arXiv:2603.21148v1 [cs.DS] 22 Mar 2026

针对 ℓ_p 度量的快速最近邻搜索

Robert Krauthgamer* NirPetruschka†魏茨曼
科学研究所

2026年3月24日

摘要

最近邻搜索（NNS）问题旨在设计一种数据结构，对位于度量空间 $\mathcal{M}$中的 n 个点的数据集 X 进行预处理，使得给定查询点 $q \in \mathcal{M}$后，能够快速返回 X 中与 q距离最小的点。此类数据结构的效率主要根据其占用的空间大小以及回答查询所需的时间来衡量。我们关注查询时间较快的情形，这对现代大规模应用至关重要：在这些应用中，数据集规模庞大，查询必须在线处理；这一情形通常以查询时间 poly($d \log\ n$) 建模。我们的主要成果是针对 ℓ_p 空间$p > 2$的 NNS 随机化数据结构，它以快速查询时间和 poly(dn) 空间实现 $p^{O(1)+\log\log p}$近似。对于查询时间较快的情形，我们的数据结构优于或无法与以下最先进水平的成果比较： [Bartal 和 Gottlieb，TCS 2019]；以及 [Krauthgamer、Petruschka 和 Sapir，FOCS2025]。

1 引言

最近邻搜索（NNS）问题旨在设计一种数据结构（也称为方案），对一个 n点数据集 X 在度量空间 $\mathcal{M}$中进行预处理，$q \in \mathcal{M}$以便给定一个查询点 X，能够快速返回使其与 q 之间距离最小的点（在近似版本中，则是近似地使该距离最小）。这类数据结构的效率主要根据其空间使用量以及回答查询所需的时间来衡量。预处理时间是次要指标，通常与空间使用量相当。由于最近邻搜索在机器学习、数据分析和信息检索等领域发挥着核心作用，它一直是广泛研究的课题，既包括实践研究，也包括理论研究（例如，参见综述 [AI17、AIR19]）。

众所周知，近似最近邻搜索可以归约为求解 $\mathrm{polylog}(n)$个近似近似邻居问题的实例 [HIM12]。因此，我们将注意力限制在后者上。

定义 1.1. 度量空间 $(\mathcal{M}, d_{\mathcal{M}})$ 上参数为$c \geq 1$, $r > 0$的近似近邻问题，简称 (c, r)-近似最近邻，定义如下：设计一种数据结构，对一个含有 n个点的子集 $X \subseteq \mathcal{M}$进行预处理，以便给定一个查询点 $q \in \mathcal{M}$，且满足 $d_{\mathcal{M}}(q, X) \leq r$，[1] 时，报告一个 $x \in X$ 使得

$$d_{\mathcal{M}}(q, x) \leq cr.$$

在随机化 数据结构，报告的结果以至少 $2/3$ 的概率满足这一条件 $x \in X$。

*Harry Weinrebe计算机科学讲席教授。部分工作获得以色列科学基金会第 #1336/23 号资助。电子邮件：robert.krauthgamer@weizmann.ac.il
†电子邮件：nir.petruschka@weizmann.ac.il

[1]如果 $d_{\mathcal{M}}(q, X) > r$，则可以报告任意结果；通常，$d_{\mathcal{M}}(q, X) := \min_{x^* \in X} d_{\mathcal{M}}(x^*, q)$。

近似	空间	参考文献
$O(\log^{1/p} n \log\log d)$	poly(nd)	[AIK09，和09]
$2^{O((\log d)^{2/3}(\log\log d)^{1/3})}$	多项式(nd)	[近似最近邻+18b]
$2^{O(p)}$	多项式(nd)	[BG19]
$p^{O(1)}$	$n^{O(\log p)}$	[KPS25]
c	$n^{O(p/c)\cdot\log\log n}$	[BBM+25]
$p^{O(1)+\log\log p}$	poly(nd)	定理 1.2

Tab表 1： ℓ_p、 $p > 2$ 中已知的近似最近邻数据结构，适用于快速查询时间范围 e.

我们关注查询时间较短的情形，这对于现代大规模应用至关重要：在这些应用中，数据集规模庞大，必须在线处理查询；这一情形通常用查询时间 poly$(d\log n)$ 来建模。在 ℓ_p 空间中，对于 $1 \le p \le 2$ [IM98、 KOR00、 HIM12]，以及 $p = \infty$ [Ind01、 ACP08、 KP12]，该情形下的近似最近邻（ANN）问题已得到充分理解。对于 $2 < p < \infty$，情况则不太明朗：目前只有少数几种数据结构，分别适用于不同范围的 p，详情见表 1。我们提出了一种适用于 ℓ_p、 $p > 2$ 的近似最近邻新方案，该方案具有较短的查询时间，并在近似与空间之间实现了更优的权衡，具体如下。

定理 1.2。 设 $p > 2$, $d \ge 1$。那么对于每个 $r > 0$，都存在一个随机化数据结构，用于 (c, r)-近似最近邻，定义在 ℓ_p^d 上，其中 $c = p^{O(1)+\log\log p}$；其查询时间为 poly$(d\log n)$，预处理（空间和时间）开销为 poly(dn)。[2]

研究 $\ell_p, p > 2$ 中的近似最近邻问题，在实践和理论上都很重要。现实世界的数据和应用可能会促使人们采用强调离群值的范数（例如用于异常检测），或改变“枢纽点”的出现情况（例如影响分类）；事实上，这类数据结构已用于时间序列分类 [YYW11]，，更多参考文献见 [BG19] 。从理论角度看， ℓ_p 空间的几何性质削弱了现有算法技术的效用，因此需要开发新的技术。一个关键挑战是弥合 $p = 2$ 与 $p = \infty$ 之间的差距。在快速查询时间的情形下，这意味着要在 ℓ_2 [HIM12] 中经典的 $(O(1)\ r)$- 近似最近邻与 ℓ_∞ [Ind01]，中的 $(O(\log\log\ d)\ r)$- 近似最近邻之间进行插值；两者都只使用 poly(dn) 空间。自然可以推测， $2 < p < \infty$ 在这两种保证之间存在某种插值；又由于根据赫尔德不等式， ℓ_∞^d 与 $\ell_{\log d}^d$ 是 $O(1)$-等价的，因此推测这种插值数据结构能够以 poly(nd) 空间实现 $O(\log p)$ 近似。针对这一猜想的第一步是 [AIK0909]，提出了一种从 ℓ_p 到 ℓ_∞ 的归约，并得到具有 $O(\log^{1/p} n \log\log\ d)$-近似的数据结构，它主要适用于 p 较大的情况。 [BG19]，提出了一种非平凡的归约，将 ℓ_p 归约为 ℓ_2，并获得 $2^{O(p)}$-近似；对于 p 较小的情况，这是重大改进，不过其近似效果比猜想差了双指数级。最近， [KPS25]，提出了一种更复杂的归约，实现了 poly(p) 近似，带来了指数级改进。然而，该方法通过递归论证经过多个中间 ℓ_t 空间（$2 \le t < p$），使空间需求增至 $n^{O(\log p)}$，远高于猜想值。我们的定理 1.2 基本完成了对 [KPS25]，的改进：将空间复杂度降回猜想中的 poly(dn)，但近似率略微增至 $p^{\log\log p}$。具体而言，该定理解决了 [KPS 25]，提出的问题：递归能否避免这种更高的空间复杂度?

定理 1.2 的证明基于对已知近邻搜索方案的一项简单而有力的改进，该方案来自 [BG19、KPS 25]，，利用了 [AP90、ABCP98] 关于稀疏

[2]时间和空间上界均与 p.

覆盖。我们在第 [BG19、KPS25] 节概述了3中的算法，并在第 4节开头直观地解释了我们的方法。

1.1 相关工作

现有关于 ℓ_p 空间中近似最近邻搜索的许多研究都聚焦于 $1 \le p \le 2$ [KOR00、IM98、HIM12、DIIM04、AI06、And09、Ngu13]这一情形。在该设定下， $O(1)$-近似可通过$\mathrm{poly}(dn)$ 预处理（空间和时间）实现，查询时间则为关于 $d\log\, n$ [KOR00、HIM12]的多项式。

近年来，针对 $p \in (2,\infty)$ 这一情形取得了重大进展，现有结果大致可分为三个方向。第一个方向的数据结构使用近线性空间，达到适中的近似因子和查询时间 [And09、ANN$^+$18a、ANN$^+$18b、ANRW21、KNT21、AN25]。第二个方向的近似因子较小，例如 $O(1)$，甚至是$1+\varepsilon$；在这种情况下，查询时间和预处理需求（空间和时间）通常都非常大 [BG19、BBM$^+$25]。第三个方向是本文的重点，其查询速度很快，即 poly($d\log\, n$)；现有结果要么以 poly(dn) 空间实现 $2^{O(p)}$-近似 [BG19], ，要么以大得多的 $n^{\omega(1)}$空间 [为代价，实现更好的近似 $o(2^p)$ [BBM$^+$25、KPS25]。我们的结果首次同时实现了 $o(2^p)$近似和 poly(dn) 空间。表 1 对这一方向中所有已知的数据结构进行了比较。

我们指出，这里使用的递归嵌入技术源自 [KPS25], ，也已成功应用于其他涉及 ℓ_p 空间的问题，例如利普希茨分解的构造 [KP25、KPS25、NR26], 几何伸展图 [KP25、KPS25], 以及低失真嵌入 [KPS25、NR26]。

2 预备知识

给定度量空间 $\mathcal{M}=(X,\ d_{\mathcal{M}})$，我们将 $B_{d_{\mathcal{M}}}(x,\ r):=\{y\in X:\ d_{\mathcal{M}}(x,\ y)\le r\}$ 记为以点 $x\in\mathcal{M}$为中心、半径为 $r>0$ 的球。

对于每个 $p,\ q\in[1,\infty)$，马祖尔映射 $M_{p,\ q}:\ \ell_p^d\to\ell_q^d$ 的计算方式是对每个坐标取绝对值的 p/q次幂，同时保留原符号。我们的算法关键依赖于该映射的以下性质。

定理 2.1 ([BL98、BG19])。令 $1\le q<p<\infty$ 和 $C_0>0$，并令 M 为马祖尔映射 $M_{p,\ q}$按因子 $\frac{p}{q}C_0^{p/q-1}$缩小后的映射。那么对于所有满足 x、$y\in B_{\ell_p}(0,\ C_0)$的 x、$y\in\ell_p^d$，

$$\tfrac{q}{p}(2C_0)^{1-p/q}||x-y||_p^{p/q}\le||M(x)-M(y)||_q\le||x-y||_p.$$

3 [BG19 的KPS25]算法概述

本节回顾 [BG19] 和 [KPS25] 在 ℓ_p中的近似最近邻算法 $p>2$。在本节余下部分，固定一个数据集 $X\subset\ell_p^d$，其中 $|X|=n$，对于某个 $p>2$。

3.1 $(2^{O(p)},r)$-近似最近邻，使用 $\mathrm{poly}(dn)$ 空间 [BG19]

在预处理阶段，考虑一组 $k=\frac{p}{2}\cdot O(\mathrm{loglog}\ d)$ 可能的近似因子$\hat{\mathcal{C}}=\{\hat{c}_i\}_{i=0}^k$，其中 $\mathrm{poly}(d)=\hat{c}_0\ge\hat{c}_1\ge\cdots\ge\hat{c}_k=2^{O(p)}$。首先，使用 [Cha98], 为 X 计算一个初始的最近邻搜索数据结构 A_{init}，它提供近似 $\hat{c}_0$ =poly(d) ，查询时间为

poly($d$log n)，空间复杂度为 poly(d) $\tilde{O}(n)$。[3] 接着，对每个数据点 $x \in X$ 以及每个近似因子 $\hat{c} \in \hat{\mathcal{C}}$，为集合 $B_{\ell_p}(0, \hat{c}r) \cap (X - x)$中的点计算一个（缩放后的）马祖尔映射 $M_{x,\hat{c}}$：$\ell_p^d \to \ell_2^d$。最后，对 ℓ_2^d 中的映射点，使用 [HIM12], 计算 $(2, r)$ 近似最近邻数据结构 $A_{x,\hat{c}}$，其查询时间为 poly(d log n)，空间复杂度为 poly(dn)。对于每个点 $x \in X$和每个近似因子 $\hat{c} \in \hat{\mathcal{C}}$，我们都有这个数据结构；显然 $|B_{\ell_p}(x, \hat{c}r)| \le n$，因此总空间需求为 $O(p \cdot \log\log d)n \cdot \text{poly}(dn) = \text{poly}(dn)$。

在查询时，给定查询点 $q \in \ell_p^d$，使用 A_{init} 找到一个 $\hat{c}_0$-近似解 x_0。关键观察是，由于马祖尔映射保证的畸变取决于点集的直径（定理 2.1），因此来自 $A_{x_0.\hat{c}_0}$ 的答案是一个 $\hat{c}_1$-近似解 x_1。迭代应用此过程，近似因子下降的速度甚至快于几何级数，约为 $\hat{c}_i = \hat{c}_{i-1}^{1-2/p}$。因此，经过 $k = O(p\log\log c_0)$ 次迭代后，我们得到一个满足 $\hat{c}_k = 2^{O(p)}$ 的近似解 x_k，此时近似因子不再进一步改善。

3.2 $(\text{poly}(p)\ r)$-近似最近邻，使用 $\text{poly}(d)n^{O(\log p)}$ 空间 [KPS25]

在 [KPS25], 中，马祖尔映射的像空间从 ℓ_2 改为 ℓ_t，以适用于一般的 $1 \le t < p$。将 [BG19], 的结果推广后，使用 ℓ_t^d 中一个具有 $n(Q)$ 查询时间和 $n(S)$ 空间的 (c_t, r)-近似最近邻数据结构，来构造一个适用于 ℓ_p^d、具有 $d(O) + {}_t^pO(\log\log d)Q(n)$ 查询时间和 $n \cdot S(\log\log d)\ {}_t^pO(n)$ 空间的 $(2^{O(p/t)} \cdot c_t, r)$-近似最近邻数据结构。结合 $t = p/2$ 使用上述结果，并递归应用该结果以将 p 降至 2（实际上这是双重递归，因为我们也对 $\hat{c}_i$ 进行迭代），可得到一个具有 poly($d$log n) 查询时间的 (poly(p) r)-近似最近邻数据结构。需要注意的是，每次应用递归步骤都会使数据结构的空间乘以因子 n，因此最终得到的数据结构具有 poly(d)$n^{O(\log p)}$ 空间。

4 $(2^{O(\log p)},\ r)$-近似最近邻，使用 $\text{poly}(dn)$ 空间

本节中，我们给出定理 1.2 的证明。我们首先说明其中的直觉。为简单起见，我们将讨论范围限定为降低 [BG19] 的空间需求；降低 [KPS25] 的空间需求在思路上类似，但技术细节更多。

回顾第 3.1 节，最终解 x_k 是通过迭代求得一系列中间解 x_0，x_1，……，x_{k-1} 得到的。每次迭代 $i < k$ 都会通过找到一个点 x_i并将搜索区域限制到 $B_{\ell_p}(x_i, \hat{c}_i r)$ 来取得进展。该区域具有有界直径，因此在此区域上应用马祖尔映射具有畸变保证。由此可知，查询在 $B_{\ell_p}(x_i, \hat{c}_i r) \cap X)$ 上计算得到的数据结构$A_{x_i,\ \hat{c}_i}$ 可以找到一个点 x_{i+1}，从而将搜索区域进一步限制到直径为 $\hat{c}_{i+1}r$ 的范围内。

预处理阶段会考虑这样一种可能性：每个点 $x \in X$ 都将在查询时作为 $\hat{c}_i$-近似解，也就是说，搜索区域将被限制在 $B_{\ell_p}(x, \hat{c}_i r)$。为了取得进展并进一步缩小搜索区域，将为该区域中的点构建一个数据结构 $A_{x,\hat{c}_i}$。定理 1.2 中的关键思想是，算法不为每个搜索区域分别准备一个数据结构，而是构建一个全局数据结构集合，由这些数据结构共同覆盖所有可能的搜索区域。对于每个 $\hat{c}_i$，算法都会构建一组近似最近邻数据结构，这些结构基于一组子集 $\mathcal{S} \subseteq 2^X$ 计算得到，并满足：对于每个点 $x \in X$，都存在某个 $S \in \mathcal{S}$，使其包含搜索区域 $B_{\ell_p}(x, \hat{c}_i r) \cap X$。此外，每个 $S \in \mathcal{S}$ 的直径至多为 $\beta\hat{c}_i r$，其中某个 $\beta > 1$ 满足条件。我们还希望 $\mathcal{S}$ 中的点总数（计入重复项）尽可能少。预处理算法只需为每个 $x \in X$ 存储一个指向集合 $S_x \in \mathcal{S}$ 的引用，该集合满足 $B_{\ell_p}(x, \hat{c}_i r) \cap X \subseteq S_x$；在查询时，如果 x 作为 $\hat{c}_i$-近似解，

[3] 全文中，记号 $\tilde{O}(f)$ 隐去了关于 f 的对数因子。

算法就查询为 S_x构建的近似最近邻数据结构。由于 S_x 的直径至多为$\beta\hat{}c_i r$，这仍会使搜索区域的直径在下一次迭代中缩小（尽管缩小比例略小）。由于 $\mathcal{S}$ 中的点总数较少，所有近似最近邻数据结构所占用的总内存也会较少。

接下来需要证明，预处理阶段确实能够高效地找到一组具有上述性质的 X 子集。幸运的是，[AP90、ABCP98], 已证明这是可行的，而且这种方法已成为一种基础算法工具，广泛应用于分布式计算、网络设计、路由、图算法和度量嵌入。

定义 4.1（稀疏邻域覆盖 [AP90, ABCP98]）。度量空间 $\mathcal{M} = (X, d_{\mathcal{M}})$ 的 (β, r)-稀疏覆盖是由子集（称为簇） $\mathcal{S} \subseteq 2^X$ 构成的集合，其中每个子集的直径至多为βr，且对每个 $x \in X$，都存在 $S \in \mathcal{S}$，使得 $B_{d_{\mathcal{M}}}(x, r) \subseteq S$。 $\mathcal{S}$中点的总数称为其稀疏度。

定理 4.2 ([ABCP98])。 存在一种算法，给定一个 n点度量空间 $\mathcal{M}$ 以及参数 $\beta > 1$ 和 $r > 0$，可输出一个 (β, r)-稀疏覆盖 $\mathcal{M}$，其稀疏度为 $O(n^{1+1/\beta})$，运行时间为$O(\beta n^{2+2/\beta})$ 。

现在我们可以证明定理1.2，证明结构主要沿用 [KPS25的定理 1。8]。

定理 *1.2*的证明。 设 $X \subset \ell_p^d$ 是一个 n点数据集，其中 $p \in (2, \infty)$。为便于阐述，我们假设 p 是 2的幂；这一点很容易处理，见 [KPS25，定理 1。2]。此外，应用赫尔德不等式后，我们可以假设 $p \le \log d$。

我们使用一个双重递归过程来构造近似最近邻方案。第一次递归假设可以访问一个针对 ℓ_p^d 的近似最近邻方案，该方案的近似因子为 c_{base}，并为 ℓ_p^d 提供一个新的近似最近邻方案，其近似因子得到改进（变小），为 c_{new}。这一步关键依赖于另一个近似最近邻数据结构，该数据结构针对 ℓ_t^d，用于 $t = p/2$，而且实际上也是通过同一种方法构造的。这引出了第二次递归：为中间空间 ℓ_p^d，$\ell_{p/2}^d$，……，ℓ_2^d构造近似最近邻数据结构，其中已知空间 ℓ_2^d 具有$O(1)$ 近似的近似最近邻数据结构。

接下来我们描述第一次递归，即如何在已知一个针对 ℓ_p^d 的 (c_{base}, r)-近似最近邻方案以及一个针对 ℓ_t^d 的 (c_t, r)- 近似最近邻方案的情况下，为 ℓ_p^d 构造一个改进的 (c_{new}, r)- 近似最近邻方案，其中 $t < p$。在预处理阶段，使用定理 4.2 为 X 构造一个 $(\beta, 2c_{\text{base}} r)$-覆盖 $\mathcal{S}$，其稀疏度为 $\tilde{O}(n^{1+1/\beta})$，适用于 $\beta = \log p$。在构造 $\mathcal{S}$ 时，为每个 $x \in X$ 存储一个指向集合 $S_x \in \mathcal{S}$ 的引用，该集合“覆盖”它，即 $B_{\ell_p}(x, 2c_{\text{base}} r) \cap X \subseteq S_x$；稀疏覆盖保证这样的集合存在。此外，为每个 $S \in \mathcal{S}$ （任意地）指定一个中心点 $y \in S$，在 $B_{\ell_p}(0, 2\beta c_{\text{base}} r) \cap (X - y)$ 上应用一个缩小因子为 $\frac{p}{t} \cdot (2\beta c_{\text{base}} r)^{p/t-1}$ 的马祖尔映射 M^y：$\ell_p^d \to \ell_t^d$，并为这些像点构造一个 (c_t, r)-近似最近邻方案 A_S。最后，为 X 构造一个 (c_{base}, r)-近似最近邻方案 A_{base}，并通过标准的独立重复方法，将这两个数据结构的成功概率放大到 $5/6$。给定一个保证满足 $x^* \in X$ 且 $\|x^* - q\|_p \le r$ 的查询 $q \in \ell_p^d$，使用 A_{base} 查询点 q，并获得答案 $x_{\text{base}} \in X$。然后找到它所属的簇 $S_{x_{\text{base}}}$ 及该簇指定的中心 y，使用 $A_{S_{x_{\text{base}}}}$ 查询点 $M^y(q - y) \in \ell_t^d$，再利用其答案 $M^y(z_{\text{out}} - y) \in M^y(X - y)$ 输出相应的 $z_{\text{out}} \in X$。

下一个断言类似于 [KPS25，断言 4。1], 主要区别在于使用了稀疏覆盖。

断言 4.3。 以概率 至少为 $2/3$ 时，我们有 $\|z_{out} - q\|_p \le c_{new} r$，其中 $c_{new} = \binom{p}{t}^{t/p} c_t^{t/p} (4\beta c_{base})^{1-t/p}$。

证明。 以至少 $\frac{5}{6}$ 的概率，数据结构 A_{base} 会输出一个满足 $\|x_{\text{base}}-q\|_p \le c_{\text{base}} r$ 的点 x_{base}。令 $S_{x_{\text{base}}}$ 为 x_{base} 所引用的覆盖中的集合，并令 $y \in S_{x_{\text{base}}}$ 为其指定中心点。由于

$$\|x^* - x_{\text{base}}\|_p \le \|x^* - q\|_p + \|q - x_{\text{base}}\|_p \le 2c_{\text{base}} r,$$

我们得到 $x^* \in B_{\ell_p}(x_{\text{base}}, 2c_{\text{base}} r) \cap X \subseteq S_{x_{\text{base}}}$。注意，根据定理 2.1，$\|M^y(x^*) - M^y(q)\|_t \le r$ 以至少 $\frac{5}{6}$，$A_{S_{x_{\text{base}}}}$ 输出一个点 $z_{\text{out}} \in M^y(S_{x_{\text{base}}}) \subseteq M^y(X)$，满足 $\|M^y(z_{\text{out}}) - M^y(q)\|_t \le c_t r$。应用并集界可知，以至少 $2/3$ 的概率，这两个事件同时成立。在这种情况下，根据定理 2.1，我们有

$$\frac{t}{p} \cdot (4\beta c_{\text{base}} r)^{1-p/t} \cdot \|z_{\text{out}} - q\|_p^{p/t} \le \|M^y(z_{\text{out}}) - M^y(q)\|_t \le c_t r,$$

重新排列后，我们得到 $\|z_{\text{out}} - q\|_p \le (\frac{p}{t})^{t/p} c_t^{t/p} (4\beta c_{\text{base}})^{1-t/p} r = c_{\text{new}} r$。 □

记 $\hat{c}_i$ 为通过应用定理 4.3 i 次得到的近似最近邻方案的近似因子，其中初始近似最近邻方案是来自 [Cha98]，的方案，其近似因子为 $\hat{c}_0 = \text{poly}(d)$。我们还记 c_t 为针对 ℓ_t 的近似最近邻方案的近似因子，该方案采用相同的方法（即递归地）构造，只不过对于 ℓ_2^d，我们使用来自 [HIM12] 的一个 $(2, r)$-近似最近邻方案，其查询时间为 $\text{poly}(d \log n)$，空间和预处理时间均为 $\text{poly}(dn)$。使用定理 4.3（其中 $t = p/2$），并进一步递归应用该方法 $k = \lceil \log(\log \hat{c}_0) \rceil = O(\log \log d)$ 次，我们得到

$$\hat{c}_k \le \sqrt{8\beta c_{p/2} \hat{c}_{k-1}} \le \sqrt{8\beta c_{p/2} \sqrt{8\beta c_{p/2} \hat{c}_{k-2}}} \le \ldots \le 8\beta c_{p/2} \cdot \hat{c}_0^{1/2^k} \le 16\beta c_{p/2},$$

也就是说，该方案的近似比为 $c_p = \hat{c}_k \le 16\beta c_{p/2}$。通过标准方法进行 $O(\log \log p) = O(\log \log \log d)$ 次独立重复，可以将该方案的成功概率提升至 $1 - \frac{1}{3 \log p}$。现在，对 p 递归 $\log p$ 层，可得

$$c_p \le (16\beta)^{\log p} = (16 \log p)^{\log p} = p^{4 + \log \log p},$$

根据并集界，总体成功概率至少为 $\frac{2}{3}$。

接下来分析该算法的查询时间、空间和预处理时间。第二层递归的每一层总共会调用 $= O(\log \log d)$ 次用于 ℓ_t的近似最近邻方案 $k \cdot O(\log \log p)$，对应不同的中间值 t。由于 [HIM12] 中的针对 ℓ_2 的 $(2, r)$-近似最近邻查询时间为 $\text{poly}(d \log n)$，并且回顾 $p \le \log d$，总体查询时间为 $\tilde{O}(\log \log d)^{\log p} \cdot \text{poly}(d \log n) = \text{poly}(d \log n)$。

为分析空间使用量和预处理时间，我们证明以下命题。

命题 4.4。 存在一个绝对常数 $D > 2$，使得在 m 个点上构建用于 ℓ_t^d, $t = 2^i$ 的数据结构时，其总空间使用量和预处理时间为 $\tilde{O}(\log \log d)^i \text{poly}(d) O(m)^{D + i/\log p}$。

证明。 我们只分析该数据结构的空间使用量；预处理时间的分析类似，因为计算马祖尔映射需要 $O(d)$ 时间，而计算稀疏覆盖需要 $O(\log p \cdot m^{2(1+1/\log p)}) \le O(\log \log d \cdot m^{2(1+1/\log p)})$ 时间。

证明通过对 $i \ge 0$ 进行归纳法来展开。对于 $i = 0$，该结论成立，因为在 m 个点上计算时，[HIM12]，的 $(2, r)$-近似最近邻使用的空间至多为 $\text{poly}(d) m^D$，其中某个绝对常数为 $D > 1$。现在，假设该结论对 $i - 1 \ge 0$ 成立。递归第 i 层的近似最近邻方案由两类近似最近邻方案组成。第一类是在全部 m 个点上计算的、来自 [Cha98] 的近似最近邻方案，其空间使用量为 $\text{poly}(d)\ \tilde{O}(m) \le \text{poly}(d)\ O(m)^{D(1+1/\log p)}$。第二类是在不同的 m 个点子集上计算的多个第 $i-1$ 层近似最近邻方案。

对于每个 $j = 0, 1, \ldots, k = \tilde{O}(\log\log d)$，令 $\mathcal{S}^j$ 为针对第一次递归第 j 层的点计算得到的、稀疏度为 $O(m^{1+1/\log p})$ 的稀疏覆盖。对于第一次递归的每一层 j 和每个簇 $S \in \mathcal{S}^j$，算法都会在 S 上计算一个第 $i-1$ 层的数据结构。根据归纳假设，该数据结构的空间使用量上界为 $\tilde{O}(\log\log d)^i \operatorname{poly}(d) O(m)^{D+i/\log p}$。注意，函数 $f\colon x \mapsto \tilde{O}(\log\log d)^i \operatorname{poly}(d) x^{D+i/\log p}$ 满足：对于所有 a、$b \geq 1$，均有 $f(a) + f(b) \leq f(a+b)$；并且，由于每个数据结构都是在某个至多包含 m 个点的 $S \in \mathcal{S}^j$ 上构建的，我们可以将每个点 $p \in S$ 对数据结构总空间使用量的贡献上界为 $f(m)/m$。由于对于每个 j，$\mathcal{S}^j$ 的稀疏度至多为 $O(m^{1+1/\log p})$，因此，在 $\mathcal{S}^j$ 的所有簇上计算的全部近似最近邻方案的总空间使用量至多为

$$O(m^{1+1/\log p})\frac{f(m)}{m} \leq \tilde{O}(\log\log d)^i \operatorname{poly}(d) O(m)^{D+(i+1)/\log p}.$$

因此，总空间使用量为

$$\begin{aligned}&\tilde{O}(\log\log d)\cdot\left(\tilde{O}(\log\log d)^i \cdot \operatorname{poly}(d)\cdot O(m)^{D+(i+1)/\log p}\right) + \operatorname{poly}(d)\cdot\tilde{O}(m)\\ &\leq \tilde{O}(\log\log d)^{i+1}\cdot\operatorname{poly}(d)\cdot O(m)^{D+(i+1)/\log p},\end{aligned}$$

证毕。 □

最后，我们使用定理 4.4 对于 $m = n$ 和 $i = \log p$，得到整个近似最近邻数据结构的空间使用量上界为

$$\tilde{O}(\log\log d)^{\log p}\operatorname{poly}(d)\cdot n^{D+\log p/\log p} \leq \operatorname{poly}(d)\cdot n^{D+1} \leq \operatorname{poly}(dn),$$

由此定理 1.2的证明得证。 □

注 4.5。 将定理 β 证明中稀疏覆盖的参数1.2 从$\log p$改为 $\log_\delta p$，对于 $0 < \delta < 1$可得到一个近似因子稍大的数据结构$p^{O(1)+\log(1/\delta)+\log\log p}$，但其空间需求与 [HIM12] 相匹配，误差至多为关于 d 的次多项式因子，以及一个额外的 n^δ 项。

证明定理时使用的相同技巧1.2，即应用定理 4.2在预处理阶段构造覆盖，也可用于改进一般赋范空间中近似最近邻的空间需求，从 [近似最近邻+18b，定理 3]。更具体地说，对于每个 $0 < \delta < 1$，可以将其数据结构的空间减少一个 $\Omega(n^{1-\delta})$ 因子，代价是近似因子额外增加$O(\delta^{-1})$ 倍。

致谢。 感谢 Shay Sapir 和 Asaf Petruschka 富有帮助的讨论。

参考文献

[ABCP98] Baruch Awerbuch、Bonnie Berger、Lenore Cowen 和 David Peleg。稀疏邻域覆盖的近线性时间构造。《*SIAM* 计算杂志》，28(1):263–277，1998年。doi:10.1137/S0097539794271898。[ACP08] Alexandr Andoni、Dorian Croitoru 和 Mihai Pătraşcu。L∞下最近邻问题的困难性。载于第*49*届 *IEEE* 计算机科学基础年会，*FOCS 2008*，第424–433页。IEEE计算机学会，2008年。doi:10.1109/FOCS.2008.89。[AI06] Alexandr Andoni 和 Piotr Indyk。用于高维空间中近似最近邻的近最优哈希算法。载于第*47*届 *IEEE* 计算机科学基础年会，第459–468页。IEEE，2006年。doi:10.1109/FOCS.2006.49。

[AI17] A. Andoni 和 P. Indyk。高维空间中的最近邻。载于《离散与计算几何手册》第43章，第1135–1150页。CRC出版社，第3版，2017年。doi:10.1201/9781315119601.[AIK09] A. Andoni、P. Indyk 和 R. Krauthgamer。克服 l_1 不可嵌入性障碍：乘积度量算法。载于第19届ACM-SIAM离散算法年度研讨会，第865–874页。工业与应用数学学会，2009年。doi:10.1137/1.9781611973068.94.[AIR19] Alexandr Andoni、Piotr Indyk 和 Ilya Razenshteyn。高维空间中的近似最近邻搜索。载于《国际数学家大会论文集》（ICM 2018），第3287–3318页，2019年。doi:10.1142/9789813272880_0182.[AN25] Alexandr Andoni 和 Negev Shekel Nosatzki。用于最近邻搜索的相似性度量嵌入。载于第66届IEEE计算机科学基础年度研讨会，FOCS，第864–893页。IEEE，2025年。doi:10.1109/FOCS63196.2025.00045.[And09] Alexandr Andoni。最近邻搜索：过去、现在与不可能。博士论文，麻省理工学院，2009年。URL：https://hdl.handle.net/1721.1/55090.[ANN+18a] Alexandr Andoni、Assaf Naor、Aleksandar Nikolov、Ilya P. Razenshteyn 和 Erik Waingarten。通过非线性谱间隙实现数据依赖哈希。载于第50届ACM SIGACT计算理论年度研讨会论文集，STOC 2018，第787–800页。美国计算机协会，2018年。doi:10.1145/3188745.3188846.[ANN+18b] Alexandr Andoni、Assaf Naor、Aleksandar Nikolov、Ilya P. Razenshteyn 和 Erik Waingarten。霍尔德同胚与近似最近邻。载于第59届IEEE计算机科学基础年度研讨会，FOCS 2018，第159–169页。IEEE计算机学会，2018年。doi:10.1109/FOCS.2018.00024.[ANRW21] Alexandr Andoni、Aleksandar Nikolov、Ilya P. Razenshteyn 和 Erik Waingarten。超越空间划分的近似最近邻。载于2021年ACM-SIAM离散算法研讨会论文集，SODA 2021，第1171–1190页。工业与应用数学学会，2021年。doi:10.1137/1. 9781611976465.72.[AP90] B. Awerbuch 和 D. Peleg。稀疏划分。载于第31届IEEE计算机科学基础年度研讨会，第503–513页，1990年。doi:10.1109/FSCS.1990.89571.[BBM+25] Yiqiao Bao、Anubhav Baweja、Nicolas Menand、Erik Waingarten、Nathan White 和 Tian Zhang。平均失真草图。载于第66届IEEE计算机科学基础年度研讨会，FOCS，第277–302页。IEEE，2025年。doi:10.1109/FOCS63196.2025.00019.[BG19] Yair Bartal 和 Lee-Ad Gottlieb。通过嵌入实现 ℓ_p 空间（$2 < p < \infty$）中的近似最近邻搜索。《理论计算机科学》，757:27–35，2019年。doi:10.1016/j.tcs.2018.07.011.[BL98] Yoav Benyamini 和 Joram Lindenstrauss。《几何非线性泛函分析》，第48卷。美国数学学会，1998年.[Cha98] Timothy M. Chan。重新探讨近似最近邻查询。《离散与计算几何》，20(3):359–373，1998年。doi:10.1007/PL00009390.[DIIM04] Mayur Datar、Nicole Immorlica、Piotr Indyk 和 Vahab S. Mirrokni。基于p稳定分布的局部敏感哈希方案。载于第20届计算几何年度研讨会，第253–262页。美国计算机协会，2004年。doi:10.1145/997817.997857.[HIM12] Sariel Har-Peled、Piotr Indyk 和 Rajeev Motwani。近似最近邻：迈向消除维数灾难。《计算理论》，8(1):321–350，2012年。doi:10.4086/TOC. 2012.V008A014.[IM98] P. Indyk 和 R. Motwani。近似最近邻：迈向消除维数灾难。载于第30届ACM计算理论年度研讨会，第604–613页，1998年。doi:10.1145/276698.276876.

[Ind01] Piotr Indyk. 关于 ℓ_∞ 范数下的近似最近邻。计算机与系统科学杂志，63(4):627–638，2001年。doi:10.1006/JCSS.2001.1781。[KNT21] Deepanshu Kush、Aleksandar Nikolov 和 Haohua Tang。通过高效的平均畸变嵌入进行近邻搜索。载于第37届国际计算几何研讨会（SoCG 2021），LIPIcs第189卷，第50:1–50:14页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2021年。doi:10.4230/LIPICS.SOCG.2021.50。[KOR00] E. Kushilevitz、R. Ostrovsky 和 Y. Rabani。在高维空间中高效搜索近似最近邻。SIAM 计算机杂志，30(2):457–474，2000年。doi:10.1137/S0097539798347177。[KP12] Michael Kapralov 和 Rina Panigrahy。通过度量扩张研究 ℓ_∞ 和地球移动距离的最近邻搜索下界。载于第39届自动机、语言与编程国际研讨会（ICALP 2012），计算机科学讲义第7391卷，第545–556页。施普林格，2012年。doi: 10.1007/978-3-642-31594-7_46。[KP25] Robert Krauthgamer 和 Nir Petruschka。有限 ℓ_p 度量的利普希茨分解。载于第41届国际计算几何研讨会（SoCG 2025），莱布尼茨国际信息学论文集（LIPIcs）第332卷，第66:1–66:14页。Schloss Dagstuhl – Leibniz- Zentrum für Informatik，2025年。doi:10.4230/LIPIcs.SoCG.2025.66。[KPS25] Robert Krauthgamer、Nir Petruschka 和 Shay Sapir。递归嵌入在ℓ_p 度量中的威力。载于第66届IEEE计算机科学基础年度研讨会（FOCS），第2546–2554页。电气电子工程师学会，2025年。doi:10.1109/FOCS63196.2025.00132。[Ngu13] Huy L. Nguyen。 ℓ_p中的近似最近邻搜索。CoRR, abs/1306.3601，2013年。URL: http://arxiv.org/abs/1306.3601。[NR26] Assaf Naor 和 Kevin Ren。当 $p > 2$时， L_p 子集的最优随机聚类。载于2026年 ACM-SIAM 离散算法年会论文集（SODA），第984–995页。工业与应用数学学会，2026年。doi:10.1137/1.9781611978971.40。[YYW11] Daren Yu、Xiao Yu 和 Anqi Wu。让最近邻对时间序列分类更有意义。第4届国际图像与信号处理大会论文集，CISP 2011，5:2481–2485，2011年10月。doi:10.1109/CISP.2011.6100672。