1145/2976749.2978381]

[26] Pinkas B, Rosulek M, Trieu N, Yanai A. SpOT-light: Lightweight private set intersection from sparse OT extension. In: Boldyreva A, Micciancio D, eds. Advances in Cryptology (CRYPTO 2019). Lecture Notes in Computer Science, vol. 11694. Santa Barbara: Springer, 2019. 401–431. [doi: 10.1007/978-3-030-26954-8_13]

[27] Pinkas B, Schneider T, Weinert C, Wieder U. Efficient circuit-based PSI via cuckoo hashing. In: Nielsen J, Rijmen V, eds. Advances in Cryptology (EUROCRYPT 2018). Lecture Notes in Computer Science, vol. 10822. Tel Aviv: Springer, 2018. 125–157.

[28] Rindal P, Rosulek M. Improved private set intersection against malicious adversaries. In: Coron JS, Nielsen J, eds. Advances in Cryptology (EUROCRYPT 2017). Lecture Notes in Computer Science, vol. 10210. Paris: Springer, 2017. 235–259. [doi: 10.1007/978-3-319-56620-7_9]

[29] Goldreich O. Foundations of Cryptography: Vol. 2, Basic Applications. London: Cambridge University Press, 2004. 599–764.

[30] Lindell Y. How to simulate it—A tutorial on the simulation proof technique. In: Lindell Y, ed. Tutorials on the Foundations of Cryptography. Information Security and Cryptography. Springer, 2017. 277–346. [doi: 10.1007/978-3-319-57048-8_6]

[31] Boudot F, Schoenmakers B, Traoré J. A fair and efficient solution to the socialist millionaires' problem. Discrete Applied Mathematics, 2001, 111(1–2): 23–36. [doi: 10.1016/S0166-218X(00)00342-5]

[32] Paillier P. Public-key cryptosystems based on composite degree residuosity classes. In: Stern J, ed. Advances in Cryptology (EUROCRYPT 1999). Lecture Notes in Computer Science, vol. 1592. Prague: Springer, 1999. 223–238. [doi: 10.1007/3-540-48910-X_16]

附中文参考文献:

[18] 窦家维, 刘旭红, 周素芳, 李顺东. 高效的集合安全多方计算协议及应用. 计算机学报, 2018, 41(8): 1844–1860. [doi: 10.11897/SP.J.1016.2018.01844]

[19] 窦家维, 陈明艳. 多重集的保密计算及应用. 电子学报, 2020, 48(1): 204–208. [doi: 10.3969/j.issn.0372-2112.2020.01.025]

[20] 陈振华, 李顺东, 黄琼, 丁勇, 刘娅茹. 非加密方法安全计算两种集合关系. 软件学报, 2018, 29(2): 473–482. http://www.jos.org.cn/1000-9825/5262.htm [doi: 10.13328/j.cnki.jos.005262]

李顺东(1963－), 男, 博士, 教授, 博士生导师, 主要研究领域为公钥密码学, 安全多方计算.

杨晨(1996－), 女, 硕士生, 主要研究领域为密码学, 信息安全.

张凯鑫(1996－), 女, 硕士生, 主要研究领域为密码学与信息安全.

汪榆淋(1997－), 女, 硕士生, 主要研究领域为应用数学, 密码学.