# 一种有效且差分隐私的安全分布式基数估计协议

Pinghui Wang*西安交通大学 中国西安
w
angpinghui369@gmail.com

Chengjin Yang 西安交通大学 中国西安
cjyang2643@stu.xjtu.edu.cn

Dongdong Xie 西安交通大学 中国西安
xdddd1
@stu.xjtu.edu.cn

Junzhou Zhao 西安交通大学 中国西安
junzhou.zhao@xjtu.edu.cn

Hui Li 西安电子科技大学 中国西安
hli
@xidian.edu.cn

Jing Tao 西安交通大学 中国西安
jtao@mail.xjtu.edu.cn

Xiaohong Guan 西安交通大学 中国西安
xhguan
@mail.xjtu.edu.cn

arXiv:2302.02158v1 [cs.CR] 4 Feb 2023

## 摘要

对分布在多个数据持有方中的不同元素进行计数，是一项基础问题，在人群计数、网络监控等诸多实际应用中发挥着重要作用。尽管已有多种空间和计算效率高的草图方法（例如 Flajolet-Martin 草图和 HyperLogLog 草图）被提出，用于解决上述基数估计问题，但在考虑与使用各数据持有方个人数据集相关的隐私问题时，这些草图方法并不安全。尽管近期有研究提出了一种协议，在基于秘密共享的多方计算（MPC）框架上成功实现了著名的 Flajolet-Martin（FM）草图，以解决隐私保护分布式基数估计（PDCE）问题，但我们发现该 MPC-FM 协议并不具备差分隐私。此外，MPC-FM 协议的计算开销较大，因此难以应用于计算资源有限的数据持有方。为解决上述问题，本文提出了一种新型协议 *DP-DICE*，该协议计算效率高且具有差分隐私，可用于解决 PDCE 问题。实验结果表明，在相同安全要求下，与当前最先进的方法相比，我们的 DP-DICE 实现了数个数量级的加速，并将估计误差降低了数倍。

**ACM 参考文献格式：**
Pinghui Wang、Chengjin Yang、Dongdong Xie、Junzhou Zhao、Hui Li、Jing Tao 和 Xiaohong Guan。2023年。一种有效且差分隐私的安全分布式基数估计协议。载于 *ACM* 会议论文集（*Conference'17*）。ACM，纽约，纽约州，美国，15 页。 https: //doi.org/10.1145/nnnnnnn.nnnnnnn

*Pinghui Wang 是通讯作者。

*Conference'17*，*2017*年7月，美国华盛顿哥伦比亚特区©2023年计算机协会。本文为作者版本，仅供个人使用。请勿再分发。最终版本发表于*ACM* 会议论文集（*Conference'17*），https://doi.org/10.1145/nnnnnnn.nnnnnnn。

## 1 引言

分布式基数计算（DCC）是指统计多个数据集的并集中不同元素数量的任务，这对许多现实应用而言都至关重要。下面举几个例子来说明其背景。

**物流监测**。RFID 系统已越来越多地部署于物流和交通运输领域 [1]。RFID 系统通常会在分散的位置部署 RFID读写器，读写器通过 RFID标签收集附近货物的信息。在一段时间内，位于不同地点的 RFID读写器可能会观测到相同的货物，因为货物（例如车辆）会随时间移动。计算物流统计数据（例如某一区域内 RFID读写器在一段时间内观测到的不同车辆数量），可以反映该区域的拥堵程度。

**疾病流行病学**。 医疗机构、政府机构和保险公司中的研究人员可能想知道一个国家中患有糖尿病等特定疾病的患者人数 [2]。医疗记录分散在各家医院中，而患者往往会在多家医院就医。因此，上述问题可以表述为一项 DCC 任务。

**受众覆盖报告**。 在互联网上，广告主通常会在多个发布商平台上开展广告活动，这些发布商代表广告主展示广告。广告活动的覆盖人数定义为至少通过一个发布商接触到该广告活动的独立用户数量，这是评估广告活动效果的一项关键指标 [3] [4]。不同发布商覆盖的用户集合可能彼此重叠。因此，覆盖人数的计算可以表述为一项 DCC 任务。

**互联网流量统计的收集**。计数在线客户端中不同客户端的数量，是网络测量与监控的一项基础任务。有时，关注的流量可能分布在属于不同持有方的多个设备上（例如，由不同互联网服务提供商持有的路由器，以及匿名通信网络中的匿名网络出口节点 [5]），而同一客户端可能出现在多个此类设备上。

当所有数据都集中收集并存储时，基数计算任务很容易完成。然而，有时数据可能分布在多个数据持有方（DH）之间，由于数据量庞大且 DH 的资源有限，收集全部数据并不可行。为解决分布式基数估计问题，一种直接的方法是分别在本地计算每个 DH 数据集的基数，然后将所有 DH 的基数相加。然而，由于不同 DH 的数据集中可能有许多共同元素，使用这种聚合方式来估计并集数据集的基数可能会产生较大的估计误差。为应对这一挑战，已有多种草图方法 [6] 被提出，例如 Flajolet-Martin（简称 FM）草图 [7] 和 HyperLogLog（简称 HLL）草图 [8]。这些方法的关键在于，为每个 DH 的数据集构建紧凑且可合并的草图（即数据摘要）。有了多个 DH 数据集对应的可合并草图后，就可以轻松将这些草图合并为一个涵盖所有 DH 数据集并集的草图。随后，便可依据合并后的草图估计并集的基数。这些特性显著降低了分布式基数估计任务所需的内存、计算和网络开销。

除了效率和准确性之外，数据隐私对许多应用也至关重要。在上述示例中，众所周知，RFID 系统收集的商品标签以及医院和互联网发布者收集的个人记录可能造成严重的隐私风险，因为这些数据可能会泄露个人的机密信息（例如，轨迹）。因此，我们需要一种解决方案，能够有效解决分布式基数估计问题，同时不泄露各数据持有方收集的个人隐私信息。

考虑到隐私问题，上述草图方法都无法胜任，因为它们生成的草图包含与个人隐私相关的机密信息。尽管 [9, 10]发现，许多流行的草图方法（包括 FM 和 HLL）只要 $n$（感兴趣集合的基数）大于某个常数，就能保护差分隐私，但这些方法假设过程中使用的函数也应保密；而在隐私保护分布式基数估计（PDCE）场景中，这一要求并不容易满足。

为解决上述问题，Hu 等人近期 [11]设计了一种协议，该协议基于秘密共享的多方计算（MPC）框架，成功实现了著名的 FM 草图，从而显著提升了现有方法的性能。尽管该 MPC-FM 协议采用安全计算来防范收集和合并从数据持有方收集的 FM 草图时遭受的攻击，但诚实但好奇的数据持有方仍可从协议输出中推断出与个人隐私相关的机密信息（详情请参见第 3.5 节）。此外，MPC-FM 协议给数据持有方带来了沉重的计算负担，限制了其在计算资源有限的边缘设备（例如物联网设备）上的应用。为应对上述挑战，本文提出了一种新协议 *DP-DICE*，用于解决 PDCE 问题。我们证明了 DP-DICE 协议是安全且满足差分隐私的。此外，与先进协议 MPC-FM 相比，它具有更高的准确性和计算效率 [11]。我们的主要贡献总结如下。

- 我们揭示，先进协议 MPC-FM [11]并不满足差分隐私，这与原论文中的声明不一致 [11]。为保证差分隐私，可以在计算基数估计的过程中加入一定量的噪声。但遗憾的是，这会导致较大的估计误差。

- 我们提出了一种新颖的协议 *DP-DICE* 来解决 PDCE 问题。我们的 DP-DICE 基于本文提出的一种快速且准确的草图方法 *FMS*。与草图方法 FM 和 HLL 相比，*FMS* 结合差分隐私的加性噪声机制时，估计误差显著更小。为使用 FMS 解决 PDCE 问题，我们设计了一种新的分布式加性噪声机制，以确保 DP-DICE 协议满足差分隐私。

- 我们在多种数据集上进行了广泛的实验。与最先进的方法相比，我们的 DP-DICE 将数据持有方上的计算速度提升了数个数量级，并将估计误差降低了数倍，同时达到相同的安全要求。

## 2 问题定义

### 2.1 PDCE问题

给定 $d$ 数据持有方（DH），其中每个 DH $j \in \{1, \ldots, d\}$ 持有一个集合 $S_j$，该集合由全集 $U$ 中的元素组成。令 $S$ 为所有集合 $S_j$ 的并集，即 $S = \bigcup_{j=1}^{d} S_j$，集合 $S$ 的基数记为 $n$，即 $n = |S|$。例如，某个 DH 可以代表一家医院，$S_j$ 是医院 $j$ 中的患者集合，那么 $n$ 就是所有这些医院中出现的不同患者数量。

我们的目标是在不泄露元素 $e$ 属于集合 $S_j$ 这一信息的情况下估计 $n$。在许多现实应用中，此类成员关系信息属于机密，且通常与个人隐私有关。例如，泄露某人是某家医院患者的信息，可能会损害该人的健康隐私。

我们将这项任务称为隐私分布式基数估计（PDCE）问题。该问题具有挑战性，因为集合 $S_1, \ldots, S_d$ 中的不同集合可能包含共同元素，因此我们无法仅通过对每个集合 $S_j$、$j = 1, \ldots, d$ 的基数求和来获得 $n$。

### 2.2 威胁模型与隐私目标

在现实应用中，可能存在大量计算和网络资源有限的 DH。因此，直接在这些 DH 上执行安全多方计算（MPC）是不可行的。遵循 [11], 的框架，我们要求满足以下假设。

- **假设 1**。存在 $c$ 计算方（CP），其中至少有一个 CP 是可信的。不可信的 CP 被建模为已被腐化，并由单一对手控制。对手可以在这些不可信的 CP 上任意行事（例如，篡改任何不可信 CP 上的数据）。
- **假设2**。每个 DH $j$ 负责收集集合 $S_j$，并在各 CP 之间安全共享 $S_j$（或 $S_j$ 的摘要）。相反，所有 CP 负责根据其收集的数据进行协作式安全近似 $n$。

• **假设3**。 基数 $n$ 的估计结果对所有 DH、CP 和对手公开可用。对于许多现实应用而言，这一假设是切实可行的，因为出于研究和商业目的，基数估计可能会向公众公开。

• **假设4**。 数据持有方会忠实地执行协议，但可能会试图合法地获取所有可能的信息。值得注意的是，[11] 假定每个数据持有方都是诚实且 不窥探隐私的，这对于某些现实应用并不切实际。例如，数据持有方（如医院）可能会好奇某个元素 $e \in U$（例如，公民身份证号码为 $e$的某个人）是否由其他数据持有方持有。

基于上述假设，我们旨在设计一种安全协议，以估计并集 $n$ 的基数 $S$，并实现以下目标：

• **目标1**。 对手无法从执行协议中获知任何信息，除协议输出外，并且无法更改该输出。

• **目标2**。 对手若不被发现，就无法破坏计算的正确性。 • **目标3**。 该协议符合差分隐私。具体而言，给定协议的输出，对手无法推断任何元素 $e \in U$ 是否属于集合 $S$。此外，某个数据持有方也无法推断任何元素 $e \in U$ 是否由其他数据持有方持有；

• **目标 4.** 所有 CP 和 DH 上的计算，以及这些参与方之间的通信，均尽可能少。

## 3 预备知识

本节首先介绍经典的基数估计方法：Flajolet-Martin（FM）草图 [7] 和 HyperLogLog（HLL）草图 [12][8]。随后，我们介绍差分隐私的定义。最后，我们简要回顾用于解决 PDCE问题的先进协议 [11]，并讨论其缺陷；本文将针对这些缺陷提出解决方案。

### 3.1 FM 草图

Flajolet-Martin（FM）草图是一种经典的数据结构，可高效估计大型集合 $S$的基数（即不同元素的数量）；该集合可能以包含重复元素的序列形式给出。如图 1 的第一列所示，FM 草图 $\{B_i, h_i\}_{i=1,\ldots,m}$由 $m$个位数组 $B_1, \ldots, B_m$和 $m$个相互独立的哈希函数 $h_1, \ldots, h_m$ 组成。每个位数组 $B_i$ 包含 $w$ 位，初始值均为零。用于更新数组 $B_i$的每个哈希函数 $h_i$都会将一个元素 $e \in U$映射为从集合 $\{0,1\}^{w-1}$ 中均匀随机选取的一个（$w-1$）位位串 $h_i(e)$。集合 $S$的 FM 草图按如下方式计算：枚举每个元素 $e \in S$，并将每一位 $B_i[\rho(h_i(e))] = 1, i = 1, \ldots, m$设为相应值，其中函数 $\rho(h_i(e))$ 返回 $h_i(e)$ 的尾随零个数。[1]集合 $S$的基数 $n$可估计为 $\hat{n} \approx 2^{\frac{Z^*}{m}} \varphi^{-1}$，其中 $\varphi$ 约为 0.77351，是一个校正因子；变量 $Z^*$ 定义如下：

$$Z^* \triangleq \sum_{i=1}^{m} z_i. \tag{1}$$

FM 草图可用于解决分布式基数估计问题，即近似估计集合 $S_1$、……、$S_d$ 的并集基数。给定每个集合的 FM 草图 $\{(B_{(j)i}, h_i)\}_{i=1,\ldots,m}$ 集合 $S_j, j = 1$、……、$d$。注意，所有这些集合都使用相同的哈希函数 $h_1$、……、$h_m$。然后，集合 $S = S_1 \cup \cdots \cup S_d$ 的并集的 FM 草图 $\{(B_i, h_i)\}_{i=1,\ldots,m}$ 计算为 $B_i = B_{(1)i} \vee \cdots \vee B_{(d)i}, i = 1, \ldots, m$，其中 $\vee$ 表示按位或运算。

[1]例如，$\rho(1011) = 0$，$\rho(1010) = 1$，$\rho(1100) = 2$，$\rho(1000) = 3$，$\rho(0000) = 4$。

### 3.2 HLL 草图

集合基数。与早期的 FM 草图相比，HLL 草图所需的内存空间和计算操作更少。其更新过程如图 1 第二列所示。该过程发现，具有 $B_i[z_i] = 1$ 的最大索引 $z_i$ 也可用于预测基数。HLL 草图不直接在内存中存储数组 $B_i$，而是使用一个寄存器跟踪每个数组 $B_i$ 的 $z_i$，从而显著降低内存使用量，将其从 $w \times m$ 位降至 $\lceil \log_2 w \rceil \times m$ 位。此外，HLL 草图采用随机平均技术，将处理每个元素的复杂度从 $O(m)$ 降至 $O(1)$。形式上，集合 $S$ 的 HLL 草图由 $m$ 个寄存器 $R[1]$、……、$R[m]$ 组成，初始值均为 0。为简单起见，令 $m$ 为 $2^r$，其中 $r$ 是一个正整数。此外，使用哈希函数 $h$ 将元素 $e \in S$ 映射为一个从集合 $\{0,1\}^{r+w-1}$ 中随机均匀选取的 $(r+w-1)$ 位串 $h(e)$。对于每个元素 $e \in S$，HLL 草图将其哈希串 $h(e)$ 分为两部分：$h^{(\mathrm{L})}(e)$ 和 $h^{(\mathrm{H})}(e)$，分别由字符串 $h(e)$ 的最低 $r$ 位和最高 $w-1$ 位组成。然后，HLL 草图按如下方式更新一个特定寄存器 $R\left[h^{(\mathrm{L})}(e)+1\right]$：

$$R\left[h^{(\mathrm{L})}(e)+1\right] \leftarrow \max\left(R\left[h^{(\mathrm{L})}(e)+1\right], \rho(h^{(\mathrm{H})}(e))\right),$$

其中，$\rho(h^{(\mathrm{H})}(e))$ 返回 $h^{(\mathrm{H})}(e)$ 的尾随零数量。

给定集合 $S$ 的 HLL 草图，基数 $S$ 估计为 $\hat{n} = \alpha_m m^2 \left(\sum_{i=1}^{m} 2^{-R[i]}\right)^{-1}$，其中 $\alpha_m = \left(m \int_0^\infty \left(\log \frac{2+u}{1+u}\right)^m du\right)^{-1}$。当 $\alpha_m m^2 \left(\sum_{i=0}^{m-1} 2^{-R_{v,a}[i]}\right)^{-1} < \frac{5}{2}m$ 时，上述估计值 $\hat{n}$ 会产生较大的估计误差。在这种情况下，基数 $n$ 可更准确地估计为 $\hat{n} = -m \ln \frac{E}{m}$，其中 $E$ 定义为 $R[1], \ldots, R[m]$ 中值为 0 的寄存器数量。

与 FM 草图类似，HLL 草图也可合并。具体而言，给定每个集合 $S_j, j = 1, \ldots, d$ 的 HLL 草图 $R_{(j)}$。同样，所有合并后的草图共享同一个哈希函数 $h$。然后，并集 $S = S_1 \cup \cdots \cup S_d$ 的 HLL 草图 $R$ 计算为$R[i] = \max_{j=1,\ldots,d} R_{(j)}[i], i = 1, \ldots, m$。

### 3.3 差分隐私

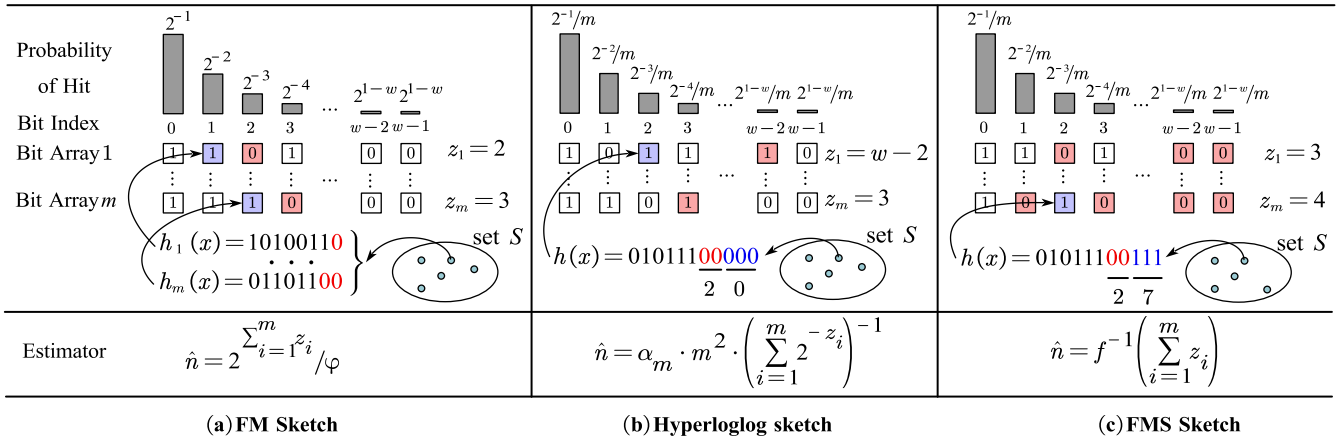
差分隐私是一种优雅的隐私数学定义，其形式如下：

**定义 1. 差分隐私（数据提供方）** [13]. 随机化算法 $M: \mathrm{D} \to \mathrm{Y}$ 满足 $(\varepsilon, \delta)$-差分隐私，当且仅当对于所有成对的 $x, x' \in \mathrm{D}$（至多相差一个元素），以及所有事件 $E \subseteq \mathrm{Y}$，均有

$$P(M(x) \in E) \leq \exp(\varepsilon) P(M(x') \in E) + \delta.$$

**定义 2. 集中差分隐私（CDP）** [14,15]。随机化算法 $M: \mathrm{D} \to \mathrm{Y}$ 满足 $\frac{1}{2}\varepsilon^2$-集中差分隐私，当且仅当对于所有最多相差一个元素的 $x, x' \in \mathrm{D}$ 对，我们有

$$\sup_{\alpha \in (1,\infty)} \frac{1}{\alpha} D_\alpha(M(x) || M(x')) \leq \frac{1}{2}\varepsilon^2,$$




(a) FM Sketch (b) Hyperloglog sketch (c) FMS Sketch

**图 1：对比 FM 草图和 HLL 草图，展示我们的 FMS 草图 [7]。 [8]（a）FM 草图。蓝色方框表示被更新为 1 的位；每个数组中的红色方框表示位索引最小的零，记作 $z_i$， $i = 1, \ldots\ldots, m$。红色数字表示尾随零。（b）HLL 草图。蓝色方框表示被更新为 1 的位；每个数组中的红色方框表示位索引最大的 1，记作 $z_i$， $i = 1, \ldots\ldots, m$。红色数字表示尾随零，蓝色数字表示位数组索引的二进制形式。（c）我们的 FMS 草图。蓝色方框表示被更新为 1 的位；红色方框表示所有值为零的位；红色数字表示尾随零，蓝色数字表示所选位数组索引的二进制形式。数组 $i$中的零位数量记作 $z_i$， $i = 1, \ldots\ldots, m$。**

其中， $D_\alpha(M(x)||M(x\prime))$ 是 $M(x)$ 的分布相对于 $M(x\prime)$ 的分布的 $\alpha$-Rényi 散度。形式上，我们将 $D_\alpha(M(x)||M(x\prime))$ 定义为：

$$D_\alpha(M(x)||M(x')) = \frac{1}{\alpha-1}\ln \mathrm{E}_{Y\sim Q}\left(\left(\frac{Q(Y)}{Q'(Y)}\right)^{\alpha-1}\right),$$

其中，函数 $Q(\cdot)$ 和 $Q\prime(\cdot)$ 分别表示 $M(x)$ 和 $M(x\prime)$ 的分布。

**差分隐私与集中差分隐私之间的关系。**Kairouz 等人 [16] 发现，若随机化算法 $M$ 满足 $(\varepsilon,0)$-差分隐私，则它满足 $\frac{1}{2}\varepsilon^2$-集中差分隐私。若算法 $M$ 满足 $\frac{1}{2}\varepsilon^2$-集中差分隐私，则对于任意 $\delta > 0$， $M$ 满足 $(\varepsilon^*_\delta, \delta)$-差分隐私，其中 $\varepsilon^*_\delta$定义如下：

$$\varepsilon^*_\delta = \inf_{\alpha>1} 0.5\varepsilon^2\alpha + \frac{\ln(1/\alpha\delta)}{\alpha-1} + \ln(1-1/\alpha) \le 0.5\varepsilon(\varepsilon + 2\sqrt{-2\ln\delta}).$$

**定义3. 离散高斯分布 [17]。**添加从高斯分布 [18] 中抽取的噪声是保护差分隐私的常用方法。然而，有限计算机无法精确表示连续噪声，使用有限精度近似可能会导致隐私保护失效 [19]。因此，Canonne 等人 [17] 提出了离散高斯分布 $\mathrm{N}_{\mathbb{Z}}(\mu, \sigma^2)$，定义如下：

$$P(X=x) = \frac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sum_{y\in\mathbb{Z}}\exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)}, \quad x \in \mathbb{Z}.$$

并得出以下定理 .

**定理** 1. *[17]* 设 $f: \mathcal{D} \to \mathbb{Z}$ 是一个敏感度为 $\Delta$的函数。敏感度 $\Delta$ 定义为： $\max|f(x) - f(y)|$，其中最大值取遍所有在 $\mathcal{D}$ 中相差至多一个元素的数据集 和。定义随机化算法 $M: \mathcal{D} \to \mathbb{Z}$为 $M(x) = f(x) + X$，其中 $X \leftarrow \mathcal{N}_{\mathbb{Z}}(0, \Delta^2/\varepsilon^2)$。那么， $M$ 满足 $\frac{1}{2}\varepsilon^2$- 集中差分隐私。

此外， [16] 中给出的以下定理表明，独立离散高斯变量之和也可用作加性噪声，以实现差分隐私。

**定理**2. 定义一个随机化算法 $M: \mathcal{D} \to \mathbb{Z}$ 为$M(x) = f(x) + \sum_{i=1}^{d} X_i$，其中 $X_i \leftarrow \mathcal{N}_{\mathbb{Z}}(0, \sigma^2)$ $i = 1, \ldots\ldots, d$ 且 $\sigma > \frac{1}{2}$。那么， $M$满足 $\frac{1}{2}\varepsilon^2$-差分隐私，其中 $\varepsilon$等于

$$\min\left\{\sqrt{\frac{\Delta^2}{d\sigma^2} + 5\sum_{k=1}^{d-1} e^{-2\pi^2\sigma^2\frac{k}{k+1}}}, \frac{|\Delta|}{\sqrt{d}\sigma} + 10\sum_{k=1}^{d-1} e^{-2\pi^2\sigma^2\frac{k}{k+1}}\right\},$$

且 $\Delta$ 是函数 $f$的敏感度。

## 3.4 MPC-FM 协议

Hu 等人 [11]提出了用于解决隐私保护分布式基数估计（PDCE）问题的先进协议。我们将其协议称为 MPC-FM。MPC-FM 的工作方式如下：每个 DH $i = 1, \ldots, d$为其私有集合 $S_i$构建 FM 草图 $F_i$。然后，每个 DH $i$将其草图 $F_i$（一个秘密）拆分为 $c$份，每个 CP 持有一份。这里的每一份都称为一个份额。单独的份额毫无用处，只有在所有 $c$个 CP 串通时，才能重构秘密草图 $F_i$。所有 DH 使用相同的哈希函数集合来生成 FM 草图，因此所有 $F_1, \ldots, F_d$ 都可合并。换句话说，可以根据草图 $F_1, \ldots, F_d$ 的值，计算并集 $S$的 FM 草图 $F$。基于这一性质，所有 CP 通过协作并安全地合并 DH 发来的所有草图$F_1, \ldots, F_d$的份额，计算草图 $F$的份额。请注意，每个 CP 仅持有草图 $F$的一个份额，因此，只要对手未腐化至少一个 CP，对手就无法重构秘密 $F$

。最后，基于草图 $F$的份额，所有 CP 安全地计算变量 $Z^*$（定义见式（1）），这就是 MPC-FM 下基数估计的结果。为实现上述流程，MPC-FM 采用了 SPDZ [20]框架。SPDZ 是一种基于秘密份额的多方计算（MPC）方案，支持在有限域上进行安全计算，并且即使对手静态地任意腐化大量参与方，该方案仍然安全。

## 3.5 挑战

本文旨在解决以下尚未解决的挑战。

**•挑战1：MPC-FM 对 DH 而言开销高昂**。对于 MPC-FM 协议，我们发现，每个 DH $j$都需要执行 $m \times |S_j|$次哈希运算，才能为集合 $S_j$生成 FM 草图。对于较大的集合 $S_j$，其开销很高，因为 $m$通常设为数千。

**•挑战2：当 DH 诚实但好奇时，MPC-FM 会泄露个人隐私**。我们先举例说明：当哈希函数和草图的最终完整值公开时，FM 会泄露隐私。考虑协议聚合后集合 $S$的 FM 草图 SK1。攻击者可以尝试找到一个元素 $e$，使得 $S \cup \{e\}$生成不同的 FM 草图 SK2，即 $e$至少将 SK1 中的一个零位填为非零。随后，对手便能轻易推断出 $e$ 肯定不在 $S$中。当然，如果对手不知道哈希函数，就无法推断 $e$是否属于该集合。在我们的问题设定中，所有 DH 都知道同一个哈希函数，而某个 DH 可能会好奇其他 DH 的隐私信息。如果像 MPC-FM 这样的协议仅仅在安全计算平台 SPDZ 上实现 FM 草图来解决我们的问题，就会出现隐私泄露。例如，设 $Z$为集合 $S$的估计量（即 FM 草图中首个零位的索引之和）。当 DH $i$（持有集合 $S_i$）发现集合 $S_i \cup \{e\}$ 的估计量 $Z_e$大于 $Z$时，DH $i$便可推断出 $e$不属于任何其他数据持有方的集合。更一般地，当 DH $i$发现集合 $S_i \cup E$的估计量 $Z_e$等于 $Z + t$时，其中 $t$为正整数，那么 DH $i$便可推断出，集合 $E$中至少有 $t$ 个元素不属于其他任何 DH 持有的集合。

以第 1 节讨论的互联网流量统计私有数据集为例。路由器等互联网网络设备的计算资源有限，且在这些设备上收集的网络流量属于私有数据。我们可以尝试对 MPC-HLL 进行修改以解决挑战1，即用 HLL 草图替换 FM 草图。显然，HLL 草图可以显著降低计算 FM 草图的复杂度 $F_i$从 $O(m|S_i|)$降低至 $O(|S_i|)$。遗憾的是，合并 HLL 草图需要执行取最大值运算，而在 SPDZ 等安全多方计算框架中，这类运算在顶层计算方上实现起来复杂且代价高昂。此外，与 MPC-FM 类似，我们很容易发现 MPC-HLL 也不具备差分隐私。为解决挑战2，可以通过加性噪声机制增强 MPC-FM。使用 MPC 合并本地数据，并向聚合输出添加加性噪声，是满足差分隐私的标准方法，也已应用于联邦学习等其他领域 [16, 21]。遗憾的是，在后续实验结果（见图 4 和图 5）中，我们观察到，采用加性噪声机制的 MPC-FM（以及 MPC-HLL）会产生显著的估计误差。

# 4 我们的方法

本节首先介绍一种新方法——*FMS* 草图，它是我们协议 *DP-DICE* 的构建模块。随后，我们将详细阐述协议 DP-DICE 并讨论其性能。

## 4.1 我们的 FMSSketch

**数据结构**。如图 1 所示，我们的 FMS 草图 $\{B_i\}_{i=1,\dots,m}$由 $m$ 个位数组 $B_1, \dots, B_m$ 组成，其中 $m = 2^r$，且 $r$ 为正整数。每个 $B_i, i = 1, \dots, m$ 都是一个包含 $w$ 位的一维数组，其中所有位 $B_i[0], \dots, B_i[w-1]$均初始化为 0。此外，FMS 草图使用单个哈希函数 $h(\cdot)$，将元素 $e \in U$映射为一个随机的（$r + w - 1$）位串 $h(e) \in \{0,1\}^{r+w-1}$。

**草图生成**。对于集合 $S$中的每个元素 $e$，与 HLL 草图 [8], 类似，根据哈希串 $h(e)$，我们首先计算

$$(i, j) \leftarrow \left(h^{(\mathrm{L})}(e) + 1, \rho\left(h^{(\mathrm{H})}(e)\right)\right),$$

其中，$h^{(\mathrm{L})}(e)$ 和 $h^{(\mathrm{H})}(e)$分别由字符串 $h(e)$ 的最低 $r$ 位和最高 $w-1$ 位组成，而 $\rho(h^{(\mathrm{H})}(e))$ 返回 $h^{(\mathrm{H})}(e)$ 的尾随零个数。然后，我们按如下方式更新位数组 $B_i$：$B_i[j] \leftarrow 1$。

**基数估计**。给定 FMS 草图 $B_1, \dots, B_m$，接下来我们将介绍估计 $n$（即集合 $S$ 的基数）的方法。定义 $Z$为所有位数组 $B_1, \dots, B_m$ 中零位的数量，即

$$Z = mw - \sum_{i=1}^{m}\sum_{x=0}^{w-1} B_i[x] = \sum_{i=1}^{m}\sum_{x=0}^{w-1}(1 - B_i[x]). \tag{2}$$

**定理** 3. 定义 $p_x \triangleq \begin{cases} 2^{-x-1} & /m, \quad 0 \le x \le w-2, \\ 2^{-w+1} & m \quad x = w-1 \end{cases}$ / 和 $V \triangleq \frac{Z}{mw}$。那么，期望为 $\mathbb{E}(V) = \frac{1}{w}\sum_{x=0}^{w-1}(1-p_x)^n$，而 $V$ 的方差为

$$\mathrm{Var}(V) = \frac{1}{mw^2}\sum_{x=0}^{w-1}(1-p_x)^n\left(1 - m\sum_{x=0}^{w-1}(1-p_x)^n\right) + \frac{2}{mw^2}\sum_{x=0}^{w-1}\sum_{y=x+1}^{w-1}(1-p_x-p_y)^n + \frac{m-1}{mw^2}\sum_{x=0}^{w-1}\sum_{y=0}^{w-1}(1-p_x-p_y)^n.$$

证明。我们很容易得出

$$P\left(\rho(h^{(\mathrm{H})}(e)) = x\right) = p_x. \tag{3}$$

定义变量 $\mathbf{1}_{j,x} \triangleq 1 - B_j[x]$. 集合 $S$中的所有元素更新后，根据式 (3)，可得

$$P(\mathbf{1}_{j,x} = 1) = P(B_j[x] = 0) = (1-p_x)^n,$$

因此，$Z$的期望计算如下：

$$\mathbb{E}(Z) = \sum_{j=0}^{m-1}\sum_{x=0}^{w-1}\mathbb{E}(\mathbf{1}_{j,x}) = m\sum_{x=0}^{w-1}(1-p_x)^n.$$

类似地，我们定义 $Z_j \triangleq \sum_{x=0}^{w-1} \mathbf{1}_{j,\ x}$. 于是有

$$\begin{aligned}
\mathbb{E}(Z^2) &= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\mathbb{E}(Z_iZ_j) \\
&= \sum_{i=0}^{m-1}\mathbb{E}(Z_iZ_i) + 2\sum_{i=0}^{m-1}\sum_{j=i+1}^{m-1}\mathbb{E}(Z_iZ_j) \\
&= m\sum_{x=0}^{w-1}\mathbb{E}(\mathbf{1}_{j,x}) + 2m\sum_{x=0}^{w-1}\sum_{y=x+1}^{w-1}\mathbb{E}(\mathbf{1}_{j,x}\mathbf{1}_{j,y}) \\
&\quad + 2\sum_{i=0}^{m-1}\sum_{j=i+1}^{m-1}\sum_{x=0}^{w-1}\sum_{y=0}^{w-1}\mathbb{E}(\mathbf{1}_{i,x}\mathbf{1}_{j,y}) \\
&= m\sum_{x=0}^{w-1}(1-p_x)^n + 2m\sum_{x=0}^{w-1}\sum_{y=x+1}^{w-1}(1-p_x-p_y)^n \\
&\quad + m(m-1)\sum_{x=0}^{w-1}\sum_{y=0}^{w-1}(1-p_x-p_y)^n,
\end{aligned}$$

其中，最后一个等式成立是因为对于任意两个不同的元组（$i,\ x$）和（$j,\ y$），我们有

$$P(\mathbf{1}_{i,x}\mathbf{1}_{j,y}=1) = P(B_i[x]=B_j[y]=0) = (1-p_x-p_y)^n.$$

于是，我们有

$$\begin{aligned}
&\mathrm{Var}(Z) = \mathbb{E}(Z^2) - \mathbb{E}(Z)^2 \\
&= 2m\sum_{x=0}^{w-1}\sum_{y=x+1}^{w-1}(1-p_x-p_y)^n + m(m-1)\sum_{x=0}^{w-1}\sum_{y=0}^{w-1}(1-p_x-p_y)^n \\
&\quad + m\sum_{x=0}^{w-1}(1-p_x)^n\left(1 - m\sum_{x=0}^{w-1}(1-p_x)^n\right).
\end{aligned}$$

于是，我们很容易推导出 $\mathbb{E}(V)$ 和 $\mathrm{Var}(V)$ 的公式分别为 $\mathbb{E}(V) = \mathbb{E}(Z)\,m w$ 以及 $\mathrm{Var}(V) = \mathrm{Var}(Z)\,m^2w^2$。□

基于上述定理，我们将函数 $f(n)$ 定义为：

$$f(n) \triangleq \frac{1}{w}\sum_{x=0}^{w-1}(1-p_x)^n \approx \frac{1}{w}\sum_{x=0}^{w-1}\mathrm{EXP}(-np_x), \tag{4}$$

最后一个近似等式成立，是因为 $p_x$ 小于 $\frac{1}{2m}$，而后者通常是一个较小的数。

接下来，我们通过 $n$ 估计 $\hat{n}$，使得 $f(\hat{n}) = V$。不难发现，$f(n)$ 随 $n$ 的增大而减小。我们可以使用二分查找法计算 $\hat{n} = f^{-1}(V)$，其中 $f^{-1}(V)$ 是式（4）中定义的函数 $f(n)$ 的反函数。

**误差分析**。根据定理3 和式（4），我们有

$$f(n) = \mathbb{E}(V) = \frac{1}{w}\sum_{x=0}^{w-1}(1-p_x)^n.$$

由于 $f(n)$ 是单调函数，我们对函数 $f(n) = \mathbb{E}(V)$ 执行逆操作。之后，我们有

$$n = f^{-1}(\mathbb{E}(V)).$$

鉴于 $\mathbb{E}(V)$、$\mathrm{Var}(V)$ 和 $f(\cdot)$ 的形式较为复杂，不容易推导出 $\mathrm{Var}(\hat{n}) = \mathrm{Var}(f^{-1}(V))$。

为解决这一问题，受 HLL [8], 分析的启发，我们在泊松模型下进行渐近分析。回顾一下，$n$ 将 $S$中的元素独立地使用哈希函数 $h(\cdot)$ 映射到草图 $B_1,\ldots,B_m$ 中。令 $n_i$表示 $S$中被哈希到 $B_i$, $i = 1,\ldots,m$的元素数量。我们看到 $\sum_{i=1}^{m} n_i = n$，因此 $n_1,\ldots,n_m$并不相互独立，这使得 $\mathrm{Var}(\hat{n})$ 的分析颇具挑战。当 $m$较大时，$n_1,\ldots,n_m$可近似为根据 $\mathrm{Pois}(\frac{n}{m})$ [22]生成的同分布独立泊松变量。定义$V_i = \frac{Z_i}{w}$，其中

$$Z_i = w - \sum_{x=0}^{w-1}B_i[x] = \sum_{x=0}^{w-1}(1-B_i[x]). \tag{5}$$

因此，我们有 $V = \frac{\sum_{i=1}^{m}V_i}{m}$。假设 $n_1$、……、$n_m$是根据分布 $\mathrm{Pois}(\frac{n}{m})$ 生成的相同且相互独立的泊松变量。那么，所有 $V_1$、……、$V_m$都是相同且相互独立的随机变量，具有相同的期望 $\mu\mathrm{pois}$ 和方差 $\sigma^2\mathrm{pois}$。对于散列到草图 $B_i$ 中的 $n_i \sim \mathrm{Pois}(\frac{n}{m})$ 个元素，每个元素被散列到 $B_i[x]$ 的概率为 $p_x \times m$。因此不难发现，散列到 $B_i[x]$ 中的元素数量是一个服从分布 $\mathrm{Pois}(np_x)$ 的随机变量。因此，我们有 $B_i[x]$ 以概率 $\mathrm{EXP}(-np_x)$ 取值为零，以概率 $1-\mathrm{EXP}(-np_x)$ 取值为一。基于上述分析，不难得到每个 $V_1$、……、$V_m$ 的期望 $\mu\mathrm{pois}$：

$$\mu_{\mathrm{pois}} = \frac{1}{w}\sum_{x=0}^{w-1}\mathrm{EXP}(-np_x) \approx f(n), \tag{6}$$

其中，函数 $f(n)$ 定义于公式（4）中。同样地，每个 $V_1,\ldots,V_m$ 的方差 $\sigma^2\mathrm{pois}$ 计算如下：

$$\begin{aligned}
\sigma^2_{\mathrm{pois}} &= \frac{1}{w^2}\sum_{x=0}^{w-1}\mathrm{EXP}(-np_x) - \mathrm{EXP}(-2np_x) \\
&\approx \frac{1}{w^2}\left(\mathrm{EXP}\left(-\frac{n}{2^w m}\right) - \mathrm{EXP}\left(-\frac{n}{m}\right)\right).
\end{aligned} \tag{7}$$

根据林德伯格–列维中心极限定理，当 [23], as $m$ 趋于无穷大时，随机变量 $\sqrt{m}(V-\mu\mathrm{pois})$ 依分布收敛于正态分布 $\mathrm{N}(0, \sigma^2\mathrm{pois})$，即

$$\sqrt{m}(V-\mu_{\mathrm{pois}}) \xrightarrow{D} \mathcal{N}(0, \sigma^2_{\mathrm{pois}}). \tag{8}$$

为推导 $\mathrm{Var}(f^{-1}(V))$（即 $\mathrm{Var}(\hat{n})$），使用 德尔塔方法 [24]，我们得到

$$\sqrt{m}(g(V) - g(\mu_{\mathrm{pois}})) \xrightarrow{D} \mathcal{N}(0, \sigma^2_{\mathrm{pois}} \times (g'(\mu_{\mathrm{pois}}))^2),$$

其中函数 $g(\cdot) = f^{-1}(\cdot)$，即式（4）中定义的函数 $f(n)$ 的反函数。于是，我们有 $g(\mu\mathrm{pois}) \approx n$ 因为$\mu\mathrm{pois} \approx f(n)$ 且 $g'(\mu\mathrm{pois}) \approx f'(n) \approx -\frac{1}{\sum_{x=0}^{w-1}p_x^{\,w}\mathrm{EXP}(-np_x)}$，其中$f'(n)$ 是函数 $f(n)$ 的导数，因此，我们的估计量 $\hat{n}$ 的均方误差可近似为 $m\times\frac{\sigma^2_{\mathrm{pois}}}{n}(f'(\,))^2 = \mathrm{EXP}\left(-\frac{n}{2^w m}\right)-\mathrm{EXP}\left(-\frac{n}{m}\right)m(\sum_{x=0}^{w-1}p_x\mathrm{EXP}(-np_x))^2$。此外，我们有 $\frac{\sum_{x=0}^{w-1}p_x\mathrm{EXP}(-np_x)}{\mathrm{EXP}(-\frac{n}{m}2^{-x-1})} \approx \sum_{x=0}^{w-1}\frac{n}{m}2^{-x-1}\mathrm{EXP}(-\frac{n}{m}2^{-x-1}) \approx \int_0^w \frac{n}{m}2^{-x-1}$

$$\frac{1}{2}\left(\mathrm{EXP}\left(-\frac{n}{2^w m}\right)-\mathrm{EXP}\left(-\frac{n}{m}\right)\right)。$$

当设置 $w \geq \lceil \log_2(n/m)+6 \rceil$ 时，我们有 $\mathrm{EXP}\left(-\frac{n}{2^w m}\right)$ 很接近于 1。基于以上

观测结果，我们有 $\mathrm{STDErr}(\hat{n}) \approx \ln\frac{2}{\sqrt{m}} \cdot \sqrt{1 - \frac{1}{\mathrm{EXP}(-\frac{n}{m}) \cdot \frac{n}{m}}} \geq 3$ 时，我们有 $\mathrm{STDErr}(\hat{n}) \approx \ln\frac{2}{\sqrt{m}} \approx \frac{0.69}{\sqrt{m}}$ 标准误差与图 3 中的实验结果吻合良好。

正如我们在第 3.5 节中提到的，公开 $Z = mwV$（在公式（2）中定义）的值会泄露本文所研究的 PDCE 场景中的隐私。为了解决这个问题，我们向变量 $N \sim \mathcal{N}\left(0, \sigma^2_{\text{noise}}\right)$ 添加一个随机变量 $Z$，以实现差分隐私。也就是说，我们定义 $V_{\text{diff}} \triangleq \frac{Z+N}{mw} = V + \frac{N}{mw}$，并由公式（8）可得

$$\sqrt{m}(V_{\text{diff}} - \mu_{\text{pois}}) \xrightarrow{D} \mathcal{N}\left(0, \sigma^2_{\text{pois}} + \frac{\sigma^2_{\text{noise}}}{mw^2}\right).$$

再次使用德尔塔方法，可得

$$\sqrt{m}(g(V_{\text{diff}}) - g(\mu_{\text{pois}})) \xrightarrow{D} \mathcal{N}\left(0, \left(\sigma^2_{\text{pois}} + \frac{\sigma^2_{\text{noise}}}{mw^2}\right) \times (g'(\mu_{\text{pois}}))^2\right).$$

因此，根据 $V_{\text{diff}}$ 推断出的估计值 $\hat{n}^*$ 的均方误差可近似为 $m \times \left(\sigma^2_{\text{pois}} + \frac{\sigma^2_{\text{noise}}}{mw^2}\right) \times (f'(\frac{n}{m}))^2 = \frac{\mathrm{EXP}\left(-\frac{n}{w}\right)\left(1-\mathrm{EXP}\left(-\frac{n}{m}\right)\right)}{m\left(\sum_{x=0}^{w-1} p_x \left(1-\frac{x}{w}\right)^{\frac{n}{m}}\right)^2} + \frac{\sigma^2_{\text{noise}}}{mw^2} \cdot \left(\mathrm{EXP}\left(\frac{n}{m}\right)\right)^2$。与近似计算 $\mathrm{STDErr}(\hat{n})$ 的过程类似，我们有 $\mathrm{STDErr}(\hat{n}^*) \approx \frac{0.69}{\sqrt{m}} \cdot \sqrt{1 + \frac{\sigma^2_{\text{noise}}}{mw^2}}$。

## 4.2 我们的 DP-DICE 协议

4.2.1 概述。我们的 DP-DICE 旨在安全地估计集合 $S = \cup_{j=1,\ldots,d} S_j$ 的基数，其中集合 $S_j$ 由 DH $j$ 在本地持有，且该 DH 诚实但好奇。我们的 DP-DICE 概述如图2所示。每个 DH $j$ 负责收集其集合 $S_j$，并为 $S_j$ 构建 FMS 草图 $\{B(j)_i\}_{i=1,\ldots,m}$。此外，它还会生成一个变量 $N_j$，用于生成加性噪声，以满足差分隐私要求。随后，DH $j$ 将 $\{B(j)_i\}_{i=1,\ldots,m}$ 和 $N_j$ 安全地发送给所有 CP。所有 CP 收到所有 DH 的 FMS 草图后，会协作合并这些草图，以安全地计算集合 $S$ 的 FMS 草图 $\{B_i\}_{i=1,\ldots,m}$。为避免隐私泄露，每个 CP 仅持有并集 $S$ 的 FMS 草图 $\{B_i\}_{i=1,\ldots,m}$ 的一个份额。最后，所有 CP 安全地计算 FMS 草图 $\{B_i\}_{i=1,\ldots,m}$ 的变量 $Z$（定义见公式 (2)），然后将 $Z + \sum_{j=1}^{d} N_j$ 输出给公众，以估计集合 $S$ 的基数 $n$；其中，$\sum_{j=1}^{d} N_j$ 是为满足差分隐私要求而添加的噪声。我们的 DP-DICE 协议包含三个阶段：离线准备阶段、数据收集阶段和数据聚合阶段。我们基于 SPDZ 框架 [25]，构建 DP-DICE，该框架使用认证份额。接下来，我们将详细介绍各个阶段。

*4.2.2* 离线准备阶段。 该阶段主要用于初始化协议中使用的一些参数，并根据多方计算协议生成供后续在线步骤使用的随机数。

**参数初始化。** 所有 DH 和 CP 在参数初始化阶段协商协议中使用的参数。首先，他们商定一个有限域 $\mathbb{F}_p = \{0, 1, \ldots\ldots, p-1\}$，该有限域将作为数据表示、秘密共享和所有计算的基础。模数 $p$ 是一个（$\lambda + \tau$）位素数，其中 $\lambda$（例如 $\lambda = 40$）是统计安全参数，而 $\tau$（例如 $\tau = 32$）由明文域决定。接下来，所有 CP 和 DH 商定 FMS 草图的参数 $m$ 和 $w$，并由所有 CP 运行

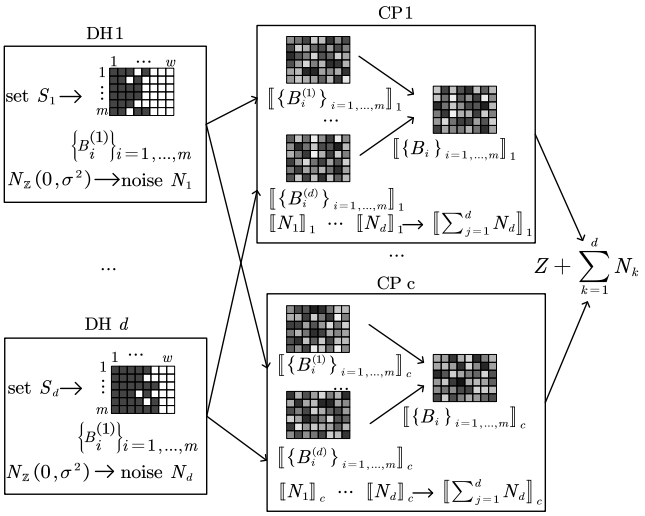



**图 2：DP-DICE 概览。$\{B(k)_i\}_{i=1,\ldots,m}, k = 1, \ldots, d$是数据持有方 DH $k$ 所持集合 $S_k$ 的 FMS 草图，$N_k$ 是一个服从离散高斯分布 $N_{\mathbb{Z}}(0, \sigma^2)$的随机变量。$\{[\![B(k)_i]\!]_j\}_{i=1,\ldots,m,\ j=1,\ldots,c}$ 和计算方 CP $j$ 所持的 $[\![N_k]\!]_j$ 分别是 $\{B(k)_i\}_{i=1,\ldots,m}$ 和 $N_k$的加法份额。**

SPDZ 设置协议以获取各自的参数和密钥。此外，所有 DH 选择一个带密钥哈希函数 $H$，并使用 [26]中的认证群组密钥交换协议建立密钥 $\mathbb{k}$。然后，将 FMS 草图实现的哈希函数 $h$ 定义为 $h(e) = H(\mathbb{k}||e)$，其中 $||$ 是字符串拼接操作。

**随机数生成。** 所有 CP 都运行若干离线协议，生成一些将在后续数据收集和聚合阶段使用的随机数。请注意，每个 CP 仅持有任一生成随机数的一部分（即一个份额），只有在所有 CP 都被攻陷并控制时，对手才能获得该随机数。我们使用的随机数生成离线协议包括 Rand、Rand2 和 RandExp，这些协议均可在 SPDZ 框架中使用。

*4.2.3* 数据收集阶段。 此阶段由 DH 负责，DH 需要完成数据集的收集，并以隐蔽方式将生成的 FMS 草图共享给所有 CP。每个 DH 还会发送一个变量，以便 CP 生成加性噪声，从而满足差分隐私要求。

FMS 草图生成。使用相同的哈希函数 $h(e) = H(\mathbb{k}||e)$，每个 DH $j$ 使用第 4.1 节的方法计算集合$S_j$的 FMS 草图 $\{B(j)_i\}_{i=1,\ldots,m}$。FMS 草图 $\{B(j)_i\}_{i=1,\ldots,m}$ 将安全地发送给所有 CP，以计算并集 FMS 草图。分布式离散噪声生成。SPDZ 等多方计算框架处理整数比处理浮点数更方便、高效。此外，我们注意到并集 FMS 草图 $\{B_i\}_{i=1,\ldots,m}$（在式 (2) 中定义）的变量 $Z$ 是一个正整数，该变量将公开，以便计算

并集为 $\mathbb{Z}$ 的基数，添加可控的离散噪声，因此我们采用最先进的安全聚合方法：分布式离散高斯噪声机制 [16]，该机制最初旨在安全聚合联邦学习中客户端的模型更新。具体而言，每个数据持有方 $j$ 生成一个随机变量 $N_j \in \mathbb{Z}$（即 $N_j$ 是一个随机整数），以防御差分攻击。$N_j$ 是从离散高斯分布中采样得到的随机变量 $\mathcal{N}_{\mathbb{Z}}(0, \sigma^2)$，即：

$$P(N_j = x) = \frac{\exp\left(-\frac{x^2}{2\sigma^2}\right)}{\sum_{y\in\mathbb{Z}} \exp\left(-\frac{y^2}{2\sigma^2}\right)}, x \in \mathbb{Z}.$$

每个数据持有方 $j$ 秘密地将其噪声变量 $N_j$ 发送给所有计算方，所有数据持有方的噪声变量之和 $N_j$，即 $\sum_{j=1}^{d} N_j$，之后将被加到变量 $Z$ 上。

安全数据共享。在 SPDZ 中，共享形式的整数 $x$ 定义为 $[\![x]\!] \triangleq (x_1, \ldots, x_c, m(x)\,1, \ldots, m(x)c, \Delta_1, \ldots, \Delta_c)$，每个计算方 $j$ 持有一个元组 $[\![x]\!]_j = (x_j, m(x)\,j, \Delta_j)$，其中元组的三部分分别是 $x$的加法份额、消息认证码和消息认证码密钥。消息认证码和消息认证码密钥用于验证变量 $x$的真实性，以便在对手篡改 $x$时及时发现。为将数据持有方持有的变量 $x$ 安全地共享给所有计算方，我们采用了 [11] 中使用的方法。具体而言，所有计算方以共享形式向数据持有方公开一个随机数 $[\![a]\!] = (a_1, \ldots, a_c, m(a)\,1, \ldots, m(a)c, \Delta_1, \ldots, \Delta_c)$，其中 $a \in \mathbb{F}_p$ 由 SPDZ 提供的函数 Rand 在线下生成。详细来说，每个计算方 $j$ 将其$a_j$ 发送给数据持有方，随后数据持有方得到 $a = \sum_{j=1}^{c} a_j$。给定 $a$后，数据持有方计算 $x - a$ 并将其广播给所有计算方。然后，所有计算方计算 $[\![x]\!] = [\![a]\!] + x - a.$。也就是说，每个计算方 $j$ 计算

$$[\![x]\!]_j = \begin{cases} (a_j + x - a, m_j^{(a)} + (x-a)\Delta_j, \Delta_j) & \text{if } j = 1, \\ (a_j, m_j^{(a)} + (x-a)\Delta_j, \Delta_j) & \text{otherwise.} \end{cases}$$

我们很容易看出，对控制最多 $c-1$ 个计算方的对手而言，$a$ 的值是未知的。因此，上述过程是安全的。每个随机变量 $a \in \mathbb{F}_p$ 都由 SPDZ 框架提供的 Rand 函数生成；为防范潜在攻击，我们在使用后立即将其丢弃。换言之，DH 需要使用不同的随机数，将不同变量安全共享给所有计算方。采用上述方法，每个 DH $j$ 都会将其 FMS 草图中的每个条目 $B_{(j)\,i}\ [l], i = 1, \ldots, m, l = 1, \ldots, w$ 以及噪声变量 $N_j$ 安全地发送并共享给所有计算方。

*4.2.4* 数据聚合阶段。此阶段面向计算方。计算方负责基于 $\{B_i\}_{i=1,\ldots,m}$ 的份额，安全地计算并集 $S$ 的 FMS 草图 $B_{(j)\,i}$，其中 $1 \le i \le m$，$1 \le j \le d$，并计算式（2）中定义的变量 $Z$。在向公众公开 $Z$ 的值之前，所有计算方都会安全地向 $Z$ 中加入噪声 $\sum_{j=1}^{d} N_j$，以满足差分隐私的要求。

**合并 FMS 草图**。为了计算并集的 FMS 草图 $\{B_i\}_{i=1,\ldots,m}$的秘密份额，我们首先定义变量 $B_i^*[l]$ 如下：

$$B_i^*[l] = \sum_{j=1}^{d} B_i^{(j)}[l], \quad i = 1, \ldots, m, \quad l = 1, \ldots, w.$$

我们很容易看出，当 $B_i^*[l] > 0$时 $B_i[l] = 1$，否则 $B_i[l] = 0$。在 SPDZ 框架下，对于任意整数 $x$ 和 $y$，我们有

$$[\![x + y]\!]_j = [\![x]\!]_j + [\![y]\!]_j = (x_j + y_j, m_j^{(x)} + m_j^{(y)}, \Delta_j).$$

**算法 1：** 函数零测试($[\![x]\!]$)

**输入：** $[\![x]\!]$，$x \in \mathbb{F}_p$

**输出：** $[\![b]\!]$，其中 $b = 0$ 当 $x = 0$ 时，否则为 $b = 1$

1 **离线阶段：**
2 **对于每个** $t = 0, 1, \ldots\ldots, L-1$，**执行**
3 $\quad [\![a_t]\!] \leftarrow$ Rand2(); // 生成一个随机比特的份额
4 $[\![a]\!] \leftarrow \sum_{t=0}^{L-1} 2^t [\![a_t]\!]$; // $a_t$ 是 $a$的第 $t$位
5 $(\beta_0, \ldots, \beta_\tau) \leftarrow$ 插值$(\phi(x))$; // $\phi(x) = \sum_{t=0}^{\tau} \beta_t x^t$
6 $([\![R^{-1}]\!], [\![R]\!], [\![R^2]\!], \ldots, [\![R^{\tau}]\!]) \leftarrow$RandExp($\tau$);

7 **在线阶段：**
8 $[\![r]\!] \leftarrow [\![a]\!] + [\![x]\!]$ 并向所有计算方公开 $r$;
9 $[\![h]\!] \leftarrow \sum_{t=0}^{L-1}([\![a_t]\!] + r_t - 2r_t \cdot [\![a_t]\!])$; // $r_t$ 是 $r$的第 $t$位
10 $[\![\gamma]\!] \leftarrow [\![R^{-1}]\!] \cdot [\![1 + h]\!]$ 并向所有计算方公开 $\gamma$;
11 $[\![b]\!] \leftarrow \beta_0$;
12 **对于每个** $t = 1, 2, \ldots, \tau$ **do**
13 $\quad [\![b]\!] \leftarrow [\![b]\!] + \beta_t \cdot \gamma^t [\![R^t]\!]$; // 注意 $\gamma^t \cdot [\![R^t]\!] = [\![(1+h)^t]\!]$
14 **返回** $[\![b]\!]$;

因此，给定 $[\![B_{(1)}{}_i\ [l]]\!]_j$、……、$[\![B_{(d)}{}_i\ [l]]\!]_j$持有的所有元组，计算方 $j$计算方 $j$计算 $[\![B_i^*[l]]\!]_j$如下：$[\![B_i^*[l]]\!]_j = \sum_{k=1}^{d} [\![B_i^{(k)}[l]]\!]_j$。为了安全地计算 $[\![B_i[l]]\!]_j$ 对于计算方 $j$的值，我们应用零测试 协议，该协议由 [27]提出，用于 $[\![B_i^*[l]]\!]$，可以安全地比较 $B_i^*[l]$ 是否等于零。零测试协议 的伪代码如算法 1所示。

为了测试整数 $x \in \mathbb{F}_p$ 是否为零，零测试首先计算 $[\![r]\!] = [\![a + x]\!] = [\![a]\!] + [\![x]\!]$，并将 $r$ 公开给所有计算方（第 8 行）。其中，$a \in \mathbb{F}_p$ 是在离线准备阶段生成的随机整数（第 2–4 行）。由于没有任何计算方知道 $a$，因此，公开 $r$不会泄露 $x$的信息，却有助于在共享形式下计算 $r$与 $a$之间的汉明距离 $h$。具体而言，我们有 $[\![h]\!] = \sum_{t=0}^{L-1}([\![a_t]\!] + r_t - 2r_t \cdot [\![a_t]\!])$，其中$L = \lceil \log_2 p \rceil$ 是 SPDZ 框架所用模数 $p$ 的位长度，$a_t$ 和 $r_t$ 分别是随机变量 $a$和 $r$ 的第 $t$位，即 $a = \sum_{t=0}^{L-1} 2^t a_t$和 $r = \sum_{t=0}^{L-1} 2^t r_t$。注意，$a_t$通过 SPDZ 提供的函数 Rand2 在离线阶段计算。

我们很容易看出，当 $x = 0$ 时，$h = 0$；否则为 $h \in \{1, \ldots, \tau\}$，其中 $\tau$（例如 $\tau = 32$）是 SPDZ 明文域设置的位长度。零测试使用一个查找函数，该函数是多项式 $\phi(\cdot)$，满足 $\phi(0) = 0$，且对所有其他 $x \in \{1, \ldots, \tau\}$，$\phi(x) = 1$。据此对查找多项式进行插值，例如使用拉格朗日插值（第 5 行）。给定 $[\![h]\!]$，直接计算查找多项式中包含的 $[\![h^t]\!]$ 效率低下。为解决这一问题，零测试在 SPDZ 上使用函数 RandExp($\tau$) 生成 $([\![R^{-1}]\!], [\![R]\!], [\![R^2]\!], \ldots, [\![R^{\tau}]\!])$，即非零随机数 $R \in \mathbb{F}_p \backslash \{0\}$ 的份额，以及其 $t$ 次幂的份额 $t \in \{-1, 1, 2, \ldots, \tau\}$；这些份额均在离线阶段生成（第 6 行）。随后，它计算 $[\![\gamma]\!] = [\![R^{-1}]\!] \cdot [\![1 + h]\!]$，并向所有 CP 公开 $\gamma$（第 10 行）。这里我们使用 $h+1$ 而非 $h$，因为当 $R^{-1}h = 0$ 时很容易推断出 $h = 0$，从而泄露 $h$ 的信息。由于 $\gamma^t = R^{-t}(1 + h)^t$，我们有 $\gamma^t \cdot [\![R^t]\!] = [\![(1 + h)^t]\!]$。形式化地，我们定义 $\phi(x) = \sum_{t=0}^{\tau} \beta_t x^t$，使得 $\phi(1) = 0$，且当 $x \in \{2, \ldots, \tau + 1\}$ 时，$\phi(x) = 1$。给定 $\gamma, \beta_0, \ldots, \beta_\tau$，以及 $[\![R]\!], [\![R^2]\!], \ldots, [\![R^{\tau}]\!]$，则零测试的结果为：$[\![b]\!] = [\![\phi(1 + h)]\!] = \sum_{t=0}^{\tau} \beta_t \gamma^t \cdot [\![R^t]\!]$.

对每个 $[\![B_i^*[l]]\!]$ 应用零测试后，我们最终得到每个 $1 \le i \le m$ 对应的 $[\![B_i[l]]\!]$。

**合并噪声变量**。给定计算方 $j$ 持有的所有元组 $[\![N_1]\!]_j$、……、$[\![N_d]\!]_j$，计算方 $j$ 以份额形式计算 $[\![N]\!]_j$：$[\![N]\!]_j = \sum_{k=1}^{d} [\![N_k[l]]\!]_j$，其中 $N = \sum_{k=1}^{d} N_k[l]$ 是用于差分隐私保护的最终加性噪声。

估计基数。给定 $[\![B_i[l]]\!]_j$ 和 $[\![N]\!]_j$，每个计算方 $j$ 首先计算 $[\![Z+N]\!]_j = [\![N]\!]_j + \sum_{i=1}^{m}\sum_{l=1}^{w}[\![B_i[l]]\!]_j$.。在 SPDZ 框架以共享形式公开值 $x = Z + \sum_{k=1}^{d} N_k[\![x]\!] =$（$x_1$、……、$x_c$、$m$（$x$）1、……、$m$（$x$）$_c$、$\Delta_1$、……、$\Delta_c$）之前，首先按如下方式检查 $x$ 的真实性：令每个计算方 $j$ 计算 $\sigma$（$x$）$_j$ $= m$（$x$）$_j$ $- x\Delta_j$ 并广播 $\sigma$（$x$）$_j$，然后检查 $\sum_{j=1}^{c}\sigma$（$x$）$_j$ $= 0$ 是否成立。若 $\sum_{j=1}^{c}\sigma$（$x$）$_j$ $\neq 0$，则中止；否则输出 $x$。给定 $x = Z + \sum_{k=1}^{c} N_k$，我们使用第 4.1 节中的方法估计并集 $S$ 的基数 $n$。

## 4.3 协议安全性与差分隐私

我们考察协议的安全属性，包括机密性和正确性。正如我们提到的，对手可以腐化除一个计算方之外的所有计算方。然而，对手无法篡改 $Z + \sum_{j=1}^{d} N_j$ 的计算正确性而不被发现，这是 SPDZ 框架的显著特点。此外，除协议输出 $Z + \sum_{j=1}^{d} N_j$ 外，对手无法从执行协议的过程中获知任何信息；该输出具有差分隐私，后文将对此进行讨论。遵循 [11]，中给出的 MPC-FM 安全性证明，我们也在通用可组合框架 [28]，下证明了 DP-DICE 的安全性，从而使我们的协议能够作为更大系统的组件，同时不损失其安全属性。

接下来，我们讨论协议的差分隐私。为了为离散高斯分布 $\mathrm{N}_{\mathbb{Z}}(0,\sigma^2)$ 设定适当的尺度 $\sigma$，以保证协议输出 $Z+N$ 的差分隐私，我们首先计算变量 $Z$ 的敏感度，其定义为 $\Delta_Z = \max |Z(S) - Z(S')|$，其中最大值取遍所有集合 $S$ 和 $S'$ 的组合；它们是全集 $U$ 的两个子集，且至多相差一个元素；$Z(S)$ 和 $Z(S')$ 分别是集合 $S$ 和 $S'$ 对应的 $Z$ 值。对于式（2）中给出的 $Z$ 定义，我们发现 $\Delta_Z = 1$。定义变量 $\varepsilon_d = \min\left(\sqrt{\frac{1}{d\sigma^2}\ln\frac{\tau_d+1}{\tau_d}}, \frac{1}{\sqrt{d}\sigma}+\tau_d\right)$，其中 $\tau_d = 10\sum_{k=1}^{d-1}\exp\left(-\frac{2k\pi^2\sigma^2}{k+1}\right)$。根据第 3.3 节的定理 2，我们发现，当 $\sigma \ge 0.5$ 时，将 $d$ 个独立离散高斯变量 $N_1$、……、$N_d \sim \mathrm{N}_{\mathbb{Z}}(0,\sigma^2)$ 加到变量 $Z$ 上，其结果满足 $\frac{1}{2}\varepsilon_d^2$-集中差分隐私。根据第 3.3 节，我们发现 $Z$ 实现了 $(\varepsilon_\delta^*, \delta)$-差分隐私，其中 $\varepsilon_\delta^* \le 0.5\varepsilon_d(\varepsilon_d + 2\sqrt{-2\ln\delta})$。除了对手之外，好奇的 DH $i$ 也可能从协议输出 $Z + \sum_{j=1}^{d} N_j$ 中减去其本地变量 $N_i$。然而，除此输出 $Z + \sum_{1\le j\le d \wedge j\neq i} N_j$ 外，它仍然一无所知。同样，我们发现变量 $Z + \sum_{1\le j\le d\wedge j\neq i} N_j$ 满足 $\frac{1}{2}\varepsilon_{d-1}^2$-集中差分隐私和 $(\varepsilon_\delta^*, \delta)$-差分隐私，其中 $\epsilon_\delta^* \le 0.5\varepsilon_{d-1}(\varepsilon_{d-1} + 2\sqrt{-2\ln\delta})$。

## 4.4 协议复杂度

与最先进的 MPC-FM 相比，[11], 我们的 DP-DICE 使用 FMS 草图，而 MPC-FM 协议使用 FM 草图。因此，我们的 DP-DICE 在每个 DH 上所需的内存使用量与 MPC-FM 协议相同，同时将 DP 处理一个元素的计算复杂度从 $O(m)$ 降低至 $O(1)$，其中 $m$ 通常设为数千。此外，我们的 DP-DICE 还降低了 CPs 上的计算量，因为与 MPC-FM 所需的变量 $Z$（在式（2）中定义）相比，我们的 DP-DICE 所使用的变量在 MPC 框架上计算起来更简单。后续实验表明，由于这一简化，网络通信开销也显著降低。

# 5 评估

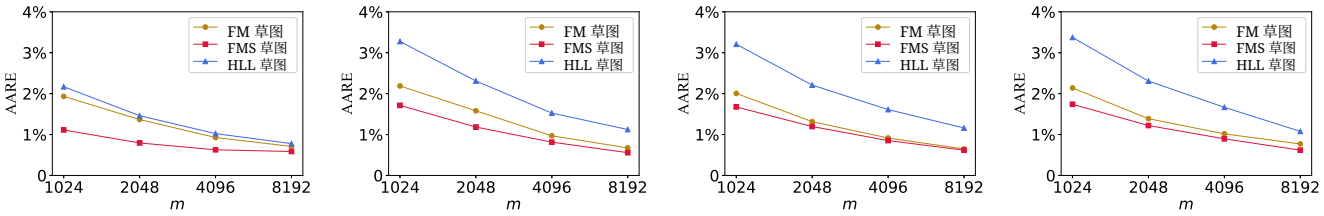
我们通过实验评估协议 DP-DICE 的性能，并将其与当前最先进的 MPC-FM [11] 和 PSC [29] 进行比较，这两种方案的代码均已公开。我们参照 MPC-FM，使用 C++ 实现了 DP-DICE 协议原型。为促进未来的研究，我们将向公众发布代码。

## 5.1 实验设置

•参数设置。我们在相同数量的位数组（或寄存器）$m$ 下，将 FMS 草图与 FM 草图和 HyperLogLog 草图进行比较。如 [7], 所述，对于 FM 草图，只需将 $w$ 设为 $\lceil \log_2 n + 6 \rceil$ 即可。对于 HLL 草图和 FMS 草图，只需将 $w$ 设为 $\lceil \log2(n/m) + 6 \rceil$，因为每个寄存器（或位数组）预计会接收 $n/m$ 个元素。在评估 FMS 草图的准确性时，我们改变差分隐私参数 $\varepsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$、位数组数量 $m \in \{1024, 2048, 4096, 8192\}$ 以及 DH 数量 $d \in \{5, 10, 15, 20, 25\}$。在评估 DP-DICE 的运行时间和通信开销时，我们改变 CP 数量，即 $c \in \{2, 3, \ldots, 7\}$。默认情况下，我们将 DH 数量 $d = 20$、CP 数量 $c = 5$、基数值 $n = 10^6$、草图数量 $m = 4096$、隐私参数 $\varepsilon$ 及其对应的 $\varepsilon_d$ 分别设为 0.1 和 0.012。除非另有说明，我们均采用这些默认值作为实验参数。

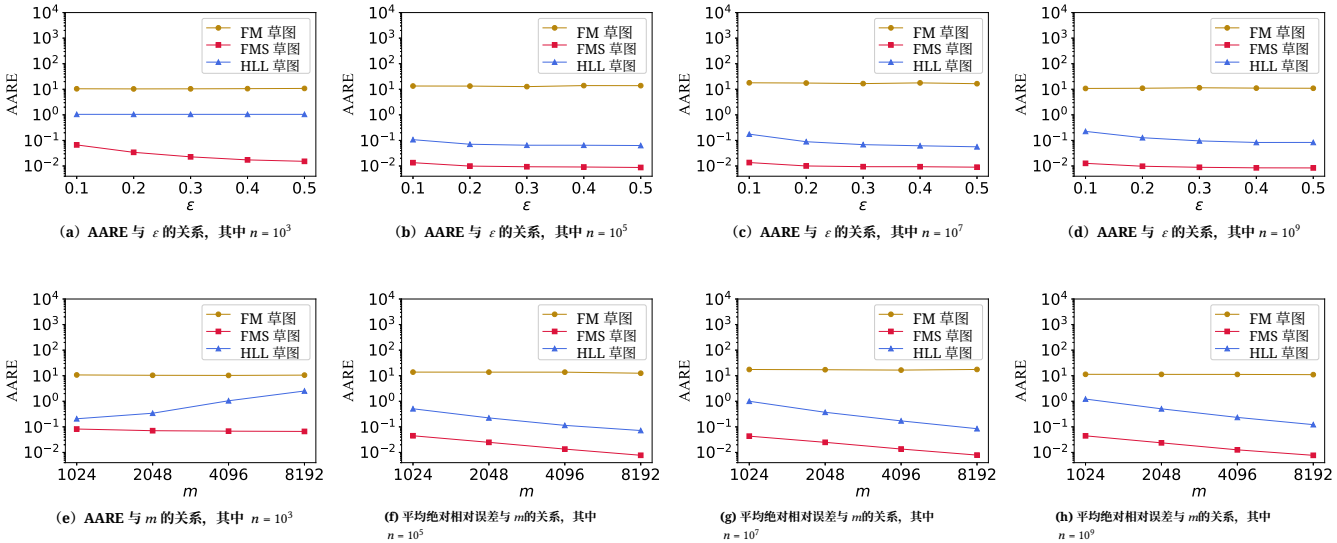
•**计算与网络环境**。我们在局域网（Local Area Network）和广域网（Wide Area Network）环境中进行了大量实验。在广域网环境中，CP 和 DH 分别均匀分布在中国北京、广州和上海的三个不同华为数据中心。华为云服务器使用 c7.8xlarge.2 类型的 ECS 实例，每个实例配备 32 个 vCPU（16 个物理核心），采用基于 Intel Ice Lake 3.0GHz 系列 CPU 的处理器、64GB 随机存取存储器，且每个网络接口的最大带宽为 30 Gbit/s。在局域网环境中，每个 CP 使用一台配备 2 颗四核 Intel(R) Xeon(R) Gold 6140 @ 2.30GHz CPU 的 Linux 服务器，每个 DH 则使用一台配备常见硬件配置的台式机，其处理器为 Intel 四核 i7-10400F CPU，随机存取存储器为 16GB。

•**数据集与度量指标**。我们随机生成多个大小为 $\{10^3, 10^5, 10^7, 10^9\}$ 的独立集合，并将其用作并集 $S$。我们使用度量指标 平均绝对相对误差 (平均绝对相对误差) 来衡量估计值 $\hat{n}$ 相对于真实值的准确性




（a）平均绝对相对误差与 $m$的关系，其中 $n = 10^3$　（b）平均绝对相对误差与 $m$的关系，其中 $n = 10^5$　（c）平均绝对相对误差与 $m$的关系，其中 $n = 10^7$　（d）平均绝对相对误差与 $m$的关系，其中 $n = 10^9$

**图 3：未考虑隐私时，我们的 FMS 草图与 FM 草图和 HLL 草图的准确度对比。**




（a）AARE 与 $\varepsilon$ 的关系，其中 $n = 10^3$　（b）AARE 与 $\varepsilon$ 的关系，其中 $n = 10^5$　（c）AARE 与 $\varepsilon$ 的关系，其中 $n = 10^7$　（d）AARE 与 $\varepsilon$ 的关系，其中 $n = 10^9$

（e）AARE 与 $m$ 的关系，其中 $n = 10^3$　（f）平均绝对相对误差与 $m$的关系，其中 $n = 10^5$　（g）平均绝对相对误差与 $m$的关系，其中 $n = 10^7$　（h）平均绝对相对误差与 $m$的关系，其中 $n = 10^9$

**图 4：在中心差分隐私设置下，我们的 FMS 草图与 FM 草图和 HLL 草图的准确性比较。**

基数值 $n$。形式上，我们将 $\hat{n}$ 的平均绝对相对误差定义为 $\mathbb{E}\left(\frac{|\hat{n}-n|}{n}\right)$。此外，我们还评估 DH 与 CP 之间的运行时间和通信开销。默认情况下，所有实验结果均为 100 次独立运行的平均值。

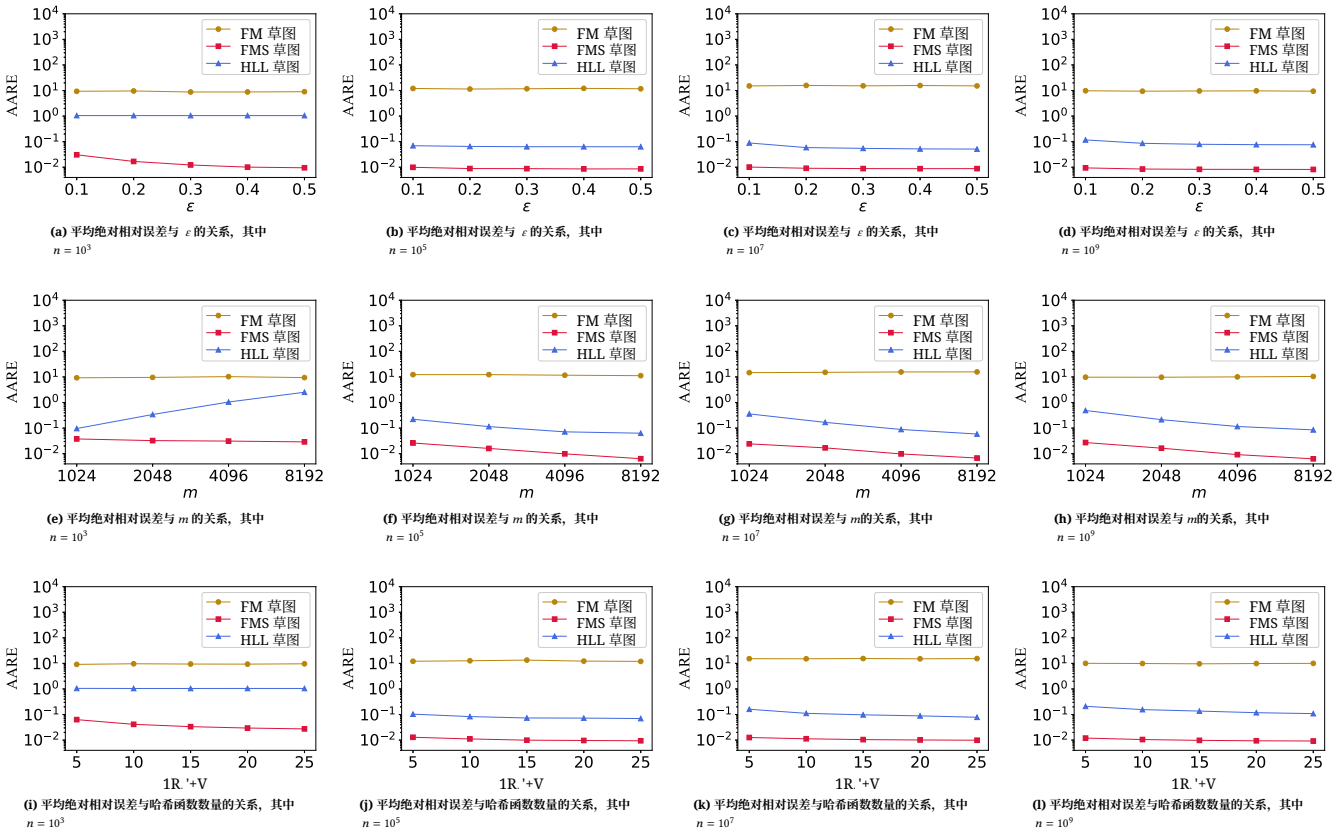
## 5.2 结果

5.2.1 FMS 与 FM 和 HLL 对比。在图 3 中，我们展示了未考虑隐私时 FMS 草图与 FM 草图和 HLL 草图的准确性对比。我们比较了在不同$m$（寄存器数量从 {1024, 2048, 4096, 8192} 中选取）和不同基数规模 $n \in \{10^3, 10^5, 10^7, 10^9\}$ 下的草图方法。正如我们在第 3.1、3.2 和 4.1 节中讨论的，这些草图方法的标准误差都与 $1/\sqrt{m}$ 成正比。当基数规模固定时，随着 $m$ 增大，它们的 AARE 会降低，这与理论分析十分吻合。我们的 FMS 草图的 AARE 始终低于其他方法。在图 3(a) 中，$n = 10^3$，当 $m$ 为 1024、2048、4096、8192 时，我们的 FMS 草图的 AARE 分别比 FM 草图低 1.7、1.7、1.5、1.2 倍，比 HLL 草图低 2.0、1.8、1.6、1.3 倍。当 $n = 10^5$ 和 $m$ 分别为 1024、2048、4096、8192 时，FMS 草图的 AARE 为分别比 FM 草图低 1.3、1.3、1.2、1.2 倍，比 HLL 草图低 1.9、1.9、1.9、2.0 倍。对于 $n \in \{10^7, 10^9\}$，也呈现出类似的规律。

需要指出的是，在使用相同的 $m$时，HLL 所需的内存空间比 FMS 和 FM 草图少 5 倍。在内存使用量相同的情况下，我们的 FMS 比 FM 草图更准确，但不如 HLL 草图准确。尽管如此，正如我们稍后讨论的那样，在我们的协议中，草图的内存使用量并非瓶颈。

*5.2.2 DP-DICE 与 MPC-FM 对比。*然后，我们展示在添加加性噪声变量以满足差分隐私时，所提出的 FMS 草图的准确性，并进一步验证 DP-DICE协议相较于 MPC-FM 的效率。

**准确性。**图4 展示了在相同差分隐私条件下，我们的 FMS 草图与 FM 和 HLL 草图的准确性对比。正如我们在第3节中所述，HLL 的估计量包含复杂的幂运算，这限制了它在 SPDZ 框架中的实现。因此，我们采用 HLL 早期版本中的估计量（即 LogLog 草图 [30]），即 $\tilde{n} = \tilde{\alpha}_m m 2^{Z^\#}$，




(a) 平均绝对相对误差与 $\varepsilon$ 的关系，其中 $n = 10^3$

(b) 平均绝对相对误差与 $\varepsilon$ 的关系，其中 $n = 10^5$

(c) 平均绝对相对误差与 $\varepsilon$ 的关系，其中 $n = 10^7$

(d) 平均绝对相对误差与 $\varepsilon$ 的关系，其中 $n = 10^9$

(e) 平均绝对相对误差与 $m$ 的关系，其中 $n = 10^3$

(f) 平均绝对相对误差与 $m$ 的关系，其中 $n = 10^5$

(g) 平均绝对相对误差与 $m$的关系，其中 $n = 10^7$

(h) 平均绝对相对误差与 $m$的关系，其中 $n = 10^9$

(i) 平均绝对相对误差与哈希函数数量的关系，其中 $n = 10^3$

(j) 平均绝对相对误差与哈希函数数量的关系，其中 $n = 10^5$

(k) 平均绝对相对误差与哈希函数数量的关系，其中 $n = 10^7$

(l) 平均绝对相对误差与哈希函数数量的关系，其中 $n = 10^9$

**图5：我们的 FM S 草图在分布式差分隐私设置下与 FM 草图和 HLL 草图的比较 s.**

其中 $\tilde{\alpha}_m \approx 0.783$ 和 $Z\# = \frac{1}{m}\sum_{i=1}^{m} R[i]$。我们直接向 HLL 的变量 $Z\#$、FM 的变量 $Z^*$（定义见公式 (1)）以及我们的 FMS 变量 $Z$（定义见公式 (2)）添加加性高斯噪声，以实现 $(\varepsilon, \delta)$-差分隐私，其中我们按照 [11] 中的设置令 $\varepsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$，并令 $\delta = 10^{-12}$。

在图 4 15 5, 30 8, 44 5, 57 0, 68 2 $n = 10^3$ (a)中，我们可以看到，当 $\varepsilon = 0.1$ 分别取 0.2、0.3、0.4 和 0.5 时，我们的 FMS 的 AARE 分别比 HLL 小.....倍。当 $\varepsilon = 0.1$ 分别取 0.2、0.3、0.4 和 0.5 时，与 FM 相比，我们的 FMS 将 AARE 分别降低了 155、308、454、575 和 700 倍。当 $n = 10^5$ 时，如图4(b)所示，我们的 FMS 的 AARE 比 HLL 小 7.6、7.2、7.2、6.8 和 7.2 倍，比 FM 小 963、1337、1461、1498 和 1617 倍。当 $n = 10^7$ 和 $n = 10^9$ 时，也呈现类似的规律。FM 的 AARE 始终较高，因为其变量 $Z^*$ 的敏感度为 $\Delta_{Z^*} = wm$，若要达到相同的差分隐私级别，就需要加入显著更大的噪声。

在图4(e)–(h)中，我们比较了不同 $m$ 下 FMS、FM 和 HLL 的准确性。当 $n = 10^3$ 时，AARE 对估计不准确更为敏感，而我们的 FMS 的 AARE 仍小于 0.08，而 FM 和 HLL 的 AARE 则非常大。HLL 在基数规模较小时表现不佳，例如，当 $n = 0$ 时，HLL 的输出约为 $0.7m$。当 $n$ 分别设为 $10_5$、$10_7$ 和 $10_9$ 时，对于任意 $m$，FMS 的 AARE 大约比 HLL 小 10 到 15 倍，比 FM 小 300 倍以上。

如前所述，在我们的问题设定中，在 SPDZ 框架下直接向 FMS 的变量 $Z$ 添加连续噪声并不容易，而且不存在可信中央服务器。因此，我们在分布式离散高斯机制设定下评估 FMS、FM 和 HLL 的 AARE。在图 5(a)–(d)中，我们比较了使用分布式离散高斯机制时，不同 $\varepsilon$ 值下 FMS、FM 和 HLL 的 AARE，其中 $\varepsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$。在图 5(a)中，当 $n = 10^3$ 时，对于 $\varepsilon = 0.1$、0.2、0.3、0.4、0.5，FMS 的 AARE 分别比 HLL 小 34.4、63.6、83.3、100.6 和 111.1 倍，比 FM 小 310、591、758、883 和 931 倍。在图 5(b)中，当 $n = 10^5$ 时，与 HLL 相比，对于 $\varepsilon = 0.1$、0.2、0.3、0.4、0.5，FMS 将 AARE 分别降低了 7.4、7.2、7.2、7.3 和 7.3 倍。与 FM 相比，对于 $\varepsilon = 0.1$、0.2、0.3、0.4、0.5，FMS 将 AARE 分别降低了 1198、1321、1288、1425 和 1360 倍。此外，$n = 10^7$ 和 $n = 10^9$ 也呈现类似的模式。

在图 5(e)–(h) 中，我们将 $m$ 从 1,024 变化到 8,192，以进一步研究 $m$的影响。在图 5(e) 中，当 $n = 10^3$ 时，FMS 的 AARE 始终低于 0.038，而 FM 则始终大于 9。与连续噪声设置中的情况相同，当 $m$ 增大时，HLL 的 AARE 甚至会增大。在图 5(f) 中，对于 $m = 1024$，2048、4096、8192，FMS 的 AARE 分别比 HLL 小 . . . 倍，比 FM 小 1810 倍。

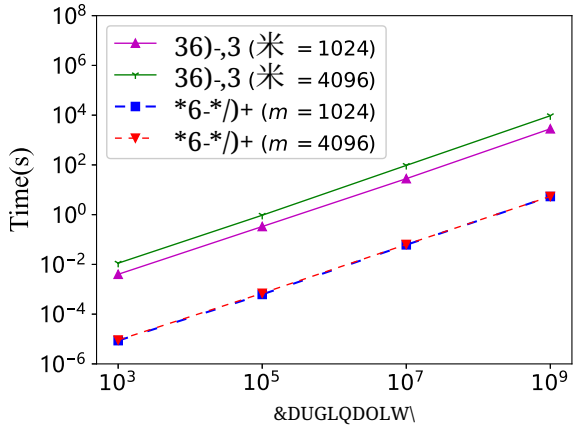



**图 6：DP-DICE 中 CP 的草图生成时间。**

。此外，FMS 和 HLL 的 AARE 大致与$1/\sqrt{m}$成正比。图 $n = 10^7$ 和 $n = 10^9$ 中的类似规律也适用于图 5(g)–(h)。

在图 5(i)–(l) 中，我们将 DH 的数量从 5 增至 25，以观察在 DH 数量不同时，FMS 的平均绝对相对误差是否稳定。我们默认设置 $m = 4096$ 和 $\varepsilon = 0.1$。如图 5(i) 所示，当 $n = 10^3$时，随着 DH 数量从 5 增至 25，FMS 的平均绝对相对误差从 0.062 降至 0.026。相比之下，FM 和 HLL 的平均绝对相对误差始终分别大于 8 和 1.04。当 $n = 10^5$时，随着 DH 数量从 5 增至 25，FM 的平均绝对相对误差略微从 0.012 降至 0.010。与此同时，FMS 将 HLL 和 FM 的平均绝对相对误差分别降低了 6 倍以上和 10 倍以上。图 $n = 10^7$ 和 $n = 10^9$ 中的 5(k)–(l) 也呈现出类似的规律。

**内存开销**。不难发现，我们的 DP-DICE 与 MPC-FM 协议具有相同的内存复杂度。这两个协议都包含在线阶段和离线阶段，因此我们分别记录它们在在线阶段和离线阶段的内存开销。实验表明，DP-DICE 和 MPC-FM 所需的内存空间都很小。DP-DICE 在离线阶段的内存开销约为每个计算方 413MB。DP-DICE 在线阶段的协议分为计算方执行和数据持有方执行。每个计算方执行协议约消耗 302MB 内存，每个数据持有方执行协议约消耗 21MB 内存。在实际应用中，计算方运行在服务器上，413MB 的内存开销对服务器而言并不算大。协议的大部分内存开销来自 SPDZ 协议，而 FMS 草图仅消耗数百 KB，几乎可以忽略不计。每个数据持有方只需为 FMS 草图分配内存，因此内存开销也很小。

**运行时间和通信开销**。我们比较了在不同设置下 DP-DICE 与 MPC-FM 的性能$n \in \{10_3,\ 10_5,\ 10_7,\ 10_9\}$，$m \in \{1024,\ 4096\}$，以及 $c \in \{2,\ 3,\ \ldots\ldots,\ 7\}$。在数据收集阶段，每个数据提供方首先计算其持有集合的草图；当集合较大时，这项计算开销很大。在图6中，当 $m = 1024$，我们看到，DP-DICE 将 MPC-FM 的草图生成时间分别缩短了 465.12 倍、536.13 倍，457.23 倍和 526.44 倍，分别对应 $n = 10^3$、$10_5$、$10_7$、$10_9$。当 $m = 4096$时，DP-DICE 的优势更加显著。从图7(a)-(d) 中可以看出，DP-DICE 将 MPC-FM 的在线通信开销降低了 1.7 至 2.4 倍，并将离线通信开销降低了 1.4 至 3.7 倍。

我们评估了 DP-DICE 和 MPC-FM 在两种不同网络环境（局域网和广域网）下的离线准备时间和在线运行时间。局域网环境下的结果如图 7(e)-(h) 所示，广域网环境下的结果如图 7(e)-(h) 所示。在局域网环境下，我们发现 DP-DICE 将在线运行时间缩短了 1.6 至 2.3 倍，将离线准备时间缩短了 1.6 至 3.7 倍。在广域网环境下，我们发现 DP-DICE 将在线运行时间缩短了 1.4 至 2.0 倍，将离线准备时间缩短了 1.6 至 2.6 倍。

5.2.3 DP-DICE 与 PSC。与 MPC-FM 和 DP-DICE 不同，PSC 发布的代码 [29] 使用 Go 编写，且不完全支持多线程。为公平比较，Hu 等人 [11] 使用 C++ 重新实现了 PSC 协议，并指出新实现的性能远优于 PSC 原论文中报告的性能。虽然这一 C++ 实现并未公开，但我们采用了 Hu 等人 [11]给出的 PSC 这一改进实现的结果。对于在 {20, 000, 30, 000, 40, 000, 50, 000} 中变化的基数 $n$，PSC 的平均绝对相对误差在 $\epsilon = 0.1$ 和 $\epsilon = 0.3$时分别约为 0.05 和 0.04（结果来自 MPC-FM[2]完整版中的图10）。对于 $n \in$ {20, 000, 30, 000, 40, 000, 50, 000}，当 $\epsilon = 0.1$ 时，我们的 DP-DICE 的 AARE 在 0.0079 至 0.0097 之间变化；当0090 $\epsilon = 0.3$ 时，变化范围为 0.0064 至 0.。我们可以看到，DP-DICE 的准确度比 PSC 高出数倍。Hu 等人证明，MPC-FM 将 PSC 的在线运行时间缩短了多个数量级。从图 7 的结果可以看出，DP-DICE 比 MPC-FM 更快，因此其效率也显著高于 PSC。

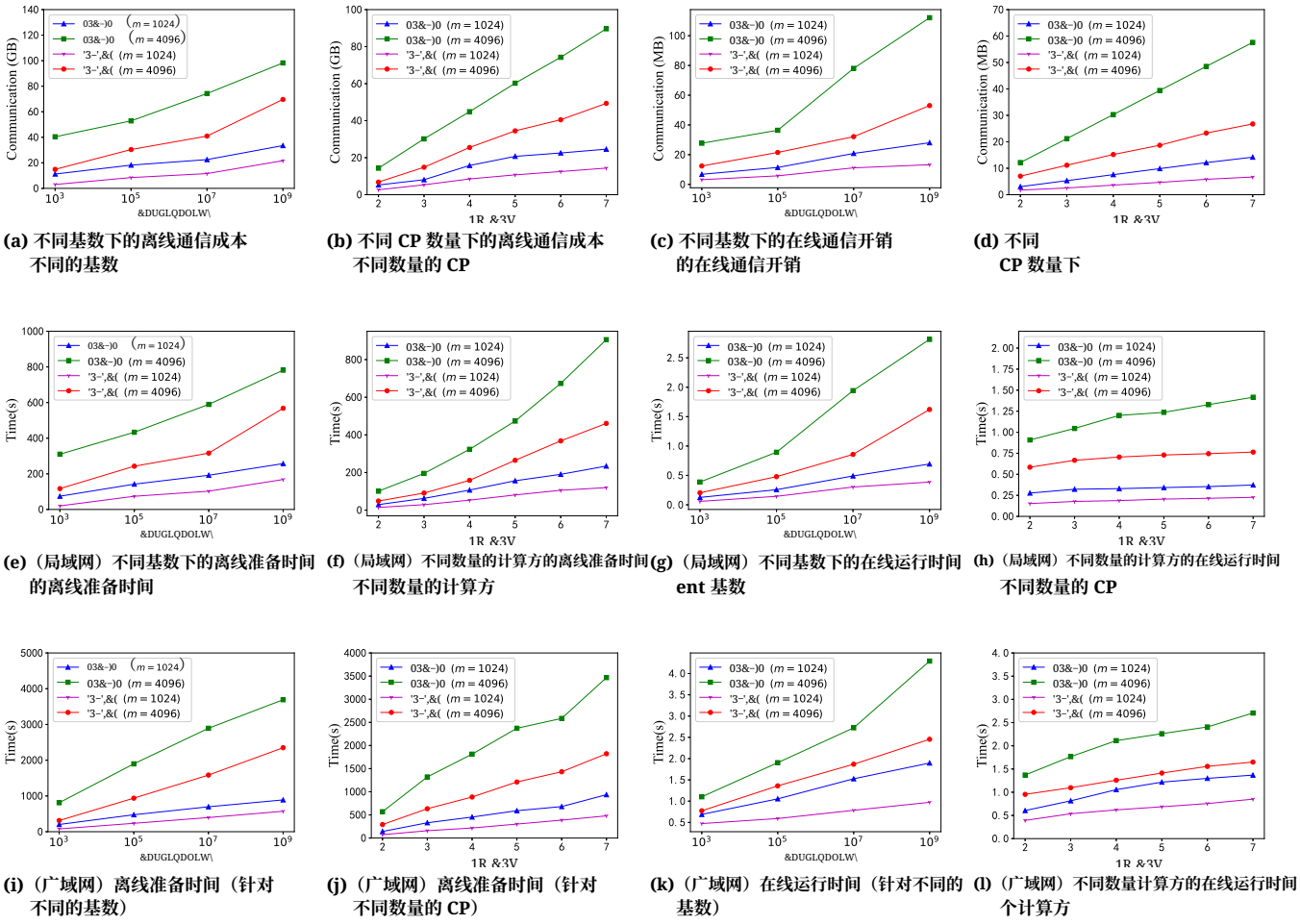
*5.2.4 DP-DICE 与 MPC* 排序与比较。在多方计算下计算不同元素数量的一种直接方法是对元素进行排序并去重。我们在相同的实验环境中，将这种 MPC 排序与比较方法与我们的 DP-DICE 进行比较。我们评估了三台 CP 的 MPC 排序协议的效率。当存在 $n = 10^2$ 个不同元素时，MPC 排序协议需要 561 秒，比我们的 DP-DICE 慢 9 倍。当 $n = 10^5$时，它需要约 20 小时，比我们的 DP-DICE 慢 468 倍。显而易见，MPC 排序协议的执行时间会随着 $n$显著增加。当 $n \geq 10^6$时，MPC 排序协议需要超过一周的时间。上述实验表明，我们的 DP-DICE 协议比 MPC 排序与比较协议高效得多。

## 6 相关工作

PDCE 协议分为两种：DH-PDCE 和 CP-PDCE。对于 DH-PDCE 协议，DH 会收集数据，同时也作为 CP 参与计算。与 DH-PDCE 不同，CP-PDCE 协议中的 DH 只负责收集数据，而 CP 则负责计算估计值。

**DH-PDCE协议**。据我们所知，Private SetUnionCardinality（PSU-CA） [31] 是第一个DH-PDCE协议。基于容斥原理，它将

[2]https://changyudong.site/assets/pdf/USENIX_21_FULL.pdf




**(a) 不同基数下的离线通信成本** **(b) 不同 CP 数量下的离线通信成本** **(c) 不同基数下的在线通信开销** **(d) 不同**

**不同的基数** **不同数量的 CP** **的在线通信开销** **CP 数量下**

**(e)（局域网）不同基数下的离线准备时间** **(f)（局域网）不同数量的计算方的离线准备时间** **(g)（局域网）不同基数下的在线运行时间** **(h)（局域网）不同数量的计算方的在线运行时间**

**的离线准备时间** **不同数量的计算方** **ent 基数** **不同数量的 CP**

**(i)（广域网）离线准备时间（针对** **(j)（广域网）离线准备时间（针对** **(k)（广域网）在线运行时间（针对不同的** **(l)（广域网）不同数量计算方的在线运行时间**

**不同的基数）** **不同数量的 CP）** **基数）** **个计算方**

**图 7：通信成本 我们的 DP-DICE 与 MPC-FM 的通信成本、离线准备时间和在线运行时间对比 .**

原始PDCE问题转化为对一系列集合交集基数的私有计算，这可以通过成熟的隐私集合求交（PSI）协议来解决。PSU-CA的输出是精确基数，而这对于许多现实应用来说并非必要。Davidson和Cid [32] 利用成员查询草图方法 布隆过滤器 来私下估计集合交集的基数，大幅提升了PSU-CA的速度。然而，容斥原理涉及的加数数量随DH数量呈指数增长，因此当数据持有方众多时，计算仍然十分密集。为解决这一问题，[33, 34] 提出了基于FM 草图的协议 [7],，这是一种广为人知的基数估计方法。然而，上述协议均不具备差分隐私。近期，Chen等人 [35] 针对每个数据持有方仅持有一个元素的特定场景，提出了一种DH-PDCE协议；Jia等人 [36] 则提出了一种私下计算两个集合并集的协议，这与我们研究的问题不同。

**CP-PDCE协议。** Ashok 和 Mukkamala [37] 提出了一种基于布隆过滤器的 CP-PDCE 协议。Egert等人 [38]揭示指出该协议并不安全，并提出了一种更安全的协议变体。遗憾的是，这两种协议都不支持差分隐私。在 [39], 中，每个 DH 都将其本地集合表示为位向量，使用随机响应法扰动该向量以实现差分隐私，然后将向量发送给 CP，以估计并集的基数。然而，该协议的估计误差很大（与全集大小处于同一量级），无法满足许多应用的需求。在[40], 中，作者使用 LogLog 草图和拉普拉斯机制来估计基数。然而，由于 LogLog 算法的敏感度较高，需要加入大量噪声才能满足差分隐私要求，因此同样会产生较大的估计误差。PSC 协议 [29, 41] 为每个 DP 使用公开哈希函数维护一个哈希表。对于由多个 DH 持有的元素，PSC 会将其哈希到哈希表中的同一个桶，从而便于消除重复项。为了达到合理的准确度，PSC 需要将哈希表的大小设置得远大于并集的基数，这严重限制了其效率和可扩展性。Hu 等人 [11] 提出了一种 CP-PDCE 协议，实现了安全的

基于 FM 草图的计算。他们利用 FM 草图固有估计方差所带来的不确定性，生成差分隐私输出 [9]。然而，当 DH 诚实但好奇时，该协议不满足差分隐私（详见第3.5 节）。Ghazi等人 [4] 提出同时估计并集的基数和频率直方图（即在所有 DH 中出现给定次数的元素所占的比例）。他们的协议基于一种布隆过滤器变体，包含多个阶段，并通过注入噪声来保证差分隐私。该协议较为复杂，且涉及许多需要设置的参数，因此不易实现。

## 7 结论与未来工作

本文提出了一种安全高效的协议 DP-DICE，用于解决 PDCE 问题。实验结果证明了 DP-DICE 的效率和效果。与 MPC-FM 协议相比，DP-DICE 的草图生成速度提升了千倍，在线运行时间缩短了约一半，估计误差降低了一千多倍。与 PSC 协议相比，DP-DICE 的在线运行时间缩短了几个数量级，估计误差降低了数倍。与近期提出的采用安全聚合协议的联邦学习类似 [16]，我们注意到，在 SPDZ 框架上聚合分布式离散高斯噪声变量时，DP-DICE 也可能出现模回绕，这不会损害安全性和精度，但会影响准确性。幸运的是，由于我们的协议通常采用很大的模数 $p$，因此这种模裁剪发生的概率极低。未来，我们计划彻底解决这一问题，并扩展协议以处理更复杂的情况。例如，某些应用中可能存在少数恶意DH。此外，DP-DICE 使用的 MPC 框架是 SPDZ，该框架并不具备鲁棒性。换言之，如果任何计算方中止，DP-DICE 协议就会终止且不返回结果。鲁棒MPC是一个活跃的研究领域，待鲁棒MPC框架可用后，我们将把 DP-DICE 协议迁移至该框架。

## 参考文献


[1] Wang Zhengxia 和 Xiao Laisheng。基于物联网的现代物流监测平台。载于 *2010*年智能计算技术与自动化国际会议，第2卷，第726–731页。IEEE，2010年。[2] Alex Berke、Michiel A. Bakker、Praneeth Vepakomma、Ramesh Raskar、Kent Larson 和 Alex 'Sandy' Pentland。利用位置历史评估疾病暴露风险并保护隐私：应对全球大流行病的密码学方法。*CoRR*，abs/2003.14412，2020年。[3] Andreas Ulbrich、Evgeny Sergeevich Skvortsov、Jeffrey Scott Wilhelm、Josh Bao、Lawrence Tsang 和 Will Bradbury。利用 HyperLogLog 跟踪受众统计数据。2021年。[4] Badih Ghazi、Ben Kreuter、Ravi Kumar、Pasin Manurangsi、Jiayu Peng、Evgeny Skvortsov、Yao Wang 和 Craig Wright。多方覆盖人数与频次直方图：私密、安全且实用。*PoPETs*，2022年(1)：373–395，2022年。[5] Roger Dingledine、Nick Mathewson 和 Paul F. Syverson。Tor：第二代洋葱路由器。载于 Matt Blaze（编），第*13*届 *USENIX* 安全研讨会论文集，*2004*年*8*月*9*日至*13*日，美国加利福尼亚州圣迭戈，第303–320页。USENIX，2004年。[6] Hazar Harmouch 和 Felix Naumann。基数估计：一项实验性综述。*VLDB*基金会论文集，11(4)：499–512，2017年。[7] Philippe Flajolet 和 G Nigel Martin。面向数据库应用的概率计数算法。计算机与系统科学杂志，31(2)：182–209，1985年。[8] Stefan Heule、Marc Nunkesser 和 Alexander Hall。实践中的 HyperLogLog：最先进基数估计算法的算法工程。载于*EDBT*，2013年。

[9] Adam Smith、Shuang Song 和 Abhradeep Guha Thakurta。Flajolet-Martin 草图本身即可保护差分隐私：以最小空间实现隐私计数。《神经信息处理系统进展》，33：19561–19572，2020年。[10] Charlie Dickens、Justin Thaler 和 Daniel Ting。（几乎）所有基数估计器都满足差分隐私，2022年。[11] Changhui Hu、Jin Li、Zheli Liu、Xiaojie Guo、Yu Wei、Xuan Guang、Grigorios Loukides 和 Changyu Dong。如何让隐私保护分布式基数估计切实可行，并免费获得差分隐私。载于《USENIX Security》，2021年。[12] Philippe Flajolet、Éric Fusy、Olivier Gandouet 和 Frédéric Meunier。HyperLogLog：近乎最优基数估计算法的分析。载于《离散数学与理论计算机科学》，第137–156页。离散数学与理论计算机科学，2007年。[13] Cynthia Dwork、Krishnaram Kenthapadi、Frank McSherry、Ilya Mironov 和 Moni Naor。我们的数据，由我们做主：通过分布式噪声生成实现隐私。载于《EUROCRYPT》，2006年。[14] Cynthia Dwork 和 Guy N Rothblum。集中差分隐私。arXiv 预印本 arXiv:1603.01887，2016年。[15] Mark Bun 和 Thomas Steinke。集中差分隐私：简化、扩展与下界。载于《密码学理论会议》，第635–658页。施普林格，2016年。[16] Peter Kairouz、Ziyu Liu 和 Thomas Steinke。用于安全聚合联邦学习的分布式离散高斯机制。载于《国际机器学习大会》，2021年。[17] Clément L Canonne、Gautam Kamath 和 Thomas Steinke。用于差分隐私的离散高斯分布。《神经信息处理系统进展》，33：15676–15688，2020年。[18] Cynthia Dwork、Aaron Roth 等。差分隐私的算法基础。《理论计算机科学基础与趋势》，9 407，2014 3 4 211( – ): – 。[19] Ilya Mironov。最低有效位对差分隐私的重要性。载于《2012年ACM计算机与通信安全会议论文集》，第650–661页，2012年。[20] Ivan Damgård、Valerio Pastro、Nigel P. Smart 和 Sarah Zakarias。基于部分同态加密的多方计算。载于《CRYPTO》，2012年。[21] Bargav Jayaraman、Lingxiao Wang、David Evans 和 Quanquan Gu。无忧分布式学习：隐私保护经验风险最小化。载于 Samy Bengio、Hanna M. Wallach、Hugo Larochelle、Kristen Grauman、Nicolò Cesa-Bianchi 和 Roman Garnett 编，《神经信息处理系统大会》，第6346–6357页，2018年。[22] Michael Mitzenmacher 和 Eli Upfal。《概率与计算：随机算法与概率分析》。剑桥大学出版社，2005年。[23] Patrick Billingsley。《概率与测度》。约翰·威立父子出版公司，第二版，1986年。[24] Jay M. Ver Hoef。谁发明了德尔塔方法？《美国统计学家》，66(2)：124–127，2012年。[25] Marcel Keller、Valerio Pastro 和 Dragos Rotaru。Overdrive：让SPDZ再创辉煌。载于《EUROCRYPT》，2018年。[26] Jonathan Katz 和 Moti Yung。可扩展的认证群组密钥交换协议。载于《CRYPTO》，2003年。[27] Helger Lipmaa 和 Tomas Toft。具有次线性在线复杂度的安全相等性与大于测试。载于《国际自动机、语言与编程研讨会》，2013年。[28] Ran Canetti。通用可组合安全。《美国计算机协会会刊》，67(5)：28:1–28:94，2020年。[29] Ellis Fenske、Akshaya Mani、Aaron Johnson 和 Micah Sherr。具有隐私集合并集基数的分布式测量。载于《计算机与通信安全会议》，2017年。[30] Marianne Durand 和 Philippe Flajolet。大基数的LogLog计数（扩展摘要）。载于 Giuseppe Di Battista 和 Uri Zwick 编，《欧洲算法研讨会》，2003年。[31] Emiliano De Cristofaro、Paolo Gasti 和 Gene Tsudik。快速且隐私保护的集合交集与并集基数计算。载于《密码学与网络安全会议》，2012年。[32] Alex Davidson 和 Carlos Cid。用于计算隐私集合运算的高效工具包。载于《亚太信息安全与隐私会议》，2017年。[33] Florian Tschorsch 和 Björn Scheuermann。一种隐私保护分布式用户统计的算法。《计算机网络》，57(14)：2775–2787，2013年。[34] Changyu Dong 和 Grigorios Loukides。以对数复杂度近似估计隐私集合并集／交集基数。《IEEE信息取证与安全汇刊》，12(11)：2792–2806，2017年。[35] Lijie Chen、Badih Ghazi、Ravi Kumar 和 Pasin Manurangsi。关于分布式差分隐私与不同元素计数。载于《创新理论计算机科学会议》，第185卷，2021年。[36] Yanxue Jia、Shi-Feng Sun、Hong-Sheng Zhou、Jiajun Du 和 Dawu Gu。基于洗牌的隐私集合并集：更快、更安全。《IACR密码学电子印本档案》，第157页，2022年。[37] Vikas G. Ashok 和 Ravi Mukkamala。利用布隆过滤器实现可扩展且高效的隐私保护全局项集支持度近似。载于《数据库安全会议》，2014年。[38] Rolf Egert、Marc Fischlin、David Gens、Sven Jacob、Matthias Senker 和 Jörn Tillmanns。通过布隆过滤器私密计算集合并集与交集基数。载于《亚太信息安全与隐私会议》，2015年。[39] Rade Stanojevic、Mohamed Nabeel 和 Ting Yu。带差分隐私的集合运算分布式基数估计。载于《隐私增强技术会议》，第37–48页，2017年。[40] Seung Geol Choi、Dana Dachman-Soled、Mukul Kulkarni 和 Arkady Yerukhimovich。面向大规模统计的差分隐私多方草图。

密码学电子印本档案，2020年。[41] Ellis Fenske、Akshaya Mani、Aaron Johnson 和 Micah Sherr。面向分布式测量的可问责的私有集合基数。*ACM* 隐私事务与安全，2022年。