ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.6, June 2008, pp.1401−1412
DOI: 10.3724/SP.J.1001.2008.01401

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于聚类分解的高维度量空间索引 B$^+$-Tree*

张军旗, 周向东+, 王 梅, 施伯乐

(复旦大学 计算机与信息技术系,上海 200433)

Cluster Splitting Based High Dimensional Metric Space Index B$^+$-Tree

ZHANG Jun-Qi, ZHOU Xiang-Dong+, WANG Mei, SHI Bai-Le

(Department of Computing and Information Technology, Fudan University, Shanghai 200433, China)

+ Corresponding author: E-mail: xdzhou@fudan.edu.cn

Zhang JQ, Zhou XD, Wang M, Shi BL. Cluster splitting based high dimensional metric space index B$^+$-Tree. *Journal of Software*, 2008,19(6):1401−1412. http://www.jos.org.cn/1000-9825/19/1401.htm

Abstract: In order to improve the query efficiency, *K*-means cluster approach is often used to estimate the data distribution in the context of high dimensional metric space index. But in previous work, the parameters of clustering are usually selected according to some heuristic manner. This paper presents a new high dimensional index approach—cluster splitting based high dimensional B$^+$-tree. Through cluster splitting, the data space is partitioned more finely to reduce the cost of data access. The relationship between cluster and the query cost is discussed, and based on the query cost model, this paper give formulas to compute the "optimal" parameters of the cluster which can minimize the query cost in theory. Experiment results show that the efficiency of the methods is better than iDistance, M-Tree and sequence scan, and the parameters computed by the formulas are very close to the real optimal one.

Key words: high dimensional space; index structure; query cost model; cluster partition

摘 要: 为了提高索引性能,高维度量空间索引通常采用 *K*-Means 等聚类技术来获取数据的分布信息.但是,已知的工作需要根据经验来确定聚类参数,缺乏对聚类与查询性能之间关系的理论分析.提出了一种基于聚类分解的高维度量空间 B$^+$-tree 索引,通过聚类分解,对数据进行更细致的划分来减少查询的数据访问.对聚类与查询代价的关系进行了讨论,通过查询代价模型,给出了最小查询代价条件下的聚类分解数目等理论的计算方法.实验显示,提出的索引方法明显优于 iDistance 等度量空间索引,最优聚类分解数的估计接近实际最优查询时所需的聚类参数.

关键词: 高维空间;索引结构;查询代价模型;聚类分割

中图法分类号: TP391 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant No.60403018, 60773077 (国家自然科学基金); the National Basic Research Program of China under Grant No.2005CB321905 (国家重点基础研究发展计划(973)); the Postdoctoral Science Foundation Funded Project of China under Grant No.20070420257 (中国博士后科学基金); the Natural Science Foundation of Shanghai of China under Grant No.04ZR14011 (上海市自然科学基金); Collaboration Plan of AMD with Universities (AMD 大学合作计划)

Received 2006-07-28; Accepted 2006-12-06

高维数据索引技术在图像、视频、时间序列等基于内容查询(content-based search)中扮演着十分重要的角色,是数据管理领域一个倍受长期关注的研究热点.目前,已知的高维索引技术大致可分为:向量空间索引,如Kd-tree(1975 年)[1],R-tree[2],X-tree[3]等;度量空间索引,如 M-tree[4]等.另外,还有如 VA-file[5]等技术.一般认为,R-tree 奠定了高维索引的技术框架.国内也开展了相应的研究,如周学海等人[6]提出 ER-Tree 动态索引结构、冯玉才等人[7]提出的一种基于距离的相似索引结构 opt-树及其变种、周项敏等人[8]提出的一种高维数据空间分割策略.

由于在高维向量空间中进行距离计算的代价十分高昂,随着数据维数的增加,R-tree 的查询性能会迅速下降[3].后来出现的度量空间索引,选取一定量的距离参考点(reference point)来提前计算数据对象到各参考点的距离,按照距离对数据进行索引,查询时根据距离三角不等式来筛选数据,减少计算代价.如 M-tree 采用最优查询候选队列实现 KNN 查询,对与查询覆盖区域相交的数据区域进行搜索;Ciaccia 等人根据数据对象间的距离分布建立了 M-tree 的查询代价模型[9].但是,多数度量空间索引对数据的分布考虑不足[10],仅根据距离三角不等式进行数据过滤,查询效率并不理想.

近年来,通过把高维数据对象映射到一维距离空间,采用 B^+-tree 进行索引得到了较深入的研究[10−13].为了获得数据分布信息,iDistance[10,11]采用 *K*-means 聚类方法对数据分布进行分析,并在此基础上进行 B^+-tree 索引.由于高维空间中普遍存在聚类相互重叠,聚类内部数据分布比较稀疏等现象,传统查询算法——凡是与查询覆盖区域相交的据数据区域(如聚类)都进行数据搜索,会引起对大量聚类子类的搜索,查询效率不高[15].iDistance 通过选择合适的初始查询半径,并进行逐步扩展来克服上述问题.但是,其查询半径的扩展需要根据经验(或反复试验)来确定,缺乏理论与系统化的方法.已知的工作虽然对如何选取最优参考点进行了大量的研究,但是,聚类与查询效率的关系等问题并未得到充分的讨论.例如在索引参考点确定后,选取不同的子聚类数目将如何影响查询效率?另一方面,鉴于高维数据集上进行 *K*-mens 聚类时,子类越多(*K* 越大),聚类的计算代价越高昂,如何在 *K* 值一定的情形下最小化查询代价,也是采用聚类分析的高维度量空间索引所必须解决的问题.

针对上述问题,本文提出了一种新的高维度量空间 B^+-tree 索引:1) 为了克服聚类数据稀疏、相互重叠等问题,提出了聚类分解的概念,即按聚类中心把聚类超球分解为不同半径的空腔超球体(也称为聚类环),对这些空腔超球体分别进行索引,有利于提高数据过滤效率;2) 对聚类进行分解其实是根据数据分布对数据进行更细致地划分.显然,过度地划分数据并不能带来更高的查询效率.为了考察聚类、聚类分解与查询代价的关系,本文给出采用聚类分解的高维度量空间 B^+-tree 索引的查询代价模型,即通过对数据点间距离的概率分布的估计,求得 KNN 查询代价的数学期望,并进一步推导出使得查询代价最小化的聚类分解数的计算公式;3) 对查询代价最小条件下的数据聚类数目进行了讨论.

本文方法在包含 60 000 余幅图像数据的通用测试数据集上进行了充分的实验.实验结果显示,新的索引方法的查询效率明显优于 iDistance,M-tree 等索引结构.按照本文公式计算出的聚类分解数目建立的索引,查询代价与实际最小查询代价的误差在 3%以内.

本文第 1 节给出相关工作与背景知识.第 2 节提出新的索引结构.第 3 节介绍最优聚类数目的估计方法.第 4 节给出实验结果与分析.最后是本文的结论.

1 相关工作

高维索引主要分为向量空间索引与度量空间索引两类.向量空间索引,如 R 树,用区域来表示划分后的数据对象子集,采用类 B^+-tree 的索引结构对这些数据区域进行索引[2].当维数增高时,索引区域之间的重叠现象加剧,距离计算的代价十分高昂.

为了减少高维向量间的计算代价,度量空间索引通过选取一定的距离参考点(reference point),提前计算数据对象到各参考点的距离.按照此距离,对数据对象进行索引.度量空间索引,如 M-tree 是基于页面的动态平衡树存取结构,通过层次聚类,把数据对象聚集到度量空间中对应的超球体中(球体的实际形状与所定义的度量空间相关),并采用最优 KNN 查询策略[15]来最小化 I/O 与计算代价.由于对树节点的利用率较低,影响了 M-tree 的

查询效率.

Filho 等人[12]提出了 Omni 方法,选择一些数据对象作为全局索引参考点并计算所有数据对象与每个参考点的距离.如有 m 个参考点,每个数据对象就会有 m 个距离,称为 Omni 坐标.Omni 索引可以建立在 B+-tree 或 R-tree 等结构上,如 Omni B+-tree 用 m 个 B+-tree 来索引对象的 m 个 Omni 坐标,但是查询需要在 m 个 B+-tree 上进行.Omni 方法通过减少查询操作的距离计算量来提高查询性能,然而每个数据点有多个坐标增加了页面访问,另一方面,计算 m 个候选集合的交集带来了额外的代价.Cui 等人[11]提出了考虑数据分布的 iDistance 高维索引方法,即通过聚类获取数据的分布信息,并提出了参考点的选取方法,iDistance 将各个聚类中的数据点到参考点的距离映射到 B+-tree 进行一维索引.通过聚类获取数据对象在向量空间中的部分位置信息是 iDistance 提高索引性能的重要手段之一.图 1 显示了通过聚类与选取参考点的方法提高过滤效果的情况,p 是最佳参考点,查询点为 q,查询半径为 rq.当聚类 c 与查询区域相交时,通过度量空间中三角不等式进行过滤,只需查询聚类 c 中以 p 为圆心、内外半径为 $r1$,$r2$ 的圆环与聚类 c 相交的深色区域中的点.

高维空间中的聚类效果并不理想,表现为聚类不够紧密,聚类内数据分布不均匀,在相当广的聚类边缘区域数据密度非常稀疏,使得整个聚类半径显得"过大",造成大量聚类相互重叠.因此,单纯地根据查询区域与聚类半径的相交与否来确定整个聚类是否进入查询候选集,不能有效地过滤与查询无实质关联的聚类[16].iDistance 的搜索算法通过实验(或经验)设定初始查询半径,然后逐步扩大查询半径进行 KNN 查询.但是,这种逐步扩大查询半径的方法面临如下问题:初始查询半径和查询半径的递增值不能预知;初始查询半径与递增值过大会引起不必要的查询,过小则需要多次重复查询,导致查询效率降低;需用通过反复实验才能获得最佳的初始查询半径以及查询扩展的递增值.

本文提出一种新的基于聚类分解的高维度量空间 B+-tree 索引.为了克服前述问题,本文提出了聚类分解的概念,即按聚类中心把聚类超球分解为不同半径的空腔超球体(也称为聚类环),对这些空腔超球体分别进行索引,提高数据过滤效率.图 2 给出了聚类分解为两个聚类环的图示说明,聚类被分为核心聚类环与边缘聚类环.使用聚类分解前,只要边缘聚类环与查询区域相交,整个聚类中全部数据点就都需要进行访问;聚类分解后,此情况下只需查询边缘聚类环中的点.

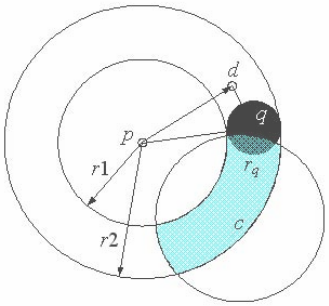


Fig.1 Cluster based filtering

图 1 基于聚类的数据过滤

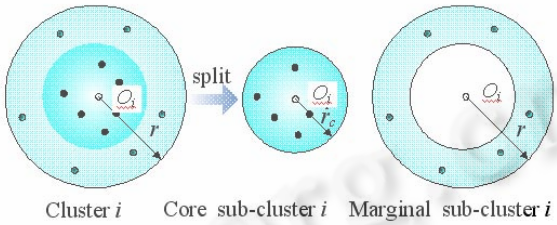


Fig.2 Cluster splitting

图 2 聚类分解

与 iDistance 的查询半径扩展一样,单纯的聚类分解只是一种启发式的方法,在实际应用中,聚类应该分解到什么程度才能获得最佳(最小查询代价)的查询效率是一个必须解决的问题.本文采用 Ciaacia 等人[14]提出的度量空间索引查询代价的估计方法,建立了基于聚类分解的 B+-tree 查询代价模型,对具有最小查询代价的聚类分解、聚类与查询代价的关系等问题进行了研究.

Ciaacia 等人[9,14]根据度量空间中数据对象间的距离分布提出了度量空间索引查询代价模型,并基于代价模型来调整 M-tree 的节点大小,以提高索引性能.利用此代价模型,可以预测度量空间中范围查询与最近临查询的 I/O 与 CPU 代价.

2 基于聚类分解的索引结构

本文提出的高维索引结构首先对数据集进行聚类分析,然后根据最佳聚类分解数目进行“聚类分解”.由于距离是单维值,数据可以按照与参考点的距离并结合所属的聚类环编号进行排序,然后索引到 B^+-tree 中.下面首先给出索引建立时需要注意的几个问题:

(1) 索引参考点选取:本文采用 Shen[15]等人提出的方法进行参考点选取.即通过 PCA 分析,在方差最大的 Principal Component 方向上选择最佳参考点,根据数据对象与最佳参考点的距离建立索引.

(2) 聚类分解:根据聚类分解公式以及聚类分解方法(将在第 2.1 节详述)进行聚类分解.

(3) 索引键值(index key)的计算:用于建立 B^+-tree 索引的 Index key 的计算公式[11]:

对于数据点 $p(x1,x2,\ldots,xd)$,且 $0\le xj\le 1,0\le j\le d$,则数据点 p 具有 index key

$$y=i\times c+dist(p,O) \tag{1}$$

其中,$i=1,2,\ldots,m$,m 为聚类环总数,O 为全局参考点 reference point,C 是常数,应取足够大的数值,避免 y 值出现重叠,$dist(p,O)$是数据点 p 与 O 的距离.

建立索引的步骤:

1. 使用 K-means 方法对数据点进行聚类;
2. 根据式(2)计算最优聚类环总数,根据聚类分解步骤进行聚类分解,计算各聚类环内外半径,为每个聚类环分配序号;
3. 选取最佳全局参考点 O,计算各个数据点到全局参考点的距离,结合各个数据点所属的聚类环序号,根据式(1)计算各数据点的 index key;
4. 根据 index key,把所有数据点插入 B^+-tree.索引建立完成.

2.1 聚类分解方法

为了避免对查询半径等的估计,本文采用最优 KNN 查询算法,通过聚类与查询范围是否相交来过滤聚类.然而在高维空间中,聚类数据分布稀疏,聚类间相互重叠的情况严重,大量的聚类参与查询导致查询效率不高.本文提出“聚类分解”方法,通过把聚类分解为聚类环,对数据进行更细致的划分.如图 3 所示,根据三角不等式得到的由 $r1,r2$ 确定的环形区域中的数据按聚类环分组后,只有与查询覆盖区域相交的聚类环中的数据点才有可能成为查询结果(如图 4 中聚类环 c 与环形区域相交的深色区域中的数据点才是最终的可能查询结果).在基于“聚类分解”的索引结构中,同一个聚类中的数据按所在区域的不同,分别存储在 B^+-tree 中不同的距离区段内的叶节点中,在计算查询范围时,可以有效地避免由于聚类边缘相交而引起的对整个聚类的搜索.

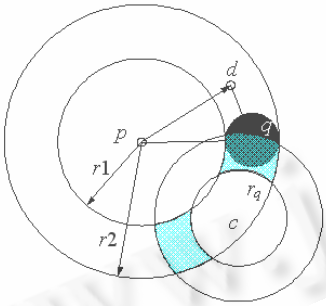


Fig.3 Clsuter splitting based filtering

图 3 基于聚类分解的数据过滤

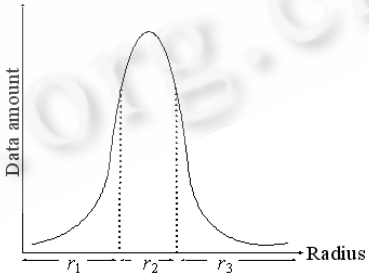


Fig.4 Distance distribution

图 4 距离分布

聚类分解可以采用如下两种方法:

按聚类半径平均分解:

把聚类分为 m 个聚类环,聚类半径为 R,每个聚类环的内外半径为 $r1,r2$,如将聚类按聚类半径 R 平均分割,每个聚类环的半径差$|r2-r1|$就等于 R/m.这种情况会产生以下问题:某些聚类环内存在大量的点,而某些聚类环

中的点稀少甚至没有,使得聚类环在查询过程中的优化作用下降.

按聚类中数据点数平均分解:

每个聚类环中分配相同数目的数据点.当数据的距离分布接近于钟形分布(如图 4 所示),则在数据分布密集的部分分配的聚类环的内外半径差较小(如 $r2$),在数据分布稀疏的部分分配的半径差较大(如 $r1,r3$).当数据的距离分布接近于均匀分布时,此分解方法等同聚类半径的平均分解.

设|cluster|为聚类内部的数据点个数,m 为每个聚类需要分解的聚类环个数,每个聚类环内的数据点数为 $|cluster|/m$,聚类环的内外半径分别为 $r1,r2$,按聚类中数据点数平均分解的步骤如下:

Step 1. 按每个数据点与聚类中心的距离升序排序组成队列 Q;

Step 2. 令第一个聚类环的内半径 $ri_1=0$,第$|cluster|/m$ 个点到聚类中心的距离为该聚类环的外半径 ro_1;

Step 3. 令第 i 个聚类环的内半径 $ri_i=ro_{i-1}$,Q 中第 $i\times(|cluster|/m)$个点到聚类中心的距离为第 i 个聚类环的外半径 ro_i;

Step 4. 重复 Step 3,直到第 m 个聚类环被分出.

显然,聚类环划分的数量是一个关键问题,划分多少才能带来最优的查询效率呢?我们有如下的断言:

断言 1(最优聚类环总数 M). 对于给定数据集 D,N_c 为初始聚类个数,$|D|=N$,H 为 B^+-树中间节点高度,u 为节点的平均扇出,则需要分配的最优聚类环总数为

$$M=\sqrt{\frac{2N_cN}{Hu}} \tag{2}$$

断言 1 将在第 3 节通过查询代价模型进行证明.

当聚类环总数确定后,如何为每个聚类分配相应的聚类环个数是一个十分重要的问题.KNN 算法根据聚类与查询区域相交与否对聚类进行过滤,半径越大,越容易引起聚类与查询区域相交;数据点数越多,因聚类边缘相交而引起的整个聚类搜索代价越大.因此,本文对每个聚类分解的聚类环数的定义如下:

定义 1(聚类环分配). 给定聚类环总数 M,每个聚类分配聚类环个数 m_i 为

$$m_i=M\times\left(r_i\times NUM_i\Big/\sum_{i=1}^{n}r_i\times NUM_i\right) \tag{3}$$

其中,M 代表聚类环总数,NUM_i 为某个聚类中数据点的个数,r_i 为聚类的半径.定义 1 的含义是:对于半径小数据点数少的聚类划分的聚类环个数少,相反聚类半径大数据点数多的聚类划分的聚类环个数多.

例 1:图 5 给出本文索引结构的图示.关于聚类 $O1$ 与 O_i 在 B+-tree 上的索引,首先对聚类进行分解(每个聚类都分为核心子聚类与边缘子聚类两部分),如聚类 $O1$ 分解为核心子聚类 1 与边缘子聚类 1,相应的数据点分别存储于编号为 $c\times1$ 与 $c\times2$ 的叶节点中.聚类 O_i 与此类似.对于区域查询 Q,可以发现只有边缘子聚类 1,边缘子聚类 i 以及核心子聚类 i 与查询区域相交,因此,相应地只需对 B+-tree 叶节点 $C\times2$,$C\times(2i-1)$,$C\times(2i)$中的数据进行访问,而包含无关数据点的叶结点 $C\times1$ 被过滤掉了.

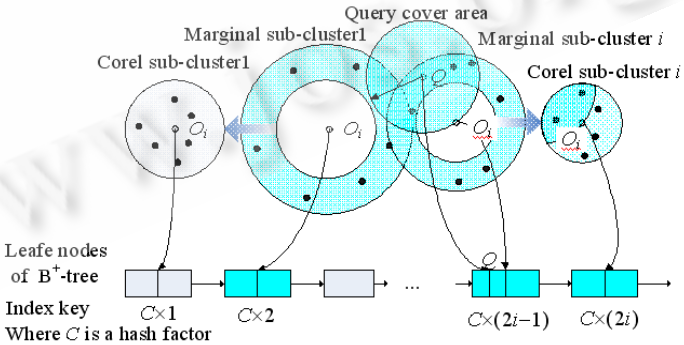


Fig.5 Index construction

图 5 建立索引的例子

2.2 KNN查询算法

本文采用最优KNN查询策略[17],所有聚类环按与查询点的距离$d(Q,C)$升序排序加入队列PQ,其中,C为PQ中任意一个以O为中心的聚类环,r为聚类半径,$d(,)$为距离函数.令$d(Q,C)$表示查询点到聚类环外圈的距离,$d(Q,O)$为查询点到聚类环中心的距离,且有$d(Q,C)=\max\{0,d(Q,O)-r\}$.按序判断聚类环区域与查询区域是否相交,如果相交,则对该聚类环中的数据进行搜索,并计算到查询Q的实际距离.即当优先级队列中的聚类环的外圈与查询点的距离小于查询半径时,对该聚类环进行范围查询,并且更新查询半径;否则,查询结束.本文KNN查询的具体算法与[17]类似:

输入:查询点Q,整数K.

1. C_i为聚类环序号,M为聚类环总数,聚类环按$d(Q,C_i)$升序排序并初始化优先级队列PQ,优先级队列中第i个聚类环用二元组表示,$[C_i,d(Q,C_i)],i=1,\ldots,M$.
2. 令结果数组$RL[j]=[_,\infty],j=1,\ldots,K;nn_{Q,k}=\infty,nn_{Q,k}$为第$k$个最近邻数据对象与查询$Q$的距离
3. While $PQ\neq\varnothing$ do:
4. 从队列PQ中按序取出一个聚类C_i
5. If $d(Q,C_i)\geq nn_{Q,k}$ then exit,
   else do:
6. 对C_i中的每个数据项O_j,if $|d(Q,P)-d(Q_j,P)|<nn_{Q,k}$ then:
7. 计算$d(Q_j,Q)$
8. if $d(Q_j,Q)<nn_{Q,k}$ then:
9. 用$\lfloor O_j,d(Q,O_j)\rfloor$更新$RL$中离查询点最远的数据项
10. 用RL中数据项离查询点的最远距离更新$nn_{Q,k}$
11. 结束

3 代价模型与聚类分解个数的确定

为了给出上述最优聚类分解个数,本文在文献[14]的基础上建立了基于聚类分解的度量空间B^+-tree查询代价模型,并通过此代价模型来考察聚类、聚类分解与查询代价的关系.

本文提出的度量空间索引建立在有界随机度量空间(bounded random metric space)$M=(U,d,d^+)$中[14],其中,U为数据空间,d为距离函数,d^+为距离值的有限上界.在M中,基于聚类分解的查询代价可以通过查询超球体(查询区域)与聚类空腔超球体(聚类环)交叉重叠的概率来估计.设聚类环C的圆心为Oc,查询点为Q,r_{cc},r'_{cc},r_q分别为聚类环外半径、内半径与查询半径,则聚类环与查询区域相交的概率分布F_{cc}为

$$\begin{aligned}F_{cc}(C,Q)&=\Pr\{r'_{cc}\leq d(O_c,Q)\leq r_{cc}+r_q\}\\&=F(r_{cc}+r_q,Q)-F(r'_{cc},Q)\end{aligned}\tag{4}$$

其中,$\Pr(.)$表示概率,$F(r,Q)$为以r为聚类半径的聚类与查询Q的相交的概率分布.为了方便,在不混淆的情况下,后面$F()$中省略了Q.

3.1 范围查询代价模型

查询的代价主要由查询中需要访问的节点数量决定.对于查询半径为r_q的区域查询$range(Q,r_q)$,其访问的节点数可以通过计算聚类环上进行区域查询时的节点访问量的数学期望来估计:本索引结构中,对聚类环C_i进行访问的代价为从B^+-tree搜索该聚类环以及对聚类环中所有点进行遍历的代价之和.设M为聚类环总数,H为B^+-tree的内部节点高度,Num_i为第i个聚类环的平均叶子节点数,N为数据总数,u是树的扇出(根据文献[18],一般$u=0.69\times b$,其中,b是节点的容量).令$Nodes(x)$表示查询x需要访问的节点数,则查询聚类环C_i需要访问的节点数为

$$Nodes(C_i)=H+Num_i\tag{5}$$

其中,$H=\lceil \log_f N/u \rceil$, $Num_i=\dfrac{N}{M\times u}$,则区域查询 $range(Q,r_q)$的查询代价的估计为

$$Nodes(range(Q,r_q))=\sum_{i=1}^{M}(F(r_{cc}+r_q)-F(r'_{cc}))\times\left(H+\frac{N}{M\times u}\right) \tag{6}$$

3.2 K-NN查询代价模型

KNN 查询可以通过范围查询来实现.但是,KNN 查询的查询半径不能像区域查询那样预先给定,因此,要对 KNN 查询半径进行估计:在有界随机度量空间中,可以用二项式分布来描述 KNN 查询半径的距离分布[14],进而求导得到其概率密度,并通过在距离空间上的积分得到 KNN 查询半径的期望.然后,将 KNN 查询半径的期望代入到范围查询代价模型就可以得到 KNN 查询代价估计.

对于本索引结构,设 $nn_{Q,k}$ 为第 k 个最近邻数据对象与查询 Q 的距离,则 $nn_{Q,k}$ 小于等于距离 r 的概率就是在距离 r 内至少有 k 个对象的概率.则 KNN 查询关于距离 r 的概率分布可以用二项式分布描述为

$$\begin{aligned}P_{Q,k}(r)&=\Pr\{nn_{Q,k}\le r\}\\&=\sum_{i=k}^{n}\binom{n}{i}\Pr\{d(O_{cc},O_q)\le r\}^i\Pr\{d(O_{cc},O_q)>r\}^{n-i}\\&=1-\sum_{i=0}^{k-1}\binom{n}{i}F(r)^i(1-F(r))^{n-i}\end{aligned} \tag{7}$$

其中,$\Pr\{d(O_{cc},O_q)\le r\}$表示以 O_{cc} 为中心的聚类环与查询 Q(查询半径等于 r)相交的概率.

$P_{Q,k}(r)$的概率密度 $p_{Q,k}(r)$为

$$p_{Q,k}(r)=\frac{\mathrm{d}P_{Q,K}(r)}{\mathrm{d}r}=\sum_{i=0}^{k-1}\binom{n}{i}F(r)^{i-1}f(r)(1-F(r))^{n-i-1}(nF(r)-i) \tag{8}$$

则关于 Q 的 KNN 查询代价的数学期望为

$$\begin{aligned}Nodes(KNN(Q))&=\int_0^{d^+}Nodes(range(Q,r))\,p_{Q,k}(r)\mathrm{d}r\\&=\int_0^{d^+}(F(r_{cc}+r_q)-F(r'_{cc}))\times\left(H+\frac{N}{M\times u}\right)\sum_{i=0}^{k-1}\binom{n}{i}F(r)^{i-1}f(r)(1-F(r))^{n-i-1}(nF(r)-i)\mathrm{d}r\end{aligned} \tag{9}$$

为了简化代价模型,设每个聚类平均分为 $n=M/N_c$ 个聚类环,其中,N_c 为初始聚类个数,则有:

聚类的平均半径 r_c 为

$$r_c=\frac{\sum_{i=0}^{N_c}r_{ci}}{N_c}.$$

其中,r_{ci} 为第 i 个聚类的半径.则每个聚类环的平均外半径 r_{cc} 为

$$r_{cc}=\frac{\sum_{i=1}^{n}r_{cci}}{n}\times r_c=\frac{\frac{1}{n}+\frac{2}{n}+\ldots+\frac{n}{n}}{n}\times r_c=\left(\frac{1}{2}+\frac{N_c}{2M}\right)\times r_c \tag{10}$$

同理,每个聚类环的平均内半径 r'_{cc} 为

$$r'_{cc}=\frac{\sum_{i=1}^{n}r'_{cci}}{n}\times r_c=\frac{\frac{0}{n}+\frac{1}{n}+\frac{2}{n}+\ldots+\frac{n-1}{n}}{n}\times r_c=\left(\frac{1}{2}-\frac{N_c}{2M}\right)\times r_c \tag{11}$$

其中,r_{cci} 为聚类环 i 的外半径.

3.3 最小查询代价聚类分解数的确定

为了得到代价最小时聚类环分解数 M,对式(9)求最小值.不失一般性,这里用均匀分布来描述数据间的距离分布,即 $F(r)=r/d^+$,r 为距离.由于仅对 M 求导,在不影响最优聚类环总数的求导结果的前提下,可以去掉式(9)中与聚类环总数 M 无关的项,简化后的式(9)为

$$Nodes(KNN(Q)) \cong \int_0^{d^+}\left(\sum_{i=1}^{M}(F(r_{cc}+r)-F(r'_{cc}))\times\left(H+\frac{N}{M\times u}\right)\right)\mathrm{d}r \cong (1/d^+)\int_0^{d^+}\left(\sum_{i=1}^{M}(r_{cc}-r'_{cc}+\mathrm{r})\times\left(H+\frac{N}{M\times u}\right)\right)\mathrm{d}r \tag{12}$$

代入聚类环全局平均内外半径式(10)、式(11)有

$$Nodes(KNN(Q)) \cong \left(MH+\frac{N}{u}\right)\left(\frac{d^+}{2}+\frac{N_c.r_c}{M}\right) \tag{13}$$

通过求导,令 $\frac{\mathrm{d}(Nodes(nn_{Q,k}))}{\mathrm{d}M}=0$,得到

$$M=\sqrt{\frac{2r_cN_cN}{Hud^+}} \tag{14}$$

在高维情况下(当维数高于 15 时),最近邻之间的距离和最远距离 d^+接近[16],即 $\frac{d^+}{r_c}\approx 1$,则使查询代价最小化的聚类分解数目的理论计算公式为

$$M=\sqrt{\frac{2N_cN}{Hu}} \tag{15}$$

3.4 使得查询代价最小化的聚类数目

对于给定的聚类数目 N_c,通过式(15)可以计算出使得查询代价最小时需要进行聚类分解的数目.由于聚类分解是在聚类的基础上进行的,显然有 $M\geq N_c$,那么,如果出现下述情况:对于给定的 N_c,根据式(15)计算出 $M=N_c$ 的结果,则表示即使不进行聚类分解,该 N_c 也已经使得查询代价最小化.因此,把 $M=N_c$ 带入式(15),有

$$N_{opt}=2N/Hu \tag{16}$$

其中,N_{opt} 表示使查询代价最小时的最优聚类个数.式(16)的意义在于,我们可以在索引前计算出使得查询代价最小所需要进行聚类的数目.这为我们在应用如 *K*-Means 聚类方法进行数据分析时,对于聚类参数 *K* 的选择提供了理论指导.另一方面,在实际的大规模数据集应用中,式(16)计算出的 *K* 一般都比较大,相应的 *K*-means 聚类的代价非常高昂.此时,使用本文提出的聚类分解方法,可以通过相对廉价的聚类分解计算获得接近最优的查询效率.

4 对比实验

为了验证聚类分解方法与利用代价模型预测最优聚类分解程度的有效性,本文从以下 3 个方面进行了对比实验:

(1) 聚类分解与单纯的聚类方法的查询效率比较;

(2) 利用代价模型预测聚类环个数的性能与普通聚类性能、最优聚类分解性能的比较;

(3) 聚类分解方法的查询性能与其他相关索引结构性能的比较.

预测聚类分解性能是指采用代价模型预测的聚类环总数进行聚类分解得到的性能,最佳聚类分解性能是指实验中采用人工调试得出的最优查询性能,普通聚类性能是指只采用普通的聚类方法(如 *K*-means)进行聚类得到的性能.

实验在包含 32 维 68 040 余幅图像数据的通用测试数据集(此数据集可从 http://kdd.ics.uci.edu/databases/CorelFeatures/CorelFeatures.data.html 得到)以及 58 维接近 60 000 余幅图像数据集上进行.实验所用硬件系统为 P4 2.8GHz CPU,768MB Memory 的 PC.建立 B^+-tree 的页面大小为 4K,树的高度 *h* 为 4,节点的存储能力 *u* 为 20,实验中根据式(16),对本数据集最优的聚类分解总数为 2 268.

4.1 相同聚类环数与聚类数性能比较

为了比较聚类分解与单纯的聚类对查询性能的影响,本文对如下两种情况进行了比较:

1) 初始聚类个数为 64 时采用不同聚类环个数进行分解;

2) 令聚类个数等于聚类环个数,不进行聚类分解.

实验结果如图 6 所示,当聚类环与聚类个数小于式(16)的最优值时(在 60 000 的数据集上此值等于 2 268),聚类分解比单纯的聚类方法有效,并且当聚类个数接近聚类最优值时,两者对查询性能的影响的差别减小.这是因为聚类环中点的相互距离较远,利用最佳参考点与三角不等式进行过滤的能力高于普通聚类方法.另一方面,由于聚类数据分布稀疏,相互重叠情况严重,单纯进行简单的聚类不能有效地过滤与查询无实质关联的数据而影响查询速度,“聚类分解”方法通过聚类环把数据分布进行更细致的划分来减少查询时的数据访问量,在计算查询范围时,可以有效地避免由于聚类边缘相交而引起的对整个聚类的搜索,从而提高了查询效率.

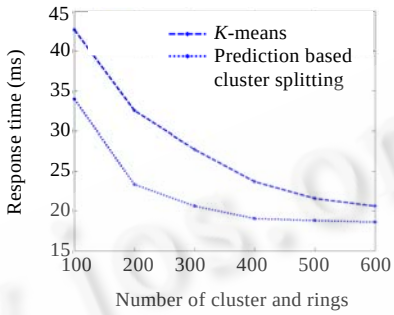
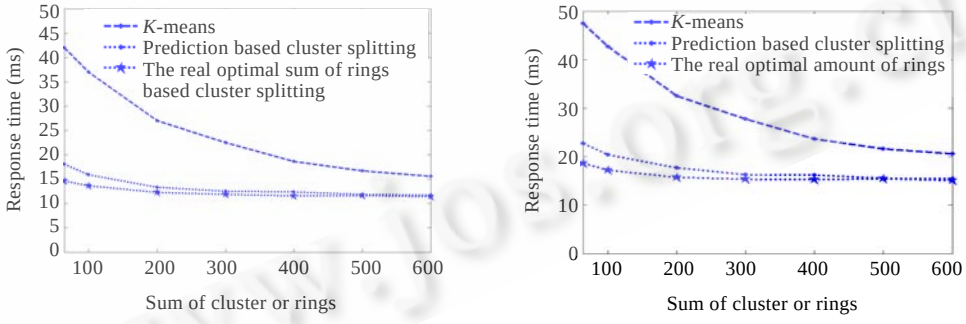


Fig.6 Comparison between rings and clusters with the same amount

图 6 相同个数的聚类环与聚类性能比较

4.2 预测聚类分解性能与实际最优性能、聚类性能比较

我们对普通聚类、预测聚类分解和最优聚类分解方法统计了对普通聚类个数从 64 到 600、K 从 10~50 的 KNN 查询效率,如图 7 所示.实验结果表明,使用聚类分解与代价模型的预测可以大大提高检索效率,代价模型的预测聚类分解性能与实际的聚类分解性能接近.随着聚类个数的增加,聚类分解对检索性能的提高逐渐降低,接近代价模型对聚类分解有效性的预测.



(a) K=10 (b) K=50

Fig.7 The prediction with different initial cluster number

图 7 基于不同初始聚类数的聚类分割预测

为了测试在不同数据大小与不同数据维度情况下预测聚类分解方法的有效性,本文分别在 32 维 30 000 个数据(如图 8 所示)与 58 维 60 000 个数据(如图 9 所示)的情况下测试了普通聚类、预测聚类分解和最优聚类分解方法的效果,采用理论计算的聚类分解与实际实验的最小查询代价的平均误差分别为 2.58%与 1.54%,都在

3%以内.

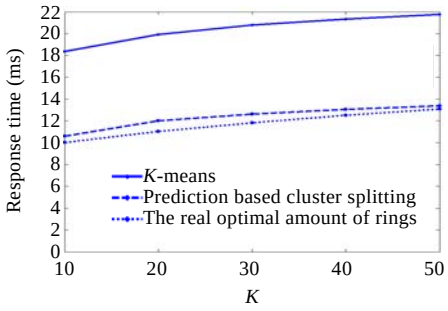


Fig.8 The prediction on the 30 000 32-dimensional data set

图 8 30 000 个 32 维数据的预测效果

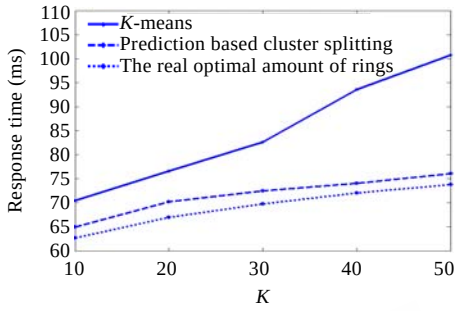


Fig.9 The prediction on the 58-dimensional data set

图 9 58 维数据的预测效果

4.3 聚类环分解方法性能与其他索引方法性能比较

我们分别选择了有代表性的度量空间索引方法,如 M-tree,Omini,iDisance 以及顺序查找等进行了查询效率对比实验,图 10 显示了聚类环分解方法性能明显高于其他索引方法的效率.Omni 方法通过减少查询操作的距离计算量来提高查询性能,然而,每个数据点有多个坐标增加了页面访问,搜索多个 B 树需要更多的 CPU 时间,计算 m 个候选集合的交集带来了额外的代价.实验显示,其查询效率不如 iDistance.在 iDistance 的实验中,我们使用了 64 个参考点[10],并通过反复验证各种初始半径与递增值后,获得了 iDistance 最高的查询效率曲线.实验结果显示,本文聚类分解方法在查询效率上是 iDistance 的 2 倍,是顺序扫描的 4~6 倍.由于 M-tree 节点的利用率较低,在高维空间中节点的覆盖区域相互重叠,引起不必要的查询代价,实验中的查询效率低于顺序扫描.

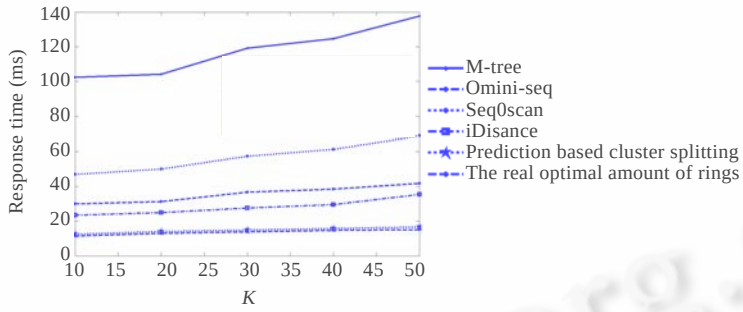


Fig.10 Comparison with other indexes

图 10 与其他索引结构性能比较

需要指出的是,iDistance 搜索算法通过实验预测并设定初始查询半径,然后逐步扩大查询半径进行 KNN 查询,初始查询半径和查询半径的递增值不能预知,当初始查询半径与递增值过大时,会引起不必要的查询;过小则需要多次重复查询,导致查询效率降低,只能通过实验来确定查询参数.

5 结 论

本文对高维索引中使用的聚类方法与查询代价的关系进行了讨论,提出了一种新的基于聚类分解的 B^+-tree 高维度量空间索引结构.本文的贡献在于,与以往对聚类方法的启发式运用不同,通过索引的查询代价模型的估计,给出了使得查询代价最小化的 K-Means 聚类数目以及最小查询代价条件下的聚类分解数目的理论计算方法.实验结果显示,本文的相关理论计算与实际结果的误差在 3%以内,本文提出的索引结构在查询效率

上明显优于传统的 M-tree,Idistance 等方法.

References:

[1] Robinson J. The k-d-b-tree: A search structure for large multidimensional dynamic indexes. In: Edmund YL, ed. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 1981. 10−18.

[2] Guttman A. R-trees: A dynamic index structure for spatial searching. In: Youmark B, ed. Proc. of the ACM Int'l Conf. on Management of Data. Boston: ACM Press, 1984. 47−57.

[3] Berchtold S, Keim D, Kriegel HP. The X-tree: An index structure for high-dimensional data. In: Proc. of the Int'l Conf. on Very Large Databases. 1996. 28−39.

[4] Ciaccia P, Patella M, Zezula P. M-Tree: An efficient access method for similarity search in metric spaces. In: Jarke M, Carey MJ, Dittrich KR, *et al*., eds. Proc. of the 23rd Int'l Conf. on Very Large Data Bases (VLDB'97). Athens: Morgan Kaufmann Publishers, 1997. 426−435.

[5] Webber R, Schek JJ, Blott S. A quantitative analysis and performance study for similarity-search methods in high-dimensional space. In: Gupta A, Shmueli O, Widom J, eds. Proc. of the 24th ACM Int'l Conf. on Very Large Data Bases (VLDB'98). New York: Morgan Kaufmann Publishers, 1998. 194−205.

[6] Zhou XH, Li X, Gong YC, Zhao ZX. Research on dynamic indexing structure for multi-dimensional vectors. Journal of Software, 2002,13(4):768−773 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/768.htm

[7] Feng YC, Cao K, Cao ZS. A multidimensional index structure for fast similarity retrieval. Journal of Software, 2002,13(8): 1678−1685 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/1678.htm

[8] Zhou XM, Wang GR. Key dimension based high-dimensional data partition strategy. Journal of Software, 2004,15(9):1361−1374 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/1361.htm

[9] Ciaccia P, Nanni A, Patella M. A query-sensitive cost model for similarity queries with M-tree. In: Proc. of the 10th Australasian Database Conf. (ADC'99). 1999. 65−76.

[10] Jagadish HV, Ooi BC, Tan KL, Yu C, Zhang R. iDistance: Adaptive B^+-tree based indexing method for nearest neighbor search. ACM Trans. on Data Base Systems, 2005,30(2):364−397.

[11] Yu C, Ooi BC, Tan KL, Jagadish HV. Indexing the distance: An efficient method to KNN processing. In: Apers PMG, Atzeni P, Ceri S, Paraboschi S, Ramamohanarao K, Snodgrass RT, eds. Proc. of the 27th Int'l Conf. on Very Large Data Bases. Roma: Morgan Kaufmann Publishers, 2001. 421−430.

[12] Filho RFS, Traina A, Traina C, Faloutsos C. Similarity search without tears: The omni family of all-purpose access methods. In: Proc. of the 17th Int'l Conf. on Data Engineering. 2001. 623−630.

[13] Berchtold S, Bohm C, Kriegel HP. The pyramid-technique: Towards breaking the curse of dimensionality. In: Haas LM, Tiwary A, eds. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. Seattle: ACM Press, 1998. 142−153.

[14] Beyer K, Goldstein J, Ramakrishnan R, Shaft U. When is nearest neighbors meaningful? In: Beeri C, Buneman P, eds. Proc. of the 7th Int'l Conf. on Database Theory. LNCS 1540, Jerusalem: Springer-Verlag, 1999. 217−235.

[15] Ciaccia P, Patella M, Zezula P. A cost model for similarity queries in metric spaces. In: Proc. of the 17th ACM Conf. on Principles on Database Systems. New York: ACM Press, 1998. 59−68.

[16] Shen HT, Ooi BC, Zhou X, Huang Z. Towards effective indexing for very large video sequence database. In: Proc. of the 24th ACM SIGMOD Int'l Conf. on Management of Data. Baltimore: ACM Press, 2005. 730−741.

[17] Berchtold S, Bohm C, Keim DA, Kriegel H. A cost model for nearest neighbor search in high-dimensional data space. In: Proc. of the 16th ACM PODS Symp. on Principles of Database Systems. Tucson: ACM Press, 1997. 78−86.

[18] Tao Y, Zhang J, Papadias D, Mamoulis N. An efficient cost model for optimization of nearest neighbor search in low and medium dimensional spaces. IEEE Trans. on Knowledge and Data Engineering (TKDE), 2004,16(10):1169−1184.

附中文参考文献:

[6] 周学海,李曦,龚育昌,赵振西.多维向量动态索引结构研究.软件学报,2002,13(4):768−773. http://www.jos.org.cn/1000-9825/13/768.htm

[7] 冯玉才,曹奎,曹忠升.一种支持快速相似检索的多维索引结构.软件学报,2002,13(8):1678−1685. http://www.jos.org.cn/1000-9825/13/1678.htm

[8] 周项敏,王国仁.基于关键维的高维空间划分策略.软件学报,2004,15(9):1361−1374. http://www.jos.org.cn/1000-9825/15/1361.htm

张军旗(1979－),男,河南许昌人,博士,讲师,主要研究领域为多媒体数据库,信息检索,群体智能,进化计算.

王梅(1980－),女,博士生,主要研究领域为多媒体数据库.

周向东(1969－),男,博士,副教授,主要研究领域为数据库,信息检索.

施伯乐(1935－),男,教授,博士生导师,主要研究领域为数据库理论与应用.

****

第 16 届全国网络与数据通信学术会议(NDCC 2008)

征 文 通 知

由中国计算机学会网络与数据通信专业委员会主办，东南大学计算机科学与工程学院、中国人民解放军理工大学指挥自动化学院、计算机网络和信息集成教育部重点实验室、江苏省网络与信息安全重点实验室承办的第 16 届全国网络与数据通信学术会议(NDCC2008)定于 2008 年 11 月上旬（具体时间另行通知）在东南大学召开。

一、**会议主题**:下一代网络的创新和发展

二、**征文范围**(但不限于)

网络体系结构　透明计算　协议工程　网络安全　网络管理　分布式计算　网格计算　普适计算　移动和无线网络　传感器网络　服务计算和 Web 服务　网络运行与管理　宽带多媒体通信　光纤通信技术　各种网络应用

三、**投稿须知**

请见会议网址：http://cse.seu.edu.cn/ndcc2008

四、**投稿方式**

论文投稿通过电子邮件的方式提交，并在邮件标题注明“NDCC2008 投稿”。

投稿邮箱：ndcc2008@seu.edu.cn

五、**重要日期**

论文提交截止日期：2008 年 6 月 20 日

论文录用通知日期：2008 年 7 月 20 日

会议注册截止日期：2008 年 8 月 20 日

六、**联系方式**

通信地址：210096 南京市四牌楼 2 号东南大学计算机学院

联 系 人：罗军舟，李伟　　联系电话：025-8379 1010　　传真：025-8379 1010

邮件地址：xchlw@seu.edu.cn