arXiv:2603.08615v1 [cs.DS] 9 Mar 2026

欧几里得聚类的分布式算法

Vincent Cohen-Addad* LiudengWang† David P.Woodruff‡ Samson Zhou§2026年3月10日

摘要

我们研究在分布式环境中构造用于欧几里得(k,z)聚类的$(1+\varepsilon)$-核心集这一问题，其中n个数据点分布在s个站点上。我们重点关注两种重要的通信模型：协调器模型和黑板模型。在协调器模型中，我们设计了一种协议，可构造$(1+\varepsilon)$-强核心集，总通信复杂度为$\tilde{O}\left(sk+\frac{dk}{\varepsilon^4}\min(,\varepsilon^{2+z})+dk\log(n\Delta)\right)$比特。与先前的工作（Chen 等人，NeurIPS 2016）相比，该协议无需在所有服务器之间以明文形式传输显式点坐标。在黑板模型中，我们进一步将通信复杂度降至$\tilde{O}\left(s\log(n\Delta)+dk\log(n\Delta)+\frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\right)$比特，在优于以往方法的同时，将保证从常数因子近似提升为$(1+\varepsilon)$-近似。我们的技术结合了实现常数因子近似的新策略、高效的核心集构造和紧凑编码方案，从而得到最优协议；这些协议的通信成本与已知最优离线核心集构造以及现有下界（Chen 等人，NeurIPS 2016；Huang 等人，STOC 2024）相匹配，误差仅相差多对数因子。

1 引言

聚类是通过将具有相似属性的点分组、并将属性不同的点分开来划分数据集的过程。对聚类的研究可追溯至20世纪50年代 [S+56, Mac67], 及其众多变体在生物信息学、组合优化、计算几何、计算机图形学、数据科学和机器学习等领域中均有应用。在欧几里得$(k,\ z)$聚类问题中，输入包括集合X，其中包含n个点$x_1,\ .....,\ x_n\in\mathbb{R}^d$，以及簇数$k>0$和指数$z>0$。目标是找到集合$\mathcal{C}$，至多包含$k$个中心，并使聚类代价最小化：

$$\min_{\mathcal{C}\subset\mathbb{R}^d,|\mathcal{C}|\leq k}\mathrm{cost}(X,\mathcal{C}):=\min_{\mathcal{C}\subset\mathbb{R}^d,|\mathcal{C}|\leq k}\sum_{i=1}^{n}\min_{c\in\mathcal{C}}\|x_i-c\|_2^z.$$

当$z=1$和$z=2$时，(k,z)-聚类问题分别可归约为经典的k-中位数和k-均值聚类问题。这些模型仍是实践中应用最广泛的表述形式，许多研究致力于其算法基础 [CCFM97、CGTS99,

*Google Research。 cohenaddad@google.com
†德克萨斯农工大学。 eureka@tamu.edu
‡卡内基梅隆大学。 dwoodruf@andrew.cmu.edu§德克萨斯农工大学。 samsonzhou@gmail.com

CG99, Mey01, COP03, FMS07, FMSW10, FL11, FSS13, CKM16, CS17, CLN+20, BFL+21, CSS21, BCJ+22, CLSS22,EFS+22,TWZ+22,BLST23, CSS23, WZZ23, BDJ+25, CJY+25, CLS25, KLR+25].

现代数据集规模的大幅增长，使研究重点转向能够跨多台机器处理数据、且无需集中访问完整数据集的大规模计算模型。分布式计算模型已成为处理此类大规模数据的常用框架。在分布式环境中，数据集 X 中的点 x_1、……、x_n 被分布到 s 台不同的机器上。给定输入精度参数 $\varepsilon > 0$，目标是让这些机器协同找出 X 的一个聚类 $\mathcal{C}$，其代价与 X 的最优聚类代价之间的比值不超过 $(1+\varepsilon)$，同时尽可能减少机器间的总通信量。与其他模型一样，中心集合 $\mathcal{C}$ 可隐式定义聚类，因为每个点都被分配给其最近中心。传输整个数据集，或为每个点显式传输簇标签，都会导致与 $n = |X|$ 成线性关系的通信量；因此，设计只交换简洁摘要（例如小型代表点集合或核心集）的协议更为可行。还需注意，由于有限精度，假设输入点位于网格 $\{1、\ldots\ldots、\Delta\}^d$ 上，每个坐标只需少量比特即可高效传输。

高效分布式聚类的一种标准策略是让每台机器对其本地点构造一个较小的加权子集，即核心集，并使其能够保留任意 k 个中心选择下的聚类代价。随后，可以在协调器处合并这些本地核心集，或通过分层聚合来近似整个数据集上的聚类目标。自然，核心集越小，通信成本越低，集中式处理也越快。在离线环境中，完整数据集 X 可在一台没有资源限制的机器上使用，已知的核心集构造可以选取 $\tilde{O}\left(\min\left(\frac{1}{\varepsilon^2}\cdot k^{2-\frac{z}{z+2}}, \frac{1}{\min(\varepsilon^4,\varepsilon^{2+z})}\cdot k\right)\right)$ 个加权点，见 X [CSS21, CLSS22, CLS+22, HLW24][1][2]。按照惯例，~~我们假设 $\Delta = \text{poly}(n)$~~，因此每个坐标都能用 $O(\log(nd\Delta))$ 比特存储，从而实现高效通信和存储。因此，离线核心集构造的大小与 n无关；鉴于现代数据集通常包含数亿个点，这一特性至关重要。

1.1 我们的贡献

我们研究分布式环境中欧氏（k, z）聚类的（$1+\varepsilon$）-核心集构造，重点关注协调器与黑板通信模型。

协调器模型。在协调器（消息传递）模型中，s 各站点（即服务器）仅通过私有信道与协调器通信，并使用私有随机性。不失一般性，假设协议是顺序执行、基于轮次且自定界的；也就是说，在每一轮中，协调器与若干参与方通信，并等待他们回复后再开始下一轮，且所有参与方都知道每条消息何时发送完毕。给定 $\varepsilon > 0$，目标是设计一个协议 Π，使协调器输出一个$(1+\varepsilon)$-核心集，用于欧几里得$(k,\ z)$-聚类，并使通信成本（即最坏情况下交换的比特总数）最小化。我们为该模型中的聚类实现了通信成本最优的协议，达到与已知下界相匹配的结果，下界来自 [CSWZ16、HLW24]:

[1]我们用 ~~$\tilde{\mathcal{O}}\left(f\left(n,d,k,\Delta,\frac{1}{\varepsilon}\right)\right)$ 表示 $\mathcal{O}\left(f\left(n,d,k,\Delta,\frac{1}{\varepsilon}\right)\right)\cdot$多重对数 $\left(f\left(n,d,k,\Delta,\frac{1}{\varepsilon}\right)\right)$~~。[2]该界可进一步改进为 $\tilde{\mathcal{O}}\left(\min\left(\frac{k^{4/3}}{\varepsilon^2}, \frac{k}{\varepsilon^3}\right)\right)$（针对 k-中位数）。

定理 1.1（协调器模型中的通信最优聚类，非正式表述）。给定精度参数 $\varepsilon \in (0, 1)$，存在一个协议，可对分布在 s 个站点上的 n 个点进行处理，生成一个用于 (k, z)-聚类的 $(1+\varepsilon)$-强核心集，其在协调器模型中的通信总比特数为 $\tilde{O}\left(sk + \frac{dk}{\varepsilon^4}\min(\varepsilon^{2+z}) + dk\log(n\Delta)\right)$。

我们指出，我们的结果也给出了 k-中位数问题的最优界，与 [HLW24、BCP+24]相匹配；为清晰起见，本文其余部分省略这一特例。相比之下，将 [BEL13], 的协议与较新的核心集构造（例如 [CSS21、CLSS22、BCP+24]）结合，可使用 $O\left(\frac{skd}{\varepsilon^4}\min(,\varepsilon^{2+z})\log(n\Delta)\right)$ 通信比特构造一个 $(1+\varepsilon)$-强核心集。乍看之下，人们可能会认为，我们的界可以通过直接推广这些方法得到；然而，正如我们在附录 3 中所解释的，事实并非如此。例如，我们的结果表明，无需在站点之间以“明文”传输任何点——通信复杂度中不含 $O(sd\log(n\Delta))$ 这一项。这令人惊讶，因为人们或许会认为，常数因子近似的坐标需要广播给所有用户。此外，我们的界中的 $\frac{1}{\varepsilon}$ 因子既不与站点数量 s 相乘，也不与传输单个坐标的 $\log(n\Delta)$ 成本相乘。最后，我们的结果可推广到任意连通通信拓扑，即使某些站点对之间不允许交互。形式化陈述见附录 C 中的定理 C.7。

协调器模型	通信成本（比特）
合并归约，搭配 [BCP+24] [BEL13]，$z=2$	$\mathcal{O}\left(\frac{skd}{\varepsilon^4}$ 最小值$(,\varepsilon^{2+z})$ 对数$(n\Delta)\right)$
[BEL13] 与 [BCP+24]	$\mathcal{O}\left(\frac{d^2k}{\varepsilon^4}\log(n\Delta) + sdk\log(sk)\log(n\Delta)\right)$
定理 1.1 (本工作)	$\mathcal{O}\left(\frac{dk}{\varepsilon^4}\min(,\varepsilon^{2+z})\log(n\Delta) + sdk\log(sk)\log(n\Delta)\right)$
	$\tilde{O}$ 欧 $\left(sk + \frac{dk}{\varepsilon^4}\min(,\varepsilon^{2+z}) + dk\log(n\Delta)\right)$
黑板模型	**通信成本（比特）**
[CSWZ16]	$\tilde{\mathcal{O}}\left((s+dk)\log^2(n\Delta)\right)$
定理 1.2（本文工作）	$\tilde{\mathcal{O}}\left(s\log(n\Delta) + dk\log(n\Delta) + \frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\right)$

图 1：分布式环境中 (k, z)-聚类算法表。我们指出，[CSWZ16]仅实现常数因子近似，而我们实现了 $(1+\varepsilon)$-近似。

黑板模型。接下来，我们考虑通信的黑板模型，其中每个 s 站点都可以广播消息，供所有其他站点查看。形式上，在黑板模型中，通信通过所有服务器均可见的共享公共黑板进行。每台服务器也都可以使用私有随机源。与协调器模型不同，这里没有指定的协调器来转发消息；相反，任何服务器都可以直接向黑板写入消息。所有服务器都能随时立即查看黑板上的全部内容。与之前一样，我们不失一般性地假设协议是顺序执行且基于轮次的，也就是说，每一轮中一台或多台服务器向黑板写入消息，随后所有服务器都可以读取新发布的消息，然后才开始下一轮。因此，消息必须采用自定界格式，以便所有服务器都能正确判断一条消息何时已完整发布。

给定精度参数 ε，目标是执行协议 Π，使得 $(1+\varepsilon)$- 欧几里得 (k,z)-聚类的核心集被显式写入黑板。协议 Π 的通信成本定义为协议执行期间写入黑板的比特总数，并按最坏情况衡量。我们注意到，与协调器模型相比，黑板模型有时能降低通信代价，因为消息只需写入一次，所有服务器便可同时访问。事实上，我们为黑板模型实现了一种分布式协议，其通信量远低于我们的协调器模型协议，并且与已有下界相匹配 [CSWZ16、HLW24]。

定理 1.2 （黑板模型中的通信最优聚类，非正式表述）。 存在一种协议，对于分布在 s 个站点上的 n 个点，生成一个用于(k,z) 聚类的$(1+\varepsilon)$ 强核心集，且在黑板模型中的通信总比特数为 $\tilde{O}\left(s\log(n\Delta)+dk\log(n\Delta)+\frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\right)$。

相比之下，最先进的协议通过$\tilde{O}\left((s+dk)\log^2(n\Delta)\right)$ 总通信量实现常数因子近似 [CSWZ16]。因此，与 [CSWZ16], 的工作相比，我们不仅在黑板设置中实现了 $(1+\varepsilon)$-核心集构造，还消除了多余的 $\log(n\Delta)$ 因子。从概念上说，定理 1.2 的要点与定理 1.1 类似：为了实现常数因子的通信量，需要传输一定数量的精确坐标（在协调器设置中，或许只需传输给少数几个站点），但要提升到 $(1+\varepsilon)$-近似，无需额外开销。最后，我们指出，对于黑板模型和协调器模型，在敏感度采样 [BCP+24] 的帮助下，对于重要的 k-中位数和 k-中位数情形，我们的核心集大小为 $\tilde{O}\left(\frac{k}{\varepsilon^4}\right)$；一般而言，我们的结果不仅达到已知下界 [HLW24]，而且在某些 n 和 k [的取值范围内还能进一步优化 22]CSS21, CLSS。

技术与算法创新。 我们指出，我们的工作提出了多项可能具有独立研究价值的算法与技术创新。例如，我们在附录 3中介绍了若干现有方法，并解释了它们为何行不通。

传统的敏感度采样会按照数据点的重要性，以相应概率选取完整的数据点。尽管这种方法在集中式环境中有效，但难以高效应用于分布式环境：传输完整数据点或高维中心会产生高昂的通信代价，而朴素的自适应采样则需要所有站点频繁更新。

为应对这些挑战，我们提出了几项新颖技术。首先，在黑板模型中，我们证明自适应采样能够适应过时信息，并由此提出一种“惰性”自适应采样协议：只有当站点的局部权重估计发生显著变化时，才更新黑板。即使从这一过时分布中进行采样，仍能保证常数因子近似，同时减少每轮传输的站点数量和总轮次。此外，额外的 L_1 采样子程序无需查询所有站点，就能进一步检测全局权重是否发生显著变化，从而提升通信效率。

在协调器模型中，我们提出了一种基于逐坐标采样的通信高效子程序。协调器和站点无需传输完整的高维中心，而是在站点排序后的坐标上执行分布式二分搜索，以找到最接近的匹配项。只需传输一个较小的偏移量，从而使通信量不再依赖于维度 d。

总体而言，我们将这些技术与逐坐标敏感度采样相结合：将每个点按其坐标分解，并根据各维度的重要性进行采样。这样

协调器便可以向各站点发送精简的摘要，服务器仅在必要时才请求更多信息。然而，重构后的样本可能并不对应数据集中的任何实际点，因此需要仔细分析，以证明整体聚类成本不会受到显著扭曲。我们认为，这些技术也有望惠及其他分布式环境，例如回归和低秩近似。这项工作属于一类更广泛的研究，旨在通过数据量化提升内存和通信效率；对于大型语言模型及其他模型而言，内存和通信效率往往是瓶颈。

2 预备知识

对于正整数 n，我们将 $[n]$ 记为集合 $\{1,\ldots,n\}$。我们用 poly(n) 表示关于 n 的任意多项式函数，并用 polylog(n) 表示关于 $\log n$ 的多项式函数。若某事件发生的概率至少为 $1-1/\text{poly}(n)$，则称该事件以高概率发生。

本文始终关注欧几里得 (k,z)-聚类。给定向量 $x,y\in\mathbb{R}^d$，我们用 dist(x,y) 表示它们之间的欧几里得距离，定义为 $\|x-y\|_2$，其中 $\|x-y\|_2^2=\sum_{i=1}^d(x_i-y_i)^2$。对于点 x 和集合 $S\subset\mathbb{R}^d$，我们将记号 dist(x,S) 扩展为：$=\min_{s\in S}\text{dist}(x,s)$。我们还用$\|x\|_z$表示 x 的 L_z范数，其定义为 $\|x\|_z^z=\sum_{i=1}^d x_i^z$。给定固定的指数 $z\geq 1$ 和有限集合 $X,C\subset\mathbb{R}^d$，其中 $X=\{x_1,\ldots,x_n\}$，我们将聚类代价 cost(X,C) 定义为

$$\text{cost}(X,C):=\sum_{i=1}^n\text{dist}(x_i,C)^z.$$

现在我们回顾三角不等式的广义形式。

事实 2.1（广义三角不等式）。对于任意 $z\geq 1$ 以及任意点 x、w、$y\in\mathbb{R}^d$，均有

$$dist(x,y)^z\leq 2^{z-1}(dist(x,w)^z+dist(w,y)^z).$$

事实 2.2（[SW18]中的命题 5）。设 $z\geq 1$，x，$y\geq 0$，且 $\varepsilon\in(0,1]$。则

$$(x+y)^z\leq(1+\varepsilon)\cdot x^z+\left(1+\frac{2z}{\varepsilon}\right)^z\cdot y^z.$$

接下来，我们定义 (k,z)-聚类的强核心集概念。

定义 2.3（核心集）。设 $\varepsilon>0$ 为一个近似参数，$X=\{x_1,\ldots,x_n\}\subset\mathbb{R}^d$为一个点集。对于（$k$, z）-聚类，核心集 是指一个加权集合（S,w），使得对于每个由 k 个中心构成的集合 $C\subset\mathbb{R}^d$，

$$(1-\varepsilon)\sum_{t=1}^n dist(x_t,C)^z\leq\sum_{q\in S}w(q)dist(q,C)^z\leq(1+\varepsilon)\sum_{t=1}^n dist(x_t,C)^z.$$

我们将使用以下已知的 (k,z)-聚类核心集构造：

定理 2.4。 *[CLSS22、HLW24、CLS+22]* 对于任意 $\varepsilon\in(0,1)$，存在一种针对欧几里得(k,z)聚类的核心集构造，可从输入数据集中抽取 $\tilde{O}\left(\min\left(\frac{1}{\varepsilon^2}k^{2-\frac{z}{z+2}},\frac{k}{\varepsilon^4\min(1,\varepsilon^{2+z})}\right)\right)$ 个加权点。

我们还回顾经典的 Johnson-Lindenstrauss 引理，它能够在近似保持两两距离的同时实现降维：

定理2.5 (Johnson-Lindenstrauss 引理)。 *[JL84]* 设 $\varepsilon \in (0,\ 1/2)$ 且 $m = \mathcal{O}\left(\frac{1}{\varepsilon^2}\log n\right)$。给定一个由 $X \subset \mathbb{R}^d$ 个 n 点组成的集合，存在一族随机线性映射 $\Pi:\ \mathbb{R}^d \to \mathbb{R}^m$ 使得对 $\pi \sim \Pi$ 的选择而言，以高概率对所有 $x,\ y \in X$，

$$(1-\varepsilon)\|x-y\|_2 \le \|\pi x - \pi y\|_2 \le (1+\varepsilon)\|x-y\|_2.$$

我们指出，对于 $(k,\ z)$-聚类，存在更高效的降维技术 [MMR19，ISZ21]；不过就我们的保证而言，Johnson-Lindenstrauss 已经足够。

最后，我们回顾一下标准的集中界 Hoeffding不等式：

定理2.6 (Hoeffding不等式)。 设 X_1、…、X_n 是相互独立的随机变量，对每个 i 均满足 $a_i \le X_i \le b_i$。设 $S_n = \sum_{i=1}^n X_i$。则对任意 $t > 0$，

$$\mathbb{P}\left(|S_n - \mathbb{E}[S_n]| > t\right) \le 2\exp\left(-\frac{t^2}{\sum_{i=1}^n (b_i-a_i)^2}\right).$$

接下来，我们回顾如下敏感度采样范式的表述，见 算法 1。

算法 1 敏感度采样

输入： 数据集 $X = \{x_1, \ldots, x_n\} \subset [\Delta]^d$ 和整数 k

输出： 加权集合 $A = \{(a_i, w_{a_i})\}$

1: 计算一个 $\mathcal{O}(1)$-近似 $S = \{s_1, s_2, \cdots\ s_k\}$。令 $C_j \subset X$ 为以s_j为中心的簇。对于点 $x \in C_j$，令 $\Delta_p := \mathrm{cost}(C_j, S)/|C_j|$ 表示 C_j的平均代价2: 对于 $x \in C_j$，令

$$\mu(x) := \frac{1}{4}\cdot\left(\frac{1}{k|C_j|} + \frac{\mathrm{cost}(x,S)}{k\,\mathrm{cost}(C_j,S)} + \frac{\mathrm{cost}(x,S)}{\mathrm{cost}(X,S)} + \frac{\Delta_x}{\mathrm{cost}(X,S)}\right).$$

3: $m \leftarrow \tilde{\mathcal{O}}\left(k/\varepsilon^2 \cdot \min(\varepsilon^{-2}, \varepsilon^{-z})\right)$, $A \leftarrow \emptyset$4: **对于** $i \leftarrow 1$ 至 m **执行**5: 从分布 a_i 中独立采样点 μ, $A \leftarrow A \cup \{(a_i,\ m^{-1}\cdot\mu(a_i))\}$ 6: **结束循环**7: **返回** A

定理 2.7。 *[BCP+24]* 敏感度采样，参见算法 1，以至少0.99的概率，为欧几里得(k, z)聚类生成一个大小为$(1+\varepsilon)$的核心集$\tilde{O}\left(\frac{k}{\varepsilon^4}\min(,\varepsilon^{2+z})\right)$。

接下来，我们回顾以下用于确定两个整数中较大者的分布式协议。

定理 2.8。 *[Nis93]* 给定两个$\mathcal{O}(\log n)$ 比特整数 X 和 Y，存在一个协议$\textsc{GreaterThan}$，它使用$\mathcal{O}(\log\log n)$ 通信总比特数，并判断 $X = Y$，$X < Y$还是$X > Y$。

我们回顾以下一种以较低通信代价应用 Johnson-Lindenstrauss 变换的协议。

定理 2.9。 *[KN10]* 对于任意整数 $d>0$，以及任意 $0<\varepsilon$，$\delta<\frac{1}{2}$，存在一个矩阵族$\mathcal{A}$，其中的矩阵属于 $\mathbb{R}^{\ell\times d}$，并且满足 $\ell=\Theta\left(\varepsilon^{-2}\log\frac{1}{\delta}\right)$，使得对于任意 $x\in\mathbb{R}^d$，

$$\Pr_{A\in\mathcal{A}}\left[\|Ax\|_2\notin[(1-\varepsilon)\|x\|_2,(1+\varepsilon)\|x\|_2]\right]<\delta.$$

此外，$A\in\mathcal{A}$ 可使用 $O\left(\log\left(\frac{l}{\delta}\right)\log d\right)$ 个随机比特进行采样，且每个矩阵 $A\in\mathcal{A}$ 的每列至多有 $\alpha=\Theta\left(\varepsilon^{-1}\log\left(\frac{1}{\delta}\right)\log\left(\frac{l}{\delta}\right)\right)$ 个非零项，因此如果在内存中显式存储 A，则可在 $O(\alpha\cdot\|x\|_0)$ 时间内计算 Ax。

自适应采样。 现在，我们介绍 [ADK09] 提出的自适应采样算法，并将其分析推广到 (k,z)-聚类。

算法 2 自适应采样

输入：数据集 $X=\{x_1,\ldots,x_n\}\subset[\Delta]^d$**；近似参数** $\gamma\geq 1$**输出：针对** X**1 上** (k,z)**-聚类的双准则近似** S**：** $S\leftarrow\emptyset, N\leftarrow O(k)$**，** $\gamma\leftarrow\Theta(1)$ **2：对于** $t\leftarrow 1$ **到** N**，执行 3：以概率** $\frac{d_i}{\sum d_i}$ **选择** s_t **为** $x_i\in X$**，其中** $\frac{1}{\gamma}\cdot$ **代价**$(x_i,S)\leq\gamma\cdot d_i$ **4：** $S\leftarrow S\cup\{s_t\}$ **5：结束循环 6：返回** S

[ADK09] 证明了自适应采样能够实现 k-均值的双准则近似，即$z=2$。为完整起见，我们将扩展证明，以说明该方法适用于加权情形和任意$z\geq 1$，不过我们指出，所用技术都是标准的。

定理 2.10。 存在一个算法，见 算法 2，该算法输出一个包含 S 个 $\mathcal{O}(k)$ 点的集合，使得以 0。99 的概率，代价$(S,X)\leq\mathcal{O}(1)\cdot$代价$(C_{OPT},X)$，其中 C_{OPT}是 (k,z)-聚类的一个最优解，该聚类针对 X。

如前所述，算法 2 中的自适应采样版本需要在每次更新后重新计算 代价(x,S)，导致通信成本很高。我们将介绍惰性采样算法。将更新后的自适应采样算法与惰性采样算法结合，可以减少所需的更新次数，从而降低总体通信成本，同时仍能保证实现$(\mathcal{O}(1),\mathcal{O}(1))$-双准则近似。

假设我们有 s 个站点，每个站点都有一个本地数据集 X_i，且每个数据点 x 都有一个采样权重 d_x。令 $D_i=\sum_{x\in X_i}d_x$ 和 $D=\sum_{i=1}^s D_i$。我们的目标是以概率 $p_x=\frac{d_x}{D}$ 对点 x 进行采样。在自适应采样的情况下，d_x 就是 cost(x,S)。为避免通信代价过高，我们的策略是让协调器仅保存 D_i，即实际 D_i 的一个近似，它 $D_i\leq\widetilde{D_i}<\lambda D_i$，并且仅当 $\widetilde{D_i}$ 与 D_i 相差较大时才更新 $\widetilde{D_i}$。然后我们可以使用 $\frac{d_x}{\widetilde{D}}$ 来采样该点，其中 $D=\sum_{i=1}^s D_i$。

由于 D 可能大于 D，的概率是多数，而在未返回 $\perp$ 的条件下采样 x 的概率恰好是 $\frac{d_x}{D}$。因此，根据马尔可夫不等式，我们至少可以采样 N 个点，满足

算法 3 惰性采样

输入： $\{d_x\}_{x\in X}$，采样权重；$\{D_i\}^{i\in[s]}$，$\{\widetilde{D_i}\}^{i\in[s]}$ 的一个常数近似，其中 $D_i \le \widetilde{D_i} \le \lambda D_i$，且对于每个站点 i，$D_i = \sum_{x\in X_i} d_x$。**输出：** 返回 $\perp$ 或根据分布 $\frac{d_x}{D}$ 采样得到的一个点 x，其中 $D = \sum_{x\in X} d_x$

$\widetilde{D} \leftarrow \sum_{i=1}^{s} \widetilde{D_i}$，$x \leftarrow \perp$

协调器以概率 $= \frac{\widetilde{D_i}}{\widetilde{D}}$ i 生成一个索引 i；协调器向站点 $x \leftarrow y$ 发送采样请求，概率为 $\frac{d_y}{\widetilde{D_i}}$

返回 x

分布 $\frac{d_x}{D}$，重复执行惰性采样共 $\mathcal{O}(N)$ 次后，概率至少为 0.99。

引理 2.11. 设 x 为 算法 3 返回的结果，$\mathcal{E}$ 为 算法 3 未返回 $\perp$这一事件。若 $\sum D_j \le \sum \widetilde{D_j} < \lambda \sum D_j$，$\mathbf{Pr}[\mathcal{E}] \ge \frac{1}{\lambda}$且 $\mathbf{Pr}[x = y|\mathcal{E}] = \frac{d_y}{D}$。此外，存在某个常数 $\gamma \ge 1$，使得运行 算法 3 共 γN 次后，它将返回至少 N 个由分布 $\frac{d_x}{D}$ 采样得到的点，概率至少为 0.99。

证明。令 $\mathcal{F}_i$ 表示结果由站点 i 采样得到这一事件。由于 $\mathbf{Pr}[\mathcal{F}_i] = \frac{\widetilde{D_i}}{\widetilde{D}}$ 且$\mathbf{Pr}[\mathcal{E}|\mathcal{F}_i] = \sum_{x\in X_i} \frac{d_x}{\widetilde{D_i}} = \frac{D_i}{\widetilde{D_i}}$，根据全概率定律，

$$\mathbf{Pr}[\mathcal{E}] = \sum_{i=1}^{s} \mathbf{Pr}[\mathcal{E}|\mathcal{F}_i] \cdot \mathbf{Pr}[\mathcal{F}_i] = \sum_{i=1}^{s} \frac{D_i}{\widetilde{D_i}} \cdot \frac{\widetilde{D_i}}{\widetilde{D}} = \frac{D}{\widetilde{D}} \ge \frac{D}{\lambda D} = \frac{1}{\lambda}.$$

对于任意 $y \in X_i$，有

$$\mathbf{Pr}[x = y|\mathcal{E}] = \frac{\mathbf{Pr}[x = y]}{\mathbf{Pr}[\mathcal{E}]} = \frac{1}{\mathbf{Pr}[\mathcal{E}]} \cdot \frac{\widetilde{D_i}}{\widetilde{D}} \cdot \frac{d_y}{\widetilde{D_i}} = \frac{\widetilde{D}}{D} \cdot \frac{\widetilde{D_i}}{\widetilde{D}} \cdot \frac{d_y}{\widetilde{D_i}} = \frac{d_y}{D}.$$

此外，令

$$Z_i = \begin{cases} 0, \text{if LazySampling returns } \perp \\ 1, \text{otherwise.} \end{cases}$$

在运行惰性采样 γN 次后，惰性采样未返回$\perp$的次数恰好为 $\sum_{i=1}^{\gamma N} Z_i$。由于 $\mathbb{E}[Z_i] = \mathbf{Pr}[\mathcal{E}] \ge \frac{1}{\gamma}$，我们有 $\mathbb{E}\left[\sum_{i=1}^{\gamma N} Z_i\right] \ge \frac{\gamma}{\lambda} N$。根据马尔可夫不等式，存在某个 $\gamma \ge 1$，使得 $\mathbf{Pr}\left[\sum_{i=1}^{\gamma N} Z_i < N\right] \le 0.01$。因此，惰性采样将以至少 0.99 的概率采样至少 N 个点。

□

最后，在 算法 4 中，我们回顾一种标准算法，该算法通过在将一个数舍入到固定底数的幂之后存储该数的指数，从而提供对该数的高效近似编码 λ。

定理 2.12。 算法 4 的输出可用于计算一个数 $\widehat{m}$ 使得$m \le \widehat{m} < \lambda m$。

算法 4 PowerApprox

输入： 待编码的数字 m，基数 $\lambda > 1$**输出：** 一个整数 i，
它 λ^i 是 λ的近似值 m1: 令 i 为满足条件的整数
$m \le \lambda^i < \lambda m$2: **返回** i

用于 (k, z)-聚类的核心集构造的高效编码。我们回顾以下高效编码：给定由 [CAWWZ25] 给出的、用于 (k, z)-聚类的核心集。给定数据集 X（它可能表示原始输入，也可能表示从某个更大的数据集导出的加权核心集），我们首先计算 X 上 (k, z)-聚类问题的一个常数因子近似 C'。对于每个 $x \in X$，令 $\pi_{C'}(x)$ 表示 C' 中距离 x 最近的中心。于是，每个点 x 都可以分解为 $x = \pi_{C'}(x) + (x - \pi_{C'}(x))$，将其拆分为最近中心和一个残差向量。

要获得 (k, z)-聚类的 $(1+\varepsilon)$-近似，只需将偏移向量 $x - \pi_{C'}(x)$ 的每个坐标舍入到最接近的 $(1+\varepsilon')$ 次幂，其中 $\varepsilon' = \text{poly}\left(\varepsilon, \frac{1}{d}, \frac{1}{\log(n\Delta)}\right)$。令 y' 表示这个舍入后的向量。

然后，我们将每个点编码为 $x' = \pi_{C'}(x) + y'$，同时存储中心 $\pi_{C'}(x)$ 的索引以及舍入后偏移量的指数值。此表示每个点使用 $\mathcal{O}\left(\log k + d \log\left(\frac{1}{\varepsilon}, d, \log(n\Delta)\right)\right)$比特。完整算法详见算法 5。

算法 5 在 (k, z)-聚类中用于核心集生成的紧凑编码

输入：数据集 $X \subset [\Delta]^d$ 及其权重 $w(\cdot)$、精度参数 $\varepsilon \in (0, 1)$、簇数 k、参数 $z \ge 1$、失败概率 $\delta \in (0, 1)$ 输出：$(1+\varepsilon)$-近似核心集，用于 (k, z)-聚类 1: $\varepsilon' \leftarrow \text{poly}(\varepsilon^z)$ $\text{poly}(k, \log(nd\Delta))$ 2: 在 X 上计算一个常数因子近似的 (k, z)-聚类解 C' 3: 对每个 $x \in X$ 执行 4: 将 $c'(x)$设为 C' 中距离 x 最近的中心 5: 计算残差向量 $y' = x - c'(x)$ 6: 将 y' 的每个坐标舍入到最接近的 $(1+\varepsilon')$ 的幂，得到 y 7: 定义 $x' = (c'(x)\ y)$，其中 y 存储每个坐标对应的指数 8: 将 x' 添加到新集合：
$X' \leftarrow X' \cup \{x'\}$ 9: 结束循环 10: 返回 (C', X')

我们对 [CAWWZ25]中的高效编码有如下保证：

引理 2.13。 *[CAWWZ25]* 设 $\varepsilon \in \left(0, \frac{1}{2}\right)$，并设 X' 为加权数据集 S，它使用中心集 C' 中的偏移量构造而成，具体定义见 算法 5。那么，对于任意中心集 $C \subset [\Delta]^d$满足 $|C| \le k$，均有：

$$(1-\varepsilon) \cdot \text{cost}(C, X) \le \text{cost}(C, X') \le (1+\varepsilon) \cdot \text{cost}(C, X).$$

引理 2.14. *[CAWWZ25]* 设 X 是一个核心集，其点权重范围为[1,poly$(nd\Delta)$]。那么，变换后的集合 X' 构成$(1+\varepsilon)$-强核心集，用于 X，其总空间需求为$\mathcal{O}(dk\log(n\Delta)) + |X| \cdot \text{polylog}\left(k, \frac{1}{\varepsilon}, \log(nd\Delta), \log\frac{1}{\delta}\right)$ 比特。

3 技术概述

本节将从技术层面概述我们的分布式协议，涵盖黑板模型和协调器/消息传递模型。

以往方法及其失效原因。 一个自然的问题是，简单的方法能否达到类似的通信上界。例如，人们或许会希望，将 [ZTHH25] 的空间上界与现有分布式聚类算法相结合就足够了。然而，如何将空间上界转化为更优的分布式聚类上界，并非显而易见。例如，作为其空间上界的一项应用， [ZTHH25]实现了一种总通信量为 $\tilde{\mathcal{O}}\left(\frac{sdk}{\varepsilon^4}\right)$的分布式协议，适用于 k和 ε的某些范围；但这一结果实际上比图1 第三行更差，因此也比我们在相同范围内的 $\tilde{\mathcal{O}}\left(\cancel{sk}+\frac{dk}{\varepsilon^4}+dk\log n\right)$ 上界更差。

—

另一种方法是让每台服务器计算一个本地核心集，例如 [HV20、CSS21、CLSS22、CLS+22、BCP+24]，还可以结合降维，例如 [SW18、MMR19、ISZ21]，，然后在黑板模型中广播这些核心集，或在消息传递模型中将这些核心集发送给协调器。然而，这种方法需要 $\mathcal{O}\left(\frac{sdk}{\varepsilon^4}\log n\right)$ 比特的通信量，而我们的上界为 $\tilde{\mathcal{O}}\left(sk+\frac{dk}{\varepsilon^4}+dk\log n\right)$。类似地，任何基于局部敏感哈希的方法，即便借助降维或其他压缩技术，例如 Kashin 表示 [LV10、AFN+24]，也至少需要每台服务器计算一个本地核心集，因此会遭遇相同的缺陷。这些局限表明，克服朴素策略固有的开销远非易事，也凸显了对新技术的需求。我们的框架正是为应对这些挑战而设计的：它通过通信最优原语模拟集中式算法，从而突破令这类简单方案受阻的瓶颈。

黑板模型的$(1+\varepsilon)$-核心集。我们的出发点是[CSWZ16], 的分布式协议，该协议首先改造 [MP04] 的中心化 RAM 算法，以发布一个由 O $(k\log n)$ 个中心构成的加权集合S，它是 (k,z)-聚类的常数因子双准则近似。可以证明，对 S 上 (k,z)-聚类的常数因子近似能够达到对输入集 X上最优聚类的常数因子近似 C 。由于 S 已在黑板上可用，此步骤是直接的。其算法的总通信成本为 O $\left((s+kd)\log^2 n\right)$。

我们注意到，各站点可以轻松计算每个中心所服务的点数 $X\ c\in C$。我们回顾一下，利用这些信息以及代价(X,C)，可以求出每个点 x 的敏感度 $s(x)$的常数因子近似，参见 [CWZ23]。因此，各站点随后可以使用敏感度采样采样 $\tilde{\mathcal{O}}\left(\min\left(\frac{k^2}{\varepsilon^2},\ \frac{k}{\varepsilon^4\min(,\varepsilon^{2+z})}\right)\right)$ 个点 X，以获得一个 $(1+\varepsilon)$-核心集。然后，我们可以使用高效编码方案，以$\mathcal{O}\left(\log k+d\log\left(\frac{1}{\varepsilon},d,\log(n\Delta)\right)\right)$ 比特表示每个采样点。这满足了通过敏感度采样获取的点所需的依赖关系，但常数因子近似产生的通信量并非最优。因此，仍需为常数因子近似寻找一种通信效率更高的协议。

黑板模型中的常数因子近似。 尽管黑板模型中的常数因子近似有许多选择，但它们似乎都各有不足。例如，前述的 Mettu-Plaxton 协议 [MP04] 需要 $\mathcal{O}(\log n)$轮采样，这会产生 $\mathcal{O}(k \log n)$ 个点。一个自然的做法是在每个站点本地构造核心集，使加权点的总数为 $\mathcal{O}(sk)$ ，而不是 n。遗憾的是，将 Mettu-Plaxton 推广到加权点后，需要 $\mathcal{O}(\log W)$ 轮，其中 W 是点的总权重；就我们的目的而言，该值为$\text{poly}(n)$。

另一个众所周知的原型是自适应采样框架 [ADK09, BEL13], ，该框架可用于在分布式设置 [BMV+12]中实现著名的 `kmeans++` 算法 [AV07]。该方法迭代地以与此前采样点到各点的距离的 z 次幂成正比的概率，采样固定数量的点；自适应采样使用 O (k) 个点即可实现 O (1) 近似，而 `kmeans++` 使用 k 个点即可实现 O $(\log k)$ 近似。然而，由于每轮之后所有站点 $i \in [s]$ 都必须报告其点到采样点的距离的 z次幂之和 D_i ，这些方法朴素地需要 O $(sk \log n)$ 个通信比特，这对于我们的目标而言是无法接受的。

相反，我们对其点到采样点的距离的 z次幂之和 D_i 采用惰性更新。黑板上仅保存基于站点 i 上次报告其值时刻的 D_i估计值 D_i 。随后，各站点尝试进行自适应采样，其中站点 i 以与 D_i成正比的概率被$\widehat{}$ $\widehat{}$采样。被采样时，站点 i 使用 D_i 在本地尝试采样$\widehat{}$一个点，但这可能失败，因为 $D_i \geq D_i$，这意味着在$D_i \frown$中随机选择的整数可能不对应于 D_i中的整数。幸运的是，如果$\frown D_i$ 是 D_i的常数近似，则失败概率将为常数，因此根据马尔可夫不等式，采样轮数总数仍为 $\mathcal{O}(k)$。如果站点 i 未能采样到一个点，我们预计 D_i 会比 D_i小一个常数比例$\widehat{}$，从而促使站点 i 在黑板上更新其值 D_i 。这种情况最多会发生 $\mathcal{O}(s\log\ n)$ 次。如果每个站点 i 将 D_i 舍入到 2 的幂，那么只需传输舍入结果的指数，使用 $\mathcal{O}(\log \log n)$ 个比特，即可将 D_i 近似到 2 倍因子以内。因此，常数因子近似的总通信量为 $\mathcal{O}(s \log\ n + kd\log\ n)$。

主要缺点是，通信轮数总数可能达到 $\mathcal{O}(s\log\ n)$，因为每个 s 站点最多可以更新其 D_i 值 $\mathcal{O}(\log n)$ 次。

黑板模型中的通信轮数削减。 一种自然的轮数削减方法是：当我们未能采样到一个点时，让所有 s 站点报告其更新后的 D_i 值。然而，由于我们采样 $\mathcal{O}(k)$ 个点，这会导致通信复杂度为 $\mathcal{O}(sk)$。

相反，我们检查全局权重 $\sum_{i\in[s]} D_i$ 是否相对于黑板上的$\widehat{}$数量 $\sum_{i\in[s]} D_i$减少了 $1\ 64$ 。为此，我们首先在每轮中执行 L_1 采样。$\frown$也就是说，黑板以概率 $p_i = \frac{D_i}{\sum_{i\in[s]} \widehat{D_i}}$选择站点 i，并计算 $\frac{D_i}{p_i}$。注意，该量的期望为 $\sum_{i\in[s]} D_i$ ，其方差至多为 $(\sum_{i\in[s]} D_i)^2$，（相差一个常数因子）。因此，重复 $\mathcal{O}(1)$ 次并取平均值，我们便可得到 $\sum_{i\in[s]} D_i$ 的 $\sum_{i\in[s]} D_i$ 加性近似，这足以确定该总和是否从 $\sum_{i\in[s]} \widehat{D_i}$减少了 $1\,64$ 倍。随后，我们可以取 $\mathcal{O}(\log\log\ n)$ 个实例，通过并集界覆盖 $\mathcal{O}(\log\ n\log\ k)$ 次迭代。

—

另一个观察是，只要我们的采样概率没有降低$1\ 64$，我们就可以同时进行多次采样。因此，我们从 $i = 0$ 开始，使用自适应采样分布从 s 个站点收集 2^i 个样本。然后检查

总权重 $\sum_{i\in[s]} D_i$ 是否小于 $\frac{1}{64}\sum_{i\in[s]} \widehat{D_i}$；如果不是，我们就递增 i 并采样 2^i 个样本，然后重复此过程，直到 $2^i > k$ 或总权重 $\sum_{i\in[s]} D_i$ 小于 $\frac{1}{64}\sum_{i\in[s]} \widehat{D_i}$。

请注意，这需要 $\mathcal{O}(\log\ n\log\ k)$ 轮采样，这足以满足 L_1 采样过程的失败概率。此外，由于我们可以通过将 D_i 的值舍入到 2 的幂并返回指数，使用 $\mathcal{O}(\log\log n)$ 个通信比特来近似 L_1 采样。由于每次总权重 $\sum_{i\in[s]} D_i$ 都小于 $\frac{1}{64}\sum_{i\in[s]} \widehat{D_i}$时，我们都需要所有 s 个站点更新~它们的权重，因此在 $\mathcal{O}(\log\ n\log\ k)$ 轮通信中，通信量为 $\mathcal{O}(s\log\ n + kd\log\ n)$。

在黑板模型中应用敏感度采样。要应用 [BCP+24] 的敏感度采样框架，需要对每个簇 $|C_j|$ 的数量以及每个簇的代价 cost(C_j, S) 作常数近似，其中 S 是最优解的 (O (1) , O (1)) 双准则近似。每个站点需要 O ($\log\log n$) 个比特，才能上传每个簇的这种常数近似。由于有 s 个站点和 O (k) 个簇，这将导致 O ($sk\log\log n$) 个通信比特，对于我们的目标而言代价过高。作为替代，我们将 Morris 计数器应用于分布式环境；这类计数器用于流模型中的近似计数。我们用一个计数器 r 存储簇的数量/代价的对数，并在每次对簇的数量/代价进行计数时，以概率$\frac{1}{2^r}$ 将计数器加 1。每个站点依次应用 Morris 计数器，并且只将增量上传到黑板上。与 Morris 计数器类似，这种子程序可以返回簇的数量/代价的常数近似。如果某个站点处理后所有计数器都保持不变，该站点只需用 O (1) 个比特告知下一个站点无需更新，最多使用 O (1) 个比特。否则，更新发生变化的计数器需要 O ($\log\log n$) 个比特。由于每个计数器最多增加 O ($\log n$) 次，并且需要计数的簇数量/代价共有 O (k) 项，因此上传所有更新~需要 O ($k\log n$) 个比特。因此，上传每个簇数量 $|C_j|$ 的常数近似以及每个簇代价 cost(C_j, S) 的常数近似，最多需要 O ($s + k\log n$) 个比特。

每个站点在获知每个簇数量的常数近似值 $|C_j|$ 以及每个簇的代价 cost(C_j, S) 后，都可以在本地应用敏感度采样。结合适用于 $(1+\varepsilon)$-核心集的高效编码，我们可以用 $\mathcal{O}\left(\frac{(d\log\log n+\log k)k}{\min(\varepsilon^4,\ \varepsilon^{2+z})}\right)$ 比特上传核心集，从而达到黑板模型所需的界。

协调器/消息传递模型。 对于消息传递/协调器模型，我们的出发点仍然是自适应采样，它可以在 $\mathcal{O}(k)$ 轮内完成，以实现常数因子近似 C，使用 $\mathcal{O}(skd\log\ n)$ 通信总比特数 [BEL13]。我们还可以对 C 应用敏感度采样，并结合我们的高效编码方案，为后续的 $(1+\varepsilon)$-核心集实现所需的通信量。因此，核心问题仍然是如何改进常数因子协议。

为此，我们首先介绍一个名为高效通信的子程序，它能够以较低的通信成本发送点的位置。高效通信背后的直觉与高效编码类似：利用某个站点所拥有的点对我们想要发送的点进行编码。唯一的区别在于，我们使用不同的点来高效编码每个维度的坐标，而不是仅使用单个点来编码想要发送的点。

回想一下，存在一个GreaterThan 协议，给定两个$\log n$ 比特整数后，可以用常数概率判断~哪个更大，所需的通信比特数为$\mathcal{O}(\log\log n)$。此外，这个过程可以重复$\mathcal{O}(\log(sk))$ 次，从而使失败概率达到[1] poly(sk)

从而可以对 poly(sk) 个可能的步骤应用并集界。借助 GreaterThan，我们可以比较各站点所拥有的点在第 i 维上的坐标 $x_j^{(i)}$ 与待接收的点的坐标 $y^{(i)}$。通过二分搜索，我们可以找到最接近 $y^{(i)}$ 的坐标 $x_j^{(i)}$，这需要$\mathcal{O}(\log n)$ 次搜索，因为该站点最多拥有 n 个点。再通过一次二分搜索，我们可以进一步找到 $\Delta y^{(i)}$，使其为 $|x_j^{(i)} - y^{(i)}|$ 的 $(1+\varepsilon)$-近似。由于 $|x_j^{(i)} - y^{(i)}| = \text{poly}(n)$，我们需要$\mathcal{O}(\log \log n)$ 次搜索来找到 $\Delta y^{(i)}$。由于协调器需要将自适应采样所采样点的近似坐标发送给每个站点，因此自适应采样的总通信成本~为$\mathcal{O}(dsk \log n)$ 比特，这对于我们的目标而言仍然代价过高。

为了解决这一问题，我们首先为每个站点生成一个 $(1+\mathcal{O}(\varepsilon))$-核心集，使得每个站点的数据集大小为 $\mathcal{O}(k)$，从而只需进行 $\mathcal{O}(\log n)$ 次搜索即可找到最接近 $x_j^{(i)}$ to$y^{(i)}$ 的坐标。此外，我们注意到，自适应采样只需要每个点的近似代价。因此，我们可以应用 Johnson-Lindenstrauss 将数据集投影到 $\mathcal{O}(\log(sk))$ 维空间。因此，自适应采样的总通信成本为 sk polylog(log n, s, k) 比特。

在获得最优解的 $(\mathcal{O}(1), \mathcal{O}(1))$-双准则近似后，我们可以用总计$\mathcal{O}(sk \log \log n)$ 通信比特数来传递每个簇的数量和代价的常数近似。然而，每个站点可能会将点分配到错误的簇，因为它只能获得 S的近似位置。幸运的是， [BCP+24] 的敏感度采样过程仍会返回一个 ()-核心集，前提是我们将每个点分配给一个中心，使其与该中心的距离接近该点到离它最近的中心的距离。因此，结合高效通信，我们可以用min($\varepsilon^4,\varepsilon^{2+z}$) polylog(log $n, k, \frac{1}{\varepsilon}$) 比特上传该 $(1+\varepsilon)$-核心集，从而达到黑板模型所需的界。

4 黑板模型中的分布式聚类协议

回顾一下，在黑板通信模型中，每个 s 站点都可以使用私有随机性，并能按顺序分轮次直接向公共平台广播，自定界消息会立即对所有人可见。本节中，不失一般性地假设有一名中央协调器负责管理整个过程。我们的目标是在这一设置下设计高效的(k, z)-聚类协议。

4.1 常数因子双准则算法

本节将介绍一种新算法，可实现(O (1) , O (1))-双准则近似；我们将在第4.2节中利用它构造一个($1+\varepsilon$)-核心集。由此得到的方案可实现$(1+\varepsilon)$-近似，其通信量为 ~O $\left(s \log n + dk \log n + \right.$ ^{dk} min($\varepsilon^4,\varepsilon^{2+z}$)$\left.\right)$ 通信比特，轮数为O (log n log k)；附录中还针对 $k = \text{O}(\log n)$的情况给出了额外优化。黑板模型中的现有双准则算法受通信瓶颈所限，例如经典的Mettu–Plaxton协议 [MP04] 及后续改进方案 [CSWZ16] ，会在O (log n)轮中每轮采样O (k)个点，导致通信量达到O $\left(dk \log^2 n\right)$ 比特。这些代价对于我们所追求的保证而言过高。

为克服这些障碍，我们改造了最初为集中式设定下的 k-中位数问题开发的自适应采样框架 [ADK09、BEL13]。该过程以类似于 `kmeans++` [AV07、BMV+12]的方式反复采样点，但现在分布在 s 多个服务器上。在

每次迭代中，我们首先按与 D_j 成正比的概率采样一个服务器 j，其中 D_j 是其各个点到当前采样点的距离的z次方之和；然后根据自适应分布，在服务器 j 内采样点。朴素实现需要在每次迭代后报告每个 D_j，导致通信量达到 $\mathcal{O}(sk\log n)$，开销过高。为高效执行这一步骤，我们设计了子程序惰性采样（LazySampling），仅使用黑板上维护的近似值 D_j，便可从自适应$\widetilde{}$分布中抽样。我们的关键创新是惰性地更新这些估计值：只有当某个站点的真实 D_j 变化超过一个常数因子时，才报告新值；我们证明，这足以满足自适应采样的需求。

引理 4.1。 存在一种算法惰性采样（*LazySampling*），能够以至少 0.99 的概率从自适应采样$\widetilde{}$分布中进行采样，前提是 $\sum D_j \le \sum D_j < \lambda \sum D_j$ 对于固定常数 $\lambda > 1$。该算法使用 $O(\log s + d\log n)$ 个通信比特。

惰性采样算法是黑板模型中用于自适应采样的通信高效子程序，其形式化描述见附录 2 中的算法 3。每个服务器 j 都会为其本地数据集维护一个$\widetilde{}$$\widetilde{}$近似权重 D_j，满足 $D_j \le D_j \le \lambda D_j$，其中 D_j 是从其点到当前采样点的距离的z次幂之和。协调器首先以与 $D_j/\sum_i D_i$成比例的概率选择一个服务器 $j\widetilde{}$$\widetilde{}$，然后请求一个点 $y \in X_j$，并以概率 $\sim$$\sim d_y/D_j$进行采样。通过 D_j 惰性地更新，即仅在 D_j 发生显著变化时才更新，惰性采样能够确保所抽取的点接近真实的自适应分布，同时大幅减少通信量。~该算法要么返回一个采样点，要么返回 $\perp$，并且每轮仅使用 O (log $s + d$log n) 个比特，从而支持在 s 个服务器之间可扩展地执行双准则算法。这样可确保每个站点仅通信 O (log n) 次更新，因此，对于常数因子~$\widetilde{}$近似，总通信量可降至 O (s log $n + kd$ log n)，因为假设所有点都位于边长为 poly(n) 的网格中，每个服务器最多只能更新 O (log n) 次 D_j。由于惰性更新依赖于近似值，协议必须偶尔验证聚合估计值 $\sum_j D_j$是否仍然接近真实总和 $\sum_j D_j$。为此，我们使用子程序 L1Sampling，用于检验两个总和之差是否超过一个常数因子。

$\sim$ $\widetilde{}$**引理 4.2**。 存在一种算法 *L1Sampling*，其输入为 $\{D_j\}_{j\in[s]}$，使得 $D_j \ge D_j$对所有 $j \in [s]$均成立。令 $\widetilde{D} = \sum_{j\in[s]} \widetilde{D_j}$ 和 $D = \sum_{j\in[s]} D_j$，并令 $\mu > 1$ 为一个参数。若 $\mu^2 D \le \widetilde{D}$，~，则 *L1Sampling* 以至少 $1-\delta$的概率返回 `True`。若 $\mu D > D$，则它以至少 $1-\delta$的概率返回 `False`。该算法使用 ~$O\left((\log s + \log\log n)\log\frac{1}{\delta}\right)$通信比特。

~L1Sampling 通过采样少量站点，并汇总重新缩放后的采样值，检查近似站点代价之和 D 是否与真实总和 D 相差一个常数因子以内。如果~汇总值与 D足够接近，该算法就返回 `True`；否则返回 `False`，从而以高概率保证惰性估计仍然准确；该算法作为算法 9 列于附录 B.2。尽管这种惰性策略可能需要多达 O ($s$log n) 轮，我们通过延迟所有更新，直到全局总和 $\sum_j D_j$ 减少一个常数因子，进一步降低轮次复杂度。当某一轮中没有采样到新的点时，自然就能检测到这一事件；以这种方式同步更新，可将总轮数降至 O (log $n$log k)。最后，为进一步减少通信，我们使用子程序 PowerApprox，它仅用 O (log log n) 比特，就能将每个 D_j 编码到一个常数因子以内：

定理 4.3。 给定 $m = \mathrm{poly}(n)$ 和常数 $\lambda > 1$，存在一种算法 *PowerApprox*(m, λ)，该算法输出以 $O(\log\log n)$ 比特编码的 $\widetilde{m}$，使得 $m \le \widetilde{m} < \lambda m$。

给定 m 和基数 $\lambda > 1$，PowerApprox 计算满足以下条件的最小整数 i： λ^i从上方近似 m，即 $m \leq \lambda^i < \lambda m$，并返回 i。这样，每个站点只需使用 O (log log n) 比特即可通信其代价的简洁表示，随后可对其进行解码，以获得原始值的常数因子近似；完整细节见附录 2 中的算法 4。结合惰性采样和 L1Sampling，该方法在保持通信量近线性的同时确保准确性。该协议首先使用惰性采样对点进行采样，再通过 L1Sampling 验证准确性，并仅在必要时通过 PowerApprox 刷新估计值。为限制轮次复杂度，各站点仅在全局和 $\sum_j D_j$ 下降一个常数因子时才报告更新；当某一轮未能采样到新的点时，即可检测到这一情况。这样，轮次数量便从 O (s log n) 降至 O (log n log k)。图 2 对该过程进行了非正式总结，正式过程见附录 B.3。总体而言，这在黑板模型中，以近线性的通信量和多对数轮次，实现了针对 (k, z)-聚类的首个 (O (1) , O (1)) 双准则近似：

算法：(k, z)**-聚类的双准则近似（简化版）**

1. **输入：**数据集 X_i，包含每个站点 $i \in [s]$。

2. **输出：**集合 S，它是一个 ($\mathcal{O}$ (1)，$\mathcal{O}$ (1))-双准则近似。

3. 初始化：

   - 将一个点采样到 S 中，并计算每个站点的近似距离$\{D_j\}$。 $\sim$
   - 设置计数器： $N = \mathcal{O}(k)$，$M = 0$。

4. **当 $M < N$ （采样大约 k 个点）时：**

   - 使用惰性采样将新点采样到 S 中。$\sim$
   - 使用 L1Sampling 检查近似距离$\{D_j\}$的准确性。
   - 如果距离准确，则加大采样力度。$\sim$
   - 否则，使用 PowerApprox 优化近似距离$\{D_j\}$。
   - 将 M 更新为成功添加的点数。

5. **返回** S。

图 2：通过自适应采样实现双准则近似的非正式版本。

引理4.4。 存在一种算法，输出一个集合 S，使得 $|S| = O(k)$ 且 代价(S, X) $\leq O(1) \cdot$ 代价(C_{OPT}, X) 的成立概率至少为 0.98，其中 C_{OPT} 是 $\sim X$ 的最优(k, z)-聚类。该算法以至少 0.99 的概率使用 $O(s \log n + kd \log n)$ 通信比特和 $O(\log n \log k)$ 通信轮数。

4.2 $(1+\varepsilon)$-核心集构造

为在输入数据集 X 上获得针对 (k, z)-聚类的 $(1+\varepsilon)$-核心集，并给定一个 ($\mathcal{O}$ (1) , $\mathcal{O}$ (1))- 双准则近似 S，我们使用以下敏感度采样概念。对于每个中心

$s_j \in S$，令 $C_j \subset X$ 为以 s_j为中心的簇。对于一个点 $x \in C_j$，令 $\Delta_p := \text{cost}(C_j, S)/|C_j|$表示 C_j的平均代价。对于 $x \in C_j$，我们定义

$$\mu(x) := \frac{1}{4} \cdot \left(\frac{1}{k|C_j|} + \frac{\text{cost}(x,S)}{k\,\text{cost}(C_j,S)} + \frac{\text{cost}(x,S)}{\text{cost}(X,S)} + \frac{\Delta_x}{\text{cost}(X,S)} \right).$$

我们将敏感度采样定义为这样一个过程：对每个点 x，以与 $\mu(x)$ 的常数因子近似成正比的概率进行采样。于是我们得到以下保证：

定理 4.5。 *[BCP+24]* 从敏感度采样概率分布的常数因子近似中采样 $\tilde{O}\left(\frac{k}{\varepsilon^4} \min(\cdot, \varepsilon^{2+z}) \right)$ 个点，然后重新赋权，可为欧几里得 (k, z) 聚类提供一个$(1+\varepsilon)$-核心集，其成功概率至少为 0.99。

我们指出，[BCP+24] 为 k-median 得到了最优界，这些界可直接推广到我们的框架；由于推广过程十分自然，我们省略进一步讨论。要应用敏感度采样框架，我们需要根据双准则解 S，得到簇大小 $|C_j|$和代价 $\text{cost}(C_j, S)$的常数因子近似。然而，对于 s 个服务器和 $\mathcal{O}(k)$ 个簇，直接上传这些量会耗费 $\mathcal{O}(sk \log\log n)$ 比特，开销过高。因此，我们针对分布式近似计数对 Morris 计数器 [Mor78] 进行了改造：

引理 4.6。 假设每个服务器 i 持有 k 个数值 $n_{i,j}$ 并且我们有 $|N_j| = \sum_{i=1}^{s} n_{i,j} = \text{poly}(n)$。存在一个算法 *DistMorris*，使其输出 $\{\widehat{N_j}\}$，满足：对于所有 $j \in [k]$，$\tilde{}$ _ _ ，以 0.99 的概率，使用 $O(s + k \log n)$ 个通信比特。

我们的 Morris 计数器协议共同维护一个计数器r_j，并以概率$\frac{1}{2^r}$ 对簇 C_j中的每个项进行递增。这为簇大小 $|C_j|$提供了常数近似。关键在于，如果某个站点不更改全局计数器 $\{r_j\}$，那么它只需发送一个比特来表示未进行更新。否则，O (k) 个计数器中的每一个最多只能增加 O $(\log n)$ 次，因此这些近似的总上传代价为 O $(k \log n)$ 个比特。我们可以使用类似的协议来近似每个簇的代价 $\text{cost}(C_j, S)$，因此总体而言，这些$\tilde{}$近似所需的总通信量为 O $(s + k \log n)$ 个比特。基于这些近似，服务器随后可以在本地执行敏感度采样。最后，我们需要对通过敏感度采样 x 得到的每个点进行高效编码。非正式地说，每个点 x 被编码为 $x' = \pi_S(x) + y'$，其中 $\pi_S(x)$ 是 S 中距离 x最近的中心，而 y' 是偏移向量 $x - \pi_S(x)$，其坐标将舍入到 $(1+\varepsilon')$ 的最近幂次，其中 $\varepsilon' =$poly $\left(\varepsilon, \frac{1}{d}, \quad {}^{1}\log(n\Delta)\right)$。形式化细节见附录 2。综合以上内容，算法 6 达到了定理 1.2 中的保证。

5 分布 式聚类协议（协调器模式） l

现在我们转向协调器（消息传递）模型，在该模型中，每台服务器仅通过私有信道与协调器通信。直接模拟我们的黑板协议需要$\mathcal{O}(dsk \log n)$比特，因为需要在s 台服务器间执行$\mathcal{O}(k)$轮自适应采样。为避免这一高昂代价，我们设计了一种协议，无需将点显式发送给所有站点即可模拟自适应采样。我们首先应用约翰逊–林登施特劳斯变换，将维度降至 $d' = \mathcal{O}(\log(sk))$，同时将两两距离的误差控制在$(1 \pm \varepsilon)$以内。随后，我们可以在投影空间中进行自适应采样，因此选中点 s 时，只需传递其投影 $\pi(s)$。为了高效近似这些位置，我们引入

算法 6 ($1+\varepsilon$**)-核心集（黑板模型） 输入：具有常数因子近似的中心双准则集合** S**，以及** $|S| =$ **O (**k**) 输出：(**$1+\varepsilon$**)-核心集** A**1：使用 DistMorris 在黑板上获得** $|C_j|$ **的 O (1)-近似，以及所有** $j \in [k]$ **对应的 cost(**C_j**,** S**) 2：** $m \leftarrow \tilde{O}\left(\frac{k}{\varepsilon^2}\right.$ **min{**$\varepsilon^{-2}, \varepsilon^{-z}$**}** $\left.\right)$ **▷设置核心集大小 3：for** $i \leftarrow 1$ **to** s **\\ 向敏感度发送局部近似 do 4：** $A_i \leftarrow \emptyset$**5：在本地为所有** $x \in X_i$**计算** $\widetilde{\mu}(x)$**，作为** $\mu(x)$**的 O (1)-近似 6：将** $\widetilde{\mu}(X_i) = \sum_{x \in X_i} \widetilde{\mu}(x)$**上传至黑板 7：end for 8：以概率**$\frac{\widetilde{\mu}(X_i)}{\sum_{i=1}^{s} \widetilde{\mu}(X_i)}$**独立采样站点** i**，共采样** m **次。▷敏感度采样 9：令** m_i **为站点** i**被采样的次数，并将** m_i **写入黑板 10：for** $i \leftarrow 1$ **to** s **\\ 遍历站点以生成样本 do 11：** $A_i \leftarrow \emptyset$**12：for** $j \in [m_i]$ **\\ 从站点** i **采样** m_i **个点 do 13：以概率** $p_x = \frac{\widetilde{\mu}(x)}{\widetilde{\mu}(X_i)}$**采样** x **14：if** x **被采样 \\ 高效编码每个样本 then 15：令** x' **为由** S **高效编码的** x**，准确性为** $\varepsilon' =$**poly(**ε**) 16：** $A \leftarrow A_i \cup$**{(**x'**,** $\frac{1}{m\widehat{\mu}(x)}$**))}，其中** $\widehat{\mu}(x)$**是** $\widetilde{\mu}(x)$**的(**$1+\frac{\varepsilon}{2}$**)-近似 17：end if 18：end for 19：将** A_i **上传至黑板 20：end for 21：** $A \leftarrow \cup_{i=1}^{s} A_i$**22：返回** A

高效通信子程序仅使用 $d \log k \operatorname{polylog}\left(\log n, \frac{1}{\varepsilon}, \frac{1}{\delta}\right)$ 比特，即可传输任意点 y 的近似版本 $\widetilde{y}$，而精确通信则需要$\mathcal{O}(d \log n)$ 比特:

引理 5.1. 给定一个点 y 和一个数据集 X，存在一个算法高效通信，使用 $d \log k \operatorname{polylog}(\log n, \frac{1}{\varepsilon}, \frac{1}{\delta})$ 通信比特，并发送 $\widetilde{y}$，使得 $\|y - \widetilde{y}\|_2 \leq \min_{x \in X} \varepsilon \|x - y\|_2$ 的概率至少为 $1 - \delta$。

直观地说，高效通信使站点能够以极少的通信量定位协调器的点 y 的近似版本。对于每个坐标 i，站点首先通过使用 $x_{i_s}^{(i)}$ 与 $y^{(i)}$ 之间的HighProbGreaterThan 协议进行二分搜索，找出最接近的本地值。如果 $y^{(i)}$ 不完全匹配，则通过指数搜索确定一个很小的偏移量 $\Delta y^{(i)}$，使得 $x_{i_s}^{(i)} + \Delta y^{(i)}$在一个 $y^{(i)}$ 因子范围内近似于$(1+\varepsilon)$。通过逐坐标执行这一过程，站点可以高效地重建一个接近 $\widetilde{y}$ 的点 y，并保证 $|y - \widetilde{y}|_2 \leq \varepsilon |x - y|_2$ 对任意本地点 $x \in X$都成立，同时只需发送少量比特。正如图 3所示，这种方法结合了沿每个坐标搜索“最近邻”的直觉与受控的近似细化。

算法：高效通信$(X, y, \varepsilon, \delta)$ **（非正式）**

1. **输入：**
   - 一个站点所拥有的一组点 $X = \{x_1, \ldots, x_l\}$
   - 来自协调器的一个点 y。
   - 精度参数 $\varepsilon \in$ （0,1）和失败概率 δ。

2. **目标：**向站点发送一个近似位置 $\widetilde{y}$，使得对于任意$x \in X$，以概率 $\geq 1 - \delta$ 满足 $\|y - \widetilde{y}\|_2 \leq \varepsilon \|x - y\|_2$。

3. 对于从 $i = 1$ 到 d 的每个坐标：

   按第 i 个坐标对 X 中的点进行排序。
   使用 HighProbGreaterThan 进行局部二分搜索，找到最接近 $x_{(i)\ i_s}$ 的点 $y_{(i)}$。

   如果 $y_{(i)}$ 等于 $x_{(i)\ i_s}$，则设置 $\Delta y_{(i)} = 0$。

   否则：
   - 确定方向 $\gamma =$ sign$(y_{(i)} - x_{(i)\ i_s}$。
   - 使用 HighProbGreaterThan 进行指数搜索，找到一个值 $\Delta y_{(i)}$，使得$x_{(i)\ i_s} + \Delta y_{(i)}$ 在因子 $(1 + \varepsilon)$ 范围内近似 $y_{(i)}$。
   - 设置 $\widetilde{y}_{(i)} = x_{(i)\ i_s} + \Delta y_{(i)}$。

4. **返回** $\widetilde{y} = (\widetilde{y}^{(1)}, \ldots, \widetilde{y}^{(d)})$。

图 3：消息传递算法中高效通信的非正式版本。完整算法见 算法 15。

给定通过上述自适应采样的高效实现得到的双准则近似 S，接下来我们执行敏感度采样，以构造一个（$1 + \varepsilon$）核心集。不幸的是，如果 cost(x, s') 与 cost(x, s) 非常接近，每个服务器现在可能会将点 x 分配给另一个中心 s'，而非最近的中心 s。因此，服务器可能会错误地计算敏感度采样所需的大小 $|C_j|$ 和代价 cost(C_j, S)。然而，这并不会影响正确性：[BCP+24] 的敏感度分析只要求每个点都分配给一个中心，且该分配的聚类代价与最优分配的聚类代价相差至多一个常数因子。因此，我们的流程仍能通过敏感度采样实现一个（$1 + \frac{\varepsilon}{4}$）核心集。采样点选定后，每个站点使用与黑板模型中相同的高效编码方案对其采样点的坐标进行编码，参见引理 2.13，从而确保发送回协调器的内容仅为紧凑表示。近似中心分配与高效编码相结合，既保证了准确性，也保证了通信效率。我们在图 4 中完整给出该算法，它实现了定理 1.1 的保证；完整细节见附录 C。

算法：$(1+\varepsilon)$，适用于协调器模型的核心集（非正式版）

(1) 每个站点计算一个局部（$1+\varepsilon/2$）核心集 P_i。(2) 协调器向所有站点广播一个约翰逊-林登施特劳斯变换 π。(3) 用一个随机点初始化解集 $S=\{s_0\}$。(4) 对于 $i=1$ 至 $i=\mathcal{O}(k)$ 次迭代（用于获得双准则解）：

- 使用高效通信，将中心 s_i 的近似 $\widetilde{s}_{(j)\ i-1}$ 发送到站点 j。$\sim$
- 站点计算近似代价 D_j 并发送给协调器。
- 协调器使用惰性采样，将下一个中心 s_i 选入 S。(5) 站点计算簇大小和代价，并将常数近似发送给协调器。(6) 协调器计算近似值总和并广播给各站点。(7) 站点计算近似敏感度 $\widetilde{\mu}(x)$，并将总和发送给协调器。(8) 协调器根据敏感度，在各站点间采样 $m=\tilde{\mathcal{O}}\left(\frac{k}{\varepsilon^4\min(,\varepsilon^{2+z})}\right)$ 个点；站点采样点并使用高效通信进行编码。(9) 将采样点合并为最终核心集 $A\prime$ 并返回。

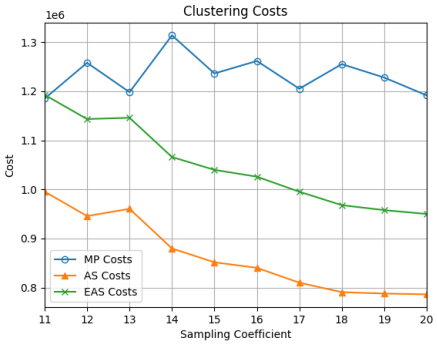
图 4：非正 式的消息传递算法。完整算法见算法16 .

6 实证评估

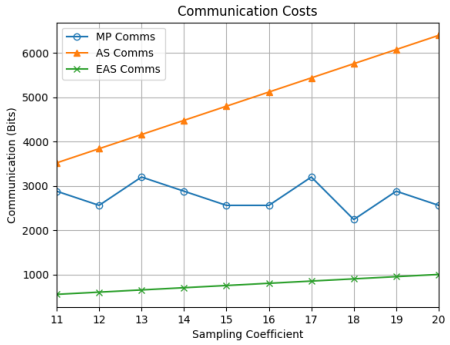
本节展示了若干针对合成数据集和真实世界数据集的简单实验结果，以补充我们的理论保证。我们考虑 k在黑板设置下进行的 -means 聚类，并以先前基于 Mettu-Plaxton 的算法 [CSWZ16] 作为基线，记为 `MP`。我们还实现了两个复杂度各异的分布式协议版本。首先，我们实现了基于自适应采样的常数因子近似算法，记为 `AS`。此外，我们还实现了自适应采样后基于紧凑编码的常数因子近似算法，记为 `EAS`。所有实验均在一台 Dell OptiPlex 7010 Tower 台式机上进行，该计算机配备 Intel Core i7-3770 3.40 GHz 四核处理器和 16 GB 内存。所有代码均可在以下位置获取：
https://github.com/samsonzhou/distributed-cluster。

6.1 真实世界数据集

为了评估我们的算法，我们在 DIGITS 数据集上进行了 k-均值聚类算法 [AK98], ，该数据集包含 1,797 张手写数字（ 0-9 ）的图像，因此与 $k=10$ 自然相关。每张图像的维度为 8×8，由 64 个特征表示，对应像素强度。该数据集可通过 `scikit-learn` 和 UCI 仓库获取。由于规模适中且类别定义明确，它是聚类任务中常用的数据集，有助于清晰地评估聚类性能。



(a) 聚类成本



(b) 通信成本

图5： DIGITS 数据集上聚类成本与通信成本的实验

对于参数 c，我们的基线算法 MP 均匀地选择 k 个初始点，然后经过 $c\log_2 n$ 轮迭代丢弃候选点。我们的 AS 算法在 ck轮迭代中，每轮迭代采样一个中心，其概率与该中心到先前采样的中心的距离成正比。最后，EAS 将所选 AS 中心的坐标量化为最接近的 2 的幂，以模拟低精度通信。我们在图5a中比较这些算法的聚类代价，并在图5b中比较其总通信成本；两者均针对 $c \in \{11$、12、.....、$20\}$。

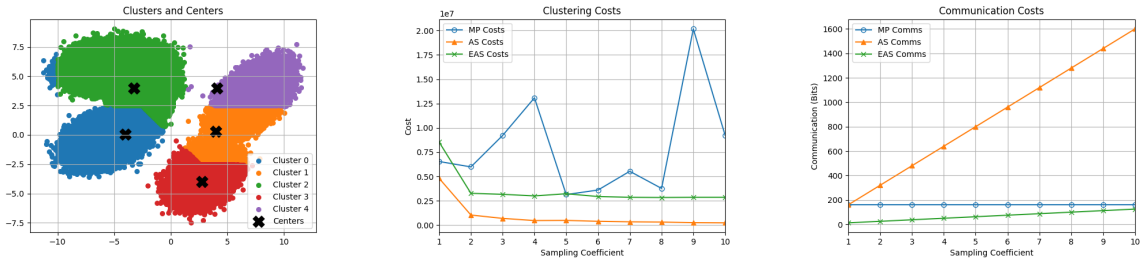
我们的结果表明，尽管自适应采样 (AS始终优于 Mettu-Plaxton (MP，但当采样数量较少时，差距相对较小，因此我们在 EAS 中采用高效编码所产生的舍入误差，使得 $c = 11$ 在图5a中的聚类成本相近。令人惊讶的是， MP 的通信成本似乎并未随 c增加，这表明所有可能的点都已被移除，无法再进行更多采样。尽管如此，对于所有$c > 11$，我们在 EAS 中的算法明显优于 MP，因此在聚类代价和通信成本两方面都优于 [CSWZ16]此前的工作，参见图5b。与我们的算法 AS相比，我们的算法 EAS 在聚类成本与通信成本之间也展现出明显的权衡，因为前者只是对后者进行舍入。

6.2 合成数据集

为便于可视化，我们生成了由二维高斯混合构成的合成数据集。选择低维度（2D）是为了能够可视化生成的簇。具体而言，我们创建了 $k = 5$ 个高斯簇，每个簇包含 $n = 100 \times 2^{10}$ 个点，总计 512,000 个数据点。每个簇均从一个不同的高斯分布中采样，其均值在范围 $[-10, 10]^2$ 内随机选取，并随机生成一个正定协方差矩阵，以确保簇的形状各不相同。

我们通过均匀采样 k 个点来实现基线算法 MP，并在 $c\log_2 n$ 轮中根据每轮翻倍的距离阈值筛除候选点，其中 c 是一个超参数。我们通过逐轮采样一个中心来实现算法 AS，共进行 ck 轮，并按照点到已有中心的距离成比例地选择点。随后， EAS 会对 AS 进行修改，将每个选定中心的坐标四舍五入到最接近的 $q = 2^{0.25}$ 次幂，以模拟低精度通信。

我们改变采样系数 $c \in \{11, \ldots, 20\}$，并测量了 k-均值聚类代价



(a) (b) (c)

图 6：合成数据集上的聚类代价和通信成本实验

（即每个点到其被分配中心的平方距离总和）。结果如图 6所示，突显了随着 c 增加，不同采样策略之间的性能权衡。我们还计算了通信代价（但未绘制），假设 MP 和 AS每个坐标占 32 比特，而 EAS 由于量化，每个坐标占 5 比特。

我们在合成数据上的结果与 DIGITS 数据集上的趋势相呼应。具体而言，对于所有 $c > 2$，我们在 EAS 中的算法在聚类代价和通信成本两方面都明显优于 MP，参见图 6c；同时，与我们的算法 AS相比，也展现了聚类代价与通信成本之间明显的权衡。我们在 EAS 中绘制了图 6a所示的聚类结果。

致谢

David P. Woodruff 部分受美国海军研究办公室资助，项目编号为 N000142112647，并获得西蒙斯研究员奖。Samson Zhou 部分受美国国家科学基金会 CCF-2335411 资助。Samson Zhou 衷心感谢橡树岭联合大学（ORAU）Ralph E. Powe 初级教职人员发展奖提供的资助。

参考文献

[ADK09] Ankit Aggarwal、Amit Deshpande 和 Ravi Kannan。k 均值聚类的自适应采样。载于《组合优化近似算法国际研讨会》，第 15–28 页。施普林格，2009年。7、11、13、26、27、28[AFN+24] Hilal Asi、Vitaly Feldman、Jelani Nelson、Huy L. Nguyen、Kunal Talwar 和 Samson Zhou。洗牌模型中的私有向量均值估计：最优速率需要大量消息。载于第四十一届国际机器学习会议，ICML，第 1945–1970 页，2024年。10[AK98] E. Alpaydin 和 C. Kaynak。手写数字光学识别。UCI 机器学习资料库，1998年。DOI：https://doi.org/10.24432/C50P49。19、50[AV07] David Arthur 和 Sergei Vassilvitskii。k-均值：谨慎选择初始种子的优势。载于《第十八届 ACM-SIAM 离散算法年度研讨会论文集》，SODA，第 1027–1035 页，2007年。11、13

[BCJ+22] Vladimir Braverman、Vincent Cohen-Addad、Shaofeng H.-C. Jiang、Robert Krauthgamer、Chris Schwiegelshohn、Mads Bech Toftrup 和 Xuan Wu。均匀采样对核心集的作用。载于第63届 IEEE 计算机科学基础年会，FOCS，第 462–473 页，2022年。2[BCP+24] Nikhil Bansal、Vincent Cohen-Addad、Milind Prabhu、David Saulpic 和 Chris Schwiegelshohn。k-均值的敏感度采样：最坏情况与稳定性最优核数据集界。CoRR，abs/2405.01339，2024年。3、4、6、10、12、13、16、18、41[BDJ+25] Vladimir Braverman、Prathamesh Dharangutte、Shaofeng H.-C. Jiang、Hoai-An Nguyen、Chen Wang、Yubo Zhang 和 Samson Zhou。弱强预言机模型中的相对误差公平聚类。载于第四十二届国际机器学习会议，ICML，2025年。2[BEL13] Maria-Florina Balcan、Steven Ehrlich 和 Yingyu Liang。一般通信拓扑结构上的分布式 k-均值和 k-中位数聚类。载于《神经信息处理系统进展 26：第27届神经信息处理系统年度会议》，第 1995–2003 页，2013年。3、11、12、13、49、50[BFL+21] Vladimir Braverman、Dan Feldman、Harry Lang、Adiel Statman 和 Samson Zhou。通过敏感度采样实现高效的核心集构造。载于亚洲机器学习会议，ACML，第 948–963 页，2021年。2[BLST23] Maria Sofia Bucarelli、Matilde Fjeldsø Larsen、Chris Schwiegelshohn 和 Mads Toftrup。关于投影聚类的泛化界。载于《神经信息处理系统进展 36：神经信息处理系统年度会议》，NeurIPS，2023年。2[BMV+12] Bahman Bahmani、Benjamin Moseley、Andrea Vattani、Ravi Kumar 和 Sergei Vassilvitskii。可扩展的 k-均值。VLDB基金会会议录，5(7)：622–633，2012年。11、13++CAWWZ[Vincent Cohen-Addad、Liudeng Wang、David P. Woodruff 和 Samson Zhou。用于聚类和子空间嵌入的快速、空间最优流式算法。CoRR，abs/2504.16229，2025年。925] CCFM[Moses Charikar、Chandra Chekuri、Tomás Feder 和 Rajeev Motwani。增量聚类与动态信息检索。载于《第二十九届 ACM 计算理论研讨会年度论文集》，第 626–635 页，1997年。297] CG[Moses Charikar 和 Sudipto Guha。设施选址和 k-中位数问题的改进组合算法。载于《第40届计算机科学基础年会》，FOCS，第 378–388 页，1999年。299] CGTS[Moses Charikar、Sudipto Guha、Éva Tardos 和 David B. Shmoys。k-中位数问题的常数因子近似算法。载于《第三十一届 ACM 计算理论研讨会年度论文集》，第 1–10 页，1999年。299] CJY kVincent Cohen-Addad、Shaofeng H.-C. Jiang、Qiaoyuan Yang、Yubo Zhang 和 Samson Zhou。滑动窗口模型中的公平聚类。载于第十三届国际学习表征会议，ICLR，2025年。2

[CKM16] Vincent Cohen-Addad、Philip N. Klein 和 Claire Mathieu。局部搜索为欧氏度量和无小子图度量中的 k-均值与 k-中位数提供了近似方案。载于《IEEE第57届计算机科学基础年会》（FOCS），第353–364页，2016年。2[CLN+20] Vincent Cohen-Addad、Silvio Lattanzi、Ashkan Norouzi-Fard、Christian Sohler 和 Ola Svensson。通过拒绝采样实现快速且精确的 k-均值。载于《神经信息处理系统进展33：神经信息处理系统年会》（NeurIPS），2020年。2[CLS+22] Vincent Cohen-Addad、Kasper Green Larsen、David Saulpic、Chris Schwiegelshohn 和 Omar Ali Sheikh-Omar。改进的欧几里得 k-均值核心集。载于 NeurIPS，2022年。2, 5, 10[CLS25] Vincent Cohen-Addad、Silvio Lattanzi 和 Chris Schwiegelshohn。近乎最优的 k-均值 PAC 学习。载于《第57届ACM计算理论年会论文集》（STOC），第2019–2030页。ACM，2025年。2[CLSS22] Vincent Cohen-Addad、Kasper Green Larsen、David Saulpic 和 Chris Schwiegelshohn。迈向 k-中位数与 k-均值核心集的最优下界。载于《STOC '22：第54届ACM SIGACT计算理论年会》，第1038–1051页，2022年。2, 3, 4, 5, 10[COP03] Moses Charikar、Liadan O'Callaghan 和 Rina Panigrahy。更优的聚类问题流式算法。载于《第35届ACM计算理论研讨会论文集》，第30–39页，2003年。2[CS17] Vincent Cohen-Addad 和 Chris Schwiegelshohn。稳定聚类实例的局部结构。载于《第58届IEEE计算机科学基础年会》（FOCS），第49–60页，2017年。2[CSS21] Vincent Cohen-Addad、David Saulpic 和 Chris Schwiegelshohn。一种新的聚类核心集框架。载于《STOC：第53届ACM SIGACT计算理论年会》，第169–182页，2021年。2, 3, 4, 10[CSS23] Vincent Cohen-Addad、David Saulpic 和 Chris Schwiegelshohn。高维空间中的确定性聚类：草图与近似。载于《第64届IEEE计算机科学基础年会》（FOCS），第1105–1130页，2023年。2[CSWZ16] Jiecao Chen、He Sun、David P. Woodruff 和 Qin Zhang。通信最优的分布式聚类。载于《神经信息处理系统进展29：神经信息处理系统年会》，第3720–3728页，2016年。2, 3, 4, 10, 13, 19, 20[CWZ23] Vincent Cohen-Addad、David P. Woodruff 和 Samson Zhou。使用 $o(\log n)$ 空间进行流式欧几里得k-中位数与 k-均值。载于《第64届IEEE计算机科学基础年会》（FOCS），第883–908页，2023年。10

[EFS+22] Jon C. Ergun、Zhili Feng、Sandeep Silwal、David P. Woodruff 和 Samson Zhou。学习增强的 k-均值聚类。载于第十届国际学习表征会议（ICLR），2022年。2[FL11] Dan Feldman 和 Michael Langberg。用于数据近似与聚类的统一框架。载于《第43届ACM计算理论研讨会论文集》（STOC），第569–578页，2011年。2[FMS07] Dan Feldman、Morteza Monemizadeh 和 Christian Sohler。一种基于弱核心集的 k-均值聚类多项式时间近似方案。载于《第23届ACM计算几何研讨会论文集》，第11–18页，2007年。2[FMSW10] Dan Feldman、Morteza Monemizadeh、Christian Sohler 和 David P. Woodruff。用于高维子空间近似问题的核心集与草图。载于《第二十一届ACM-SIAM离散算法研讨会论文集》（SODA），第630–649页，2010年。2[FSS13] Dan Feldman、Melanie Schmidt 和 Christian Sohler。将大数据转化为小数据：用于 k-均值、主成分分析和投影聚类的常数规模核心集。载于《第二十四届ACM-SIAM离散算法研讨会论文集》（SODA），第1434–1453页，2013年。2[HLW24] Lingxiao Huang、李建和 Xuan Wu。关于欧几里得 (k, z)-聚类的最优核心集构造。载于《第56届ACM计算理论研讨会论文集》（STOC），第1594–1604页，2024年。2, 3, 4, 5[HV20] Lingxiao Huang 和 Nisheeth K. Vishnoi。欧氏空间聚类的核心集：重要性采样近乎最优。载于《第52届ACM SIGACT计算理论年会论文集》（STOC），第1416–1429页，2020年。10[ISZ21] Zachary Izzo、Sandeep Silwal 和 Samson Zhou。Wasserstein重心的降维。载于《神经信息处理系统进展34：神经信息处理系统年会》（NeurIPS），2021年。6, 10[JL84] William B Johnson 和 Joram Lindenstrauss。将利普希茨映射扩展到希尔伯特空间。载于《现代分析与概率会议》（康涅狄格州纽黑文，1982年），第189–206页。《当代数学》，第26卷，1984年。6[KLR+25] Karthik C. S.、Euiwoong Lee、Yuval Rabani、Chris Schwiegelshohn 和 Samson Zhou。关于 ℓ_2^2 最小和聚类的可近似性。载于《第41届国际计算几何研讨会》（SoCG），第62:1–62:18页，2025年。2[KN10] Daniel M Kane 和 Jelani Nelson。一种去随机化的稀疏 Johnson-Lindenstrauss 变换。arXiv预印本 arXiv:1006.3585，2010年。7[LV10] Yurii Lyubarskii 和 Roman Vershynin。不确定性原理与矢量量化。《IEEE信息论汇刊》，56(7):3491–3501，2010年。10[Mac67] J MacQueen。多元观测的分类与分析。载于《第五届伯克利数学统计与概率研讨会》，第281–297页，1967年。1

[Mey01] Adam Meyerson。在线设施选址。载于第42届计算机科学基础年会（FOCS），第426–431页，2001年。2[MMR19] Konstantin Makarychev、Yury Makarychev 和 Ilya P. Razenshteyn。Johnson-Lindenstrauss变换在 k-均值和 k-中位数聚类中的性能。载于第51届ACM SIGACT计算理论年会（STOC）论文集，第1027–1038页，2019年。6, 10[Mor78] Robert Morris。用小寄存器计数大量事件。ACM通讯，21(10):840–842，1978年。16, 38[MP04] Ramgopal R. Mettu 和 C. Greg Plaxton。近似聚类的最优时间界。机器学习，56(1-3):35–60，2004年。10, 11, 13[Nis93] Noam Nisan。阈值门的通信复杂度。《组合数学：保罗·埃尔德什八十岁》，1(301-315):6，1993年。6, 42[S+56] Hugo Steinhaus 等。论物质体的分割。波兰科学院通报，1(804):801，1956年。1[SW18] Christian Sohler 和 David P. Woodruff。k-中位数和子空间近似的强核心集：告别维度。载于第59届IEEE计算机科学基础年会（FOCS），第802–813页，2018年。5, 10[TWZ+22] Murad Tukan、Xuan Wu、Samson Zhou、Vladimir Braverman 和 Dan Feldman。用于投影聚类及其应用的新核心集。载于国际人工智能与统计会议（ AISTATS ）论文集，第5391–5415页，2022年。2[WZZ23] David P. Woodruff、Peilin Zhong 和 Samson Zhou。滑动窗口模型中的近最优 k-聚类。载于神经信息处理系统进展第36卷：神经信息处理系统年度会议（NeurIPS 2023），2023年。2[ZTHH25] Xiaoyi Zhu、Yuxiang Tian、Lingxiao Huang 和 Zengfeng Huang。欧几里得聚类的空间复杂度。IEEE信息论汇刊，71(6):4515–4536，2025年。10

面向(k, z)-聚类的自适应采样

我们首先从高层概述自适应采样为何能返回双准则近似。设$\{A_j\}_{j=1}^{k}$ 为对应于某个最优（k, z）-聚类的簇。若由 S 诱导的每个簇 A_j 的代价已经是代价(A_j, C_{OPT})的$\mathcal{O}$(1)-近似，其中 C_{OPT} 是最优解的中心集合，则我们可以达到最优代价的$\mathcal{O}$(1)-近似。若簇 A_j 的代价是代价(A_j, C_{OPT})的$\mathcal{O}$(1)-近似，则我们将其定义为好簇，否则称 A_j 为坏簇。我们可以证明，以常数概率，自适应采样会采样到一个能将坏簇转化为好簇的中心。然后，由马尔可夫不等式可知，我们可以在$\mathcal{O}(k)$个采样轮数内消除所有坏簇，从而得到常数因子近似。

我们使用以下定义来描述好簇和坏簇。

定义 A.1. 令 S_i 为 S 中第 i 轮中采样得到的集合 算法 2，并令 C_{OPT} ={c_1, c_2, $\cdots$ c_k} 为 (k, z)-聚类的一个最优解。

对于 $j \in [k]$，令 A_j 为 X 中被分配给最优聚类中 c_j 的点，平局时任意打破，即 A_j ={$x \in X$: 距离 (x, c_j) $\leq$ 距离 (x, c_ℓ) $\forall \ell \in [k]$}。我们定义

$$\boldsymbol{Good}_i = \{A_j : \text{cost}(A_j, S_i) \leq \gamma_z \cdot \text{cost}(A_j, C_{OPT})\}$$
$$\boldsymbol{Bad}_i = \{A_1, A_2, \cdots, A_k\} \setminus \boldsymbol{Good}_i$$

其中 $\gamma_z = 2 + (3 + 6z)^z$。

根据定义，如果每个簇都是好簇，我们就可以保证 S是一个（$\mathcal{O}(1)$，$\mathcal{O}(1)$）双准则近似。我们将证明， S要么已经是一个（$\mathcal{O}(1)$，$\mathcal{O}(1)$）双准则近似，要么我们会以某个常数概率在每次采样后减少坏簇的数量，这意味着采样 $\mathcal{O}(k)$ 个点后，我们可以得到一个 $\mathcal{O}(1)$ 近似。

引理 A.2。 假设 代价$(X, S_i) > 2\gamma^2 \cdot \gamma_z$代价$(X, C_{OPT})$，那么

$$\mathbf{Pr}\left[|\boldsymbol{Bad}_{i+1}| < |\boldsymbol{Bad}_i|\right] \geq \delta$$

对于某个常数 $\delta > 0$。

引理 A.2 是 [ADK09]中引理 5 的推广。我们将通过把证明拆分为若干引理来证明引理 A.2 。我们首先考虑广义三角不等式的一种特定参数化形式。

引理 A.3。对于任意 x， y， $\mu \in \mathbb{R}^d$,

$$dist(x, y)^z \leq 2 \cdot dist(x, \mu)^z + (1 + 2z)^z \cdot dist(y, \mu)^z.$$

证明。该结论可由应用事实 2.2 并取 $\varepsilon = 1$得出，因此根据三角不等式，dist$(x, y)^z \leq$(dist(x, μ) + dist$(y, \mu))^z \leq 2 \cdot$ dist$(x, \mu)^z + (1 + 2z)^z \cdot$ dist$(y, \mu)^z$。 □

我们首先证明，在每一轮中，我们以常数概率从一个坏簇中采样一个点。该结论类似于 [ADK09]中的引理 1。

引理 A.4。 在我们的算法的第 i 轮中，要么 代价$(X, S_i) \leq 2\gamma^2 \cdot \gamma_z$代价$(X, C_{OPT})$，要么从坏簇$_i$ 中的某个簇采样一个点的概率至少为$_{1\ 2}$。

证明。 假设代价$(X,\ S_{i-1}) > 2\gamma^2 \cdot \gamma_z$代价$(X,\ C_{OPT})$。那么，从某个坏簇中选取 x 的概率为

$$\mathbf{Pr}\left[x \in \mathbf{Bad}_i\right] = \frac{\sum_{A_j \in \mathbf{Bad}_i} \sum_{x_l \in A_j} d_l}{\sum_{x_l \in X} d_l} = 1 - \frac{\sum_{A_j \in \mathbf{Good}_i} \sum_{x_l \in A_j} d_l}{\sum_{x_l \in X} d_l}.$$

在此，我们指出 d_l 是从 x_l 到 S_{i-1} 的距离，如算法 2中所定义。由于代价$(A_j,\ S_{i-1}) \leq \gamma_z$代价$(A_j,$ $C_{\text{OPT}})$ 如果 $A_j \in$好$_i$，并且代价$(X,\ S_{i-1}) > 2\gamma^2 \cdot \gamma_z$代价$(X,\ C_{OPT})$根据我们所持的条件，

$$\mathbf{Pr}\left[x \in \mathbf{Bad}_i\right] \geq 1 - \frac{\gamma^2 \cdot \gamma_z \sum_{A_j \in \mathbf{Good}_i} \text{cost}(A_j, C_{\text{OPT}})}{2\gamma^2 \cdot \gamma_z \,\text{cost}(X, C_{\text{OPT}})} \geq 1 - \frac{1}{2} = \frac{1}{2}.$$

□

考虑一个固定坏簇 $A \in$ **坏**$_i$。设 $m = w(A) = \sum_{x \in A} w(x)$，其中 $w(x)$是 x的权重，并定义 $r = \left(\text{代价}(A, C_{\text{OPT}}) / m \right)^{1/z}$。将 μ 记为 C_{OPT} 中对应于 A的中心。设 y为 S_{i-1}中最接近 μ 的点。我们将证明 y （从而 S_{i-1}中的每个采样点）远离 μ。

引理 A.5. 距离$(y, \mu) \geq 3r$。

证明。 由于 A 是一个固定坏簇,

$$\gamma_z \operatorname{cost}(A, C_{\text{OPT}}) \leq \operatorname{cost}(A, S_{i-1}) = \sum_{x \in A} w(x) \min_{c \in S_{i-1}} \operatorname{dist}(x, c)^z \leq \sum_{x \in A} w(x) \operatorname{dist}(x, y)^z.$$

根据引理 A.3,

$$\begin{aligned} \sum_{x \in A} w(x) \operatorname{dist}(x, y)^z &\leq \sum_{x \in A} \left(2 \cdot w(x) \cdot \operatorname{dist}(x, \mu)^z + (1 + 2z)^z \cdot w(x) \cdot \operatorname{dist}(y, \mu)^z \right) \\ &= 2 \operatorname{cost}(A, C_{\text{OPT}}) + (1 + 2z)^z \cdot m \cdot \operatorname{dist}(y, \mu)^z. \end{aligned}$$

因此

$$(1 + 2z)^z \cdot m \cdot \operatorname{dist}(y, \mu)^z \geq (\gamma_z - 2) \operatorname{cost}(A, C_{\text{OPT}}).$$

由于 $\gamma_z = 3^z(1 + 2z)^z + 2$,

$$3r = \left(\frac{\gamma_z - 2}{(1 + 2z)^z} \cdot \frac{\operatorname{cost}(A, C_{\text{OPT}})}{m} \right)^{1/z} \leq \operatorname{dist}(y, \mu).$$

□

定义 $B(\alpha) = \{x \in A :$ 距离$(x, \mu) \leq \alpha r\}$ 为固定坏簇 A 中与 μ的距离不超过 αr 的点。我们将证明，只要采样一个 A 靠近中心 x 的点 μ，坏簇就可以转化为好簇。此结论类似于 [ADK09]中的引理 2。

引理 A.6。 令 A 为由 C_{OPT} 定义的任意固定坏簇，并令 $b \in B$（α），对于 $0 \leq \alpha \leq 3$。则

$$\operatorname{cost}(A, S_{i-1} \cup \{b\}) \leq \gamma_z \cdot \operatorname{cost}(A, C_{OPT}).$$

证明。

$$\operatorname{cost}(A, S_{i-1} \cup \{b\}) = \sum_{x \in A} w(x) \min_{c \in S_{i-1} \cup \{b\}} \operatorname{dist}(x, c)^z \leq \sum_{x \in A} w(x) \cdot \operatorname{dist}(x, b)^z.$$

根据引理 A.3,

$$\sum_{x \in A} w(x) \cdot \operatorname{dist}(x, b)^z \leq \sum_{x \in A} \left(2 \cdot w(x) \cdot \operatorname{dist}(x, \mu)^z + (1 + 2z)^z \cdot w(x) \cdot \operatorname{dist}(b, \mu)^z \right).$$

由于 $b \in B(\alpha)$ 和 $0 \leq \alpha \leq 3$, 距离$(b, \mu) \leq \alpha r \leq 3r$。因此

$$\begin{aligned} \operatorname{cost}(A, S_{i-1} \cup \{b\}) &\leq 2 \operatorname{cost}(A, C_{\text{OPT}}) + m(1 + 2z)^z (3r)^z \\ &= 2 \operatorname{cost}(A, C_{\text{OPT}}) + 3^z (1 + 2z)^z \operatorname{cost}(A, C_{\text{OPT}}) \\ &= \gamma_z \operatorname{cost}(A, C_{\text{OPT}}). \end{aligned}$$

□

接下来我们证明，固定坏簇中各点的大部分权重 A 落入 $B(\alpha)$。回顾一下，我们定义了 $m = w(A) = \sum_{x\in A} w(x)$。令 $w(B(\alpha)) = \sum_{x\in B(\alpha)} w(x)$。这一结论与 [ADK09] 中的引理 3 类似。

引理 A.7。

$$w(B(\alpha)) \geq m\left(1 - \frac{1}{\alpha^z}\right), \text{ for } 1 \leq \alpha \leq 3.$$

证明。

$$\text{cost}(A, C_{\text{OPT}}) \geq \text{cost}(A\backslash B(\alpha), C_{\text{OPT}}) = \sum_{x\in A\backslash B(\alpha)} w(x) \min_{c\in C_{\text{OPT}}} \text{dist}(x,c)^z.$$

由于对于任意 $x \in A$，μ 都是 C_{OPT} 中 x 的最近中心，因此

$$\sum_{x\in A\backslash B(\alpha)} w(x) \min_{c\in C_{\text{OPT}}} \text{dist}(x,c)^z = \sum_{x\in A\backslash B(\alpha)} w(x)\text{dist}(x,\mu)^z.$$

由于对于任意 $x \in A\backslash B(\alpha)$，距离$(x, \mu) \geq \alpha r$，因此

$$\sum_{x\in A\backslash B(\alpha)} w(x)\text{dist}(x,\mu)^z \geq w(A\backslash B(\alpha)) \cdot (\alpha r)^z = \left(1 - \frac{w(B(\alpha))}{m}\right) m(\alpha r)^z.$$

由于 $r = \left(\frac{\text{代价}(A, C_{\text{OPT}})}{m}\right)^{1/z}$，

$$\text{cost}(A, C_{\text{OPT}}) \geq \left(1 - \frac{w(B(\alpha))}{m}\right) m(\alpha r)^z = \left(1 - \frac{w(B(\alpha))}{m}\right) \alpha^z \,\text{cost}(A, C_{\text{OPT}}).$$

因此，

$$w(B(\alpha)) \geq m\left(1 - \frac{1}{\alpha^z}\right).$$

□

接下来我们将证明，$B(\alpha)$ 中的点的成本至少是坏簇 A 的代价的一个常数比例，因此我们将以常数概率采样到靠近 μ 的一个点，如果我们从 A中采样一个点。该陈述类似于 [ADK09]中的引理 4。

引理 A.8。

$$\mathbf{Pr}\left[x \in B(\alpha) | x \in A \text{ and } A \in \boldsymbol{Bad}_i\right] \geq \frac{1}{\gamma^2}\left(1 - \frac{1}{\alpha^z}\right)\frac{(3-\alpha)^z}{\gamma_z}.$$

证明。 回顾可知，y 是 S_{i-1}中最接近 μ 的点。我们有

$$\text{cost}(A, S_{i-1}) = \sum_{x\in A} w(x) \min_{c\in S_{i-1}} \text{dist}(x,c)^z \leq \sum_{x\in A} w(x)\text{dist}(x,y)^z.$$

令 $d =$ 为距离(y, μ)。由引理 A.3,

$$\text{cost}(A, S_{i-1}) \leq \sum_{x\in A}\left(2w(x)\text{dist}(x,\mu)^z + (1+2z)^z w(x)\text{dist}(y,\mu)^z\right) = 2\,\text{cost}(A, C_{\text{OPT}}) + m(1+2z)^z d^z.$$

由于 $r = \left(\frac{\text{代价}(A,\ C_{\text{OPT}})}{m}\right)^{1/z}$，因此代价$(A,\ S_{i-1}) \leq 2mr^z + m(1+2z)^z d^z$。

另一方面，代价$(B(\alpha)\ S_{i-1}) = \sum_{x\in B(\alpha)}\ w(x)\ \min_{c\in S_{i-1}}$距离$(x,c)^z$。根据三角不等式，距离$(x,c)\ \geq$距离$(c,\mu)\ -$距离$(x,\mu)\ \geq$距离$(y,\mu)\ -$距离$(x,\mu)$。由于 $x \in A$且 A是一个坏簇，因此距离$(y,\mu)\ -$距离$(x,\mu)\ \geq 0$，于是

$$\text{cost}(B(\alpha), S_{i-1}) \geq \sum_{x\in B(\alpha)} w(x)(\text{dist}(c,\mu) - \text{dist}(x,\mu))^z \geq \sum_{x\in B(\alpha)} w(x)(\text{dist}(y,\mu) - \text{dist}(x,\mu))^z.$$

对于 $x \in B(\alpha)$，我们有距离$(x,\mu)\ \leq \alpha r$，因此代价$(B(\alpha)\ S_{i-1}) \geq \sum_{x\in B(\alpha)}\ w(x)$(距离$(y,\mu)\ - \alpha r)^z$。根据引理 A.7，$w(B(\alpha))\ \geq m\left(1 - \frac{1}{\alpha^z}\right)$，因此代价$(B(\alpha)\ S_{i-1})\ \geq m\left(1 - \frac{1}{\alpha^z}\right)$(距离$(y,\mu)\ - \alpha r)^z$。

由于 $\frac{1}{\gamma}\cdot\ d_i \leq$代价$(x_i, S)\ \leq \gamma\cdot\ d_i$，参见 算法 2，因此

$$\begin{aligned}\mathbf{Pr}\left[x \in B(\alpha) | x \in A \text{ and } A \in \mathbf{Bad}_i\right] &= \frac{\sum_{x_i\in B(\alpha)} d_i}{\sum_{x_i\in A} d_i} \geq \frac{\frac{1}{\gamma}\,\text{cost}(B(\alpha), S_{i-1})}{\gamma\,\text{cost}(A, S_{i-1})} \\ &\geq \frac{m}{\gamma^2}\left(1 - \frac{1}{\alpha^z}\right)\cdot \frac{(\text{dist}(y,\mu) - \alpha r)^z}{m(2r^z + (1+2z)^z \text{dist}(y,\mu)^z)}.\end{aligned}$$

通过计算导数，我们可以观察到

$$\frac{(\text{dist}(y,\mu) - \alpha r)^z}{2r^z + m(1+2z)^z \text{dist}(y,\mu)^z}$$

是距离(y,μ)的递增函数，当距离$(y,\mu)\ \geq 3r \geq \alpha r$时。因此

$$\begin{aligned}\mathbf{Pr}\left[x \in B(\alpha) | x \in A \text{ and } A \in \mathbf{Bad}_i\right] &\geq \frac{m}{\gamma^2}\left(1 - \frac{1}{\alpha^z}\right)\cdot \frac{(3r - \alpha r)^z}{m(2r^z + (1+2z)^z (3r)^z)} \\ &= \frac{1}{\gamma^2}\left(1 - \frac{1}{\alpha^z}\right)\frac{(3-\alpha)^z}{\gamma_z}.\end{aligned}$$

□

因此，如果我们在 $B(\alpha) \subset A$中为 $a \leq 3$采样一个点，我们将把一个坏簇 A 转化为一个好簇，并且我们将以常数概率在 A 中采样到这样的点，这意味着我们将以常数概率减少坏簇的数量。

引理 A.9。 假设我们的算法在第 i 轮选取的点 x 来自 $A \in$ 坏$_i$ 和$S_i = S_{i-1} \cup \{x\}$。那么

$$\mathbf{Pr}\left[\text{cost}(A, S_i) \leq \gamma_z \cdot \text{cost}(A, C_{OPT}) | x \in A \ \textit{and}\ A \in \boldsymbol{Bad}_i)\right] \geq \delta,$$

其中

$$\delta = \max \frac{1}{\gamma^2}\left(1 - \frac{1}{\alpha^z}\right)\frac{(3-\alpha)^z}{\gamma_z}.$$

证明。 该结论由 引理 A.6 和引理 A.8。 □

如果代价$(X,\ S_i)\ > 2\gamma^2\cdot\ \gamma_z$代价$(X,\ C_{OPT})$，我们将以常数概率从一个坏簇中采样一个点，并且从坏簇中采样的点将以概率将其转变为一个好簇。因此，我们将以概率减少坏簇的数量。

我们已经证明，我们将以常数概率从一个坏簇中采样一个点。我们还证明了，从坏簇中采样的中心将以常数概率将该坏簇转变为一个好簇。这两个事实使我们得出结论：对于每次采样，我们都将以常数概率把一个坏簇转变为一个好簇，这正是 引理 A.2所声称的。

引理 A.2. 假设 代价$(X, S_i) > 2\gamma^2 \cdot \gamma_z$代价$(X, C_{OPT})$，那么

$$\mathbf{Pr}\left[|\boldsymbol{Bad}_{i+1}| < |\boldsymbol{Bad}_i|\right] \geq \delta$$

其中 $\delta > 0$ 为某个常数。

证明。若代价$(X, S_i) > 2\gamma^2 \cdot \gamma_z$代价$(X, C_{OPT})$，根据引理 A.4，我们以至少 1 2 的概率从某个坏簇中采样一个点 x。根据引理 A.9，若 $x \in A$ 来自某个坏簇，则以至少 δ 的概率有代价$(A, S_i) \leq \gamma_z \cdot$ 代价(A, C_{OPT})。因此， A 以至少 $\frac{\delta}{2}$ 的概率会在 Good$_{i+1}$ 中成为一个好簇。所以，Pr $[|\mathrm{Bad}_{i+1}| < |\mathrm{Bad}_i|] \geq \frac{\delta}{2}$。 □

现在，我们证明对于一般的 $z \geq 1$，自适应采样的正确性。

定理 2.10. 存在一个算法，参见 算法 *2* ，它输出一个集合 S ，其中包含 $\mathcal{O}(k)$ 个点，使得其概率至少为 0.99， 代价$(S, X) \leq \mathcal{O}(1) \cdot$ 代价(C_{OPT}, X)，其中 C_{OPT} 是一个最优的(k, z)-聚类 X。

证明。根据引理 A.3，除非 cost$(X, S_i) \leq 2\gamma^2 \cdot \gamma_z$cost$(X, C_{OPT})$，否则每轮中坏簇的数量都会以常数概率减少。因此，根据马尔可夫不等式，以 0.99 的概率，经过 O (k) 轮后，坏簇的数量将减少至 0，除非 cost$(X, S_i) \leq 2\gamma^2 \cdot \gamma_z$cost$(X, C_{OPT})$。因此，要么不存在坏簇，要么 cost$(X, S_i) \leq 2\gamma^2 \cdot \gamma_z$cost$(X, C_{OPT})$。这两种情况都意味着，对于所有 $j \in [k]$，我们都有 cost$(A_j, S) \leq 2\gamma^2 \cdot \gamma_z \cdot$ cost(A_j, C_{OPT})，因此 S 成为最优 (k, z)- 聚类的常数近似。 □

B 黑板通信模型

B.1 黑板模型中的双准则近似

我们首先给出黑板模型中 (k, z)-聚类的初始化算法，见 算法 7，
它生成一个点 x ，并将每个站点的总代价 2-近似值 D_i 上传到黑板上，代价为 D_i 。

初始化后，我们使用惰性采样采样 $\mathcal{O}(k)$ 个点，从而得到 $(k,\ z)$-聚类的 $(\mathcal{O}(1)$ ，$\mathcal{O}(1))$- 双准则近似。

我们首先证明 算法 8 以 $(\mathcal{O}(1)$ ，$\mathcal{O}(1))$-双准则近似返回该 $(k,\ z)$-聚类，所需通信比特为 $\mathcal{O}(sk1$ og log $n + kd\log\ n)$ ，通信轮数为 $\mathcal{O}(k)$ 。

~引理 *B.1*。算法 *8* 输出一个大小为 *O (k)* 的集合 S，使用 *O* $(sk + kd\log\ n)$ 比特的通信和 *O (k)* 轮通信；此时，概率至少为 *0.98*，有 *cost*$(S, X) \leq$*O (1)* $\cdot$ *cost*(C_{OPT}, X)，其中 C_{OPT} 是 X 的最优 (k, z)-聚类。

证明。 我们证明该算法的正确性，即由 S 得到的聚类代价是最优 (k, z)-聚类代价的常数因子近似。我们还给出 S 中点的数量上界。

算法 7 初始化

输入： 数据集 X_i 分配给每个站点 $i \in [s]$

输出： 一个集合 S 包含一个采样点 x，近似代价 $\widetilde{D_i}$ 对黑板上的每个站点

协调器采样一个点 x 并将其上传到黑板上，$S \leftarrow \{x\}$
对 $i \leftarrow 1$ 到 s **执行**
  $r_i \leftarrow$ PowerApprox(代价(X_i, S), 2)
  将 r_i 写在黑板上
结束循环
$\widetilde{D_i} \leftarrow 2^{r_i}$
返回 S , $\widetilde{D_i}$

算法 8 针对 $k = \mathcal{O}(\log n)$ 的懒惰自适应 (k, z)-采样

输入： 每个站点 i 所拥有的数据集 X_i，$i \in [s]$

输出： 一个集合 S，它是 (k, z)-聚类的 ($\mathcal{O}(1)$, $\mathcal{O}(1)$) 双准则近似

运行初始化，得到 S 和 $\widetilde{D_i}$
$N \leftarrow \mathcal{O}(k)$
对于 $i \leftarrow 1$ 到 N，**执行**
  $x \leftarrow$ 惰性采样({代价(x, S)}，{ $\widetilde{D_j}$})，并将 x 上传到黑板
  $S \leftarrow S \cup \{x\}$
  每个站点 j 计算 $D_j =$ 代价(X_j, S)
  $r_j \leftarrow$ PowerApprox(D_j, 2)，每个站点 j 在黑板上更新 r_j （如果 $r_j \neq \log 2$） $\widetilde{D_j}$
  $\widetilde{D_j} \leftarrow 2^{r_j}$
结束循环
返回 S

$\widetilde{}$双准则近似保证。由于我们在每一轮都会更新 D_j，因此每次在算法中运行 LazySampling({cost(x, S)}, {D_j}) 时，都满足条件 $\sim$ $\sim D_j \leq D_j < 2D_i$。因此，根据引理 2.11，如果 LazySampling 未返回 $\perp$，我们以概率 cost(x,S) cost(x,S) 采样到一个点 x；经过 $N =$ O (k) 轮惰性采样后，我们以至少 0.99 的概率获得 O (k) 个非 $\perp$ 点。因此，我们将获得 O (k) 个采样点，它们满足 AdaptiveSampling（算法 2）中自适应采样所需的条件。因此，根据引理 A.2，我们以某个常数概率 p 在每一轮减少 Bad$_i$ 的规模。由定理 2.10 可知，算法 8 以至少 0.99 的概率为 (k, z)-聚类生成一个常数近似。

算法 8 的通信复杂度。初始化需要 O ($s$log log $n + d$log n) 个通信比特。上传采样点 x 的位置需要 O ($d$log n) 个比特，因为需要上传 d 个坐标，且每个坐标需要 O (log n) 个比特。每个站点需要 O (1) 个比特将 $\perp$ 返回给协调器。然后，所有 s 个站点都需要将 r_i 写入黑板。由于 $r_i =$ O (log n)，且编码 r_i 需要 O (log log n) 个比特，因此通信量为 O ($s$log log n) 个比特。由于我们将重复迭代 O (k) 次，总通信成本为 O (sk log log $n + kd$log n) 个比特。最后需要指出，每次迭代中，我们将在 $N =$ O (k) 轮采样中使用两轮通信，因此通信总轮数为 O (k)。

□

B.2 L_1 采样子程序

在本节中，我们介绍一种 L_1 采样算法，见算法 9。该 L_1 采样算法的主要目的是最终改进自适应采样方法的通信复杂度和轮次复杂度，使其适用于 $k = \Omega(\log n)$，参见附录 B.3。

采样算法L1Sampling的良好近似。检测 $\mu^2 D \le \widetilde{D}$时，这意味着 $\widetilde{D}$ 远非良好近似，L1Sampling 将返回 True，以通知我们更新 D的值。另一方面，当 $\mu D > D$时，这意味着 D 对 D的近似足够好，L1Sampling 将返回 False，因此我们可以避免不必要的更新，从而节省通信成本。

算法 9 L_1-采样：L1Sampling($\{D_j\}$, $\{D_j\}$ μ, δ)

输入： $D_i = \text{cost}(X_i, S)$，站点 i中各点的成本；总成本 $D = \sum_{i=1}^{s} D_i$；$\{D_i\}_{i=1}^{s}$和 D written on blackboard, which 是近似 imations to $\{D_i\}^s$ d D; $\mu > 1$, the distorti **关于参数；** $\delta > 0$, **失败概率 输出：用于指示** $\widetilde{D}$ **是否为** μ^2**-近似**D**1：** $N \leftarrow \mathbf{O}\left(\log \frac{1}{\delta}\right)$, $T \leftarrow 0$, $\lambda \leftarrow \frac{\mu}{4}$, $\alpha \leftarrow 2\mu$**2：对于** $i \leftarrow 1$ **到** N **执行**3：**以概率** $\Pr[j] = \frac{D_j}{\widetilde{D}}$ **采样一个站点** j，**在黑板上更新** $r_j \leftarrow$ **PowerApprox(**D_j, λ**)**4 $T \leftarrow T + \lambda^{r_j}$：**Pr**·：**结束循环**6：**如果** $\alpha \cdot T \le N \cdot D$ **则 7：返回 True 8：否则 9：返回 False**10：**结束 if**

现在我们证明算法 9的正确性，即我们将证明，如果总质量显著减少，那么该算法以概率 $1-\delta$返回 True；类似地，如果总质量没有显著减少，那么该算法以概率 $1-\delta$返回 False。

引理 *B.2.* 假设 $\widetilde{D_j} \ge D_j$ 对所有 $j \in [s]$均成立，且 $\widetilde{D} \ge D$。若 $\mu^2 D \le \widetilde{D}$，算法 *9* *L1Sampling($\{D_j\}$, $\{D_j\}$ μ, δ)* 将以至少 $1-\delta$ 的概率返回 *True*。若 $\mu D > D$，则 *L1Sampling($\{D_j\}$, $\{D_j\}$ μ, δ)* 将以至少 $1-\delta$ 的概率返回 *False*。

证明。令 $D_j = \lambda^{r_j}$。由于 r_j由 PowerApprox(D_j, λ) 返回，根据 定理 2.12，我们有 $D_j \le \lambda^{r_j} = D_j < \lambda D_j$。

定义随机变量

$$Z_i = \frac{1}{p_j}\widehat{D_j}, \text{ with probability } p_j = \frac{\widetilde{D_j}}{\widetilde{D}}.$$

我们将评估 Z_i 和 $\mathbb{E}[Z_i]$ 的范围，以便稍后应用Hoeffding不等式。我们有 $D_j \ge D_j$ 和 $D_j < \lambda D_j$。因此，

$$Z_i = \frac{1}{p_j}\widehat{D_j} = \frac{\widetilde{D}}{\widetilde{D_j}}\widehat{D_j} \le \frac{\widetilde{D}}{D_j}\lambda D_j = \lambda\widetilde{D}.$$

另一方面，根据定义，Z_i 必须为非负数，因此 $Z_i \in [0, \lambda D]$。对于 $\mathbb{E}[Z_i]$ 的范围，我们有

$$\mathbb{E}[Z_i] = \sum_{j=1}^{s} \frac{1}{p_j} \widehat{D_j} \cdot p_j = \sum_{j=1}^{s} \widehat{D_j}.$$

由于 $D_j \in [D_j, \lambda D_j]$，因此我们有 $\mathbb{E}[Z_i] \in [D, \lambda D]$。类似地，根据期望的线性性，由于 $T = \sum_{i=1}^{N} Z_i$，因此我们有 $\mathbb{E}[T] = N \cdot \mathbb{E}[Z_i] \in [ND, N\lambda D]$。我们根据 $\mu^2 D \leq \tilde{D}$ 或 $\mu^2 D > \tilde{D}$ 进行分情况讨论，分别对应所述保证中的两种情况。

情况 1： $\mu^2 D \leq \tilde{D}$。我们首先分析 $\mu^2 D \leq \tilde{D}$ 的情况。我们将 $\mathcal{N}$ 定义为如下事件：当 $\mu^2 D \leq \tilde{D}$ 时，L1Sampling($\{\widehat{D_j}\}, \{D_j\}$ μ, δ) 返回 False，将其称为假阴性事件。注意，当 $\mu^2 D \leq \tilde{D}$ 时，该事件在算法中发生，当且仅当 $\alpha T > N\tilde{D}$。

注意，如果 $|T - \mathbb{E}[T]| \leq \lambda ND$，则 $\alpha T > N\tilde{D}$ 不可能，因为 $\mu^2 D \leq \tilde{D}$ 和 $|T - \mathbb{E}[T]| \leq \lambda ND$ 蕴含

$$\alpha T \leq \alpha(\mathbb{E}[T] + |T - \mathbb{E}[T]|) \leq \alpha(\lambda ND + \lambda ND) \leq \frac{2\alpha\lambda}{\mu^2} N\tilde{D} = \frac{1}{\mu} N\tilde{D} < N\tilde{D}.$$

第一个不等式来自三角不等式。第二个不等式源于以下事实：$\mathbb{E}[T] \in [ND, N\lambda D]$ 和 $|T - \mathbb{E}[T]| \leq \lambda ND$。第三个不等式由条件 $\mu^2 D \leq \tilde{D}$ 得出。等式成立是因为 $\alpha = 2\mu$ 和 $\lambda = \frac{\mu}{2}$。最后一个不等式由 $\mu > 1$ 得出。

因此，在满足条件 $\mu^2 D \leq \tilde{D}$ 时，只有当 $|T - \mathbb{E}[T]| > \lambda ND$，才会出现假阴性$\mathcal{N}$。因此，

$$\mathbf{Pr}[\mathcal{N}] = \mathbf{Pr}\left[\alpha T > N\tilde{D} | \mu^2 D \leq \tilde{D}\right] \leq \mathbf{Pr}\left[|T - \mathbb{E}[T]| > \lambda ND | \mu^2 D \leq \tilde{D}\right].$$

根据 Hoeffding不等式，参见 定理2.6，

$$\mathbf{Pr}\left[|T - \mathbb{E}[T]| > \lambda ND | \mu^2 D \leq \tilde{D}\right] \leq 2\exp\left(-\frac{(\lambda ND)^2}{\sum_{i=1}^{N} (\lambda D)^2}\right) = 2\exp(-N) \leq \delta,$$

因为我们有 $N_i \in [0, \lambda D]$ 和 $N = \mathcal{O}\left(\log \frac{1}{\delta}\right)$。由于我们已经证明，事件$\mathcal{N}$发生并导致假阴性的概率至多为 δ，那么当 $\mu^2 D \leq \tilde{D}$时，L1Sampling($\{D_j\}$, $\{D_j\}$ μ, δ) 将以至少 $1 - \delta$ 的概率返回 True。

接下来我们分析 $\mu D >$的情况 $\tilde{D}$.

情况 2： $\mu D > D$。我们定义事件 $\mathcal{P}$：当 $\mu D > D$ 时，L1Sampling($\{D_j\}$, $\{D_j\}$ μ, δ) 返回 True，即发生假阳性事件。注意，当且仅当 $\alpha T \leq ND$ 且 $\mu D > D$ 时，这种情况才会发生。

注意，如果，则 $\alpha T \leq N\tilde{D}$ $|T - \mathbb{E}[T]| \leq \frac{1}{2}ND$ $\mu D > \tilde{D}$ $|T - \mathbb{E}[T]| \leq$ 不可能发生。事实 $\frac{1}{2}ND$ 上，如果 且，那么

$$\alpha T \geq \alpha(\mathbb{E}[T] - |T - \mathbb{E}[T]|) \geq \alpha(ND - \frac{1}{2}ND) = \frac{\alpha}{2}ND > \frac{\alpha}{2\mu}N\tilde{D} = N\tilde{D}.$$

第一个不等式来自三角不等式。第二个不等式是基于以下事实：$\mathbb{E}[T] \in [ND, N\lambda D]$ 和 $|T - \mathbb{E}[T]| \leq \frac{1}{2}ND$。第三个不等式源于条件 $\mu D > \tilde{D}$。第二个等式是因为 $\alpha = 2\mu$。

因此，在满足条件 $\mu D > \widetilde{D}$ 的情况下，只有当 $|T - \mathbb{E}[T]| > \frac{1}{2}ND$ 时才会出现假阳性$\mathcal{P}$。因此，

$$\mathbf{Pr}\left[\mathcal{P}\right] = \mathbf{Pr}\left[\alpha T \leq N\widetilde{D} | \mu D > \widetilde{D}\right] \leq \mathbf{Pr}\left[|T - \mathbb{E}[T]| > \frac{1}{2}ND | \mu D > \widetilde{D}\right].$$

然后根据Hoeffding不等式，参见定理2.6，

$$\mathbf{Pr}\left[|T - \mathbb{E}[T]| > \frac{1}{2}ND | \mu D > \widetilde{D}\right] \leq 2\exp\left(-\frac{(\frac{1}{2}ND)^2}{\sum_{i=1}^{N}(\lambda D)^2}\right) = 2\exp\left(-\frac{N}{4\lambda^2}\right) \leq \delta,$$

因为 $N_i \in [0, \lambda D]$ 且 $N = \mathcal{O}\left(\log \frac{1}{\delta}\right)$。由于我们已经证明事件$\mathcal{P}$的概率不超过 δ，因此若 $\mu D > \widetilde{D}$，L1Sampling($\{D_j\}, \{\widetilde{D}_j\}$ μ, δ)将以至少 $1 - \delta$的概率返回False。 □

B.3 带通信/轮次缩减的双准则近似

在本节中，我们将介绍另一种协议，其使用$\mathcal{O}(\log n \log k)$通信轮数以及$\mathcal{O}(s \log n + kd \log n)$比特的通信成本。该协议所用的通信轮数和总通信量将少于算法 8，当 $k = \Omega(\log n)$时。

我们首先应用 L_1 采样子程序L1Sampling来估计 D，其使用低通信成本。然后，若来自 L_1 采样过程的 D 估计显著下降，则更新 $\widetilde{D}$。采用该策略，我们只需在极少轮次中更新 D_j，从而节省更新 D_j的通信成本。

然而，通信轮数仍为$\mathcal{O}(k)$，这在某些设定下会过于昂贵。为了进一步减少通信轮数，我们采用批量采样策略。我们在每一轮将待采样的点数加倍，直到总代价 D 显著下降。D 最多只能显著下降$\mathcal{O}(\log n)$轮，因此我们可以将通信轮数减少到$\mathcal{O}(\log n \log k)$。

另一个必须解决的问题是，我们可能会采样到一些无效点，因为总代价 D 可能在一轮批量采样过程中显著下降。一个采样点是有效的，仅当它是在 D 为 D的$\mathcal{O}(1)$-近似这一条件下被采样时；因此，如果 D 在该采样轮次中显著下降，某些采样点可能是无效的。为确保我们采样到足够多的有效点，我们需要在采样过程中统计有效样本的数量。我们并不将待采样点数固定为 $N = \mathcal{O}(k)$，而是统计有效样本的数量，并且仅在获得至少 N 个有效样本后才终止采样。幸运的是，尽管在这种策略下我们采样的点数可能超过 N，但我们可以证明所采样的点的总数仍为$\mathcal{O}(k)$。

该算法的完整内容见算法 10。

算法 10 使用 L1Sampling 检测 D 是否显著减少，以及是否需要更新 $\widetilde{D}$。在 D 是 D 的 O (1)-近似的条件下，它使用批量采样来采样点 $\widetilde{D}$，并验证批量采样后 D 是否仍是 D 的 O (1)-近似。如果 D 仍是 D 的 O (1)-近似，这意味着所有采样点都有效，我们便将它们计入数量。如果 D 不再是 D 的 O (1)-近似，这意味着批量采样期间 O (1)-近似条件被破坏，导致部分采样点无效。不过，第一个采样点 $\widetilde{D}$ 必定有效，因为我们是在 D 是 O (1)-近似的条件下对其进行采样的。因此，我们可以将有效采样点的计数加 1。我们重复采样，直到我们有

算法 10 双准则近似算法 (k, z)

-聚类 输入：数据集 X_i**，每个站点** $i \in [s]$ **输出：集合** S**，它是 (**k, z**)-聚类** $\sim$ **的 (O (1)，O (1))-双准则近似 1: 均匀采样一个点放入** S**，并计算** $D_j =$**的常数因子近似 {**D_j**}，代价为 cost(**X_j, S**) 2:** $N \leftarrow$ **O (**k**)，**$M \leftarrow 0, \mu \leftarrow 8, \delta \leftarrow$ **O** $\left({}^{1}\log^{2} n \right)$ **3: 当** $M < N$ **\\ 采样大约** k **个点时，执行 4:** $bool \leftarrow$ **False** $i \leftarrow 1\sim$ **5: 当** $bool =$**False \\ 近似距离 {**D_j**} 准确时，执行 6: 对于** $l \leftarrow 1$ **到** 2^i **\\ 尝试将采样点采样到** S **中，执行** $\sim$ **7:** $x_l \leftarrow$ **LazySampling({cost(**x, S**)}, {**D_j**}), 将** x_l **上传到黑板 8: 结束循环 9:** $S \leftarrow S \cup ($**∪**$_{l=1}^{2^i}$**{**x_l**})** $L \leftarrow$ **#{**$x_l \neq \perp, 1 \leq l \leq 2^i$**} 将成功采样的点加入** S **10:** $M \leftarrow M + L$ **11: 每个站点** j **计算** $D_j =$ **cost(**X_j, S**) ▷ 更新到最近中心的距离** $\sim$ $\sim$ **12:** $bool \leftarrow$ **L1Sampling({**D_j**}, {**D_j**}** μ, δ**) ▷ 检查近似距离 {**D_j**} 13: 如果** $bool =$**False，则 14:** $i \leftarrow i + 1$ **▷ 增加采样数量以减少轮次 15: 否则**$\sim$ **16:** $r_j \leftarrow$ **PowerApprox(**D_j**, 2)，每个站点** j **在黑板上更新** r_j**，如果** $r_j \neq$ **log**$_2$ $D_j\sim$ **17:** $D_j \leftarrow 2^{r_j}$ **▷ 所有站点更新近似距离 18: 结束条件判断 19: 结束循环 20: 结束循环 21: 返回** S

至少计数 $N = \mathcal{O}(k)$ 个有效采样点。则返回的集合 S 将包含 $\mathcal{O}(k)$ 个点，并且是一个 $\mathcal{O}(1)$-近似。此外，通过使用 L1Sampling 和批量采样，我们可以保证算法 10具有较低的轮次复杂度和总通信量。

我们将证明分为三部分。首先，我们将证明 S 是一个 $\mathcal{O}$(1)-近似，并且$|S| = \mathcal{O}(k)$。其次，我们将证明该算法使用 $\mathcal{O}(\log\ n \log\ k)$ 轮通信。˜最后，我们将证明该算法以至少 0.99 的概率使用 $\mathcal{O}(s \log\ n + kd \log\ n)$ 比特的通信。

为便于分析，我们定义有效样本这一概念。假设 x 是通过 LazySampling({cost(x, S)},{D_i})<style id="">采样得到的点。若 x 是在条件˜ $D \leq D < \gamma \cdot D_i$ 下采样得到的，其中 γ为某个常数，则称其为有效样本。

引理 B.3. 算法 *10* 将返回 S 使得 $|S| = \mathcal{O}(k)$ 且 代价$(S, X) \leq \mathcal{O}(1)\cdot$代价$(C_{OPT}, X)$的概率至少为 0.98，其中 C_{OPT} 是 (k, z)-聚类 X的最优解。

˜证明。假设 x 是通过惰性采样({cost(x, S)}, {D_i})采样得到的一个点。根据引理 A.2，如果 x 是有效样本，那么它以常数概率减少 Bad$_i$ 的大小。如果我们的算法采样了 $N =$ O (k) 个有效点，根据定理 2.10，我们将以至少 0.99 的概率得到一个常数近似 S 。我们将证明，对于 $\gamma = \mu^2$，我们的算法确实至少采样了 N 个有效样本。

根据算法， D_j 始终是2-近似 D_j ，且 $D_j \le D_j < 2D_j$ 在我们更新 D_j之后。在对 D_j的两次更新之间， D_j 保持不变，但 D_j 可能只会减小。这意味着 $D_j \le D_j$ 在算法 10中始终成立。由于 $D_j \le D_j$ 始终成立， $D \le D$ 也始终成立。

If $\tilde{D} < \mu^2 D$ 采样 L 个点后，这意味着所有这些 L 点都是有效的。如果 L1Sampling 返回 False，我们将向计数器 M 中加上 L ；如果 L1Sampling 返回 True，则向计数器 M 中加上 1。无论哪种情况，我们采样到的有效点数量都至少等于向计数器 M中增加的数量。

如果 $\tilde{D} \ge \mu^2 D$ 采样 L 个点后，这意味着我们采样到的某些点是无效的。然而，我们采样到的第一个点必定是有效的，因为它是在条件 $\tilde{D} < \mu^2 D$下采样的。如果 L1Sampling 返回 True，我们只会向计数器 M中加上 1。因此，本轮新增的有效样本数量至少等于向计数器 M中增加的数量。

因此，如果 L1Sampling 每次 $\tilde{D} \ge \mu^2 D$都返回 True，我们采样到的有效点数量至少等于向计数器M 中增加的数量。由于当 S中只有一个点时， $D = \text{poly}(n)$，并且我们在 算法 10中每次更新 $\tilde{D}$ 时， $D \le \tilde{D} < 2D$、 $\tilde{D} \ge \mu^2 D$最多只能发生 $\mathcal{O}(\log n)$ 次。我们知道，当 $\tilde{D} \ge \mu^2 D$ 时，L1Sampling 以至少 $1 - \delta$的概率返回 True。由于 $\delta = \mathcal{O}\left(\frac{1}{\log^2 n}\right)$，因此根据并集界，L1Sampling 每次 $\tilde{D} \ge \mu^2 D$ 都返回 True 的概率至少为 0.99。

因此，以至少 0.98 的概率，我们将采样到至少 $\mathcal{O}(k)$ 个有效点，且它们构成最优解的 $\mathcal{O}(1)$-近似。

的大小上界 S**.** 我们将评估未计为有效样本的采样点数量，并证明此类点的总数为 $\mathcal{O}(k)$。假设我们总共进行了 次 D_j 更新 m 。

令 N_p 表示未计为有效样本的采样点总数，令 p_i 表示第 $(i-1)$ 次更新与第 i 次更新之间未计为有效样本的点数。则 $N_p = \sum_{i=1}^{m} p_i$。令 N_q 表示计为有效样本的采样点总数，令 q_i 表示第 $(i-1)$ 次更新与第 i 次更新之间计为有效样本的点数。则 $N_q = \sum_{i=1}^{m} q_i$。由于当我们计得 $N = \mathcal{O}(k)$ 个有效样本时，算法将终止，因此 $N_q = \mathcal{O}(k)$。第 i 次更新之前采样的点数为 $p_i + 1$，而在这 $p_i + 1$ 个点之前采样的点数为 $\frac{p_i+1}{2}$，因此，$p_i \le 2q_i$。于是 $N_p \le 2N_q = \mathcal{O}(k)$。 S 中的点要么计为有效样本，要么不计为有效样本，因此 $|S| = N_p + N_q = \mathcal{O}(k)$。 ——

□

接下来，我们给出该算法轮次复杂度的上界。

引理 B.4. 算法 *10* 以至少 $\mathcal{O}(\log n \log k)$ 的概率使用的通信轮数不超过0.99。

证明。 在初始阶段，我们需要 $\mathcal{O}(1)$ 轮通信，即 算法 7。对于 算法 10剩余部分的通信轮数，我们计算两次更新之间进行批量采样的轮数，并计算更新的次数。两者的乘积就是采样的总轮数。

对于两次更新之间进行的批量采样轮次，由于如果不进行更新 D ，我们会在隔一轮时将待采样点的数量翻倍，且待采样点的总数为$\mathcal{O}(k)$，因此其最多为$\mathcal{O}(\log k)$轮。对于进行更新 D的情况，则有两种

~情况：在满足条件 $D < \mu D$时进行一次“不必要的更新”，以及在满足条件 $D \geq \mu D$时进行一次~“必要的更新”。根据引理 B.2，L1Sampling 将返回~True，并且在满足条件 $D < \mu D$ 时，以至多 δ的概率进行一次“不必要的更新”。由于连续进行批量采样而不更新的次数至多为 O (log k)，因此在两次“必要的更新”之间进行一次“不必要的更新”的概率至多为 O (δ log k)。~ ~对于“必要的更新”，我们将更新 D，使得 $D \leq D < 2D$。由于 $\mu = 8$，这意味着每次在满足条件 $D \geq \mu D$时进行“必要的更新”，~ ~ D 至少会缩小 4 倍。因此，此类更新最多可以发生 O (log n) 次。

两次“必要更新”之间发生一次“不必要更新”的概率至多为 O (δ log k)，而“必要更新”的次数至多为 O (log n)，因此，算法 10 中发生“不必要更新”的概率至多为 O (δ log n log k)。由于 $\delta =$ O $\left({}^{1}\log^{2} n \right)$，算法 10 中不会发生“不必要更新”的概率至少为 0.99。由于“必要~更新”最多发生 O (log n) 次，这意味着以至少 0.99 的概率，D的更新次数至多为 O (log n)。

因此，以至少 0.99 的概率，算法 10 的$\mathcal{O}$ (log $n$log k) 轮批量采样。接下来，我们需要评估一轮批量采样需要多少通信轮数。对于每一轮批量采样，协调器可以同时发送 2^i 惰性采样请求，随后每个站点都可以作出响应。因此，每次采样迭代需要两轮通信。为了获得L1SAMPLING返回的结果，可以同时发出$\mathcal{O}\left(\log \frac{1}{\delta}\right) = \mathcal{O}(\log \log n)$ 个请求，因此通信轮数为$\mathcal{O}(1)$。因此，算法 10 中的通信总轮数为$\mathcal{O}$ (log $n$log k)。

最后，所有站点都可以在同一轮次中更新$\widetilde{r_j}$ 的值。由于我们最多更新$\mathcal{O}$ (log n)次，总共需要$\mathcal{O}$ (log n)通信轮数。将算法各部分的通信轮数相加，可得总通信轮数为$\mathcal{O}$ (log $n$log k)，概率至少为 0.99。

□

最后，我们分析算法的总通信量。

~**引理 B.5.** 算法 *10* 使用 $\mathcal{O}(s \log n + kd\log n)$ 通信比特，概率至少为 0.99。

证明。 该算法 7 子程序 初始化 产生 $\mathcal{O}(d\log n + s\log\log n)$ 通信比特，因为我们只写入样本点 x 和 r_i 到黑板上，并且 r_i =$\mathcal{O}$ (log log 代价(X_i, S)) $= \mathcal{O}(\log\log n)$。

我们将更新$\mathcal{O}$ (k)个样本，这将使用$\mathcal{O}$ (kdlog n)通信总比特数。

此外，我们最多运行 O (log $n$log k) 次 L1Sampling。每次运行时，协调器会选择 O $\left(\log \frac{1}{\delta}\right)$ = O (log log n) 个站点。它使用 O (log s) 比特在黑板上表示一个站点，以便该站点知道自己已被选中。每个被选中的站点需要 O (log log n) 比特进行回复。因此，在算法 10 中运行 L1Sampling 的总通信成本为 O (log $n$log k(log s + log log n)) 比特。

更新值 $\{r_j\}$ 的代价为每次迭代 $\mathcal{O}$ ($s$log log n) 比特。由于以至少 0.99 的概率，值将更新 $\mathcal{O}$ (log n) 次，因此更新值 $\{r_j\}$ 的总代价为$\mathcal{O}$ (s log $n$log log n) 比特。

因此，将 算法 10 各部分的通信成本相加，并忽略所有~关于 s 的多对数项，k、log n，总通信量为 $\mathcal{O}(s \log n + kd\log n)$。 □

我们对算法 10有如下完整保证。

引理 *B.6.* 算法 *10* 将返回 S，使得 $|S| = O(k)$ 且 $\mathrm{cost}(S, X) \le O(1)\cdot\mathrm{cost}(C_{OPT}, X)$ 的概率至少为 *0.98*，其中 C_{OPT} 是 X 的最优 (k, z)- 聚类。该算法~以至少 *0.99* 的概率使用 $O(s \log n + kd \log n)$ 比特的通信量和 $O(\log n \log k)$ 轮通信。

证明。 证明可直接由引理 B.3、引理 B.4和引理 B.5得出。 □

B.4 $(1+\varepsilon)$：黑板模型中的敏感度采样核心集

为了通过 $(\mathcal{O}(1), \mathcal{O}(1))$ 双准则近似 S，为 X 构造一个 $(1+\varepsilon)$-核心集，我们使用敏感度采样。难点在于确定 C_j 和 $\mathrm{cost}(C_j, S)$，以计算 $\mu(x)$，因为传递 C_j 和 $\mathrm{cost}(C_j, S)$ 的精确值需要 $\mathcal{O}(sk\log n)$ 比特，这对于我们的目的而言并不实际。然而，对 C_j 和 $\mathrm{cost}(C_j, S)$ 进行常数近似，就足以为敏感度采样近似$\mu(x)$；利用 Morris 计数器即可高效实现这一点，且通信开销极小。

算法 11 $\textsc{Morris}(r, n)$

输入： 初始计数索引 r，以及元素数量 n**输出：** 新的计数索引 m

$m \leftarrow r$ **对于** $i \leftarrow 1$ 到 n，**执行** 以概率 $\frac{1}{2^m}$，$m \leftarrow m+1$ **循环结束** **返回** r

定理 B.7。 *[Mor78]* 设 $X = \textsc{Morris}(0, n)$。那么 $\mathbb{E}\left[2^X - 1\right] = n$。此外，存在一个常数 $\gamma > 0$ 使得如果 $l \ge \frac{\gamma}{\varepsilon^2}\log\frac{1}{\delta}$ 并且 $Y = \frac{1}{l}\sum_{i=1}^{l}\left(2^{X_i} - 1\right)$，其中 $X_1, X_2, \ldots, X_l$ 是 l *Morris* $\textsc{Morris}(0, n)$ 的独立输出，则

$$Y \in [n - \varepsilon n, n + \varepsilon n],$$

概率至少为 $1 - \delta$。

我们可以将 Morris 改造为分布式版本 DistMorris，以降低通信成本。

引理 B.8。 设每个站点 i拥有 k个数 $n_{i,j}$，以及 $|N_j| = \sum_{i=1}^{s} n_{i,j} = \mathrm{poly}(n)$。以至少 0.99 的概率，*DistMorris*(14) 返回常数近似 $\widetilde{N}_j$，使得对于所有 $j \in [k]$，$\widetilde{N}_j \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$ 使用 $O(s + k\log n)$ 通信比特。

证明。 尽管我们将Morris 的过程拆分成分布式版本，但其准确性并不重要。这是因为Morris 的每一步只需知道上一步的状态，而站点每次使用Morris的一步时，都能获知上一步的状态：要么这两步都在该站点执行，要么上一步的状态已由上一站点准确上传。因此，定理 B.7 中原始 Morris 的陈述 对我们的分布式版本仍然成立。

算法 12 DISTMORRIS(ε):

输入：精度参数 ε S_i k $n_{i\,j}$ $j \in [k]$，每个站点拥有数字，，输出：和 $\widetilde{N}_j = \widetilde{\sum_{i=1}^{s}} n_{i,j},\ j \in [k]$ 的近似值 $(N_1, N_2, \cdots N_k)$1: $l \leftarrow O\left(\frac{1}{\varepsilon^2}\log(100k)\right)$2: $\overline{m}_{j,t} \leftarrow 0$ 对所有 $j \in [k]$ 和 $t \in [l]$3: 对 $i \leftarrow 1$ 到 s 执行 4: 对所有 $j \in [k]$ 和 $t \in [l]$，执行 $m'_{j,t} \leftarrow$ Morris($m_{j,t}$,$n_{i,j}$) 5: 如果对所有 $j \in [k]$ 和 $t \in [l]$，$m'_{j,t} = m_{j,t}$ 则 6: 站点 i将 $\perp$ 上传到黑板 7: 否则 8: 对所有满足 $m'_{j,t} \neq m_{j,t}$ 的 $j \in [k]$ 和 $t \in [l]$，站点 i 上传 $(m'_{j,t} - m_{j,t}, j, t)$ 9: 结束如果 10: $m_{j,t} \leftarrow m'_j$ 11: 结束循环 12: 对所有 $j \in [k]$~ ~ ~，$\widetilde{N}_j \leftarrow \frac{1}{l}\sum_{t=1}^{l}(2^{m_{j,t}} - 1)$ 13: 返回 $(N_1, N_2, \cdots N_k)$

–

根据 定理 B.7，$\widetilde{N_j} \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$ 概率至少为 $1 - \frac{1}{100k}$ 对于任意 $j \in [k]$，如果我们设 $l = \mathcal{O}(\log(100k))$。那么，根据并集界，$\widetilde{N}_j \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$ 对所有 $j \in [k]$ 同时成立的概率至少为 0.99。

为了评估总通信成本，我们引入 M_i 和 $M_{i,\ j}$ 如下，以便于分析。若站点 i 在黑板上上传 $\perp$，则将 M_i 设为 0；否则，将其设为通信成本，用于更新所有满足 $m'_{j,\ t} \neq m_{j,\ t}$ 的 $j \in [k]$ 和 $t \in [l]$ 对应的 $(m'_{j,\ t} - m_{j,\ t},\ j,\ t)$。我们将 $M_{i,\ j,\ t}$ 设为更新 $(m'_{j,\ t} - m_{j,\ t},\ j,\ t)$ 所需的通信成本。

每个带有 $M_i = 0$ 的站点只需上传 $\perp$，占用 $\mathcal{O}(1)$ 比特。由于总共有 s 个站点，带有 $M_i = 0$ 的站点数量至多为 s。因此，这些站点的通信总量为 $\mathcal{O}(s)$。

对于带有 $M_i \neq 0$ 的站点，所使用的通信成本为

$$\sum_{M_i \neq 0} M_i = \sum_{i=1}^{s} M_i = \sum_{j=1}^{k}\sum_{t=1}^{l}\sum_{i=1} M_{i,j,t}.$$

注意，第一个等式成立是因为我们只添加了值均为零的项，而第二个等式则是将 M_i 除以 $M_{i,j,t}$ 得到的。

由于 $\widetilde{N_j} = \frac{1}{l}\sum_{t=1}^{l}(2^{m_{j,t}} - 1) \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$，则 $2^{m_{j,t}}$ 不能大于 $2l \cdot N_j = O(\log k\, \mathrm{poly}(n))$。因此 $m_{j,t} \leq O(\log n \log\log k)$。由于 $M_{i,j,t}$ 若非零则至少为 1，对于给定的 j, t，至多存在 $m_{j,t}$ 个非零的 $M_{i,j,t}$。对于每个非零的 $M_{i,j,t}$，由于站点上传 $m'_{j,t} - m_{j,t}$，其至多为最终的 $m_{j,t}$，因此站点更新 $m'_{j,t} - m_{j,t}$ 所用比特不超过 $O(\log m_{j,t})$，并用 $O(\log k + \log l)$ 比特表示 j 和 t。因此，

$$\sum_{i=1} M_{i,j,t} = \sum_{i \in [s], M_{i,j,t} \neq 0} M_{i,j,t} \leq m_{j,t} \cdot \mathcal{O}(\log m_{j,t} + \log k + \log l).$$

˜由于 $m_{j,t} \leq \mathcal{O}(\log\ n \log\log\ k)$ 且 $l = \mathcal{O}(\log\ k)$，则 $\sum_{i=1} M_{i,j,t} \leq \mathcal{O}(\log\ n)$。因此，

$$\sum_{M_i \neq 0} M_i = \sum_{j=1}^{k} \sum_{t=1}^{l} \sum_{i=1} M_{i,j,t} \leq kl \cdot \tilde{\mathcal{O}}(\log n) = \tilde{\mathcal{O}}(k \log n).$$

将上传 $\perp$ 的站点的通信成本与˜其他站点的通信成本相加后，总通信量不超过 $\mathcal{O}(s + k \log n)$。

□

由于对于任意 $j \in [k]$, $|C_j| = \sum_{i=1}^{s} |C_j \cap X_i|$ 和 $\text{cost}(C_j, S) = \sum_{i=1}^{s} \text{cost}(C_j \cap X_i, S)$，我们可以使用 DistMorris 以较低的通信成本近似计算这些项。由于 $|C_j|$˜和 $\text{cost}(C_j, S)$均至多为 poly(n)，总通信量至多为 $\mathcal{O}(s + k \log\ n)$ 比特。每个站点 i 都可以在本地计算 $\text{cost}(x, S)$，其中 $x \in X_i$。然后，利用$|C_j|$ 和 $\text{cost}(C_j, S)$的常数因子近似，每个站点都可以在本地计算敏感度 $\mu(x)$ 的常数因子近似。

最后，我们给出一种算法，在给定具有 $\mathcal{O}(1)$ 近似、包含 $\mathcal{O}(k)$ 个点且敏感度为 $\mu(x)$ 的情况下，为 X 生成一个 $(1+\varepsilon)$-核心集，其通信量为 $\mathcal{O}\left(\frac{dk}{\varepsilon^4}(\log\ k + \log\ \frac{1}{\varepsilon} \log\log\ n)\right)$ 比特。

算法 13 $(1+\varepsilon)$-核心集（黑板模型）

输入： 一个常数近似 S, $|S| = \mathcal{O}(k)$
输出：一个 $(1+\varepsilon)$-核心集 A**1：使用 DistMorris 获取** $|C_j|$ **的 O (1)-近似，并在黑板上获取所有** $j \in [k]$ **的 cost**(C_j, S) **2：** $m \leftarrow$ ˜**O** $\left(\frac{k}{\varepsilon^2}\right)$**min{**$\varepsilon^{-2}, \varepsilon^{-z}$**}3：对于** $i \leftarrow 1$ **到** s **执行 4：** $A_i \leftarrow \emptyset$ **5：在本地为所有** $x \in X_i$**计算** $\widetilde{\mu}(x)$**，作为** $\mu(x)$ **的 O (1)-近似 6：上传** $\widetilde{\mu}(X_i) = \sum_{x \in X_i} \widetilde{\mu}(x)$ **7：结束循环 8：以概率** $\widetilde{\mu}(x_i)/\sum_{i=1}^{s} \widetilde{\mu}(x_i)$ **独立地对站点** i **进行** m **次采样。令** m_i **为站点** i**被采样的次数。将** m_i **写入黑板 9：对于** $i \leftarrow 1$ **到** s **执行 10：** $A_i \leftarrow \emptyset$**11：对于** $j \in [m_i]$ **执行 12：以概率** $p_x = \widetilde{\mu}(x)\,\widetilde{\mu}(x_i)$ **对** x **进行采样 13：如果采样到** x**，则 14：令** x' **为由** S **高效编码的** x**，准确性为** $\varepsilon' =$**poly(**ε**) 15：** $A \leftarrow A_i \cup$**{(**x'**,** $\frac{1}{m\widehat{\mu}}$**(**x**))}，其中** $\widehat{\mu}(x)$ **是** $\widetilde{\mu}(x)$ **的** $(1 + \frac{\varepsilon}{2})$**-近似 16：结束条件 17：结束循环 18：将** A_i **上传到黑板 19：结束循环 20：** $A \leftarrow \cup_{i=1}^{s} A_i$**21：返回** A

现在我们证明 算法 13 的通信正确性及其复杂度。

引理 B.9。 算法 *13* 返回 A，使得 A 是一个$(1+\varepsilon)$-核心集，适用于 X，且在已有一个0.98概率下的$(O(1), O(1))$-双准则近似 S时成立。该算法使用$\tilde{O}\left(s+k$对数 $n+\frac{dk}{\text{最小值}(\varepsilon^4,\varepsilon^{2+z})}\right)$通信比特。

证明。根据引理 B.8，我们可以以至少 0.99 的概率获得 $|C_j|$ 和 $\text{cost}(C_j, S)$ 的 O (1)-近似。每个站点也可以在本地获得 $\text{cost}(x, S)$ 的精确值。由于 $\mu(x) := \frac{1}{4}\cdot\left(\frac{1}{k|C_j|}+\text{cost}(x,S)\,k\,\text{cost}(C_j,S)\right.$ $+\text{cost}(x,S)\,\text{cost}(X,S)+\left.\frac{\Delta_x}{\text{cost}(X,S)}\right)$ 以及 $\Delta_p := \text{cost}(C_j, S)/|C_j|$，我们可以计算出 $\tilde{\mu}(x)$ 的 O (1)-近似，其中 $\mu(x)$。尽管 [BCP+24] 在 $|S| = k$ 和以概率 $\mu(x)$ 进行采样点的设置下证明了定理 2.7，但在其证明中只需要 $|S| = O(k)$ 和 $\mu^{-1}(x) \le O(1)\cdot\max\{\frac{1}{k|C_j|}, \frac{\text{cost}(x,S)\,k\,\text{cost}(C_j,S)}{\text{cost}(x,S)\,\text{cost}(X,S)}, \frac{\Delta_x}{\text{cost}(X,S)}\}$。因此，在 $|S| = O(k)$ 和 $\tilde{\mu}(x) = O(1)\cdot\mu(x)$ 的条件下，定理 2.7 仍然成立。令 $B = \{(x, m\tilde{\mu}^{-1}(x))\}$，则当我们拥有 $|C_j|$和 $\text{cost}(C_j, S)$ 的 O (1)-近似时，B 以至少 0.99 的概率是 X 的 $(1+\frac{\varepsilon}{2})$-核心集。

我们有 $A = \{(x', m\tilde{\mu}^{-1}(x))\}$。由于 B是 X的 $(1+\frac{\varepsilon}{2})$-核心集，根据引理 2.13，$A$是 X的 $(1+\varepsilon)$-核心集。因此，算法 13 以至少 0.98 的概率返回 X 的 $(1+\varepsilon)$-核心集。

我们需要 $O(s+k\log n)$ 比特来获得关于 $|C_j|$ 和 $\text{cost}(C_j, S)$ 的 O (1)-近似。我们需要$\tilde{O}(s\log m) = O(s)$ 比特在黑板上写下 m_i。我们需要 $O(\log k + d\log\log n)$ 比特来上传x'，因为我们需要指出 S 中哪个中心离 x最近，这需要 $O(\log k)$ 比特；此外，我们还需要上传幂指数，这需要 $O\left(\log\frac{1}{\varepsilon}\log\log n\right)$ 比特。我们需要 $O\left(\frac{1}{\varepsilon}\log\log n\right)$比特来上传 $m\hat{\mu}^{-1}(x)$，因为我们只需要上传 $\hat{\mu}(x)$，它是 $\tilde{\mu}(x)$ 的 $(1+\frac{\varepsilon}{2})$-近似。由于 $|A| = O\left(\frac{k}{\varepsilon^4}\min(\cdot,\varepsilon^{2+z})\right)$，总共需要 $\tilde{O}\left(s+k\log n+\frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\right)$ 比特。□

我们通过指出可以得到一个 $(1+\varepsilon)$-强核心集来完成本小节，其通信成本不超过 $\tilde{\mathcal{O}}\left(s\log(n\Delta) + dk\log(n\Delta) + \frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\right)$ 比特。

定理 B.10. 存在一个协议，对分布在 s 个站点上的 n 个点进行处理，以至少0.97 的概率为(k, z)-聚类生成一个$(1+\varepsilon)$-强核心集，且使用

$$\tilde{O}\left(s\log(n) + dk\log(n) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$$

在黑板模型中的通信总比特数。

证明。 我们可以使用算法 10 生成一个 $(\mathcal{O}(1), \mathcal{O}(1))$-双准则近似 S，然后使用算法 13 获得一个 $(1+\varepsilon)$-强核心集。根据引理 B.6 和引理 B.9，返回的集合 A 是一个 $(1+\varepsilon)$-强核心集，其概率至少为 0.97，且总共使用$\tilde{\mathcal{O}}\left(s\log(n) + dk\log(n) + \frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\right)$ 比特的通信成本。□

C 通信协调器模型

本节将讨论我们针对通信协调器模型设计的分布式算法。请注意，如果所有服务器都在黑板设置中实现分布式算法，那么仅通过在这些服务器之间进行的 s 轮自适应采样，就会产生 $\mathcal{O}(dsk\log n)$ 项通信。

C.1 协调器模型中的高效采样

为了避免通信成本中的这些 $\mathcal{O}(dsk \log n)$ 项，我们需要一种通信效率更高的方法，以便以较低的通信成本执行自适应采样并上传核心集。我们提出了一种名为 EfficientCommunication 的算法，它能够以高准确性和低代价发送一个点的位置。假设某个固定站点拥有一组点 $X = \{x_1, \cdots \ x_l\}$，而另一个站点拥有一个点 y。为了在不将每个点显式发送给所有站点的情况下模拟自适应采样，我们的目标是以较低代价将 y 的高精度近似位置发送给第一个站点。为此，EfficientCommunication 近似并发送一个点在每个维度上的坐标。对于第 i维，我们使用子程序 HighProbGreaterThan，该子程序能够以高概率检测 $y^{(i)}$ 是否大于 $x_j^{(i)}$，其中 $y^{(i)}$ 是 y 在第 i维上的坐标，而 $x_j^{(i)}$ 是 $x_j \in X$在第 i维上的坐标。

算法 14 HighProbGreaterThan($x, \ y, \ \delta$)

输入：整数 x, y**，失败参数** δ**输出：布尔值** $bool$**，用于指示** x**是否大于** y**1:** $N \leftarrow \mathbf{O}\left(\log \frac{1}{\delta}\right), r \leftarrow 0$**2: 对** $i \leftarrow 1$ **至** N **执行 3:** $r_i \leftarrow 0$ **如果 GreaterThan(**$\bar{x}, y$**) 告知我们** $x \leq y$**，否则告知我们** $r_i \leftarrow 1$ **4:** $r \leftarrow r + r_i$**5: 结束循环 6: 如果** $\frac{r}{N} \leq \frac{1}{2}$ **则 7: 返回 False 8: 否则 9: 返回 True 10: 结束判断**

— –

子程序HighProbGreaterThan 是对 GreaterThan 的改编版本，由 [Nis93]提出。该子程序的主要用途是通过二分搜索，找出$y^{(i)}$ 相对于 $\{x_1^{(i)}$、$\}$ 的位置，只需进行 $\log l$ 次比较。然后，将 $y^{(i)}$ 与 $x_j^{(i)} + (1+\varepsilon)^m$进行比较，即可找到最佳的 $x_j^{(i)}$ 和 m，以近似表示 $y^{(i)}$。协调器模型下的算法完整地列于算法 15。

我们首先证明 HighProbGreaterThan 的正确性；如有必要，可多次运行并采用多数表决，以提高正确概率。

引理 C.1. 如果 *HighProbGreaterThan*(x, y, δ) 返回 *False*，则 $x \leq y$ 的概率至少为 $1 - \delta$。如果 *HighProbGreaterThan*(x, y, δ) 返回 *True*，则 $x > y$ 的概率至少为 $1 - \delta$。此外，当 $x, y \in [-\text{poly}(n), \text{poly}(n)]$ 时，该协议使用 $O\left(\log\log n \log \frac{1}{\delta}\right)$ 通信比特。

证明。我们定义随机变量 $E_i = r_i$。根据 定理 2.8，GreaterThan 给出错误答案的概率至多为 $p < \frac{1}{2}$，因此 –

$$\mathbb{E}[E_i | x \leq y] = 0 \cdot \mathbf{Pr}[E_i = 0 | x \leq y] + 1 \cdot \mathbf{Pr}[E_i = 1 | x \leq y] \leq p,$$
$$\mathbb{E}[E_i | x > y] = 0 \cdot \mathbf{Pr}[E_i = 0 | x > y] + 1 \cdot \mathbf{Pr}[E_i = 1 | x > y] \geq 1 - p.$$

算法 15 高效通信$(X, y, \varepsilon, \delta)$ 输入：由一个站点持有的集合 $X = \{x_1, \cdots \quad x_l\}$；由另一站点持有的点 y； ε，精度参数； δ，失败概率 输出：第二个站点向第一个站点发送 $\widetilde{y}$，它将是 y 的位置的近似位置，使得对于任意 $x \in X$，$\|y - \widetilde{y}\|_2 \leq \varepsilon \|x - y\|_2$ 的概率至少为 $1 - \delta$1：对于 $i \leftarrow 1$ 到 d 执行 2：对 $X = \{x_{i_1}, \cdots \quad x_{i_l}\}$ 排序，使得 $x_{i_1}^{(i)} \leq \cdots \leq x_{i_l}^{(i)}$，其中 $x_{i_j}^{(i)}$ 是 x_{i_j} 的第 i 个坐标 3：$x_{i_0}^{(i)} \leftarrow -\Delta, x_{i_{l+1}}^{(i)} \leftarrow \Delta, \delta' \leftarrow$ O $\left(_{d \ \log l+} \quad {}^{\delta}(\log \log {}_{n+}\log {}_{\frac{1}{\varepsilon}})\right)$4：使用二分搜索和 HighProbGreaterThan$(y^{(i)}, x_{i_j}^{(i)}, \delta')$ 找到 $x_{i_s}^{(i)}$，使得对于任意 $i_j \in \{0, 1, 2, \cdots \quad l, l+1\}$，$|x_{i_s}^{(i)} - y^{(i)}| \trianglelefteq |x_{i_j}^{i} - y^{(i)}|$ 5：使用 HighProbGreaterThan 测试 $y^{(i)} = x_{i_s}^{(i)}$ 6：如果 $y^{(i)} = x_{i_s}^{(i)}$ 则 7：$\Delta y^{(i)} \leftarrow 0$ 8：否则 9：$\gamma \leftarrow$ sign$(y^{(i)} - x_{i_s}^{(i)})$ 10：使用二分搜索和 HighProbGreaterThan$(y^{(i)}, x_{i_s}^{(i)} + \gamma \cdot (1+\varepsilon)^t, \delta')$ 找到 m，使得对于任意 $t \in \mathbb{N}$，$|x_{i_s}^{(i)} + \gamma \cdot (1+\varepsilon)^m - y^{(i)}| \leq |x_{i_s}^{(i)} + \gamma \cdot (1+\varepsilon)^t - y^{(i)}|$ 11：$\Delta y^{(i)} \leftarrow \gamma \cdot (1+\varepsilon)^m$ 12：结束 if 13：$\widetilde{y}^{(i)} \leftarrow x_{i_s}^{(i)} + \Delta y^{(i)}$ 14：结束 for返回 $\widetilde{y} = (\widetilde{y}^{(1)}, \widetilde{y}^{(2)}, \cdots \quad \widetilde{y}^{(d)})$

由于 $E_i \in [0, 1]$，根据 Hoeffding不等式，参见 定理2.6，

$$
\begin{aligned}
\mathbf{Pr}\left[\sum_{i=1}^{N} E_i > \frac{N}{2} \middle| x \leq y\right] &= \mathbf{Pr}\left[\sum_{i=1}^{N} E_i > Np + \frac{N}{2} - Np \middle| x \leq y\right] \\
&\leq \mathbf{Pr}\left[\left|\sum_{i=1}^{N} E_i - N\mathbb{E}[E_i]\right| > \frac{N}{2} - Np \middle| x \leq y\right] \\
&\leq 2\exp(-\frac{\left(\frac{N}{2} - Np\right)^2}{N \cdot 1^2}) \\
&= \delta,
\end{aligned}
$$

and

$$
\begin{aligned}
\mathbf{Pr}\left[\sum_{i=1}^{N} E_i \leq \frac{N}{2} \middle| x > y\right] &= \mathbf{Pr}\left[\sum_{i=1}^{N} E_i \leq N(1-p) + \frac{N}{2} - N(1-p) \middle| x > y\right] \\
&\leq \mathbf{Pr}\left[\left|\sum_{i=1}^{N} E_i - N \cdot \mathbb{E}[E_i]\right| > \frac{N}{2} - Np \middle| x > y\right] \\
&\leq 2\exp(-\frac{\left(\frac{N}{2} - Np\right)^2}{N \cdot 1^2})
\end{aligned}
$$

$= \delta.$

Hence HighProbGreaterThan(x, y, δ) 的正确概率至少为 $1-\delta$.

关于通信成本，由于 GreaterThan 使用$\mathcal{O}(\log\log n)$ 比特进行通信，假设 x, $y \in [-\text{poly}(n), \text{poly}(n)]$,, 并且我们运行 GreaterThan 共 $N = \mathcal{O}\left(\log \frac{1}{\delta}\right)$ 次，那么总通信成本为$\mathcal{O}\left(\log\log n \log \frac{1}{\delta}\right)$ 比特。 □

高效通信$(X, y, \varepsilon, \delta)$ 在算法 15 中将发送 $\widetilde{y}$，这是 y的一个良好近似位置，且其概率至少为 $1-\delta$。此外，它仅使用 $d\log l\,\text{polylog}(\log n, \log l, \frac{1}{\varepsilon}, \frac{1}{\delta})$比特。为正式证明这些保证，我们先证明 $\|y-\widetilde{y}\|_2 \le \varepsilon\|x-y\|_2$ 其概率为 $1-\delta$，然后给出该协议总通信量的上界。

引理 C.2。 高效通信$(X, y, \varepsilon, \delta)$ 将向第一个站点发送 $\widetilde{y}$，使得 $\|y-\widetilde{y}\|_2 \le \varepsilon\|x-y\|_2$，其概率至少为 $1-\delta$。

证明。 我们假设 HighProbGreaterThan 正确。我们将证明 $\|y-\widetilde{y}\|_2 \le \varepsilon\|x-y\|_2$。首先，我们将证明对于任意 $j \in [d]$ 和 $x_j \in X$, $|y^{(i)} - \widetilde{y}^{(i)}| \le \varepsilon|x_j^{(i)} - y^{(i)}|$。

令 $X^{(i)} = \{-\Delta, x_1^{(i)}, x_2^{(i)}, \cdots x_l^{(i)}, \Delta\}$。假设 $x_s^{(i)} \in X^{(i)}$ 对任意 $x_j^{(i)} \in X^{(i)}$ 都满足 $|x_s^{(i)} - y^{(i)}| \le |x_j^{(i)} - y^{(i)}|$。如果 $y^{(i)} = x_s^{(i)}$，那么 $\widetilde{y}^{(i)}$ 就是 $y^{(i)}$，这意味着 $|y^{(i)} - \widetilde{y}^{(i)}| = |x_s^{(i)} - y^{(i)}| = 0$。

如果 $y^{(i)} \ne x_s^{(i)}$，我们有 $\widetilde{y}^{(i)} = x_s^{(i)} + \Delta y^{(i)}$ 和 $\Delta y^{(i)} = \gamma\cdot(1+\varepsilon)^m$，其中 $\gamma = \text{sign}(y^{(i)} - x_s^{(i)})$。假设 $y^{(i)} - x_s^{(i)} = \gamma\cdot(1+\varepsilon)^{m+m'}$。于是

$$|y^{(i)} - \widetilde{y}^{(i)}| = |(y^{(i)} - x_s^{(i)}) - (\widetilde{y}^{(i)} - x_s^{(i)})| = |(1+\varepsilon)^{m+m'} - (1+\varepsilon)^m|.$$

由于 $|x_s^{(i)} + \gamma\cdot(1+\varepsilon)^m - y^{(i)}| \le |x_s^{(i)} + \gamma\cdot(1+\varepsilon)^t - y^{(i)}|$ 对任意 $t \in \mathbb{N}$, $|m'|$ 必须小于**1**。因此

$$|y^{(i)} - \widetilde{y}^{(i)}| = (1+\varepsilon)^{m+m'}\cdot|1-(1+\varepsilon)^{-m'}| \le \varepsilon(1+\varepsilon)^{m+m'} = \varepsilon|x_s^{(i)} - y^{(i)}|.$$

由于 $|x_s^{(i)} - y^{(i)}| \le |x_j^{(i)} - y^{(i)}|$ 对任意 j，且 $|y^{(i)} - \widetilde{y}^{(i)}| \le \varepsilon|x_s^{(i)} - y^{(i)}|$，从而 $|y^{(i)} - \widetilde{y}^{(i)}| \le \varepsilon|x_j^{(i)} - y^{(i)}|$。因此，对任意 $x_j \in X$,

$$\|y-\widetilde{y}\|_2^2 = \sum_{i=1}^d |y^{(i)} - \widetilde{y}^{(i)}|^2 \le \sum_{i=1}^d \varepsilon^2|x_j^{(i)} - y^{(i)}|^2 = \varepsilon^2\cdot\|x_j - y\|_2.$$

因此 $\|y-\widetilde{y}\|_2 \le \varepsilon\|x-y\|_2$ 对任意 $x \in X$。

失败概率分析。接下来，只需通过统计算法中调用 HighProbGreaterThan($\cdot, \cdot$ δ') 的次数，给出失败概率的上界。对于每个维度i，我们运行 HighProbGreaterThan($\cdot, \cdot$ δ')，以找到距离 $y^{(i)}$ 最近的 $x_{i_s}^{(i)}$，并用 m 近似$y^{(i)} - x_{i_s}^{(i)}$。为了找到距离 $y^{(i)}$最近的 $x_{i_s}^{(i)}$，我们可以通过二分搜索找到 $x_{i_p}^{(i)} \le y^{(i)} \le x_{i_{p+1}}^{(i)}$，然后将其中点 $\frac{x_{i_p}^{(i)} + x_{i_{p+1}}^{(i)}}{2}$ 与 $y^{(i)}$ 比较，以确定哪个更接近 $y^{(i)}$。由于$\{x_{i_0}^i, x_{i_1}^i, \cdots x_{i_l}^i, x_{i_{l+1}}^i\}$中有 $l+2$ 个元素，因此在这一阶段需要运行 HighProbGreaterThan($\cdot, \cdot$ δ') 共 $\log(l+2)+1$ 次。

为了找到满足对于任意 $t \in \mathbb{N}$， $|x_{i_s}^{(i)} + \gamma \cdot (1+\varepsilon)^m - y^{(i)}| \le |x_{i_s}^{(i)} + \gamma \cdot (1+\varepsilon)^t - y^{(i)}|$ 的 m，我们可以通过二分搜索找到 $(1+\varepsilon)^q \le |y^{(i)} - x_{i_s}^{(i)}| \le (1+\varepsilon)^{q+1}$，然后将它们的中点与$|y^{(i)} - x_{i_s}^{(i)}|$ 比较，以确定哪个更接近 $|y^{(i)} - x_{i_s}^{(i)}|$。由于我们已经获得所有 $j \in \{0, 1, \cdots \ l+1\}$ 对应的 $x_{i_p}^{(i)} \le y^{(i)} \le x_{i_{p+1}}^{(i)}$和 $|x_{i_j}^{(i)}| \le \Delta$，因此只需在 $\{0, 1, \cdots, \log_{1+\varepsilon} \Delta\}$ 中搜索 q。由于 $\Delta =$是 poly(n)，我们可以通过二分搜索，在至多 O $\left(\log \log_{\varepsilon} n\right)$ 轮次内找到 q。因此，在这一阶段，我们需要运行 HighProbGreaterThan$(\cdot, \cdot \ \delta')$ 共 O $\left(\log \log \ n + \log \ \frac{1}{\varepsilon}\right)$ 次。因此，我们至多会调用 HighProbGreaterThan O $\left(d\left(\log \ l + \log \log \ n + \log \ \frac{1}{\varepsilon}\right)\right)$ 次。由于 HighProbGreaterThan$(\cdot, \cdot \ \delta')$ 返回正确结果的失败概率至多为 δ'，且$\delta' =$O $\left(\frac{}{d \ \log l + } \ ^{\delta}(\log \log \ n + \log \ \frac{1}{\varepsilon})\right)$，EfficientCommunication$(X, y, \varepsilon, \delta)$ 的失败概率至多为 δ。

□

接下来，我们分析该算法的通信复杂度。

引理 C.3. 高效通信$(X, y, \varepsilon, \delta)$ 总共使用 $d \log \ell \operatorname{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$ 个通信比特，其中 $\ell = |X|$ 是第一个站点拥有的点数。

证明。 我们最多需要运行 HighProbGreaterThan$(\cdot, \cdot \ \delta')$ $\mathcal{O}\left(d\left(\log \ \ell + \log \log \ n + \log \ \frac{1}{\varepsilon}\right)\right)$次。由于 HighProbGreaterThan$(\cdot, \cdot \ \delta')$ 单次运行的代价为 $\mathcal{O}\left(\log \log \ n \log \ \frac{1}{\delta'}\right)$ 比特，因此总共运行 HighProbGreaterThan$(\cdot, \cdot \ \delta')$ 需要 $d \log \ \ell \operatorname{polylog}(\log \ n, \log \ \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$ 比特。

我们还需要通过通信将 $\widetilde{y} = (\widetilde{y}^{(1)}, \widetilde{y}^{(2)}, \cdots \ \widetilde{y}^{(d)})$ 发送到第一个站点。不过，我们只需将 $x_{i_s}^{(i)}$ 和 $\Delta y^{(i)}$ 发送到第一个站点。由于第一个站点已经掌握所有 $x \in X$的位置，我们只需发送 i_s 来标识 $x_{i_s}^{(i)}$，这需要 O $(\log \ell)$ 比特。由于 $\Delta y^{(i)} = \gamma \cdot (1+\varepsilon)^m$，我们只需发送 γ和 m。发送 γ需要 O (1) 比特，因为 $\gamma \in \{-1, 0, 1\}$。发送 m需要 O $\left(\log \log \ n + \log \ \frac{1}{\varepsilon}\right)$ 比特，因为 $(1+\varepsilon)^m =$ poly(n)。因此，发送 $\widetilde{y} = (\widetilde{y}^{(1)}, \widetilde{y}^{(2)}, \cdots \ \widetilde{y}^{(d)})$ 至多需要 $d \log \ \ell \operatorname{polylog}(\log \ n, \frac{1}{\varepsilon})$ 比特。

因此，算法 15 最多需要 $d \log \ \ell \operatorname{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$ 来发送 $\widetilde{y}$给第一个站点。 □

结合引理 C.2和引理 C.3，可得：

引理 C.4。高效通信$(X, y, \varepsilon, \delta)$ 将向第一个站点发送 $\widetilde{y}$，使得 $\|y - \widetilde{y}\|_2 \le \varepsilon \|x - y\|_2$ 的概率至少为 $1 - \delta$。此外，对于总计，它仅使用 $d \log \ \ell \operatorname{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$比特的通信，其中 $\ell = |X|$ 是第一个站点拥有的点数。

C.2 $(1+\varepsilon)$-核心集通过协调器模式下的敏感度采样 1

根据附录C.1中的分析可知，我们可以在协调器处有效地执行自适应采样，以实现双准则近似。接下来还需构造一个$(1+\varepsilon)$-核心集，为此我们再次使用敏感度采样。

高效通信$(X, y, \varepsilon, \delta)$能够以较低的通信成本发送 y的位置。然而，其通信成本为 $d \log \ \ell$ polylog(log n, log $\ell, \frac{1}{\varepsilon}, \frac{1}{\delta}$)，其中 $\ell = |X|$。对于协调器模型，假设每个站点 $i \in [s]$ 拥有一个数据集 X_i。由于 $|X_i| = \mathcal{O}(n)$，并且协调器需要将全部$\mathcal{O}(k)$个样本的近似位置发送给每个站点以应用自适应采样，这仍需要 $dsk \log \ n \operatorname{polylog}(\log \ n, \frac{1}{\varepsilon}, \frac{1}{\delta})$比特来发送样本的位置

使用高效通信$(X_i, y, \varepsilon, \delta)$。幸运的是，我们可以为每个 X_i生成一个$(1+\frac{\varepsilon}{2})$-核心集 P_i，其大小为 $\tilde{\mathcal{O}}\left(\frac{k}{\varepsilon^4}\min\{,\varepsilon^{2+z}\}\right)$。由于自适应采样同样适用于加权情形，因此只需为加权核心集$P=\cup_{i\in[s]}P_i$生成一个$(\mathcal{O}(1),\mathcal{O}(1))$-双准则近似。因此，只需 dskpolylog(log $n, k, \frac{1}{\varepsilon}, \frac{1}{\delta}$)比特，即可将样本的位置发送到每个站点。

为进一步消除 d 的多重依赖，我们注意到，应用自适应采样和敏感度采样时，只需知道数据点 $x\in P_i$ 与通过自适应采样生成的中心 $s\in S$ 之间的距离。因此，我们可以使用 Johnson-Lindenstrauss 变换将 P 映射到 $\pi(P)\subset\mathbb{R}^{d'}$，其中 $d'=\mathcal{O}(\log(sk))$。通过 JL 变换，$\pi(P)$ 位于低维空间中，但映射仍保留两两距离。因此，我们可以进一步降低向每个站点发送样本位置所需的通信成本；现在该成本为 skpolylog(log $n, s, k, \frac{1}{\varepsilon}, \frac{1}{\delta}$) 比特。因此，我们可以应用自适应采样，并将采样点 s 的精确位置发送给协调器。随后，协调器可以使用 EfficientCommunication$(\pi(P_i)\ \pi(s)\ \varepsilon, \delta)$ 将 $\pi(s)$ 的近似位置发送给每个站点，其精度足以让每个站点更新点的成本的常数近似。这样，我们就能以较低的通信成本重复进行自适应采样，并获得最优 (k, z)-聚类的 $(\mathcal{O}(1), \mathcal{O}(1))$-双准则近似 S。

由于每个站点都有 π（S）的近似副本，因此它们可以向协调器发送$|C_j\cap P_i|$ 和 cost$(C_j\cap P_i, S)$ 的常数近似。不过，站点可能会将一个点分配给 S 中一个并非离它最近的中心。这是因为站点只有 π（S）的近似位置，因此当 cost(x, s') 与 cost(x, s) 非常接近时，可能会将 x 分配给另一个中心 s'；其中 s 是离 x最近的中心。幸运的是，定理 2.7 的证明并不要求所有点 x 都分配给离它最近的点。如果 cost$(x, s')\le$ O (1) · cost(x, s)，定理 2.7 仍然成立。因此，我们可以通过敏感度采样为 P生成一个（$1+\frac{\varepsilon}{4}$）-核心集。

对这些点进行敏感度采样后，每个站点都可以使用 EfficientCommunication$(S, x, \varepsilon', \delta)$ 将采样点发送给协调器。通过类似引理 2.13 中关于高效编码的讨论，我们可以证明这些采样点构成一个（$1+\frac{\varepsilon}{2}$）-核心集 A'，用于$P=\cup_{i\in[s]}P_i$。因此，A' 将是 $X=\cup_{i\in[s]}X_i$的一个（$1+\varepsilon$）-核心集，协调器可以基于核心集 A'求解（k, z）-聚类。由于 $|P_i|=\tilde{\mathrm{O}}\left(\frac{k}{\varepsilon^4}\min(,\varepsilon^{2+z})\right)$，我们可以使用 polylog(log $n, k, \frac{1}{\varepsilon}, \frac{1}{\delta}$) 个通信比特运行 EfficientCommunication$(P_i, y, \varepsilon, \delta)$，以发送每个采样点。我们在算法 16 中给出完整算法。

现在我们证明 算法16 将以 $(1+\varepsilon)$-核心集的形式返回 X，且以常数概率成立，并使用较低的通信成本。我们首先证明 A' 是 $(1+\varepsilon)$-核心集形式的 X。

引理 C.5。 *算法16* 以至少 $(1+\varepsilon)$ 核心集的形式返回 X，其概率至少为 0。96 在协调器模型中。

证明。由于我们使用 EfficientCommunication$(\pi(P_j)\ \pi(s_{i-1})\ \varepsilon', \delta)$ 来发送 $\tilde{s}_{i-1}^{(j)}$，根据引理 C.4，对于任意 $x\in P_j$，$\|\pi(s_{i-1})-\tilde{s}_{i-1}^{(j)}\|_2\le\varepsilon'\|\pi(x)-\pi(s_{i-1})\|_2$。因此，$\|\pi(x)-\tilde{s}_{i-1}^{(j)}\|_2$ 将是 $\|\pi(x)-\pi(s_{i-1})\|_2$ 的 $(1+\varepsilon')$-近似。

由于 π 2 9 $\|\pi\ x\ -\pi\ y\ \|_2\in[\frac{1}{2}\|x-y\|_2, \frac{3}{2}\|x-y\|_2]$ $1=\frac{1}{sk}$ $\|\pi\ x\ -\pi\ y\ \|_2\in[\frac{1}{2}\|x-y\|_2, \frac{3}{2}\|x-y\|_2]$ 是 Johnson-Lindenstrauss 映射，根据 定理 .，( ) ( ) 以至少 $\mathcal{O}()$ 的概率成立。因此，对于任意 $x, y\in P=\cup_{i\in[s]}P_i$，( ) ( ) 以至少 0.99 的概率成立。由于 $\|\pi(x)-\pi(y)\|_2$ 是任意 $x, y\in P$ 的常数近似，且 $\|\pi(x)-\tilde{s}_{i-1}^{(j)}\|_2$ 是 $\|\pi(x)-\pi(s_{i-1})\|_2$ 的 $(1+\varepsilon')$-近似，因此 $\|\pi(x)-\tilde{s}_{i-1}^{(j)}\|_2$ 将是 $\|x-s_{i-1}\|_2$ 的 4-近似。

算法16：用于协调器模型的 $(1+\varepsilon)$-核心集 输入：每个站点 i 所拥有的数据集 X_i，$i \in [s]$ 输出：发送给协调器的 $(1+\varepsilon)$-核心集 A' 1：每个站点 i 为 X_i 生成一个 $(1+\frac{\varepsilon}{2})$-核心集 P_i 2：协调器向每个站点发送随机种子，以生成 Johnson-Lindenstrauss π，使得对于任意 x，$\|\pi(x) - \pi(y)\|_2 \in [\frac{1}{2}\|x-y\|_2, \frac{3}{2}\|x-y\|_2]$，且 $\pi(x) \in R^{d'}$，其中$d' = O(\log(sk))$ 3：$S \leftarrow \{s_0\}$，其中 s_0 是由协调器采样的一个点 4：对于 $j \in [s]$，执行 $S_j \leftarrow \emptyset$ 5：$N \leftarrow O(k)$ $\varepsilon' \leftarrow O(\varepsilon^z)$ $m \leftarrow \tilde{O}\left(\frac{k}{\varepsilon^4}\min(\varepsilon^{2+z})\right), \delta \leftarrow O\left(\frac{1}{sk+m}\right)$ 6：对于 $i \leftarrow 1$ 到 N，执行 7：使用高效通信 EfficientCommunication$(\pi(P_j), \pi(s_{i-1}), \varepsilon', \delta)$ 将 $\tilde{s}_{i-1}^{(j)}$（$\pi(s_{i-1})$ 的近似）发送给每个站点 j。$S_j \leftarrow S_j \cup \{\tilde{s}_{i-1}^{(j)}\}$ 8：每个站点更新代价 $\text{cost}(\pi(P_j), S_j)$，并将 D_j（$\text{cost}(\pi(P_j), S_j)$ 的常数近似）发送给协调器 9：$s_i \leftarrow \text{LazySampling}(\{\text{cost}(\pi(x), S_j)\}, \{D_j\})$ 10：结束循环 11：每个站点 j 为 $l \le |S|$ 计算 $|C_l^{(j)}|$ 和 $\text{cost}(\pi(C_l^{(j)}), S_j)$，其中 $C_l^{(j)} = \{x \in P_j : \text{dist}(\pi(x), \tilde{s}_l^{(j)}) \le \text{dist}(\pi(x), \tilde{s}_p^{(j)}) \ \forall p \le |S|\}$ 12：每个站点 j 将 $|C_l^{(j)}|$ 和 $\text{cost}\left(\pi\left(C_l^{(j)}\right), S_j\right)$ 的常数近似发送给协调器 13：协调器计算 $\sum_{l \le |S|} |C_l^{(j)}|$ 和 $\sum_{l \le |S|} \text{cost}\left(\pi\left(C_l^{(j)}\right), S_j\right)$，并将它们的常数近似发送给每个站点 14：对于所有 $x \in P_j$，每个站点在本地计算 $\tilde{\mu}(x)$，作为 $\mu(x)$ 的 $O(1)$-近似，并将 $\widehat{\mu}(P_i)$（$\tilde{\mu}(P_i) = \sum_{x \in P_i} \tilde{\mu}(x)$ 的常数近似）发送给协调器 15：协调器以概率 $\widehat{\mu}(P_j)/\sum_{i=1}^{s} \widehat{\mu}(P_i)$ 独立地对站点 j 采样 m 次。令 m_j 为站点 j 被采样的次数。将 m_j 发送给站点 j 16：对于 $i \leftarrow 1$ 到 s，执行 17：$A_i \leftarrow \emptyset, A_i' \leftarrow \emptyset$ 18：对于 $j \in [m_i]$，执行 19：以概率 $p_x = \tilde{\mu}(x)/\tilde{\mu}(P_i)$ 对 x 进行采样 20：如果 x 被采样，则 21：$A_i \leftarrow A_i \cup \{x, m\widehat{\mu}^{-1}(x)\}$ $\tilde{x} \leftarrow \text{EfficientCommunication}(S, x, \varepsilon', \delta)$ $A_i' \leftarrow A_i' \cup \{\tilde{x}, m\widehat{\mu}^{-1}(x)\}$，其中 $\widehat{\mu}(x)$ 是 $\tilde{\mu}(x)$ 的 $(1+\frac{\varepsilon}{2})$-近似 22：结束条件判断 23：结束循环 24：结束循环 25：$A' \leftarrow \cup_{i=1}^{s} A_i'$ 26：返回 A'

$\tilde{}$通过发送 D_j，协调器(1)-近似代价$(\pi(P_j), S_j)$，协调器拥有一个 协调器 (1)-近似代价(P_j, S)。因此，我们可以应用自适应采样，并根据 定理 2.10，

S 将以至少 0.99 的概率，成为$P=\cup_{j=1}^{s}P_j$ 的 (k,z)-聚类最优解的 $(\mathcal{O}(1),\mathcal{O}(1))$ 双准则近似。

由于 $C_l^{(j)}=\{x\in P_j:\text{dist}(\pi(x)\ \tilde{s}_l^{(j)})\ \leq\text{dist}(\pi(x)\ \tilde{s}_p^{(j)})\ \forall p\ \leq\ |S|\}$，且 $1\ 4\cdot\text{dist}(x,s_p)\ \leq\ \text{dist}(\pi(x)\ \tilde{s}_p^{(j)})\ \leq 4\cdot\text{dist}(x,s_p)$，因此，如果 $x\in C_l^{(j)}$，则 $1\ 16\cdot\text{dist}(x,S)\ \leq\text{dist}(\pi(x)\ \tilde{s}_l^{(j)})\ \leq 16\cdot\text{dist}(x,S)$。因此，如果我们应用敏感度采样，

$$\mu(x)=\frac{1}{4}\cdot\left(\frac{1}{k\left|\bigcup_{j=1}^{s}C_l^{(j)}\right|}+\frac{\text{cost}(\pi(x),S_j)}{k\sum_{l\leq|S|}D_l^{(j)}}+\frac{\text{cost}(\pi(x),S_j)}{\sum_{l\leq|S|,j\in[s]}D_l^{(j)}}+\frac{\Delta_x}{\sum_{l\leq|S|,j\in[s]}D_l^{(j)}}\right),$$

其中 $D_l^{(j)}=\text{cost}\left(\pi\left(C_l^{(j)}\right),S_j\right)$ 和 $\Delta_p=\frac{\sum_{l\leq|S|}D_l^{(j)}}{\left|\bigcup_{j=1}^{s}C_l^{(j)}\right|}$，它将以至少 0.99 的概率返回 $P=\cup_{j=1}^{s}P_j$ 的 $(1+\frac{\varepsilon}{4})$-核心集。

由于我们通过高效通信$(S,x,\varepsilon',\delta)$将 $\tilde{x}$ 发送给协调器，因此 $\|\tilde{x}-x\|_2\leq\varepsilon'\|s-x\|_2$，其中 s 是 S 中距离 x最近的点。由于 $\|\tilde{x}-x\|_2\leq\varepsilon'\|s-x\|_2$，根据引理 2.13 的证明，对于任意解 $|C|=k$，$(1-\frac{\varepsilon}{4})\cdot$代价$(C,A)\ \leq$代价$(C,A')\ \leq(1+\frac{\varepsilon}{4})\cdot$代价$(C,A)$。

由于 P是 X的$(1+\frac{\varepsilon}{2})$-核心集， A是 P的$(1+\frac{\varepsilon}{4})$-核心集，且 A'是 A的$(1+\frac{\varepsilon}{4})$-核心集，因此， A'是 X的$(1+\varepsilon)$-核心集。

成功概率评估。 为了应用自适应采样，我们需要对每个 P_j和 s_{i-1}应用高效通信$(\pi(P_j)\ \pi(s_{i-1})\ \varepsilon',\delta)$，总共需要运行$\mathcal{O}(sk)$次。由于我们需要对敏感度采样抽取的每个点应用高效通信$(S,\ x,\ \varepsilon',\ \delta)$，总共需要运行 m 次。由于 $\delta\leftarrow\mathcal{O}\left(\frac{1}{sk+m}\right)$，所有高效通信实例均返回足够准确的近似位置的概率至少为0。99。

由于 Johnson-Lindenstrauss 变换以至少 0.99 的概率将两两距离保持在一个常数倍以内，因此自适应采样返回 $(\mathcal{O}(1),\mathcal{O}(1))$-双准则近似的概率至少为0.99，敏感度采样以至少 0.99 的概率返回一个核心集，因此，算法16 为 X 返回核心集的总概率至少为0.96。

□

引理 C.6. 算法16 使用 $\tilde{O}\left(sk+\frac{dk}{\varepsilon^4}\min(,\varepsilon^{2+z})+dk\log n\right)$ 个通信比特。

证明。由于我们应用 Johnson-Lindenstrauss 变换将 P映射到 $\mathbb{R}^{d'}$，其中 $d'=\text{O}(\log|P|)=\text{O}(\log(sk))$，根据定理 2.9，协调器需要向每个站点发送一个 $\text{O}(\log(sk\log(sk))\cdot\log d)$ 比特的随机种子，使每个站点都能生成映射 π。这需要 $s\text{polylog}(s,k,d)$ 比特。

根据引理 C.4，由于 $|P_j|=\text{O}(k)$ 且 $d'=\text{O}(\log(sk))$，我们需要使用 $\text{polylog}(\log n,s,k,\frac{1}{\varepsilon'},\frac{1}{\delta})$ 比特来应用高效通信 $(\pi(P_j)\ \pi(s_{i-1})\ \varepsilon',\delta)$。由于 $|S|=\text{O}(k)$，我们需要使用 $d\,\text{polylog}(\log n,s,k,\frac{1}{\varepsilon'},\frac{1}{\delta})$ 比特来应用高效通信$(S,x,\varepsilon',\delta)$。由于需要应用高效通信$(\pi(P_j)\ \pi(s_{i-1})\ \varepsilon',\delta)$ $\text{O}(sk)$ 次，将 $\pi(s_{i-1})$ 的位置发送给每个站点；并应用高效通信$(S,x,\varepsilon',\delta)$ $\text{O}\left(\frac{k}{\varepsilon^4}\min(,\varepsilon^{2+z})\right)$ 次，将通过敏感度采样抽取的点的位置发送给协调器，因此应用高效通信的总比特代价为

$$sk\,\text{polylog}\left(\log n,s,k,\frac{1}{\varepsilon}\right)+\frac{dk}{\min(\varepsilon^4,\varepsilon^{2+z})}\,\text{polylog}\left(\log n,s,k,\frac{1}{\varepsilon}\right).$$

我们需要将 s_{i-1} 的精确位置发送给协调器，这需要 $\mathcal{O}(dk \log n)$ 比特。每个站点 j 都向协调器发送 $|C_l^{(j)}|$ 和代价 $\left(\pi\left(C_l^{(j)}\right), S_j\right)$ 的常数近似，而协调器需要将 $\sum_{l\le|S|}|C_l^{(j)}|$ 和 $\sum_{l\le|S|}$ 代价 $\left(\pi\left(C_l^{(j)}\right), S_j\right)$ 的常数近似发送给每个站点，总计需要 $\mathcal{O}(sk \log\log n)$ 比特。每个站点都需要向协调器发送 $\widehat{\mu}(P_i)$，这需要 $\mathcal{O}(s\log\log n)$ 比特。协调器需要将要采样的点数 m_i 发送给每个站点，这需要 $\mathcal{O}(s \log k)$ 比特。我们需要 $\mathcal{O}\left(\frac{1}{\varepsilon}\log\log n\right)$ 比特，将权重 $\widehat{\mu}(x)$ 发送给协调器。

因此，算法16 的总通信成本为

$$\tilde{\mathcal{O}}\left(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk\log n\right)$$

比特。 □

现在，我们给出算法在协调器模型下的完整保证。

定理 C.7。 存在一种算法，它以至少 0.96 的概率，在协调器模型下返回一个$(1+\varepsilon)$ 的核心集，其中包含 X，并使用 $\tilde{O}\left(sk + \frac{dk}{\varepsilon^4}\min(,\varepsilon^{2+z}) + dk\log n\right)$ 比特的通信。

证明。 结论立即由引理 C.5 和引理 C.6得出。 □

C.3 一般拓扑上的聚类

我们给出一般拓扑下分布式聚类的形式化定义。设 $V = \{v_1, \ldots, v_n\}$ 为由无向图 $G = (V, E)$连接的 n 个节点的集合，其中每条边$(v_i, v_j) \in E$ 表示站点 v_i 与v_j之间的一条直接通信链路。每个节点 v_i 持有一个本地数据集 X_i，全局数据集为 $X = \bigcup_{i=1}^n X_i$。仅允许沿 G的边进行通信。

目标是计算一组 k 个中心$\mathcal{C} = \{c_1, \ldots, c_k\}$，使其在给定范数参数 $z \ge 1$下最小化全局聚类代价:

$$\mathrm{cost}(X, \mathcal{C}) = \sum_{x\in X} d(x, \mathcal{C})^z,$$

其中 $d(x,\mathcal{C}) = \min_{c_j\in\mathcal{C}} d(x, c_j)$是点 x 到$\mathcal{C}$中最近中心的距离。目标是ˆ计算一组近似中心$\mathcal{C}$，同时最小化 G各边上的通信。

在 [BEL13],中，聚类问题在这一一般拓扑设定下得到研究。他们假设s 个站点由具有 $m = |E|$ 条边的无向图 $G = (V, E)$连接。其算法构造一个$(1+\varepsilon)$-核心集，对于 k-均值的通信量为O $\left(m\left(\frac{dk}{\varepsilon^4} + sk\log sk\right)\right)$ 字长，对于 k-中位数为O $\left(m\left(\frac{dk}{\varepsilon^2} + sk\right)\right)$ 字长。由于在其模型中一个字长需要O $(d\log n)$比特，相应代价对于 k-均值为O $\left(mdk\log n\left(\frac{d}{\varepsilon^4} + s\log sk\right)\right)$ 比特，对于 k-中位数为O $\left(mdk\log n\left(\frac{d}{\varepsilon^2} + s\right)\right)$比特。

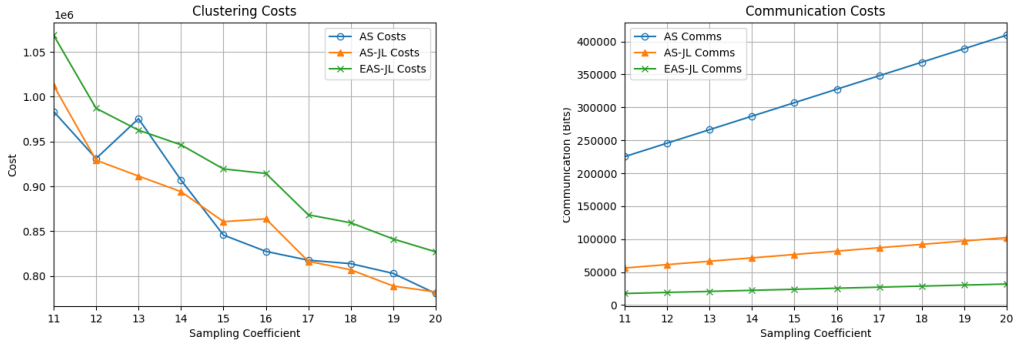
我们的协调器模型算法可以自然地扩展到一般拓扑：发送给协调器的每条消息最多经过 m 条边，使总通信量增加 m 倍。由此得到的通信成本（以比特计）为

$$\tilde{\mathcal{O}}\left(m\left(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk\log n\right)\right).$$

与 [BEL13], 相比，我们的方法改进了对 n 的依赖，将乘法因子$\log n$ 替换为加法项，同时保留了一般(k, z)-聚类的相同近似保证。

D 额外实证评估

在本节中，我们在合成数据集和真实世界数据集上进行了额外的实证评估，以进一步支持我们的理论保证。与 第 6 节中介绍的黑板模型实验不同，我们现在重点评估协调器模型下的算法。作为基线，我们采用基于自适应采样的常数因子近似算法，记为AS。第二种方法使用共享随机性（可通过公共随机性获得），对每个点应用标准降维方法，对应于通信效率更高的变体 AS-JL。具体而言，对于具有 d 个特征的数据集，我们生成一个大小为 $d' \times d$的随机矩阵，其中每个元素均从缩放正态分布 $\frac{1}{\sqrt{d'}} \cdot \mathcal{N}(0,\ 1)$ 中抽取，并将 d'设为比 d小一个常数因子，即 $d' = \frac{d}{2}$或 $d' = \frac{d}{4}$。最后，我们将紧凑编码方案应用于采样点以及自适应采样生成的中心，以此表示我们的协调器模型算法，记为 $\overline{\text{EAS}}$-JL。



(a) 聚类成本　　(b) 通信成本

图7:　DIGITS 数据集上的聚类成本和通信成本实验

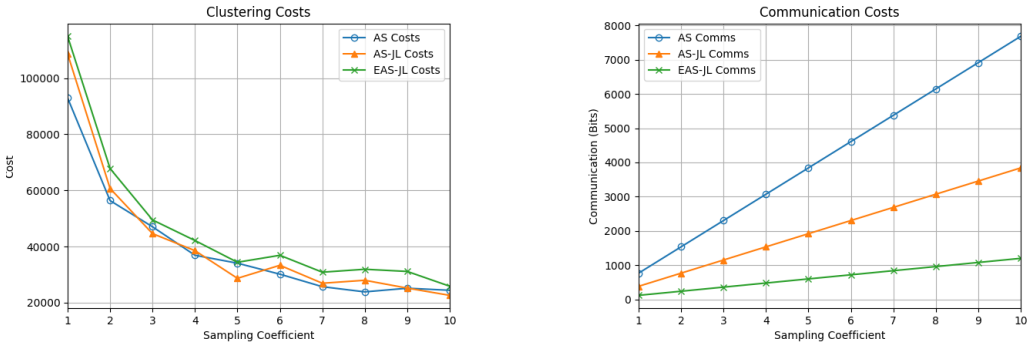
D.1 真实世界数据集

我们首先按照第 6 节中所述，在 DIGITS 数据集 [AK98], 上评估了我们的算法。与之前一样，我们使用参数 $n = 1797$、 $d = 64$ 和 $k = 10$。我们为 AS-JL 和 EAS-JL设置 $d' = \frac{d}{4} = 16$ 。对于 $c \in \{11, 12, \ldots, 20\}$，我们比较了图5a中的聚类成本和图5b中的通信成本。结果表明，我们的算法 EAS-JL 返回的中心的聚类成本始终与其他算法 AS和 AS-JL返回的中心具有竞争力，同时在 $c \in \{11, 12, \ldots, 20\}$的所有设置下，通信量都显著更少。例如，图 7 显示，在 $c = 11$时， EAS-JL 的聚类成本约为 AS 聚类成本的 1.07 倍，同时通信量减少了 $9\times$ 。这一趋势似乎在采样系数 c的整个取值范围内持续存在；例如，在 $c = 20$时， EAS-JL 的聚类成本仍约为 AS 的 1.07 倍，同时通信量仍减少了约 $9\times$ 。

D.2 合成数据集

接下来，我们按照第 6 节中的方法，在由高斯混合构成的合成数据集上评估了我们的算法。具体而言，我们生成了 $k = 3$ 个簇，每个簇包含 640 个点，总计 1920个点，每个点具有 8 个特征，因此数据集总大小为 15360。每个簇均从一个不同的高斯分布中抽取，其均值从范围 $[-10, 10]^8$内均匀随机选取，协方差矩阵则生成为随机正定矩阵，从而形成方向和形状各异的簇。

我们在采样系数取 $c \in \{1, 2, \ldots, 10\}$的范围内比较了聚类成本与通信代价，发现 EAS-JL 在通信量显著更少的情况下，始终取得与 AS 和 AS-JL 相当的聚类性能。例如，在 $c = 5$处，EAS-JL 与 AS 的聚类成本几乎相等，而 EAS-JL 的通信成本优于其超过 $4\times$ 倍。同样，在 $c = 10$ 处， EAS-JL 与 AS 的聚类成本几乎相等，而 EAS-JL 的通信成本几乎优于其 $8\times$ 倍。这些结果与我们在真实世界数据上的发现相一致，表明我们的算法在协调器模型中能够有效保持聚类质量，同时显著减少通信。



(a) (b)

图 8： 合成数据集上聚类成本与通信代价的实验