有多少用户曾在此停留很长时间？长时间聚合访问计数的高效解决方案

PEYMAN AFSHANI,奥胡斯大学，丹麦REZAUL CHO
WDHURY,石溪大学，美国INGE LI GØRTZ,丹麦技术大学，丹麦
MAYANK GOSWAMI,纽约市立大学皇后学院，美国FRA
NCESCO SILVESTRI*，帕多瓦大学，意大利MARIAFIORE
TOGNON, 帕多瓦大学，意大利

arXiv:2601.09489v1 [cs.DS] 14 Jan 2026

本文探讨长时间聚合访问计数问题，其定义如下。给定 n 个用户和 m 个区域，每位用户会花费一些时间访问其中若干区域。对于参数 k，以及由 r 个区域的一个子集构成的查询，任务是统计在查询区域内累计停留时间至少为 k 的不同用户数量。该问题的研究动机源于对大规模移动数据集进行分析时产生的查询。

我们提出了多种用于支持长时间聚合访问计数的精确和近似数据结构，并给出了条件下界和无条件下界。首先，我们介绍一种具有空间时间权衡的精确数据结构，以及基于采样和草图技术的高效近似解。随后，我们研究几何场景下的问题，其中区域是 $\mathbb{R}^d$ 中的点，查询则是超矩形，并推导出在这些结构化空间中性能更优的精确数据结构。

计算理论：• **计算理论** → **数据管理的数据结构与算法**；草图技术与采样；下界与信息复杂度。

附加关键词和短语：精确计数与近似计数、草图技术、几何算法、下界

1 引言

大规模移动数据集中一种常见的数据类型，是包含时间信息的地理定位数据 [PT14]。为简单起见，我们可以将这些数据集视为由三元组构成，其中每个三元组（u、r、t）表示某个用户 u在某个区域 r停留了一段时间 t。根据所采用的数据采集技术不同，区域的表示方式也可能有所差异。一种常见方法利用移动网络日志（即 CDR），记录每部智能手机在特定时间所连接的蜂窝基站。这些数据集覆盖广泛的人群；但是，它们无法确定用户的确切位置，只能表明用户位于某个蜂窝基站的覆盖区域内。另一种方法依赖于会定期记录 GPS 坐标及时间戳的应用，例如导航应用。尽管这些数据可能无法提供与网络日志相同的统计保证，但能够提供精确的 GPS 坐标。移动数据集通常规模非常庞大：例如，电信公司 Vodafone 每天仅从意大利的用户那里就收集约 300 亿条记录 [VF23]，而 TomTom 每天从其导航设备收集 610 亿个 GPS 数据点 [Mar19]。

这些出行数据集用于提取人类行为相关信息，例如用于城市发展、经济学和旅游业。针对这些数据集的常见查询包括：“有多少通勤者在高速公路上堵了超过 *1* 小时？”、“如何

*通讯作者：francesco.silvestri@unipd.it。

作者联系方式：PEYMAN AFSHANI，奥胡斯大学，丹麦；REZAUL CHOWDHURY，石溪大学，美国；Inge Li Gørtz，丹麦技术大学，丹麦；MAYANK GOSWAMI，纽约市立大学皇后学院，美国；FRANCESCO SILVESTRI，帕多瓦大学，意大利；MARIAFIORE TOGNON，帕多瓦大学，意大利。

CLAV 类型	空间	查询时间	保证	定理
$(k,\ r)$-CLAV，精确解	S	$\tilde{\Omega}\left(N/S^{1/r}\right)$	条件下界	2.1
(k, r)-CLAV，精确解	$O((N/\lambda)^r n + N)$	$O(r\lambda)$	对于任意 $\lambda > 0$	2.2
$(k,\ r)$-CLAV，近似解，基于采样	$O(N)$	$O\left(\frac{r^3}{\varepsilon^2}$对数$n\right)$	εn_Q 加性误差，具有高概率，$\varepsilon > 0$	3.1
(k, r)-CLAV，近似解，基于草图	$O(m\varepsilon^{-1}$日志$^2\, n$日志$\, r)$ bits $\Omega\left(\min\{n,\ (\frac{m}{2d})^{2d}\}\right)$比特	$O\left(\frac{1}{\varepsilon}\log^2 n\right)$	$\Theta(n_{Q,k} + \varepsilon n_Q)$ 加性误差 具有高概率，$\varepsilon = \Omega(n_{Q,\ k}/n_Q)$；它可能会统计累计时间为 $\geq$ 的用户 $k(1 - 1/r)$	3.2
几何 CLAV，精确	—	-	任意查询时间下的无条件下界 任意查询时间	4.1
几何 CLAV，精确	$\Omega(m^{2d-1-o(1)})$	O (1)	条件下界	4.2
几何 CLAV，精确，$d = 2$	$\Omega(m^{3-\varepsilon})$ 预处理 time	$O(m^{1-\varepsilon})$	.....的条件下界 任意 $\varepsilon > 0$	4.4
几何 CLAV，精确	O(分钟$\{Nm^{2d-2}, m^{2d}\}$)	O（对数$_w\, n_{Q,\ k}$）	对于 $\mathbb{R}^d$ 中的点	4.9

表 1。此处汇总了我们的主要结果。为便于阅读，界限以简化形式给出。除非另有说明，空间以字为单位。请注意，n 是用户数，m 是区域数，$N = |T|$、n_Q 是查询区域 Q 中的不同用户数，而 $n_{Q,\ k}$ 是查询区域 Q 中累计时间为 $\geq k$ 的不同用户数（所需输出）；$d = O(1)$ 是几何版本中点的维度。w 是字长，至少为 $\log(n + m)$。$\tilde{O}(\cdot)$ 和 $\tilde{\Omega}(\cdot)$ 符号隐藏了多对数因子。

有多少独立个体在市中心停留了一个多小时？", "有多少独立游客在威尼斯和佛罗伦萨停留了两天以上？"（例如 [LHV+14、MBGP13、CCDT+22]）。随着用户移动，这些查询需要识别不同区域中的同一用户，并计算其在目标区域内的总时长。考虑三元组（$u_1,\ r_1,\ 20h$）、（$u_2,\ r_1,\ 15h$）、（$u_1,\ r_2,\ 15h$）、（$u_3,\ r_2,\ 30h$）、（$u_2,\ r_3,\ 20h$），以及由区域 $\{r_1,\ r_2\}$ 定义的感兴趣区域：随后，两个用户$u_1,\ u_3$ 在该区域内停留了至少 30 小时，而 u_2 未达到要求的最短时长。这些问题所指定的区域可以转换为覆盖该感兴趣区域的蜂窝基站列表（用于网络日志），或转换为 GPS 坐标的边界框。此外，出于隐私原因，这些查询不要求列出用户，而是要求提供数量。请注意，在不同场景中也会出现类似查询，例如评估网站上的用户参与度时（例如 [SY18]）。

本文对上述问题给出了形式化定义，并将其命名为长时间聚合访问计数，简称*CLAV*。我们首先给出一个通用定义，该定义不对区域属性作任何假设，也可用于非几何对象（例如访问网页）。随后，我们定义几何 *CLAV*，这是一个变体，其中用户和区域属于欧几里得空间 $\mathbb{R}^d$。我们为这些问题提供了计算精确解和近似解的高效方法，以及无条件下界和条件下界。现在，我们将介绍 CLAV 和几何 CLAV 问题及其结果，这些内容也汇总在表 1 中。

1.1 (k, r)-CLAV

一般对象的长时间聚合访问计数问题的第一个定义如下：

定义 *1.1*。设 $U = \{u_0, \ldots u_{n-1}\}$ 是一个由 n不同用户组成的集合，$R = \{r_0, \ldots r_{m-1}\}$ 是一个由 m区域组成的集合，并设 $T = \{(i, j, \tau_{i,j})$，其中 $0 \leq i < n, 0 \leq j < m, \tau_{i,j} > 0\}$ 是一个由 N三元组组成的集合，其中每个三元组 $(i, j, \tau_{i,j})$ 表示用户 u_i在区域 r_j 中花费了 $\tau_{i,j}$的时间；我们假设 $\tau_{i,j} = 0$，如果不存在三元组 $(i, j, \cdot) \in T$。设 $k > 0$和 $1 \leq r \leq m$是给定的两个参数。*(k, r)-CLAV* 要求

构建一个数据结构，给定 U, R, T作为输入，对于任何满足 $|Q| = r$的查询 $Q \subseteq R$，返回在由 Q表示的区域中总计花费不少于 k时间的用户数量$n_{Q,\,k}$。即：

$$n_{Q,k} = \left| \left\{ u_i \in U : \sum_{j \in Q} \tau_{i,j} \geq k \right\} \right|.$$

不失一般性，我们假设 T 中不存在两个具有相同 i 和 j 值的三元组（$i,\ j,\ k_1$）、（$i,\ j,\ k_2$）；通过将这些三元组替换为（$i,\ j,\ k_1 + k_2$），即可轻松得到这一性质。我们使用以下记号：N 表示 T 中三元组的数量（即 $N = \|T\|$）；R_j 表示具有 τ_i 的用户 u_i 的集合 $R_j = \{i : \tau_{i,\,j} > 0$（即 $R_j = \{i : \tau_{i,\,j} > 0$）$\}$；[1] N_Q 表示查询区域 Q 中三元组的数量（即 $N_Q = \sum_{j \in Q} |R_j|$）；$n_Q$ 表示查询区域 Q 中不同用户的数量。最后，除非另有说明，空间需求均以内存字为单位；我们假设每个内存字包含 $w = \Omega$（$\log(n + m)$）比特。

显然，简单的精确查询解法 Q需要通过扫描 Q中 r集合里的用户，耗费 $O(N_Q)$ 查询时间。另一方面，如果预先计算所有可能的查询，虽然所需空间相对于 r呈指数级增长，但我们可以实现 $O(1)$查询时间。在第 2.2 节和第 2.1 节中，我们利用集合交方面的结果（例如 [GKLP17]）补全了这种权衡，并推导出紧确下界。与集合交不同，$(k,\ r)$-CLAV 不要求查询中的所有用户都属于所有区域；例如，对于查询 Q，只要某个用户 u_i 在两个区域 $r_1,\ r_2 \in Q$ 中的出现时间为 $k/2$，就足以将其计入任意 $r \geq 2$。

在第 3.1 节中，我们分析了近似解。首先，对于任意 $\varepsilon > 0$，我们给出一种线性空间解，其 $O\left(r^3 \varepsilon^{-2} \log(n)\right)$ 查询时间：它提供$n_{Q,\,k}$的无偏估计量，并且以高概率将最大加性误差控制在 $O(\varepsilon n_Q)$以内。随后，我们给出一种压缩数据结构，它使用基于 Flajolet-Martin 草图和 Count-Min 草图的草图来概括每个区域。该数据结构需要 $O\left(m\varepsilon^{-1} \log^2 n \log r\right)$ 比特空间，以及 $O\left(r\varepsilon^{-1} \log^2 n\right)$查询时间。空间和查询时间的改进，是以结果质量略有下降为代价的：估计量可能会纳入累计时间为 $\geq k(1 - 1/r)$的用户；并且，对于$\varepsilon = \Omega\left(n_{Q,\,k}/n_Q\right)$，以高概率，最大误差为 $\Theta\left(n_{Q,\,k} + \varepsilon n_Q\right)$。此外，如果已知$n_{Q,\,k}$的粗略估计，我们就能以$\varepsilon = \Theta\left(n_{Q,\,k}/n_Q\right)$实现 3-近似。

我们观察到，也可以根据数据流中检测重频项的结果推导出一个简单的草图，因为在 Q 个区域中出现 k 次的用户，其出现频率为$\tilde{}\ \tilde{}\ \tilde{}\phi T_Q$，其中 $\phi = k/T_Q$和 $T_Q = \sum_{(i,j,\tau_{i,j}) \in T, j \in Q} \tau_{i,j}$。然而，这样得到的草图所需空间取决于查询区域内所有用户的累计出现次数；相反，我们的解决方案与 T_Q 无关。

1.2 几何 CLAV

现在我们给出几何情形的定义。我们用 $\mathbb{R}^d$ 中的一个点表示每个区域，例如城市中的一个兴趣点（POI）；查询则用一个超矩形表示。

定义 *1.2*（几何 *CLAV*）。设 $U = \{u_0, \ldots u_{n-1}\}$ 是一个由 n 不同用户组成的集合，设 $R = \{x \in \mathbb{R}^d\}$ 是 $\mathbb{R}^d$ 中由 m点组成的集合，并设 $T = \{(i, x, \tau_{i,x}) with\ \tau_{i,x} > 0, 0 \leq i < n, x \in R\}$ 是一个由N三元组组成的集合，其中每个三元组 $(i, x, \tau_{i,x})$ 表示用户 u_i在点 x停留了 $\tau_{i,x}$；我们假设$\tau_{i,x} = 0$若不存在三元组 $(i, x, \cdot) \in T$。给定 U, R, T，几何 *CLAV* 要求构造一种数据结构，使其能够接收一个查询 Q（在 $\mathbb{R}^d$ 中由超矩形表示），并返回

[1]我们定义 $[n] = \{0, 1, \ldots n - 1\}$。

在查询 Q中停留时间不少于 k 的用户数量，即：

$$n_{Q,k} = \left| \left\{ u_i \in U : \sum_{x \in Q, \tau_{i,x} \in T} \tau_{i,x} \geq k \right\} \right|.$$

现在我们描述这一版本的结果。我们使用与（k， r）-CLAV 问题类似的记号；为简化记法，我们假设 d 为常数。

首先，我们在第4.1节给出一个无条件的空间下界 $\min\{n, (m/2d)^{2d}\}$ 比特，该下界与查询时间无关。当 $n > (m/2d)^{2d}$ 时，通过制表法使用查找表的解（定理 4.5）可达到这一界限，误差仅相差多对数因子。

随后，我们在第 4.2 节和第 4.3 节证明两个更强的条件下界。第一个下界是针对查询时间为多对数级的数据结构的 $\Omega(m^{2d-1-o(1)})$ 空间下界；其成立条件是，假设 r-集合不相交猜想在 [GKLP17] 中的快速查询版本成立。对于平面情形，我们证明了另一种条件下界：若 $n = m$ 且 $N = O(m^2)$，则任何数据结构都必须满足以下二者之一：查询时间为 $\Omega(m^{1-\varepsilon})$，或预处理时间为 $\Omega(m^{3-\varepsilon})$；此结论对任意 $\varepsilon > 0$均成立，并以布尔矩阵乘法猜想 [WW18, BW09] 为条件。

最后，在第 4.4 节中，我们提供了一种通过归约到带颜色的支配计数问题 [Gao23]来构建的数据结构。该数据结构需要 $O\left(\min\{Nm^{2d-2}, m^{2d}\}\right)$ 个内存字，并且查询时间为 $O(\log_w n_{Q,k})$。该数据结构几乎达到了第 4.2 节中得到的条件下界（相差 $n^{o(1)}$ 因子）。

1.3 先前工作

与我们问题相关的研究涉及多个不同领域：事实上，类似任务已经分别从精确和近似的角度被作为时空查询、重频项或不同元素计数进行研究。接下来，我们旨在概述我们的问题与文献中现有表述之间的差异。

大量工作致力于解决时空聚合查询：通常提出的问题包括时间戳 或区间 查询，要求报告、计数或计算在指定时间窗口内出现在特定（矩形）空间区域中的对象的聚合值。这实质上等同于关注在某个区域内出现超过 k 时间且具有 $k = 0$的所有用户。报告任务的示例包括 [TP01], [RBR12],[GNR+05]。计数变体也很常见：例如， [TKC+04] 研究了对满足特定时空查询的用户进行计数的问题，但其他变体还包括计算每个时间戳的访客平均数量 [TPZ02] ，或根据基数报告排名前 k 的时间区间 [XW23]。另一些工作则侧重于计算出现在低维矩形查询中的用户的基于特征的聚合，该查询同时确定一个时间区间和一个地理区域 [TBG09]。需要注意的是，这类查询与我们的问题有本质区别，因为它没有对用户的停留时间施加任何非平凡要求。因此，由于区间查询的典型输入数据由快照组成，用于记录用户在特定时间出现在特定位置的情况，所以所采用的数据结构通常难以针对一般 k情况适配我们的问题。

我们的任务涉及对出现在区域并集 Q中的不同用户进行计数，而这个并集只有到查询时间才会确定。因此，可以使用基于草图的近似来加快任务处理并降低内存需求。关于计数，过去几十年间，人们对多种基数估计草图进行了深入研究（[FM85, WVZT90, BYJK+02, HNH13, Tin14, Ert23]），实现了最优的空间—精度权衡（[KNW10]），并具备可合并性（即通过组合各个集合的草图来获得其并集的草图）。然而，据我们所知，这些研究都无法在报告目标元素基数估计的同时，筛除那些

不符合跨区域谓词的用户。

同样，现有用于重频项报告的草图似乎也不适用。Misra-Gries（[MG82, DLOM02, Zha13]）、Sticky Sampling（[MM02]）等流行的重频项算法，以及 [YZ09], 中的最优报告算法，都无法直接扩展到我们的场景，因为简单地重新适配这些算法，会导致草图的空间取决于 T 中停留时间的具体取值，正如第 1.1 节末尾所指出的。至于 CountSketch 及其变体（[AMS96, CCFC02, CM05]），它们只有通过显式查询各项的键，才能恢复最重频项的（近似）频率。

最后，现有关于监控阈值化数据的研究，要么专注于识别或监控所有超过某个阈值的键，却无法同时支持多个不同的查询（[LDW+20, KCR06]）；而我们的目标是设计更简单的数据结构，以支持计数和多个用户定义查询，从而相较于直接套用现有方案，同时改善查询运行时间和空间复杂度。

关于几何 CLAV，当 $k = 1$ 且当$\tau_{i,\ x}$为 0 或 1 时，该问题称为“广义交集搜索”（或彩色范围搜索） [JL93, GJS95] ，自 [GJRS18]以来便受到广泛关注。在与我们最相关的变体中，目标是构建一种数据结构，用于存储一个为每个点指定了颜色的输入点集；给定一个查询矩形，目标是统计查询区域内不同颜色的数量。该问题在一维情况下可以高效求解 [JL93, GH21, Gao23], ，但已知在二维及更高维度中较难求解 [KRSV08]。有趣的是，我们的条件下界高于 [KRSV08],中得出的下界，这表明我们的问题确实更加困难。

2 ($k,\ r$)-CLAV 问题的精确解

在这里，我们推导出一般（即非几何）($k,\ r$)-CLAV 问题（定义 1.1）几乎匹配的上界与条件下界。我们从下界开始。

2.1 (k, r)-CLAV 问题的下界

令 $S_0, \cdots\ S_{m-1}$ 为全集 U的子集，其总大小为 $N = \sum_{i=1}^{m} |S_i|$。 r-集合不相交问题的目标是对这组子集进行预处理，以便回答如下形式的查询：给定满足 $|Q| = r$的 $Q \subseteq [m]$，判断 $\cap_{q\in Q} S_q$ 是否为空，即对于 $q \in Q$，子集 S_q是否两两不相交。对于 $r = 2$，这就是简单的两两集合交问题，这是一个众所周知的问题，并有其自身的猜想：集合不相交猜想和强集合不相交猜想（[GLP19]中的猜想 1 和 2）。我们的困难性来源是强 r-集合不相交猜想（ [GKLP17], 中的猜想 9；该文也给出了匹配的上界）。

猜想 1 ([GKLP17]). 任何针对 r-集合不相交问题的数据结构，若使用 S空间~且查询时间为 T，则必须满足 $ST^r = \Omega(N^r)$。

我们想指出，尽管 [GKLP17], 中没有明确说明，但上述猜想只有在全集大小足够大时才可能成立，因为存在一种平凡解法（使用 N空间，仅存储输入），其运行时间为 $T = r|U|$；而如果 $|U| \ll N$，该猜想显然不成立。在第 4 节中，我们提出了上述猜想的一个版本：全集大小至少为 m的多对数级，且查询时间至多为 m的多对数级；我们称之为 Polylog-UniQuery-β-集合不相交猜想（第 4.2 节中的猜想 3）。

定理 2.1。 假设猜想 *1* 成立。那么，任何用于(k, r)-*CLAV* 问题的数据结构~，若使用 S空间且查询时间为 T，则必须满足 $ST^r = \Omega(N^r)$。

证明。我们将说明如何利用一个用于 (k,r)-CLAV 问题的数据结构，以相同的空间和时间开销解决r-集合不相交问题。考虑后一个问题的输入，即集合 $S_0,\cdots\ S_{m-1}\subseteq U$。通过重新标记，我们可以假设 $U=[n]$，其中 $n:=\left|\cup_{i=1}^{m}S_i\right|$。我们创建 m个区域 $\{r_0,\cdots\ r_{m-1}\}$ 和 n个用户。对于每个 $j\in[n]$ 以及每个满足 $j\in S_i$ 的 $i\in[m]$，我们创建一个三元组 $(j,i,1)$。这类三元组的数量恰为 $N=\sum_{i=1}^{m}|S_i|$。我们为这个 (k,r)-CLAV 问题实例预处理一个数据结构，使用 S空间。

对于 r-集合不相交问题的查询 $Q\subseteq[m]$，我们向 (k,r)-CLAV 问题的数据结构发出相同的查询 Q，并附上 $k=r$。由于任何用户在任何区域内停留的时间最多为一个单位，因此，询问在 r 个区域中累计停留了 r 时间的用户，恰好等同于求相应集合的交集大小。因此，如果 (k,r)-CLAV 问题的查询返回的计数非零，我们就报告 r-集合不相交问题的查询集不互不相交；否则，就报告它们互不相交。□

请注意，我们的归约实际上适用于 r-集合交集计数问题，其目标不仅是判断 r 个集合是否互不相交，还要返回它们的交集大小。我们在先前工作中尚未见过这个问题。r-集合交集计数问题是否比 r-集合不相交问题更难？对于 $r=2$，已有研究证明，r-集合交集计数问题具有与 r-集合不相交问题相同的紧确权衡（参见 [GLP19] 中的附录 C.4）。

不过，让我们考察他们的上界，看看它能否推广到 $r>2$。对于给定的查询时间目标 T，[GLP19] 将大小至少为 T的集合称为大集合，否则称为小集合。这意味着大集合至多有 N/T 个。对于每一对这样的集合，他们都会预先计算答案（即交集的元素个数），耗费空间 $S=O((N/T)^2)$。此外，对于每个 S_i，他们都会为 S_i中的元素存储一个成员哈希表，总共耗费 $O(N)$空间。给定一个查询 $|S_i\cap S_j|$，如果 S_i和 S_j都是大集合，他们就能在常数时间内回答。否则，假设 S_i是小集合，即至多包含 T个元素。对于 S_i中的每个元素，以 $O(1)$时间查询它是否属于 S_j，并返回 $|S_i\cap S_j|$，总查询时间为 $O(T)$。空间-查询权衡满足 $ST^2=O(N^2)$。

当 $r>2$时，同样的策略几乎可行，但现在如果至少有一个 r集合较小，我们可以在查询时选取最小的集合，其元素数量至多为 T；接下来，对于其中的每个元素，我们都必须查询所有其余的 $r-1$ 哈希表（针对 S_q，$q\neq i$）。因此，查询时间变为 $O(rT)$。对于常数 r，这与 r-SetDisjointness 的权衡相同；但对于 $r>2$，查询时间会多出一个因子 r。因此，我们提出以下猜想。

猜想 2. 任何用于解决 *r-SetIntersectionCount* 问题、使用 S 空间且$\tilde{}$查询时间为 T的数据结构，都必须满足 $ST^r=\Omega((rN)^r)$。

定理 2.1 的证明表明，猜想 2 适用于(k,r)-CLAV 问题。现在我们给出一个与猜想 2 的权衡相匹配的上界。

2.2 上界：k（k，r）-CLAV 问题

给定一个固定的 r，下面我们将展示如何构造数据结构 D_r，以便对任意 k 求解精确的（k，r）-CLAV 问题。该数据结构采用了将集合划分为大集合和小集合的方法，这一方法已在 r-SetIntersection 问题的精确数据结构 [GKLP17] 中提出。然而，我们的数据结构还需要额外的组件，以处理仅出现在查询子集中的用户。回顾一下，集合 T 是一个由 N 个三元组组成的集合，其中每个三元组（i，j，$\tau_{i,\ j}$）表示用户 u_i 在区域 r_j 中度过了 $\tau_{i,\ j}$ 时间，而 $R_j\subseteq[n]$ 表示曾在区域 r_j 中度过一段时间的用户集合。

令 $\lambda>0$ 为一个合适的参数。如果 $|R_j|>\lambda$，我们称区域 r_j大；否则称为小。注意，大区域的数量不可能超过 $\min\{m,\ N/\lambda\}$。分别令 L_λ 和 S_λ 为 R 中所有大区域和小区域构成的子集。

算法1 n_Q的精确算法，k

1：在 T_{Q_l} 上使用 k 进行二分搜索，找出在 Q_l 中至少停留 k 时间的不同用户数量 n_{Q_l}。
2 R_{Q_s} i $\tau_{i\,Q_s}$ $i \in \bigcup_{r_j \in Q_s} R_j$：计算所有元组 (，，) 的集合，其中：设置 τ_i、
3 $_{Q_s} = \sum_{r_j \in Q_s} \tau_i$、$_j$ 和 $n_{Q_s} = 0$。4：对于每个 $(i,\ \tau_i,\ _{Q_s}) \in R_{Q_s}$，执行以下操作：5：如果 U_{Q_l} 中不存在满足 τ_i、$_{Q_l} \geq k$ 的元组 $(i,\ \tau_i,\ _{Q_l})$，则执行 ▷ 使用二分搜索进行检查 U_{Q_l}6：递增 n_{Q_s}，条件为 τ_i、$_{Q_s} + \tau_i$、$_{Q_l} \geq k$ ▷，假设 τ_i、$_{Q_l} = 0$ 中不存在形如 $(i,\ *)$ 的元组 U_{Q_l}7：返回 $n_{Q_l} + n_{Q_s} (= n_Q,\ k)$

在构建时间，我们首先按键（$i,\ j$）对 T 排序，以便通过二分搜索轻松验证给定用户 u_i 是否在给定区域 r_j 内停留过；若是，则检索其在该处停留的时间 $\tau_{i,\ j}$。然后，对于每个子集 $Q \subseteq L_\lambda$，我们执行以下操作。我们计算由所有满足 $i \in \bigcup_{r_j \in Q} R_j$ 的元组（$i,\ \tau_{i,\ Q}$）组成的临时集合 V_Q，并设定 $\tau_{i,\ Q} = \sum_{r_j \in Q} \tau_{i,\ j}$。我们从 V_Q 创建两个数组 T_Q 和 U_Q，其中 T_Q 包含 V_Q 的所有元组，并按时间 $\tau_{i,\ Q}$ 非递减排序；而 U_Q 则按用户索引 i 非递减排序。我们的数据结构 D_r 由 T 的排序版本、所有 $r_j \in S_\lambda$ 对应的 R_j，以及所有 $Q \subseteq L_\lambda$ 对应的 T_Q 和 U_Q 数组组成。

现在，给定一组 $Q = Q_l \cup Q_s$ 查询区域，满足 $|Q| \leq r$、$Q_l \subseteq L_\lambda$ 和 $Q_s \subseteq S_\lambda$，我们按照算法1使用 D_r，找出在 Q 中停留时间至少为 k 的不同用户数量 $n_{Q,\ k}$，具体如算法1所述。我们首先查看 Q_l 的预计算解。然后，查看 Q_s 区域中的用户：对于每个用户 u，我们验证其累计时间是否至少为 k，并且只有当该用户尚未在 Q_l 的解中计数时才将其计入（即，对于 u，Q_l 中的累计时间已至少为 k）。

以下定理描述了内存与查询时间之间的权衡。注意，在特殊情况下，若 $r = O(1)$、$m = O(n)$、$N = O(n^2)$ 和 $\lambda = \sqrt{n}$，则可得到一种查询时间为 $\tilde{O}(\sqrt{n})$、内存为 $O(n^{3r/2+1})$ 的数据结构。

定理 2.2。上述数据结构需要 $O((\min\{m, N/\lambda\})^r rn)$ 构建时间和 $O(\min\{r\lambda, N\})$ 查询时间，并使用 $O((\min\{m, N/\lambda\})^r n + N)$ 内存字。

证明。满足 $Q \subseteq L_\lambda$ 且 $|Q| \leq r$ 的子集数量为 $O((\min\{m, N/\lambda\})^r)$，而我们需要 $O(rn)$ 时间，为每个这样的 Q 构建 T_Q 和 U_Q 数组。因此，D_r 可在 $O((\min\{m, N/\lambda\})^r rn)$ 时间内构建。我们需要 $O((\min\{m, N/\lambda\})^r n)$ 个内存字来存储所有 T_Q 和 U_Q 数组，并需要 $O(N)$ 个内存字来存储 T 的排序版本以及所有 $r_j \in S_\lambda$ 对应的 R_j 集合。因此，D_r 占用 $O((\min\{m, N/\lambda\})^r n + N)$ 个内存字。查询的开销主要由计算 R_{Q_s} 和 n_{Q_s} 所需的时间决定，这两者都可以在 $O(\min\{r\lambda, N\})$ 时间内完成。□

3 k（$r,\ r$）-CLAV 问题的近似解

这里，我们推导了一般（$k,\ r$）-CLAV 问题的近似解（定义 1.1）。

3.1 通过采样进行近似

设 Q 为一组 r 查询区域，并用 $T_Q \subset T$ 表示区域 Q 中出现的三元组集合。设 N_Q 和 n_Q 分别表示 Q 中的三元组数量和不同用户数量。为了精确计算 $n_{Q,\ k}$，需要读取查询区域中所有现有的 N_Q 三元组；不过，如果近似估算就足够，我们可以概述一种基于采样的查询过程，以便在合理的误差界内，按用户指定的概率快速估算 $n_{Q,\ k}$。首先，我们注意到，对于任意查询集 Q，可以将 $n_{Q,\ k}$ 改写为

$$n_{Q,k} = \sum_{(i,j,\tau_{i,j}) \in T_Q} \frac{\phi_i}{c_i} \tag{1}$$

算法 2 基于采样的估计$n_{Q,\ k}$

1: $s \leftarrow \frac{r^2}{2\varepsilon^2}\ln(2/\delta)$
2: $z \leftarrow 0$
3: **对** $l \leftarrow 1$ **到** s **执行**
4: $(i,\ j,\ \tau_{i,\ j})$ 从 T_Q中随机采样的随机三元组
5: $t \leftarrow \tau_{i,j},\quad c_i \leftarrow 1$
6: **对**每个 $r_q \in Q - \{r_j\}$ **执行**
7: **如果**$(i, q, \tau_{i,q}) \in T$**则** ▷ 通过二分搜索找到
8: $t \leftarrow t + \tau_{i,q},\ c_i \leftarrow c_i + 1$
9: **如果** $t \geq k$ **则** $\phi_i \leftarrow 1$
10: **否则** $\phi_i \leftarrow 0$
11: $z \leftarrow z + \frac{\phi_i}{c_i}$
12: $\hat{n}_{Q,\ k} \leftarrow z \cdot \frac{N_Q}{s}$
13: **返回** $\hat{n}_{Q,k}$

其中，c_i是Q中包含用户i的区域数量（即与i在T_Q中相关联的不同三元组数量）；而ϕ_i是一个指示变量，若$\sum_{r_j \in Q} \tau_{i,\ j} \geq k$则取值为 1，否则取值为 0。由于用户$i$在$Q$的$c_i$个区域中出现，因此需要缩放因子$c_i$。下文中，我们将$n_{Q,\ k}$的估计量记为$\hat{n}_{Q,\ k}$。

由于此过程仅旨在缩短查询时间，因此我们允许通过预处理来存储三元组，具体如下。我们将每个三元组$(i,\ j,\ \tau_{i,\ j}) \in T$放入数组$T$，并按$(i,\ j)$对$T$排序。这样便可通过二分搜索快速判断给定用户$u_i$是否在给定区域$r_j$停留过；若是，则可检索$\tau_{i,\ j}$。

经过上述预处理后，用于推导无偏估计量$\hat{n}_{Q,\ k}$的查询过程如算法 2 所示。具体而言，我们定义样本量s，根据定理 3.1，该样本量使我们能够很好地近似$n_{Q,\ k}$。我们从T_Q中均匀随机地采样s个三元组（有放回）。对于每个采样得到的三元组$(i,\ j,\ *)$，我们针对每个$r_q \in Q \setminus \{r_j\}$在$T$上执行二分搜索，以确定$u_i$是否曾在$r_q$中停留；若是，则检索$\tau_{i,\ q}$。若$\sum_{r_q \in Q} \tau_{i,\ q} \geq k$，则令$\phi_i = 1$；否则，令$\phi_i = 0$。将所有采样三元组对应的$\frac{\phi_i}{c_i}$值相加并进行适当缩放（算法 2 第 12 行），即可得到估计量$\hat{n}_{Q,\ k}$。

以下命题表明，在算法 2 第 12 行得到的$\hat{n}_{Q,\ k}$是无偏的，并说明了在该区域内采样s个条目所能达到的精度。

定理 3.1。设 $0 < \delta < 1$和 $0 < \varepsilon < 1$分别为置信参数和精度参数，Q为一组r查询区域。那么，对于任意 $s \geq \frac{r^2}{2\varepsilon^2}\ln(2/\delta)$，通过从$T_Q$中均匀随机抽取$s$个三元组（有放回），即可计算出$n_{Q,k}$的无偏估计量$\hat{n}_{Q,k}$，并保证$\hat{n}_{Q,k} \in [n_{Q,k} - \varepsilon n_Q, n_{Q,k} + \varepsilon n_Q]$以至少 $1 - \delta$ 的概率成立。

证明。设$\hat{n}_{Q,k}$按照算法 2 计算。我们需要证明它是$n_{Q,k}$的无偏估计量，并确定达到所述误差界所需的最小样本量。令$s > 0$表示样本量。然后，我们将查询 s 个三元组（从T_Q中出现的 N_Q 个三元组里抽取），以获得对$n_{Q,k}$的有界误差估计。对于 $1 \leq q \leq s$，设从 T_Q 抽取的第 q 个样本形式为$(i_q, *, *)$，也就是说，它属于用户u_{i_q}。对于每个 q，我们定义如下随机变量：

$$Y_q = \begin{cases} \frac{1}{c_{i_q}} & \text{if } \phi_{i_q} = 1; \\ 0 & \text{otherwise.} \end{cases}$$

令 $Z = \sum_{q=1}^{s} Y_q$，并注意这将是算法 2 第 12 行中存储的值。注意，$\mathbb{E}[Z] = \mathbb{E}[\sum_{q=1}^{s} Y_q] = s\mathbb{E}[Y_q]$ 且 $\mathbb{E}[Y_q] = 0 \cdot \Pr[Y_q = 0] + \sum_{h=1}^{r} \frac{1}{h} \Pr\left[Y_q = \frac{1}{h}\right] = \sum_{h=1}^{r} \frac{1}{h} \cdot \frac{v_h}{N_Q}$，其中 v_h 定义为 T_Q 中对应于访问过 Q 的恰好 h 个区域、且在这些区域总共停留至少 k 个时间单位的用户的三元组数量。因此，$\mathbb{E}[Y_q] = n_{Q,k}/N_Q$，这里利用了 $n_{Q,k} = \sum_{h=1}^{r} \frac{1}{h} \cdot v_h$ 这一事实。然后，我们可以构造一个针对 $n_{Q,k}$ 的无偏估计量：$\hat{n}_{Q,k} = z \cdot \frac{N_Q}{s}$。事实上，我们有 $\mathbb{E}[\hat{n}_{Q,k}] = \frac{N_Q}{s} \cdot s\mathbb{E}[Y_q] = n_{Q,k}$。

现在，我们可以利用霍夫丁不等式推导出如下误差上界：

$$\Pr[|\hat{n}_{Q,k} - n_{Q,k}| \geq t] = \Pr\left[\left|\sum_{q=1}^{s} \frac{Y_q N_Q}{s} - n_{Q,k}\right| \geq t\right] \leq 2\exp\left(-\frac{2t^2}{\sum_{q=1}^{s}(M_q - m_q)^2}\right) \leq 2\exp\left(-\frac{2t^2 s}{N_Q^2}\right).$$

其中，m_q 和 M_q 分别是变量 $Y_q \cdot \frac{N_Q}{s}$ 的下界和上界。最后一个不等式是根据以下事实得出的：$m_q = 0$ 和 $M_q = N_Q/s$。因此，使用任意 $s \geq \frac{N_Q^2}{2t^2}\ln(2/\delta)$，都可以以概率 $\geq 1 - \delta$ 得到 $n_{Q,k}$ 的良好估计。接着，我们可以固定 $t = \varepsilon n_Q$，从而选取任意 $s \geq \frac{N_Q^2}{2(\varepsilon n_Q)^2}\ln(2/\delta)$。现在，由于 $N_Q \leq r \cdot n_Q$，我们得出结论：对于任意 $s \geq \frac{r^2}{2\varepsilon^2}\ln(2/\delta)$，$\hat{n}_{Q,k} \in [n_{Q,k} - \varepsilon n_Q, n_{Q,k} + \varepsilon n_Q]$ 以至少 $1 - \delta$ 的概率成立。□

总的来说，采用这种方法时，查询时间相当于对 $s \geq \frac{r^2}{2\varepsilon^2}\ln(2/\delta)$ 个三元组搜索精确解：这里，查询一个三元组（i, j, $*$）意味着检查 u_i 是否存在于 Q 中除 r_j 之外的 $r - 1$ 个区域里；如果预处理阶段只生成 T，那么每个区域最多需要 $O(\log n_Q)$ 的二分搜索时间。因此，查询时间的上界为 $O\left(\frac{r^3}{\varepsilon^2}\log(1/\delta)\log n_Q\right)$，而该方法的空间复杂度为 $O(N)$，例如存储输入的开销。此外，值得注意的是，通过存储辅助数据结构，可以进一步降低期望查询时间，使单个区域内的用户查找能够以快于二分搜索的期望时间完成。最后，我们指出，这种简单的抽样方案适用于任意 r 和 k。

3.2 基于草图的近似

现在，我们为 k-CLAV 问题提出一种简洁数据结构，其中每个区域都由一个草图表示。在查询时，合并查询集 Q 中所有 r 区域的草图，即可提供 $n_{Q,k}$ 的估计值。令 n_Q 表示 Q 所有区域中的不同用户数量。不失一般性，我们假设 $k \geq r^2$：只需相应地缩放 T 和 k 中的时间即可。

为每个区域计算的草图是 Flajolet-Martin (FM) 草图[FM85, AMS96], 的一个修改版本：FM 的每个位位置都替换为一个 Count-Min 草图，该草图由一个包含 $O(1/\varepsilon)$ 个计数器的向量构成，每个计数器占用 $\Theta(\log r)$ 比特，参数为合适的 $\varepsilon > 0$。通过对向量条目逐对求和，可以轻松合并 Q 中各区域的草图。底层的 FM 草图可用于估算区域 Q 中的不同用户数量，而 Count-Min 草图则可用于从不同用户总数中区分出长时间聚合访问的不同用户数量。

现在，我们描述给定区域 r_j 的草图 S_j。令 $h\colon [n] \to [n]$，且 $g_i\colon [n] \to [1/\varepsilon]$。对于任意 $i \in [\log n + 1]$，它们分别是从二元独立哈希族 H 和 G 中随机选取的哈希函数。对于任意 $\ell \in [\log n + 1]$ 和区域 r_j，令 $V_{\ell,j}$ 为一个包含 $1/\varepsilon$ 个计数器的向量，每个计数器由 $\Theta(\log r)$ 比特组成，初始值均为 0。我们用 tail(x) 表示自然数 x 的二进制表示中尾随零的个数；换句话说，$2^{\text{tail}(x)}$ 是整除 x 的最大 2 的幂。每个区域的草图按算法3计算。区域 r_j 中的每个用户 u_i 都会使向量 $V_{\ell_i,j}$ 中的计数器 $g_{\ell_i}(i)$ 增加 $\lfloor \tau_{i,j} r^2/k \rfloor$，其中 $\ell_i = \text{tail}(h(i))$；也就是说，时间现在以 k/r^2 为单位计量（在某个区域中出现时间少于 k/r^2 的每个用户都

算法3 区域r_j的草图构造

1: **对于**每个用户 u_i 在 R_j **执行**
2: **如果**$\tau_{i,j} \geq k/r^2$ **则**
3: $c_{i,j} = \lfloor \tau_{i,j} r^2/k \rfloor$
4: $\ell_i =$ 尾部$(h(i))$
5: $V_{\ell_i,\ j}[g_{\ell_i}(i)] =$ 分钟 $\{r^2, V_{\ell_i,j}[g_{\ell_i}(i)] + c_{i,\ j}\}$

算法 4 查询过程，查询 Q

1: 令 $\tilde{V}_\ell$for $\ell \in [\log n + 1]$为一个包含 $O(1/\varepsilon)$个计数器、每个计数器占用 $\Theta(\log r)$比特的向量 ▷ 草图 S_Q
2: **对于**每个 $\ell \in [\log n + 1]$以及每个 $i \in [1/\varepsilon]$ **执行**
3: Set $\tilde{V}_\ell[i] = \min\{r^2, \sum_{j \in Q} V_{\ell,j}[i]\}$ ▷ S_Q =所有 $r_j \in Q$的草图之和 S_j
4: 令 $\tilde{\ell} \in [\log n + 1]$为最大的整数，使得向量 $\tilde{V}_{\tilde{\ell}}$至少有一个计数器不小于 $r^2 - r$
5: **返回**$2^{\tilde{\ell}}$ (= $\hat{n}_Q$)

会被忽略）。注意，计数器不会超过 r^2。我们观察到，累计时间至少为 k 的任何用户，都会使某个计数器至少增加 $r^2 - r$ 次（其中的 $-r$ 项是由舍入造成的）。

对于给定的查询集 Q，我们执行算法 4 中的操作。首先，通过合并所有 $r_j \in Q$对应的草图 S_j来构造草图S_Q。我们知道，对于每个 $\ell \in [\log n + 1]$，每个 S_j都包含一个向量$V_{\ell,j}$，其中有 $O(1/\varepsilon)$个计数器，每个计数器的长度为 $\Theta(\log r)$比特。 S_Q中对应的向量在 $\ell \in [\log n + 1]$时为 $\tilde{V}_\ell$。我们先为每个索引 $i \in [1/\varepsilon]$设置 $\tilde{V}_\ell[i] = \min\{r^2, \sum_{j \in Q} V_{\ell,j}[i]\}$。然后，搜索最大的整数 ℓ，使得向量 $V_{\tilde{\ell}}$包含一个值至少为 $r^2 - r$的计数器；当不存在累计时间低于 k的点时，该值应当与标准 FM 得到的结果相同。因此，我们的估计值为$\hat{n}_Q = 2^\ell$。我们通过标准的中位数技巧提高成功概率，即保留 $O(\log(1/\delta))$个相互独立的草图副本，并取查询结果的中位数。

定理 *3.2*。考虑一个包含 r 个区域的查询集 Q，令 $n_{Q,k}$ 和 $n^-_{Q,k}$ 分别表示在 r 个区域中累计时间至少为 k 和 k $(1 - 1/r)$ 的用户数量。令 $\hat{n}_Q$ 为算法 *4* 返回的值。那么，以至少 $1 - \delta$ 的概率，有$n_{Q,k}/3 \leq \hat{n}_Q \leq 3n^-_{Q,k}$（若 $\varepsilon = O(n_Q/n_{Q,k})$），否则 $n_{Q,k}/3 \leq \hat{n}_Q \leq 3n^-_{Q,k} + O(\varepsilon n_Q)$。该数据结构需要 $O(m\varepsilon^{-1} \log n \log r \log(1/\delta))$ 比特，查询时间为 $O(\varepsilon^{-1} \log n \log(1/\delta))$。

证明。我们称用户 u_i为 $(r^2 - r)$-长用户，当且仅当 $\sum_{r_j \in Q} \lfloor \tau_{i,j} r^2/k \rfloor \geq r^2 - r$。由于取整操作，所有累计时间至少为 k的用户都是 $(r^2 - r)$-长用户。我们观察到，对于每个 $(r^2 - r)$-长用户 u_i，都有 $\tilde{V}_{\ell_i}[g_{\ell_i}(i)] \geq r^2 - r$，其中 $\ell_i = \text{tail}(h(i))$。令 $\hat{n}$为 $(r^2 - r)$-长用户的数量。于是有 $n_{Q,k} \leq \hat{n} \leq n^-_Q$。假设我们从草图 S_j和 S_Q中移除所有由非 $(r^2 - r)$-长用户产生的更新。此时， S_Q中的所有非零计数器的值都至少为 $r^2 - r$，且索引 $\tilde{\ell}$等价于非零向量中的最大索引：这是使用相同哈希函数 h的标准 FM 草图所提取的值。因此，以常数概率 1/3 [AMS96],，我们有 $\hat{n}/3 \leq 2^{\tilde{\ell}+1/2} \leq 3\hat{n}$，从而 $n_{Q,k}/3 \leq 2^{\tilde{\ell}} \leq 3n^-_{Q,k}$。

现在我们考虑非 $(r^2 - r)$-长用户的贡献。我们注意到，只有存在一个向量 $V_{\ell'}$ 满足 $\ell' > \ell$，以及一个索引 i' 满足$\tilde{V}_{\ell'}[i'] \geq r^2 - r$时，它们才会影响 ℓ 的值。我们预计有 $n_Q/2^{\ell'}$ 个用户的尾部至少为 ℓ'；因此，根据马尔可夫不等式，以常数概率，至多有 $O(n_Q/2^{\ell'})$ 个用户更新 $\tilde{V}_{\ell'}$。此外，通过球与箱子论证，我们发现，当 n_Q 足够大时，以高概率， $\tilde{V}_\ell$中每个计数器的最大负载为 $O(n_Q\varepsilon/2^\ell)$。由于非 $(r^2 - r)$-长用户对任何计数器的增量都不超过任何

计数器 $r^2 - r$，因此，任何满足 $2^\ell > O\left(n_Q \varepsilon\right)$ 的向量 $\tilde{V}_\ell$ 中，都没有计数器能达到值 $r^2 - r$。因此，以常数概率，我们将得到大小为 $O\left(n_Q \varepsilon\right)$ 的加性误差。

每个草图由 $O(\log n)$ 个向量组成，每个向量包含 $O(1/\varepsilon)$ 个计数器，每个计数器占 $O(\log r)$ 比特，因此总计需要 $O\left(m\varepsilon^{-1} \log n \log r\right)$ 比特。由于两个计数器可以在常数时间内求和，查询时间为 $O\left(r\varepsilon^{-1} \log n\right)$。计入中位数技巧的开销后，即可得到所述界。□

我们猜想，借助 BYJK+02] 中用于计数不同项、可提供 ε 近似的草图，可以构造一个用于 (k, r)-CLAV 的草图，使其对任意 $\varepsilon > 0$ 都能提供 ϵn_Q 近似；这样一来，草图大小将与 ε^{-2} 成正比。

4 几何 CLAV

本节研究几何 CLAV 问题（定义 1.2）。为便于阐述，我们假设输入点位于 $\mathbb{R}^d$，且满足 $d = O(1)$。在第4.1节中，我们给出无条件空间下界。在第 4.2 节和第 4.3 节中，我们证明两个条件下界，分别依赖于 r-集合不相交猜想的快速查询版本和布尔矩阵乘法（BMM）猜想[2] [WW18、BW09], 。最后，在第 4.4 节中，我们给出两种数据结构：一种基于制表，另一种通过归约到带颜色的支配计数问题构造。

4.1 无条件空间下界

在这里，我们证明几何 CLAV 问题空间复杂度的无条件下界。首先定义点集（用户的位置）。用 $X_{i:s}$ 表示一个点，除第 i 个坐标设为 s外，其余坐标均设为零。令$m' = \frac{m}{2d}$。定义点集 $X_i = \{X_{i:s} | s \in \mathbb{Z},\ -m' \le s \le m',\ s \ne 0\}$。例如，对于 $d = 2$，$X_1 = \{(-m', 0), (-m'+1, 0) \ldots, (-1, 0), (1, 0) \ldots, (m', 0)\}$。每个 X_i包含 $2m'$ 个点，我们定义集合 $X = X_1 \cup \ldots \cup X_d$。$X$有 m 个点。

考虑一个由 $2d$ 非负整数值 $\sigma = (\ell_1, r_1, \ell_2, r_2, \ldots\ldots, \ell_d, r_d)$ 组成的序列。若 $1 \le \ell_i, r_i \le m'$ 对每个 $1 \le i \le d$ 都成立，则称 σ 有效。现在可以定义矩形 $Q(\sigma) = [-\ell_1, r_1] \times \ldots \times [-\ell_d, r_d]$。令 Q 为所有有效序列按此方式得到的矩形的集合。注意，$|\mathrm{Q}| = m'^{2d}$，且 Q 中每个矩形的边界上恰有 $2d$ 个点。

定理 4.1。无论查询时间如何，任何用于几何 *CLAV* 问题的数据结构都至少需要 $\min\{n, (m/2d)^{2d}\}$ 比特的存储空间。

证明。我们使用上文定义的集合 Q。令 $b = \min\{n, |\mathrm{Q}|\}$。主要思路是证明，给定一个由 b 比特组成的序列，我们可以利用数据结构对这些比特进行编码，使得每个比特都可以通过发出查询来恢复（解码）。这显然表明，该数据结构至少需要消耗 b 比特的存储空间，从而证明定理。令 $B_1, \ldots\ldots, B_b$ 为任意比特序列。

我们首先按照 Q 中矩形内部所包含的点数对矩形进行排序，若点数相同，则按字典序打破平局。具体来说，考虑对应于两个有效序列 σ_1和 σ_2 的两个矩形 $Q(\sigma_1)$ $Q(\sigma_2) \in \mathrm{Q}$。那么，如果 $Q(\sigma_1)$ 的内部点数严格少于 $Q(\sigma_2)$，则 $Q(\sigma_1)$ 排在 $Q(\sigma_2)$ 之前；但如果 $Q(\sigma_1)$ 和 $Q(\sigma_2)$ 的内部点数相同，则当 σ_1 按字典序小于 σ_2 时，$Q(\sigma_1)$ 排在 $Q(\sigma_2)$ 之前，反之亦然。

考虑这一排序中 Q 内的第 i 个矩形 Q_i。如果 $B_i = 0$，则我们什么也不做。否则，我们定义一个唯一的用户 u_Q，使其在区域 p_1、……、p_{2d}中的每个区域恰好停留一个单位时间，这些区域位于 Q_i 的边界上。由于 $b = \min\{n, |\mathrm{Q}|\}$，这一定义是明确的。这个用户集合以及 X 中的点作为输入提供给数据结构。

[2]BMM 猜想是另一个著名的猜想，并且独立于多重集合不相交性猜想。因此，它为几何 CLAV 问题的困难性提供了额外证据。

接下来需要证明，我们可以使用数据结构解码所有比特。现在考虑查询算法，并假设我们想要提取比特 B_i 的值。注意，集合 X 与 Q 是固定的（不依赖于比特序列）。因此，在解码时，可以计算出 X 和 Q，包括编码时使用的 Q 的排序。要提取 B_i，我们考虑 Q 中第 i 个矩形 Q_i。我们使用 Q_i 查询数据结构，并得到在 Q_i 中停留至少 $2d$ 时间的用户数量 z。我们先考虑 Q_i 恰好包含 $2d$ 个点（包括边界）的情况。如果 $z = 0$，那么我们可以断定 $B_i = 0$，因为没有添加任何用户在边界上的 $2d$ 点处停留 1 个单位时间。如果 $z = 1$，那么我们必然有 $B_i = 1$，因为我们不可能有另一个用户 u_Q 的 $2d$ 个实例包含在 Q_i 中。现在考虑 Q_i 内部还包含额外点的情况。在这种情况下，我们利用已提取所有比特 B_1、……、B_{i-1} 这一事实。考虑一个矩形 $Q' \in$ Q，使得 Q' 位于 Q_i 内。由于我们已经知道比特 B_1、……、B_{i-1} 的值，因此我们知道是否添加了用户 $u_{Q'}$。对于每个这样的矩形 Q'，我们利用先前解码的比特，统计已添加的用户数量。从 z 中减去该值后，我们就能判断是否添加了用户 u_{Q_i}，进而解码 B_i。□

4.2 基于快速查询集合不相交猜想的条件性下界

在定理 4.1 中，为了得到 $\Omega(m^{2d})$ 的最坏情况渐近下界（此后我们假设 $d = O(1)$），我们需要构造一个包含 $\Omega(m^{2d})$ 个用户的实例。虽然该下界排除了在 n 取较大值时获得高效解的可能性，但对于 n 取较小值时能否获得高效解，仍未有定论。这里我们证明，在维度为 2 或更高时，极其高效的解不太可能存在。我们的困难性依据是以下猜想。下面我们将把第 3 节中的 r-集合不相交改为 β-集合不相交，因为 r 也用于表示区域。

猜想 3（快速查询集合不相交猜想：）。设 $S_1, \cdots S_a$ 是来自全集 $\mathcal{U}$ 的 a 个集合。设 β 和 s 为固定整数常数。那么，存在一个常数 c（仅依赖于 s 和 β），使得以下结论成立：当 $|\mathcal{U}| \geq \log^c a$ 时，以 $O(\log^s a)$ 时间回答 β-集合不相交查询，在最坏情况下需要 $\Omega(a^{\beta - o_{(1)}})$ 空间。

上述猜想并未以上述精确形式明确提出。然而，在 [GLP19] 中已证明，对于 $\beta = 2$，上述猜想可由强集合不相交猜想推出。具体而言，已证明，要么需要 $\Omega(a^{2-o_{(1)}})$ 空间，要么查询时间为 $\Omega(|\mathrm{U}|^{0.5-o_{(1)}})$；由于在我们的表述中，可以直接选择 $c > 2s$，因此得到空间下界 [GLP19]。

我们证明，在猜想 3 的假设下，可以调整第4.1节中的构造以得到一个下界。考虑一个集合不相交问题的输入，并采用上述猜想中的记号：设 $S_1, \ldots, S_a$ 是全集 U 中的 a 个集合。令 $\bar{S}_i = \mathrm{U} \setminus S_i$ 为 S_i 的补集。我们将在第4.1节中使用相同的点集 X，但稍作修改。首先，令 $m = 4da$，从而 $m' = 2a$。定义集合 $X_i^+ = \{X_{i:s} \mid 1 \leq s \leq 2a\}$。对于 $i \neq d$，定义 $X_i^- = \{X_{i:s} \mid -2a \leq s \leq -1\}$。最后，定义 $X_d^- = \{X_{i:s} \mid -2da \leq s \leq -1\}$。我们将这些子集 X_i^+ 或 X_i^- 中的每一个称为一个块。因此，我们有 $2d$ 个块，并将 X_d^- 指定为最后一个块，其余块则从 1 到 $2d-1$ 任意排序。注意，最后一个块包含 $2da$ 个点，而所有其他块均包含 $2a$ 个点。令 X 为所有这些块的并集，因此 $|X| = O(a)$。给定一个集合 $S \subset$ U 和一个点 $q \in X$，在 q 处的编码 S 对应于以下操作：对于每个索引 $i \in S$，我们创建一个三元组，用户 u_i 在 q 处花费单位时间。根据上述定义，我们构造以下 d 维几何 CLAV 问题实例：对于每个 $q \in X_d^-$，在 q 处对 U 进行编码。换言之，在最后一个块的每个点处都对 U 进行编码。

接下来，对每个 $q \in X \setminus X_d^-$，我们执行以下操作。回想一下，根据构造，q 只有一个坐标非零。令 i 为 q 这个非零坐标的绝对值。如果 i 是奇数，那么我们

将 $S_{(i+1)/2}$ 编码在 q 处（例如，如果 $i=1$，我们就编码 S_1）。如果 i 是偶数，那么我们将 $S_{i/2}$ 编码在 q 处。直观地说，这种编码可以描述如下：对于除最后一个块以外的每个块，当我们从原点向外遍历块中的点时，i 依次递增为 1、2、3、4、...直到 $2a$，因此我们按此顺序编码集合 S_1，S_1，S_2，S_2，...，S_a。我们设定 $k=2da+1$，然后将此实例存储在用于几何 CLAV 查询的数据结构中。

现在考虑一个集合不相交问题查询，该查询由需要求交的集合的索引 i_1、.....、i_{2d-1} 给出。注意，每个集合 S_i 都被编码在 $2d-1$ 个点上，即除最后一个块外，每个块中各有一个点。我们按如下方式标记其中一些点。对于 S_{i_j}，我们考虑第 j 个块及其唯一的点 q_{i_j}，使得 S_{i_j} 被编码在 q_{i_j} 中。令 c_j 表示点 q_{i_j} 的非零坐标的大小。根据我们的构造，c_j 为奇数且 $i_j=\frac{c_j+1}{2}$。定义 $w=\sum_{j=1}^{2d-1}(i_j-1)$。注意，$0\le w\le(2d-1)(a-1)$。最后，我们标记最后一个块中非零坐标值为 $-(k-w-2d+1)$ 的点。注意，$1\le k-w-2d+1\le 2ad$，因此这样的点确实存在并且可以被标记。我们创建一个查询矩形 Q，使所有这些标记点都位于其边界上。注意，对于每个 j、$1\le j\le 2d-1$，当我们查看第 j 个块时，集合 S_1、S_1、.....、S_{i_j-1}、S_{i_j-1} 的编码位于 Q 内，因此每个用户在这些点上恰好花费 i_j-1 的时间。因此，当我们查看完全位于 Q 内且处于第 1 至 $2d-1$ 个块中的点时，每个用户总共恰好花费 w 的时间。在最后一个块中，Q 内恰好有 $k-w-2d+1$ 个点（包括边界上的点），因此每个用户恰好花费 $k-w-2d+1$ 的时间。将位于 Q 边界上且处于第 1 至 $2d-1$ 个块中的点称为重要点。因此，忽略重要点后，每个用户在 Q 中恰好花费 $k-2d+1$ 的时间。最后注意，如果 $S_{i_1}\cap S_{i_2}\cap\ldots S_{i_{2d-1}}$ 非空，则至少有一个用户在每个 $2d-1$ 重要点上花费一个时间单位，因此在 Q 中花费的累计时间为 k。另一方面，如果 $S_{i_1}\cap S_{i_2}\cap\ldots S_{i_{2d-1}}$ 为空，则根据构造，不会存在这样的用户。这意味着，如果几何 CLAV 查询返回非零答案，我们就可以得出这些集合的交集非空；否则，交集为空。现在利用快速查询集合不相交猜想（猜想 3），我们得到以下结果。

定理 4.2。 假设猜想 *3* 成立。那么，要在多对数查询时间内回答 d维几何 *CLAV* 查询，需要 $\Omega(m^{2d-1-o_{(1)}})$空间，假设 d为常数。

4.3 基于布尔矩阵乘法猜想的 $d=2$ 条件下界

这里，我们针对平面版本给出了一个不同的条件下界。这个结果比上一节给出的下界略弱，但它使用了一个不同的猜想，该猜想从另一个角度进一步表明，要为几何 CLAV 问题找到极其高效的解法并不容易。我们的下界受到 [KRSV08]下界的启发，但它使用了一个与布尔矩阵乘法（BMM） [WW18],相关的猜想，定义如下。

定义 4.3。 设 A和 B是两个元素属于$\{0,1\}$的 $m\times m$矩阵。*BMM* 问题是计算另一个 $m\times m$矩阵 C，使得 $C[i,j]=$或$_{k=1}^{m}(A[i,k]$与 $B[k,j])$成立。它本质上类似于普通矩阵乘法，只是将 + 和 × 运算替换为逻辑或 和与。

使用标准矩阵乘法算法，可以在 $O(m^3)$ 时间内解决 BMM 问题。我们将采用的主要猜想是，任何“组合”算法都无法显著改进这一解决方案。“组合”算法的确切定义并没有普遍

公认的标准 [BW09]，但一般来说，它指的是只利用组合结构，而不采用代数方法（如用于快速矩阵乘法的 Strassen 类方法）的算法 [3]。

猜想 4。 *BMM* 问题对于任意常数 $\varepsilon > 0$ *[WW18]*，都无法在 $O(m^{3-\varepsilon})$ 时间内通过组合算法求解。

现在我们将证明，针对我们问题的高效数据结构可以为 BMM 问题带来改进的组合算法。令 A 和 B 为 BMM 问题的输入。我们如下创建 $2m$ 个点。考虑连接点 (0,1) 和 (1,2) 的线段 ℓ_0，以及连接 (1, 0) 和 (2, 1) 的线段 ℓ_1。在每条线段上放置 m 个点。令 $p_1, \dots p_m$ 为线段 ℓ_0上的点，令 $q_1, \dots, q_m$为线段 ℓ_1上的点。这些点构成我们的点集 X。这些点的主要性质是，对于线段 ℓ_0 上的每个点 p_i 以及线段 ℓ_1上的每个点 q_j，都存在一个包含 p_i 和q_j且不包含 ℓ_0 或 ℓ_1中其他点的矩形。我们还创建了 m 个用户 $u_1, \dots, u_m$。

现在考虑矩阵 A。如果 $A[i,k] = 1$，那么 u_k在点 p_i 停留一个时间单位。如果 $B[k, j] = 1$，那么 u_k 在点 q_j 停留一个时间单位。这就构造出了一个至多包含 $2m^2$ 个三元组的实例。现在注意，对于任意两个点 p_i 和 q_j，我们都可以画出一个只包含这两个点的矩形。注意，如果任意用户 u_k 在这个矩形中停留 2 个时间单位，那么必有$A[i,k] = B[k, j] = 1$，进而可推出 $C[i, j] = 1$。如果不存在这样的用户，则 $C[i, j] = 0$。因此，我们可以用 m^2 次查询来求解 BMM 问题。由此得到以下下界。

定理 4.4。假设猜想 *4* 成立。那么，对于包含 m 个用户、 m 个点和 $O(m^2)$ 个三元组的输入，任何能在二维中解决几何 *CLAV* 问题的数据结构，要么具有 $\Omega(m^{1-\varepsilon})$ 的查询时间，要么具有 $\Omega(m^{3-\varepsilon})$ 的预处理时间，对任意 $\varepsilon > 0$均成立。

4.4 几何 CLAV 的上界

我们首先讨论无条件下界的紧致性，假设 n 大于 $(m/2d)^{2d}$。

4.4.1 制表法方案。 我们注意到，可以通过制表法使用查找表解决该问题。

定理 4.5. 几何 *CLAV* 可以用 $O(m^{2d})$ 的空间和 $O(\log m)$ 的查询时间解决 .

证明。我们采用经典秩空间归约：对于每个维度 i，将 R 中每个点的第 i个坐标替换为其秩。这样便将问题归约为R中的点的坐标均介于 1 和 m之间的情形。注意，在这种情况下，组合上不同的查询最多有 m^{2d} 个，因此，对于每个查询，我们只需将结果存储在一个大小为$O(m^{2d})$的数据结构中。给定查询矩形后，我们只需在O中用 m时间找到矩形各坐标的秩，然后查找答案。□

上述简单的数据结构表明，第4.1节中的无条件下界对于“较大”的 n是紧的。现在我们着眼于 n“较小”的情况。

4.4.2 使用彩色支配计数的高效数据结构。我们先考虑最简单的情形，即 $d = 1$，也就是说，几何 CLAV 输入中的所有点（位置）都位于实线上。和之前一样，我们只需对 R中的点排序，即可将其归约到秩空间。因此，可以假设 R中的点为 1 到 m。对于区间 ℓ，令$\tau_{u,\ell} = \sum_{x \in \ell, \tau_{u,x} \in T} \tau_{u,x}$表示用户 u在区间 ℓ内花费的总时间。对于 u，一个极小区间是指满足 $\tau_{u,[l,r]} \geq k$、 $\tau_{u,(l,r]} < k$且$\tau_{u,[l,r)} < k$的区间 $[l, r]$。

[3]引用 Bansal 和 Williams [BW09] 的话：“[这个术语主要只是用来区分那些不同于源自 Strassen [.....] 的代数方法的方案；我们认为，组合算法就是不调用环上的矩阵乘法预言机的算法。”

令 M_u 表示用户 u的所有极小区间构成的集合。下面的引理给出了极小区间数量的上界。

引理 4.6。 极小区间的总数至多为 N。

证明。对于每个用户 u， M_u 中的极小区间可以重叠，但 M_u 中的区间具有不同的起点；此外，在每个起点处，用户 u 花费的时间都严格大于 0。因此，极小区间的总数上界为 N。□

现在我们将一维输入“提升”到二维。给定 $\mathbb{R}^2$中的两个点 $p=(p_x,\ p_y)$和 $q=(q_x,\ q_y)$，如果 $p_x \geq q_x$ 且 $p_y \geq q_y$，我们称 p 支配 q。如果 $p \neq q$，这些不等式中必须有一个严格成立。现在，我们将区间 $\ell=[a,b]$映射为平面上的点$(-a,\ b)$。这样可以确保，如果区间 $\ell'=[a',b']$包含 ℓ，那么映射后的点$(-a',\ b')$就支配点$(-a,\ b)$。现在考虑以下问题，称为二维的彩色支配计数 。

定义 4.7（二维彩色支配计数）。考虑一个由 N 个点组成的集合 S，其中每个点 p 都被赋予一种颜色 $c(p)$，颜色取自 $[n]$。目标是将 S 存储在一个数据结构中，使得给定一个查询点 $q=(q_x,q_y)$，可以找出被 q 支配的颜色数量；更精确地说，我们需要返回

$$c(q)=|\{c(p)\mid p=(a,b)\in S, a\leq q_x, b\leq q_y\}.$$

已知该问题可用 $O(N)$ 空间和 $O(\log_w(n_{Q,k}))$ 查询时间解决 [Gao23]，其中 w 为字长，$n_{Q,k}$ 为返回的答案。因此，我们得到以下结果。

定理 4.8。几何 *CLAV* 问题在一维情况下可用 $O\left(\min\{N, m^2\}\right)$ 空间解决，查询时间为 $O(\log_w(n_{Q,k}))$，其中 w 为字长，$n_{Q,k}$ 为查询的答案。

4.4.3 高维。

定理 4.9. 几何 *CLAV* 可以在 d 维度下以 $O\left(\min\{Nm^{2d-2}, m^{2d}\}\right)$ 空间复杂度求解，查询时间为 $O(\log_w(n_{Q,k}))$ 其中 w 是字长，$n_{Q,k}$ 是查询的答案。

证明。令 Q 为一个查询矩形。注意，不失一般性，我们可以假设 Q 的每条边界都经过 X 中的一个点（否则，我们可以缩小 Q，而不改变 Q中的点数）。我们对维度 2 到 d进行制表：对于 2 到 d之间的每个维度，我们考虑 Q 两条边界的每个可能坐标。注意，每个维度至多有 m^2 种选择，总计有 m^{2d-2} 种选择。对于每种选择，我们都构建一个不同的数据结构。因此，给定 Q，我们查看 Q 在维度 2 到 d 上的坐标，并找到为 Q 构建的数据结构。这意味着我们可以直接将 Q 投影到第一维，并使用定理 4.8 的一维数据结构。□

查询的答案 $n_{Q,k}$至少需要存储 $(\log n_{Q,k})/w$ 个字，因此运行时间比最优时间多出 $w/\log w$ 倍。注意，如果用户和位置 ID 必须能放入一个字中，那么 w 至少为 $\log(n+m)$。$n_{Q,k}$ 至多为 n，因此最坏情况下的运行时间为 $O(\log n/\log\log(n+m))$。我们的条件下界表明，数据结构的开销几乎已达到最优，无法完全避免。我们的下界构造了一个包含 $N=O(m)$ 的实例，其中用户数量仍为多对数级，由此推出空间的条件下界为 $\Omega(m^{2d-1-o_{(1)}})$。因此，上述定理的空间消耗无法改进到 $O(Nm^{2d-2-\varepsilon})$，其中任意常数 $\varepsilon>0$ 均成立。

最后，尽管上述精确数据结构是针对固定的 k构建的，我们仍可为不同的 k 构建副本，从而为所有 k构建近似数据结构。也就是说，对于 $i=1$，2、.....、$\log K$中的每个 $k'=2^i$，我们都存储相应的数据结构，其中 K是最大允许值。这样便可返回在查询区域内停留至少 k/2时间的用户数量。

致谢

本文所述工作是在参加夏威夷大学马诺阿分校举办的 AlgoPARC 并行算法与数据结构研讨会期间完成的，部分经费由美国国家科学基金会第 2452276 号资助。该工作还获得以下机构的部分支持：意大利大学与研究部 PRIN 20174LF3T8 AHeAD 项目；意大利大学与研究部 PNRR CN00000013 高性能计算、大数据与量子计算国家中心；马斯登基金（MFP-UOA2226）；丹麦独立研究基金（DFF-8021-002498）。F. Silvestri 谨此感谢 Motion Analytica 的 B. Zamengo 就出行数据进行有益讨论，这些讨论启发了本项工作。

参考资料

[AMS96] Noga Alon、Yossi Matias 和 Mario Szegedy。近似频率矩的空间复杂度。载于第二十八届ACM 计算理论研讨会论文集，第20–29页，1996年。 [BW09] Nikhil Bansal 和 Ryan Williams。正则性引理与组合算法。载于2009年第50届IEEE计算机科学基础研讨会，第745–754页。IEEE，2009年。 [BYJK+02] Ziv Bar-Yossef、T. S. Jayram、Ravi Kumar、D. Sivakumar 和 Luca Trevisan。数据流中的不同元素计数。载于 José D. P. Rolim 和 Salil Vadhan（编），《计算机科学中的随机化与近似技术》，第1–10页，柏林、海德堡，2002年。Springer 柏林海德堡。[CCDT+22] Lorenzo Cavallo、Erika Cerasti、Mascia Di Torrice、Alessandra Righi、Maria Teresa Santoro、Tiziana Tuoto、Luca Valentino、Dario Di Sorte、Mauro Rossi、Andrea Zaramella、Dario Bertocchi、Glauco Mantegari 和 Bruno Zamengo。探索移动网络数据以开展旅游统计：意大利国家统计局与沃达丰意大利商业部之间的合作。《官方统计评论》，3：43–76，2022年。 [CCFC02] Moses Charikar、Kevin Chen 和 Martin Farach-Colton。寻找数据流中的频繁项。载于《自动机、语言与编程国际研讨会》，第693–703页。Springer，2002年。 [CM05] Graham Cormode 和 Shan Muthukrishnan。改进的数据流摘要：Count-Min 草图及其应用。《算法杂志》，55(1)：58–75，2005年。 [DLOM02] Erik D Demaine、Alejandro López-Ortiz 和 J Ian Munro。有限空间下互联网数据包流的频率估计。载于《欧洲算法研讨会》，第348–360页。Springer，2002年。 [Ert23] Otmar Ertl。Ultraloglog：用于近似不同元素计数的实用且更节省空间的 HyperLogLog 替代方案。arXiv预印本 arXiv:2308.16862，2023年。 [FM85] 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。 [Gao23] Younan Gao。用于二维支配关系彩色范围计数的自适应数据结构。载于《算法与数据结构研讨会》，第460–473页。Springer，2023年。 [GH21] Younan Gao 和 Meng He。空间高效的二维正交彩色范围计数。载于 Petra Mutzel、Rasmus Pagh 和 Grzegorz Herman（编），第二十九届欧洲算法研讨会（ESA 2021），《莱布尼茨国际信息学论文集》（LIPIcs）第204卷，第46:1–46:17页，德国达格施图尔，2021年。达格施图尔城堡—莱布尼茨信息学中心。 [GJRS18] Prosenjit Gupta、Ravi Janardan、Saladi Rahul 和 Michiel Smid。计算几何：广义（或彩色）交集搜索。载于 Sartaj Sahni 和 Dinesh Mehta（编），《数据结构与应用手册》。CRC 出版社，第2版，2018年。 [GJS95] Prosenjit Gupta、Ravi Janardan 和 Michiel Smid。广义交集搜索问题的进一步结果：计数、报告与动态化。《算法杂志》，19(2)：282–317，1995年9月。 [GKLP17] Isaac Goldstein、Tsvi Kopelowitz、Moshe Lewenstein 和 Ely Porat。空间与时间权衡的条件下界。载于《算法与数据结构：第15届国际研讨会，WADS 2017》，加拿大纽芬兰与拉布拉多省圣约翰斯，2017年7月31日至8月2日，论文集，第10389卷，第421页。Springer，2017年。 [GLP19] Isaac Goldstein、Moshe Lewenstein 和 Ely Porat。关于有界全集下集合不相交性与集合交的困难性。载于《第30届国际算法与计算研讨会（ISAAC 2019）》，第7–1页。达格施图尔城堡—莱布尼茨信息学中心，2019年。 [GNR+05] Gilberto A Gutiérrez、Gonzalo Navarro、Andrea Rodríguez、Alejandro González 和 José Orellana。基于快照与事件的时空访问方法。载于第13届ACM 地理信息系统国际研讨会论文集，第115–124页，2005年。 [HNH13] Stefan Heule、Marc Nunkesser 和 Alexander Hall。实践中的 HyperLogLog：先进基数估计算法的算法工程。载于第16届扩展数据库技术国际会议论文集，第683–692页，2013年。

[JL93] Ravi Janardan 和 Mario Lopez。广义交集搜索问题。《国际计算几何与应用杂志》，3(01)：39–69，1993年。[KCR06] Ram Keralapura、Graham Cormode 和 Jeyashankher Ramamirtham。阈值计数的通信高效分布式监测。载于2006年ACM SIGMOD数据管理国际会议论文集，第289–300页，2006年。[KNW10] Daniel M. Kane、Jelani Nelson 和 David P. Woodruff。不同元素问题的最优算法。载于第二十九届ACM SIGMOD-SIGACT-SIGART数据库系统原理研讨会论文集，PODS '10，第41–52页，美国纽约州纽约市，2010年。计算机协会。[KRSV08] Haim Kaplan、Natan Rubin、Micha Sharir 和 Elad Verbin。高效的彩色正交范围计数。《SIAM计算杂志》，38(3)：982–1011，2008年。[LDW+20] Meng Li、Haipeng Dai、Xiaoyu Wang、Rui Xia、Alex X. Liu 和 Guihai Chen。分布式数据流中的阈值监测。《IEEE/ACM网络汇刊》，28(3)：1033–1046，2020年6月。[LHV+14] Thomas Louail、Luis E Herrera、Maarten Vanhoof、Vanesa Frias-Martinez、Vincent D Blondel 和 José M Arias。理解城市地区的拥堵出行。《自然通讯》，5(1)：5347，2014年。[Mar19] Andy Marchant。地图背后：我们如何使地图保持最新。TomTom新闻中心，2019年10月。[MBGP13] Raul Montoliu、Jan Blom 和 Daniel Gatica-Perez。从智能手机数据中发现日常生活中的兴趣地点。《多媒体工具与应用》，62(1)：179–207，2013年。[MG82] Jayadev Misra 和 David Gries。寻找重复元素。《计算机编程科学》，2 152，1982年2 143( )：–。[MM02] Gurmeet Singh Manku 和 Rajeev Motwani。数据流上的近似频数计数。载于第28届超大型数据库国际会议论文集，VLDB '02，第346–357页。VLDB基金会，2002年。[PT14] Nikos Pelekis 和 Yannis Theodoridis。《移动性数据管理与探索》。Springer出版公司，2014年。[RBR12] Miguel Romero、Nieves Brisaboa 和 M Andrea Rodríguez。smo-index：用于时间戳与区间查询的简洁移动对象结构。载于第20届地理信息系统进展国际会议论文集，第498–501页，2012年。[SY18] Yoshifumi Seki 和 Mitsuo Yoshida。新闻应用中按类别分析用户停留时间。载于2018年IEEE/WIC/ACM国际网络智能会议（WI），第732–735页，2018年。[TBG09] Igor Timko、Michael H Böhlen 和 Johann Gamper。道路网络中的序列时空聚合。载于第12届扩展数据库技术国际会议：数据库技术进展论文集，第48–59页，2009年。[Tin14] Daniel Ting。不同元素的流式近似计数：击败最优批处理方法。载于第20届ACM SIGKDD知识发现与数据挖掘国际会议论文集，KDD '14，第442–451页，美国纽约州纽约市，2014年。计算机协会。[TKC+04] Yufei Tao、George Kollios、Jeffrey Considine、Feifei Li 和 Dimitris Papadias。使用草图进行时空聚合。载于第20届数据工程国际会议论文集，第214–225页。IEEE，2004年。[TP01] Yufei Tao 和 Dimitris Papadias。Mv3r-tree：用于时间戳与区间查询的时空访问方法。载于 VLDB，第1卷，第431–440页，2001年。[TPZ02] Yufei Tao、Dimitris Papadias 和 Jun Zhang。平面点的聚合处理。载于《扩展数据库技术国际会议》，第682–700页。Springer，2002年。[VF23] Vodafone Business 和 FS Italiane。FS Italiane 与 Vodafone Business：用于分析意大利移动性的网络数据，2023年。[WVZT90] Kyu-Young Whang、Brad T Vander-Zanden 和 Howard M Taylor。面向数据库应用的线性时间概率计数算法。《ACM数据库系统汇刊》（TODS），15(2)：208–229，1990年。[WW18] Virginia Vassilevska Williams 和 R. Ryan Williams。路径、矩阵与三角形问题之间的次立方等价关系。《ACM杂志》，65(5)，2018年8月。[XW23] Jianqiu Xu 和 Raymond Chi-Wing Wong。高效回答 top-k窗口聚合查询：计算分层结构上的覆盖数序列。载于2023年第39届IEEE数据工程国际会议（ICDE），第1300–1312页，2023年。[YZ09] Ke Yi 和 Qin Zhang。分布式重频项与分位数的最优跟踪。载于第二十八届ACM SIGMOD-SIGACT-SIGART数据库系统原理研讨会论文集，PODS '09，第167–174页，美国纽约州纽约市，2009年。计算机协会。[Zha13] Yu Zhang。分布式频繁项挖掘。《应用力学与材料》，263：1202–1207，2013年。