Indexing Metric Spaces for Exact Similarity Search 复现报告

摘要

随着社会进程的不断数字化，我们看到可用数据激增。这就是所谓的大数据。在研究环境中，数据的三个方面通常被视为试图从大数据中创造价值时的主要挑战来源：数量、速度和多样性。目前有许多研究涉及数量或速度，而涉及种类的研究较少。度量空间是解决多样性问题的理想选择，因为它可以容纳任何数据，只要这些数据可以配备满足三角不等式的距离概念。为了加速在度量空间中的搜索，人们提出了一系列针对度量数据的索引技术。然而，现有的调查报告覆盖面有限，尚未有一项全面的实证研究报告。因此，本文作者对支持精确相似性搜索的现有度量索引进行了全面调查：总结了度量索引用于支持精确相似性搜索的现有分区、剪枝和验证技术；对索引构建的时间和空间复杂性进行了分析，并对它们的查询处理性能进行实证比较。在评估度量索引性能时，实证研究非常重要，因为性能在很大程度上取决于现有剪枝和验证的有效性以及数据分布，这意味着复杂性分析通常只能提供有限的见解。本文旨在揭示不同索引技术的优缺点，为特定环境下选择合适的索引技术提供指导，并为未来的度量索引研究提供方向。

关键词：度量空间；索引和查询；度量空间相似性查询

1 引言

海量数据正在不断产生。例如，全球导航卫星系统（如 GPS）和基于通信网络的定位技术使车载设备和智能手机能够生成海量时空数据。监控摄像头、智能辅助麦克风和智能手机正在生成海量多媒体数据。此外，Facebook、Twitter 和 WhatsApp 等服务也正在生成大量社交媒体数据。数据的激增提供了创造价值的机会，使企业和社会受益。另一方面，由于数据量大、速度快、种类多，这种状况也带来了严峻的挑战。这里的“量”是指海量数据，“速”是指数据到达的速度快，“多”是指数据来源广泛，结构和数据类型多样。许多研究和产品都涉及数据量或数据速度，而对数据种类的关注则较少。更具体地说，许多平台（如 MapReduce [4]、Hadoop、Spark、Flink、和 Storm）都在解决数据量和速度问题，只有少数平台（尤其是 Azure Cosmos DB5 和 GeminiDB6）通过支持一系列模型（如表格、图形和文档）来关注多样性方面。使用度量空间可以解决多样性问题，因为任何类型的数据，只要能与满足三角形不等式的距离概念相关联，都可以被视为度量数据。因此，通过使用度量空间，可以开发出统一的解决方案，从而从各种数据中创造价值。

，度量空间中的查询在现实生活中的许多应用中都扮演着重要角色，其中相似性查询是核心 [3] [7] [1]。给定一个查询对象，相似性查询会根据相似性的定义找到相似的对象。有许多索引

技术旨在加速度量空间中的查询速度。这篇论文的主要目的是介绍一份全面的调查报告，描述并分析现有的度量索引。本报告对其中的 GHT family 相关内容进行了复现工作。

2 相关工作

此部分对 GHT family 方面内容进行了简要的描述，并对其先分析方法做简要概述。并且本次课程的论文复现工作通过扩大评测范围，将原文中未全部纳入整体评估的 GHT family 全部测试。

2.1 GHT family

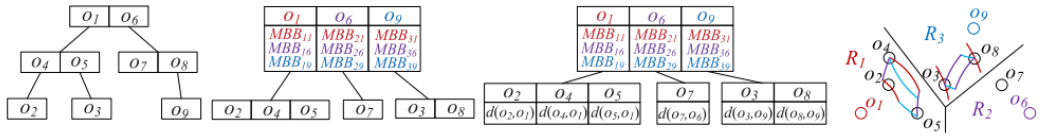

图 1. GHT 及其变体示例

GHT （Generalized Hyperplane Tree）[8] 是通过递归使用广义分区建立的二叉树。在图 1(a) 所示的 GHT 例子中，节点中的对象 o_1 和 o_6 表示两个子树的中心。

GHT 可以推广到 m-ary trees，即 GNAT [2] 。图 1(b) 是 GNAT 的一个示例。在构建 GNAT 时，每次会选择 m 个中心 $c_i(1 \leq i \leq m)$，然后将对象分配到最近的中心。此外，GNAT 还会存储每个节点相对于每个中心 c_j 的最小边界框 $MBB_{ij} = [mindist(o, c_j), maxdist(o, c_j)](o\ R_i)$ ，如图 1(d) 所示，红色圆圈表示与中心 o_1 相对的 MBB，紫色圆圈表示与中心 o_6 相对的 MBB，蓝色圆圈表示与中心 o_9 相对的 MBB。

EGNAT [5] [6] 是 GNAT 的动态版本。图 1(c) 是一个例子。它支持插入和删除，并将 GNAT 扩展为外部内存索引。此外，为了提高剪枝能力，EGNAT 叶节点中的每个条目都存储了该条目与其父条目之间的距离。

GHT 的存储成本为 $O(ns)$，构建成本为 $\Omega(nlog_2 n)$，而 GNAT 的存储成本为 $O(ns+mn)$，构建成本为 $\Omega(mnlog_m n)$，其中 n 表示对象总数，s 表示对象大小，m 表示树节点的最大子节点数量。在最初的研究中 [2]，GNAT 的 MBB（The minimum bounding box of a node）存储成本为 $O(m^2 n_{node})$ ，其中 n_{node} 表示非叶节点的数量。在这里，我们使用 $O(n/m)$ 来估算 n_{node}，这样就能用 $O(mn)$ 的空间来存储 MBB 信息。请注意，GHT、GNAT 和 EGNAT 都是不平衡树，这意味着它们的最差构建成本为 $O(n^2)$。如果我们假设树结构是平衡的，那么 GHT 的最优构建成本为 $\Omega(nlog_2 n)$，GNAT 和 EGNAT 的最优构建成本为 $\Omega(mnlog_m n)$。

3 本文方法

此部分对本文将要复现的工作进行概述，本文对原文章中 GHT family 的全部成员进行了分析，关注了原本被省略掉的索引类型。采用了三个真是的数据集，即 LA，Words 和 Color；

其中 LA 包含洛杉矶的地理位置信息，使用 $L_2 - norm$ 来衡量相似性。Words 包含专有名词、缩略词和复合词，用编辑距离计算词与词之间的距离。Color 包含从 Flickr 中提取的标准 MPEG-7 图像特征，两个特征之间的相似性用 $L_1 - norm$ 来衡量。表1汇总了数据集的统计数据关于 GHT family 各项索引的性能，最后以 PA（页面访问）次数、距离计算次数（Compdists）和运行时间（Running Time（ms））为指标衡量。

表 1. Caption

Datesets	Cardinality	Dimensionality	Intrinsic Dimensionality	Measurement
LA	1,073,727	2	5.4	$L_2 - norm$
Words	611,756	1-34	1.2	Edit distance
Color	1,000,000	282	6.5	$L_1 - norm$

4 复现细节

4.1 与已有开源代码对比

参考了原文作者关于 GHT 构建的代码以及性能分析代码，但本文新增了 GHT family 所有成员的代码实现和分析，能够更加全面的分析各个索引之间的性能差异。

4.2 实验环境搭建

实验环境设备搭建如表2所示。

表 2. 实验环境

CPU	13th Gen Intel(R) Core(TM) i7-13700KF 3.40 GHz
Memory	32GB DDR5
System	Windows 11
Language	C++

5 实验结果分析

本部分对实验所得结果进行分析，详细对实验内容进行说明，实验结果进行描述并分析。原文作者发现，在这三个数据集中，较小的支撑点个数性能最佳。因此，我们选择了五个有代表的值（3,5,10,15,20）。从可以使用 MRQ 查询进行基于主内存的度量索引比较时的数据（如图2-图7）看出，在支撑点较多时，GHT 的性能要优于 GNAT，而 GNAT 和 EGNAT 则在不同数据中有不同的表现，整体上并无明显区分。

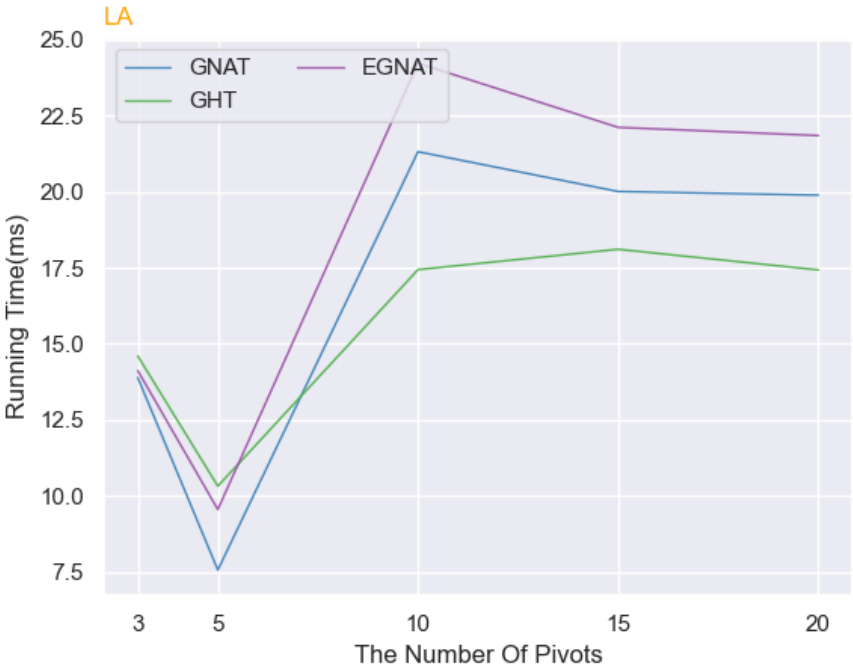

图 2. 使用 MRQ 查询进行基于主内存的度量索引比较（LA-a）

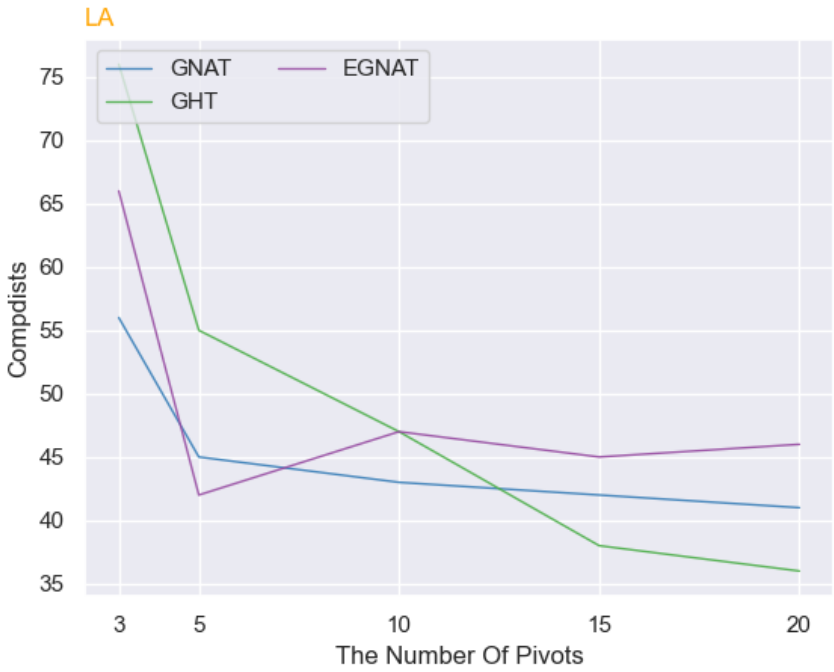

图 3. 使用 MRQ 查询进行基于主内存的度量索引比较（LA-b）

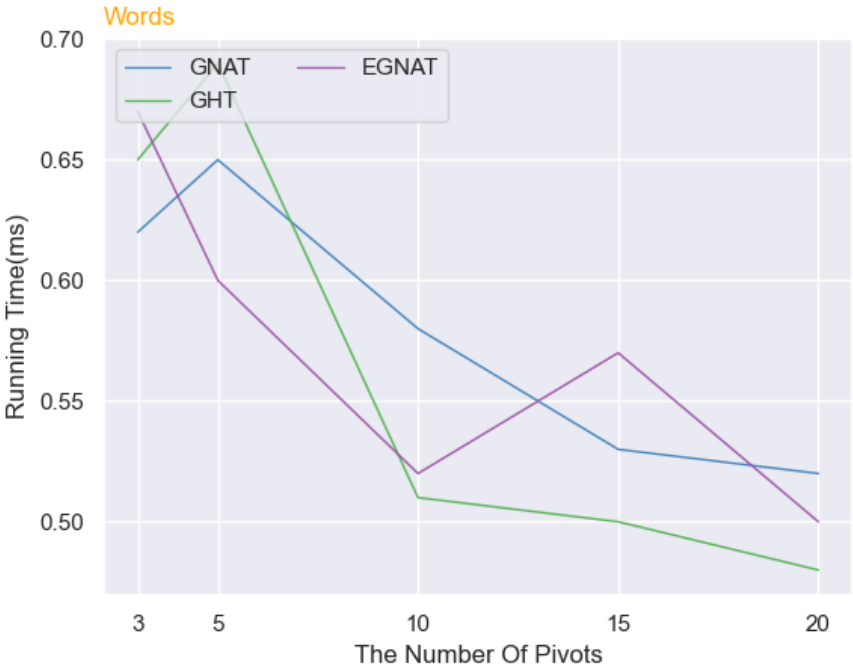

图 4. 使用 MRQ 查询进行基于主内存的度量索引比较（Words-a）

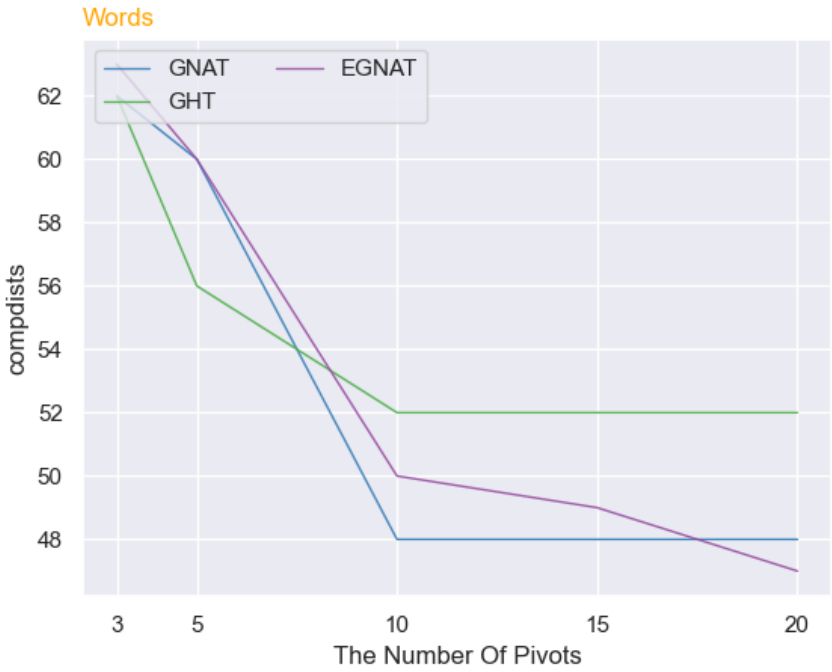

图 5. 使用 MRQ 查询进行基于主内存的度量索引比较（Words-b）

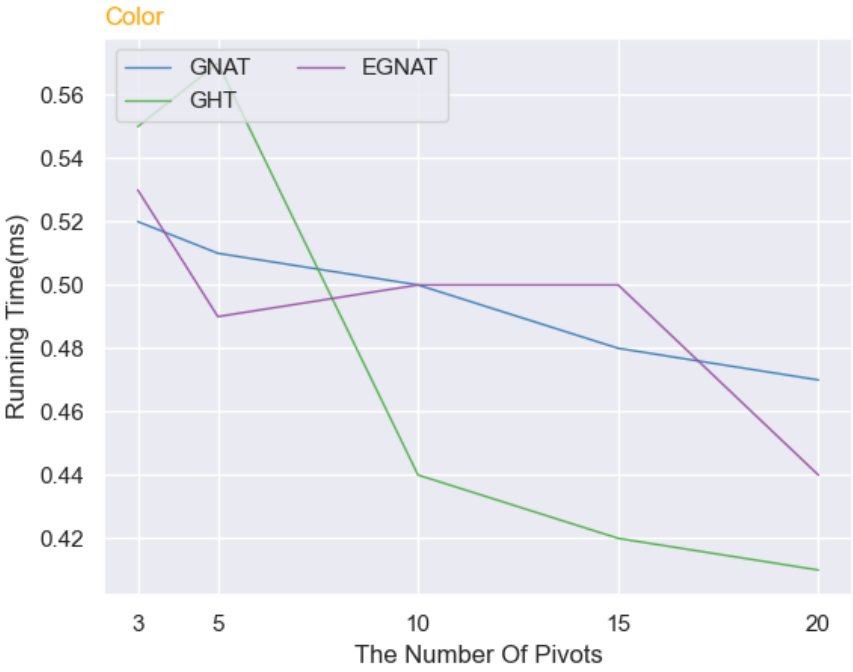

图 6. 使用 MRQ 查询进行基于主内存的度量索引比较（Color-a）

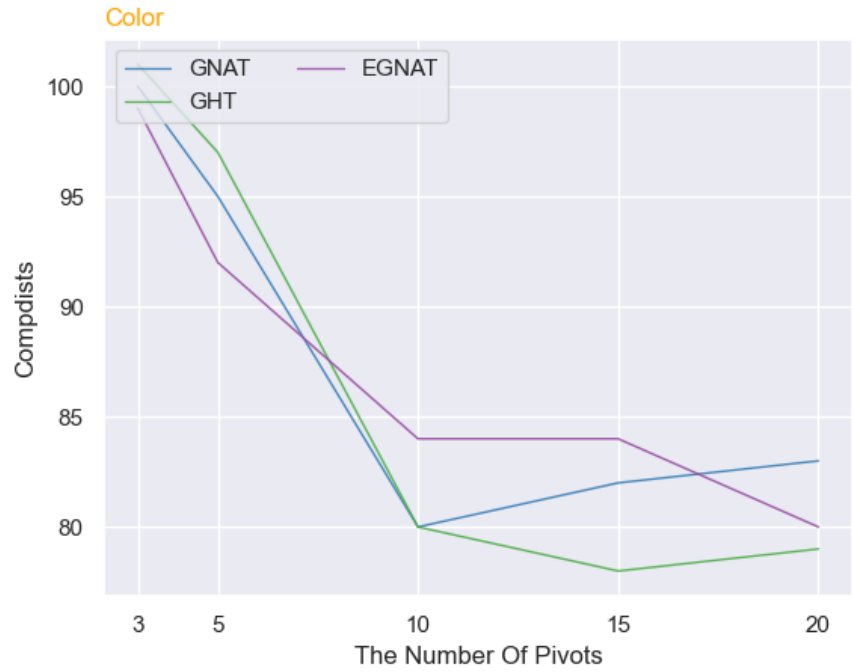

图 7. 使用 MRQ 查询进行基于主内存的度量索引比较（Color-b）

6 总结与展望

本文实现了 GHT family 的性能分析，可以基于所给数据集比较三种索引之间的性能差异，可以补齐原论文中所缺少的分析数据，补全整体逻辑框架。但总体来说，没有通过更多的、更广的数据集来验证分析结果，也仅使用 MRQ 查询进行基于主内存的度量索引比较，没有测试更多的查询算法或者基于硬盘等，这意味着分析结果可能只是针对这种实验条件下 GHT family 索引性能成立。未来可将分析扩大实验范围，从而获得更加全面的索引性能评价数据。

参考文献

[1] Vlastislav Dohnal avel Zezula, Giuseppe Amato and Michal Batko. Similarity search: The metric space approach. *Springer Science Business Media*, 2006.

[2] Sergey Brin. Near neighbor search in large metric spaces. *In Proceedings of the 21th International Conference on Very Large Data Bases*, pages 574–584, 1995.

[3] Ricardo Baeza-Yates Edgar Chávez, Gonzalo Navarro and José Luis Marroquín. Searching in metric spaces. *ACM Computing Survey*, 33(3):273–321, 2001.

[4] effrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Communication of the ACM*, 51(1):107–113, 2008.

[5] Roberto Uribe Mauricio Marin and Ricardo Barrientos. Searching and updating metric space databases using the parallel egnat. *In Proceedings of the International Conference on Computational Science*, page 229–236, 2007.

[6] Gonzalo Navarro and Roberto Uribe-Paredes. Fully dynamic metric access methods based on hyperplane partitioning. *Information Systems*, 36(4):734–747, 2011.

[7] D. A. Rachkovskij. Distance-based index structures for fast similarity search. *Cybernetics and Systems Analysis*, 53(4):636–658, 2017.

[8] Jeffrey K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179, 1991.