高维空间数据索引结构分析研究*)

古 毅 吴中福 魏 丽 钟 将 马金亮
(重庆大学计算机学院 重庆 400044)

摘 要 索引机制是数据库和多媒体领域的重要研究课题,很多在大规模数据集里进行相似性检索的应用都需要有效的高维索引结构来加速查询过程。本文总结了多维索引结构的特点、分类及查询方式,分析了影响索引结构性能的主要因素及其性能评价准则,最后介绍了索引结构的最新发展,并结合多维索引结构目前存在的问题,说明了今后研究的方向。

关键词 多媒体,多维数据,多维索引结构,聚类,金字塔树,数据挖掘

Study of High-dimensional Index Structures

GU Yi WU Zhong-Fu WEI Li ZHONG Jiang MA Jin-Liang
(College of Computer Science, Chongqing University, Chongqing 400044)

Abstract Indexing schemes are significant research issues in the domain of database and multimedia, efficient indexing schemes for high-dimensional data are required for speeding up the similar searching in the large-scale datasets. This paper summarizes the characteristics of multi-dimensional indexing structures, as well as the classification and retrieval types. Subsequently, it analyzes the factors impacting the performance and the evaluation schemes of performance. Lastly, it introduces the newest indexing structures, and combining with current open issues, it also indicates the researching trends in the future.

Keywords Multimedia, Multi-dimensional data, Multi-dimensional indexing structures, Clustering, Pyramid-tree, Data mining

1 引言

随着多媒体应用的日益广泛和多媒体数据的大量增加,如何对海量的多媒体信息进行组织、存储,以及快速有效的检索已经成为人们迫切需要解决的问题,这就需要借助支持快速检索的索引结构。由于传统的数据索引结构(如 B-tree 等)不适用于多维数据,因此多维索引结构的研究变得尤为重要。目前,研究者们已经提出了大量的多维索引结构,并在实际应用中取得了较好的效果。本文将对已有多维索引结构的相关内容进行分析和总结,并结合目前多维索引结构的研究现状提出了今后的发展方向。

2 多维数据及多维索引结构的特点

所谓多维数据,是指多维空间中的数据,一般说来,它具有以下一些特点[1]:

1) 结构复杂:对于多维数据,它有可能是多维空间中的点数据,也有可能是复杂的多边形或多面体,一般不能像传统的关系型数据库一样用固定大小的条目来保存它。

2) 稀疏性:在高维空间中,数据点是非常稀疏的,而且存在空空间现象。

3) 动态性:在数据的插入和删除过程中,还往往伴随着对数据本身的修改。

4) 海量数据:多维数据库的存储空间往往比较大,例如,一般的地图大概就需要几千兆字节的存储空间。

5) 操作多样化:对于多维数据库而言,没有标准操作,往往要根据实际应用的需要来确定。

6) 时间代价大:虽然多维数据库的操作所花费的时间各不相同,但一般远高于传统关系型数据库的操作。

7) 不能排序:无法对空间数据作线性排序以使那些在多维空间中相邻的数据仍然能够相邻。

正是由于多维数据具有以上特点,因此也要求多维索引结构具有以下一些相应的特征[1]:

1) 动态性:对空间数据目标的处理常常包括一些诸如插入和删除等操作,这就要求访问方法的设计和维护必须考虑动态特点。

2) 主从存储器协调性:虽然目前的存储技术提高得很快,但也不可能完全在主存中进行数据库操作。如何合理安排主从存储器之间的任务分配,并考虑并行处理是提高索引效率的重要方法。

3) 支持尽量多的操作:不能以牺牲其它操作来支持某种特定的操作,而且应该能保证操作的并行性和可恢复性。

4) 简单性和鲁棒性:复杂的访问方法可能有时在某一方面的性能优异,但付出的代价却是计算量大、易于出错、且往往鲁棒性较差。这样的结构用在大型系统里一般不如简单、鲁棒性强的结构更为实用。

5) 高效性:一方面要求索引方法对于空间数据的检索比

*)受重庆大学研究生创新基金"基于内容的图像检索引擎"项目的资助,项目编号:200504Y1A0070113。吴中福 教授,博士生导师,主要研究领域为计算机网络与通信、计算机网络安全。古 毅 硕士研究生,主要研究方向为图像数据库检索、多维索引技术。魏 丽 硕士研究生,主要研究方向为计算机网络、基于内容的图像检索。

较快。主要的设计目标就是要达到一维处理中的 B-Tree 性能，而且性能的下降应在对数曲线范围内。另一方面，又希望索引范围相对于整个数据库比较小，即搜索空间小，存储利用率高。

6）集成性：索引结构常常要集成到大规模的数据库系统中去，当嵌入时，它对数据库系统的影响应尽可能地小。

3 多维索引的分类

在近几十年对多维数据索引的研究过程中，已经提出了大量的多维索引结构，图 1 和图 2 给出了两类最常见的树型多维索引结构的发展历史。

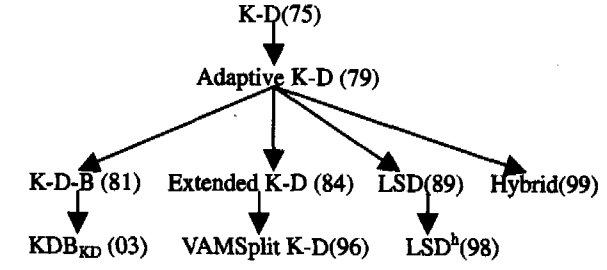

图 1 K-D 树及其变种

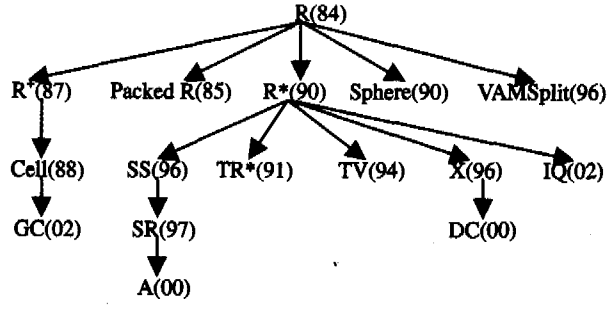

图 2 R 树及其变种

根据多维索引结构的特点，可以将其分为以下几类：

1）根据数据集的切分方法，可分为空间划分法、数据划分法以及混合型划分法。空间划分法是将整个数据空间层层划分为彼此不相交的子空间，如 grid-file[2]、quadtree[3]、K-D-B-tree[4]等；数据划分法则是根据数据所在的位置进行层次划分，如 R-tree[5]、X-tree[6]、TV-tree[7]等；混合型划分是既根据空间进行划分又根据数据进行划分，如 Hybrid-tree[20]等。

2）根据空间的划分原则，又可分为基于特征的划分方法和基于距离的划分方法。基于特征的划分方法是根据数据每个独立维上的值来划分子空间，而与用于计算对象间距离的距离函数无关，如 R-tree[5]、X-tree[6]等；基于距离的划分方法是根据对象间的距离来划分子空间，这里的距离是由一个给定的距离函数计算出来的，如 SS-tree[11]、TV-tree[7]等。

3）根据处理的数据类型则可分为点数据类和空间数据类。点数据类是指那些只能处理点数据的索引结构，如 K-D-tree、TV-tree[7]等；空间数据类是指既能处理点数据，又能处理线、矩形等具有一定形状的数据的索引结构，如 R-tree[5]、R*-tree[9]等。

4）根据索引的组织形式，还可以分为树形结构类和非树形结构类。树形结构类的索引结构是按照树的形式组织的，如 K-D-tree、R-tree[6]等；非树形结构类的索引结构不是按照树的形式组织的，如 VA-File[19]等。

5）根据包络形状，可进一步分为矩形、球形和混合形。包络为矩形的有 R-tree[5]、R*-tree[9]等；包络为球形的有 SS-tree[12]、TV-tree[7]等；将矩形和球形结合起来的则称为混合形，如 SR-tree[12]等。

4 多维索引结构的查询方式

对于空间数据库，由于没有标准的查询语言，其查询方式取决于具体的应用领域，但对于某些给定的数据库，常用的查询方式有以下几种[1]：

1）精确匹配查询：对于给定的查询对象 q，从数据库中找出所有与 q 相同的数据。

2）点查询：给定点对象 p，从数据库中找出所有包含点 p 的数据。

3）窗口查询：给定一个 d 维查询区间 I，从数据库中找出至少包含 I 中一个点的所有数据。

4）相交查询：给定具有一定形状的空间数据 q，从数据库中找出至少包含 q 中一点的所有数据。

5）包含查询：给定查询对象 q，从数据库中找出所有包含对象 q 的数据。

6）被包含查询：给定查询对象 q，从数据库中找出所有被 q 包含的数据。

7）邻接查询：给定查询对象 q，从数据库中找出所有与 q 邻接的数据。

8）范围查询：给定查询对象 q 和查询半径 r，从数据库中找出所有与 q 的距离小于 r 的数据，图 3 给出了三维数据空间中范围查询的例子。

9）K-近邻查询：给定查询对象 q 及正整数 k，从数据库中找出距离 q 最近的 k 个数据，图 4 给出了三维数据空间中 2-最近邻查询的例子。

10）空间连接：给定空间对象集合 R、S 及空间谓词 θ，对象 $i \in R$，对象 $j \in S$，从数据库中找出所有满足 $\theta(i,j)$ 的对象对 (i,j)，其中 $(i,j) \in R \times S$。

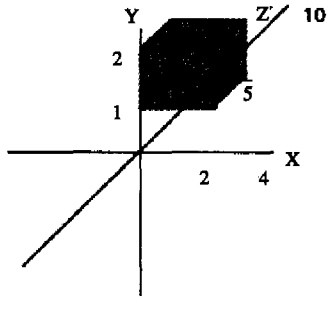

图 3 三维空间中的范围查询

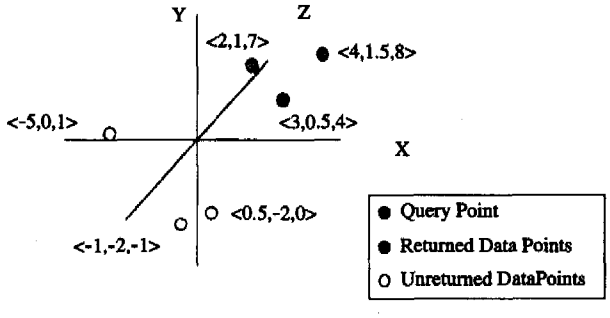

图 4 三维空间中的 2-最近邻查询

5 多维索引结构性能分析

5.1 多维索引结构在性能上的局限

Weber 等人用一种代价模型对基于边界区域(BR)的索引结构和基于空间划分(SP)的索引结构进行了性能上的分析[19],得到了如下几点结论:

1) 对任何通过数据聚类和空间划分而形成的索引结构,总存在一个维数 d,当索引结构的维数超过 d 时,其性能不如简单的顺序扫描。

2) 在任何索引结构上进行相似性查询时,随着维数的升高,其时间复杂度趋近于 $O(N)$。

3) 对任何通过数据聚类和空间划分而形成的索引结构,在进行相似性搜索时,总存在一个维数 d,当索引结构的维数超过 d 时,所有的数据块都会被访问。

5.2 影响多维索引性能的因素及性能评价方法

多维索引结构性能依赖于很多因素和参数,影响多维索引结构性能的因素主要包括使用的硬件、操作系统、页面大小和数据集等。影响多维索引结构性能的参数主要包括测试数据集的数据分布,数据模型,数据集大小等。

由于影响多维索引结构性能的因素是多方面的,目前还没有很好的评价标准,因此,衡量多维索引结构性能的因素主要包括磁盘访问次数、查询时间、存储空间利用率等。

针对多维索引结构目前存在的性能问题,研究者们提出了以下几种主要的性能优化方法:

1) 选择合适的页面大小:多维索引结构的性能随页面大小的变化而变化,因此,如何选择合适的页面大小也是一个比较重要的因素,图 5 展示了页面大小对 X-tree 性能的影响。

2) 减少重叠区域:在基于边界区域的多维索引结构中,随着维数的增加,区域间的重叠现象变得尤为突出,从而大大增加了 I/O 次数。因此,减少重叠区域对基于边界区域的多维索引结构的性能有着非常重要的作用。

3) 维数约减:由于传统的多维索引结构大多数都存在"维度灾难"问题,因此,降低高维数据的维数以适用于传统的多维索引结构也是一个比较有效的方法。

4) 减少距离计算次数:在检索过程中距离计算的开销是比较大的,因此,可以利用三角不等式剪枝等方法来减少距离计算次数,从而提高多维索引结构的性能。

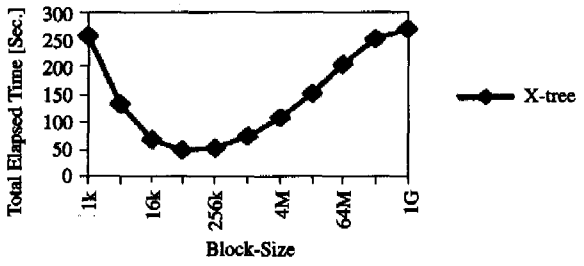

图 5 页面大小对 X-tree 性能的影响

6 多维索引结构的最新研究进展

前面已经对多维索引结构进行了详细分类,实际上,多维索引结构主要包括 R-tree 及其变种、K-D-tree 及其变种、网格文件类、空间填充曲线、VA-File 类等,它们各自的特点及适用范围在文[1]中已有详细介绍。最近研究者又提出了几种新的索引结构,下面我们就对几种比较有代表性的多维索引结构进行介绍:

1) 聚类金字塔技术

Berchtold 于 1998 年提出了金字塔方法[21],该方法基于一种特殊的优化高维数据的不均衡分割策略,其基本原理是先将 d 维空间分割成 2d 个金字塔,共享数据空间的中心点为顶点,然后再将每个金字塔分割成平行于金字塔基的数据页。这样的分割可以有效地按任何一种一维有序索引结构存储。Berchtold 从理论和实验上证明:当处理范围查询时,这种分割策略的性能优于其它的分割策略,而且采用金字塔方法的查询处理效率不会随着维数的增加而降低,因此金字塔树的性能远远超过了 R*-tree[9],X-tree[6] 等索引方法。

但是这种分析结果是基于均匀数据分布和超立方体查询的,对于那些覆盖数据空间边界的查询,或者非常偏斜的查询效果就不理想,而现实世界中的数据很少是服从均匀分布的。因此,张海勤等人于 2001 年提出了聚类金字塔树[23]的概念,将金字塔方法的应用扩展到不均匀数据集上。其基本原理是:对不均匀 d 维数据分布,首先进行聚类操作,将数据分成若干个互不相交的超矩形,保证每一个超矩形内的数据基本趋于均匀分布,每一个超矩形对应一个数据类。然后对每一个数据类建立起相应的金字塔集合,即每一个数据类对应 $2d$ 个金字塔。聚类金字塔方法使用 B^+ 树来储存数据项,充分利用 B^+ 树的优点(如快速的插入、更新和删除操作,优良的并发控制和恢复等)。两者的差别是金字塔方法只使用一棵 B^+ 树,由于聚类金字塔树可能使用几棵 B^+ 树,虽然索引的创建和查询处理复杂了许多,但在性能上得到了很大的改进。

聚类金字塔方法可以有效地索引不均匀分布的数据,适合于各种形状的范围查询,对只给定少数属性值的偏范围查询,聚类金字塔方法的处理结果稍差一点,但迄今为止还没有任何一个高维索引方法可以有效地处理偏范围查询。

2) NB-tree

目前,很多的多维索引结构都具有复杂且不易实现的结构和算法,然而算法的复杂程度与其性能的提高是不成比例的。为此,ManueJ. Fonseca 等人于 2003 年提出了 NB-tree[22] 索引结构。其基本思想是:计算特征向量的 Euclidean 距离,将 d 维特征向量映射为 1 维索引值,即降维,然后将这些值通过 B^+-tree 来组织(B^+-tree 除了时间复杂度低之外最大的优点是很容易实现顺序查找,而且目前的大部分 DBMS 系统都支持 B^+-tree),使得在高维空间中相邻的点在映射后仍然相邻。

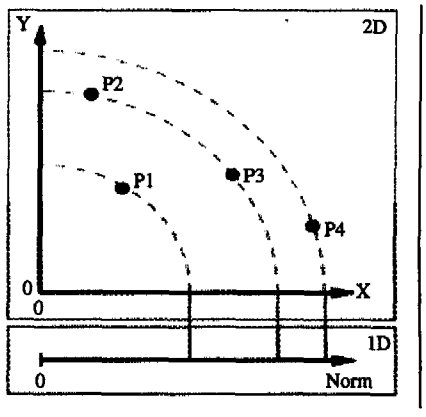

图 6 NB-tree 降维原理

NB-tree 能很好地实现顺序、最近邻和范围查找,并能够较好地适应维数和数据集大小的变化。图 6 展示了 NB-tree

将 $2d$ 空间映射到 $1d$ 的原理，图 7 给出了 NB-tree 在二维空间中的范围查询例子。

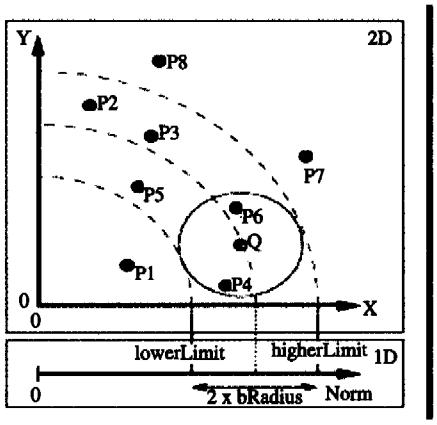

图 7　二维空间中的范围查询例子

结束语　虽然研究者已经提出了大量的多维索引结构，但是它们各有优缺点，目前，还没有证明某种索引结构在所有的指标上都优于其它索引结构，而只是在某些方面表现出优越的性能，主要原因是定义优化的原则和决定性能的参数太多。本文只是对这些索引结构进行了概括性的介绍，并对其性能进行了初步分析。从前面的介绍可以看出，多维索引结构还存在很多问题，比如数据的划分策略、性能的评价标准以及高维数据挖掘等。因此，多维索引结构还有很多问题值得研究，必将会引起人们更大的兴趣和投入更多的精力。

参 考 文 献

1 Gaede V, Gunther O. Multidimensional Access Methods. ACM Computing Surveys, 1998, 30(2)

2 Nievergelt J, Hinterberger H, Sevcik K. The grid file: An adaptable, symmetric multikey file structure. In: Proc. of the Third ECT Conf. 1981. 236～251

3 Finkel R, Bentley J. Quad-trees: A data structure for retrieval on composite keys. ACTA Information, 1974(4): 1～9

4 Robinson J T. The K-D-B-tree: A search structure for large multidimensional dynamic indexes. In: Proc. of the ACM SIGMOD Intl. Conf. on Management of Data. 1981. 10～18

5 Guttman A. R-tree: A dynamic index structure for spatial searching. In: Proc. of the ACM SIGMOD Intl Conf on Management of Data. 1984. 47～54

6 Berchtold S, Keim D, Kriegel H-P. The X-tree: An index structure for high-dimensi0nal data. In: Proc. of the 22nd Intl. Conf. on Very Large databases. 1996. 28～39

7 Lin K-I, Jagadish H, Faloutsos C. The TV-tree: An index structure for high-dimensional data. The VLDB Journal, 1994(3): 517～549

8 Weber R, Schek H-J, Blott S. A Quantitative Analysis and Performance study for Similarity-search Methods in High dimensional Spaces. In: Proc. of 24th VLDB Conf. 1998. 194～295

9 Beckmann N, Kriegel H, Schneider R, et al. The R * -tree: An Efficient and Robust Access Methods for Points and Rectangles. In: Proc. ACM SIGMOD Int. Conf. On Management of Data. Atlantic city, NJ, 1990. 322～331

10 Kamel I, Faloutsos C. On packing R-tree. In: Proc. of 2nd Intl. Conf on Information and Knowledge Management, 1993. 490～499

11 White D A, Jain R. Similarity indexing with the SS-tree. In: Intl. Conf. on Data Engineering (ICDE), New Orleans, LA, 1996, 03: 516～523

12 Katayama N, Satoh S. The SR-tree: An index structure for high dimensional nearest neighbor queries. In: Conf. on Management of Data, 1997. 369～380

13 Indyk P, Motwani R. Approximate nearest neighbors: Toward removing the cause of dimensionality. In: Proc. ACM Symp Theory of Computing, 1998. 604～613

14 Kushlievitz E, Ostrovsky R, Rabani Y. Efficient search for approximate nearest neighbor in high-dimensi0nal spaces. In: Proc. ACM Symp Theory of Computing, 1998. 614～623

15 Zezula P, Savino P, Amato G, et al. Approximate similarity retrieval with M-trees. VLDB, 1998, 7(4): 294～307

16 Sakurai Y, et al. The A-tree: An index structure for high-dimensional spaces using relative approximation. In: Proc. of the 26th VLDB Conf. 2000

17 Fonseca M J, Jorge J. Indexing High-Dimensional Data for Content-Based Retrieval in Large Databases. In: Proc. of the 8th Intl. Conf. on Database Systems for Advanced Applications (DASFAA 03), 2003

18 Cha G-H, Zhu X, Petkovic D, et al. An Efficient Indexing Method for Nearest Neighbor Searches in High-Dimensional Image Databases. IEEE Transactions on Multimedia, 2002, 4(1): 76～87

19 Weber R, Schek H-J, Blott S. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In: Proc. of the Int'l Conf. on VLDB, 1998. 194～205

20 Chakrabarti K, Mehrotra S. The Hybrid Tree: An index structure for high-dimensional feature space. In: Proc. Int. Conf. on Data Engineering, 1999. 440～447

21 Berchtold S, Böhm C, Kriegel H P. The Pyramid-Technique: Towards Breaking the Curse of Dimensionality. In: Proc. of the International Conference on Management of Data (SIGMOD'98), ACM Press, 1998

22 Fonseca M J, Jorge J A. Indexing high-dimensional data for content-based retrieval in large databases: [Technical report]. INESC-ID. http://virtual. inesc-id. pt/tr/mjf-jaj-TR-01-03. pdf, 2003

23 张海勤，欧阳为民，蔡庆生. 聚类金字塔树：一种新的高维空间数据索引方法. 中国科学技术大学学报，2001，31(6)

(上接第 141 页)

规模资源共享(即 N 很大)的网格环境中可以大大地缩短资源遍历的时间。

参 考 文 献

1 Foster I. The Grid: A New Infrastructure for 21st Century Science [J]. Physics Today, 2002, 55(2): 42～47

2 杨广文，等. 一种全局统一的层次化网格资源模型[J]. 计算机研究与发展，2003，40(12)：1763～1769

3 肖侬，等. 基于资源目录技术的网格系统软件设计与实现[J]. 计算机研究与发展，2002，39(8)：902～906

4 Tarjan R E, Vishkin U. An efficient parallel biconnectivity algorithm [J]. SIAM Journal of Computer, 1985 (14): 862～874

5 熊家军，等. 树的后根遍历的一种并行算法[J]. 小型微型计算机系统，2002，23(5)：580～582

6 Wilkinson B, Allen M. Parallel programming: techniques and applications using networked workstation and parallel computers [M]. Prentice Hall Inc., 1999

7 严蔚敏，吴伟民 编著. 数据结构(C 语言版)[M]. 北京：清华大学出版社，1997

8 Karp R M, Ramachandran V. Parallel Algorithms for Shared-Memory Machines [M]. Handbook of Theoretical Computer Science, vol A, J. van Leeuwen Ed., MIT Press, 1990. 869～941