ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.8, August 2008, pp.2054−2065
DOI: 10.3724/SP.J.1001.2008.02054

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

基于查询采样的高维数据混合索引*

张军旗[1,2,3], 周向东[3+], 施伯乐[3]

[1](北京大学 信息科学技术学院 智能科学系,北京 100871)
[2](北京大学 信息科学技术学院 机器感知与智能教育部重点实验室,北京 100871)
[3](复旦大学 计算机与信息技术系,上海 200433)

High Dimensional Hybrid Index Based on Query Sampling

ZHANG Jun-Qi[1,2,3], ZHOU Xiang-Dong[3+], SHI Bai-Le[3]

[1](Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871)
[2](Key Laboratory of Machine Perception for the Ministry of Education, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871)
[3](Department of Computing and Information Technology, Fudan University, Shanghai 200433)
+ Corresponding author: E-mail: xdzhou@fudan.edu.cn, http://www.fudan.edu.cn

Zhang JQ, Zhou XD, Shi BL. High dimensional hybrid index based on query sampling. *Journal of Software*, 2008,19(8):2054−2065. http://www.jos.org.cn/1000-9825/19/2054.htm

Abstract: In order to improve the query answering of high-dimensional database, data distribution is necessary to select appropriate indexing strategy. However, traditional data distribution models can not estimate the accurate data distribution in the complex real multimedia data of image and video. This paper presents a method to estimate the accurate data distribution based on query sampling, and proposes a novel hybrid index to speed up processing of high-dimensional *K*-nearest neighbor (KNN) queries. The proposed hybrid index improves the query efficiency by adaptively selecting different index strategies for the data with different distribution. In the first step, the cluster analysis and cluster splitting methods are applied to construct a tree-based index, and then the relationship between data distribution and index performance is derived by sampling. At last some tree branches with sparse data are extracted for linear scan, while the aggregate data remains in the tree. Extensive experiments on four real image data sets show that the proposed hybrid index structure performs better than iDistance, M-Tree and linear scan, and scales better with dimensions. The index is still faster than linear scan when the dimension reaches 336. The experiments also show that the proposed query sampling algorithm can obtain the accurate data distribution when the amount of sampling is below $\sqrt{N}$ (N is the size of data set).

Key words: nearest neighbor query; high dimensional index; marginal data; cluster partitioning

* Supported by the National Natural Science Foundation of China under Grant No.60403018 (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 Natural Science Foundation of Shanghai of China under Grant No.04ZR14011 (上海市自然科学基金); the College Cooperation Plan of AMD (AMD 大学合作计划)
Received 2007-03-26; Accepted 2007-08-03

摘 要: 为了改进高维数据库查询的效率,通常需要根据数据分布来选择合适的索引策略.然而,经典的分布模型难以解决实际应用中图像、视频等高维数据复杂的分布估计问题.提出一种基于查询采样进行数据分布估计的方法,并在此基础上提出了一种支持最近邻查询的混合索引,即针对多媒体数据分布的不均匀性,自适应地对不同分布的数据使用不同的索引结构,建立统一的索引结构.为了实现混合索引,采用构造性方法:首先通过聚类分解分割数据并建立树状索引;然后使用查询采样算法,对数据实际分布进行估计;最后根据数据分布的特性,把稀疏数据从树状索引中剪裁出来,进行基于顺序扫描策略的索引,而分布比较密集的数据仍然保留在树状索引中.在 4 个真实的图像数据集上进行了充分的实验,结果显示,该索引方法明显优于 iDistance,M-Tree 等度量空间索引,在维数达到 336 时,查询效率仍高于顺序扫描.实验结果显示,该查询采样算法在采样数据量仅为$\sqrt{N}$(N 为数据量)的情况下即可获得满足索引需要的分布估计结果.

关键词: 最近邻查询;采样;高维索引;边缘数据;聚类分解

中图法分类号: TP311 **文献标识码**: A

在图像、生物信息、医学成像、时间序列等领域需要对大数据集进行相似性查询.一般通过特征转换将数据对象映射到高维向量空间,把相似性查询转换为向量空间的最近邻查询.由于大量数据将引起较高的查询代价,因此利用各种索引结构管理特征向量.索引结构可分为树状索引与基于顺序扫描的索引.树状索引,如R-tree[1],M-tree[2]等,通过对聚集数据或空间的划分来提高对数据的过滤能力;基于顺序扫描的索引,如VA-File[3],通过扫描估计文件,减少对数据文件的访问量.国内也开展了相应的研究,如 ER-Tree 动态索引结构[4]、基于距离的相似索引结构 opt-树及其变种[5]、高维数据空间分割策略[6]、基于矢量量化的索引方法[7]、基于聚类分解的高维度量空间索引[8]等.

在高维空间,只有数据集的数据分布与聚集情况足够好,基于分割或聚类的索引方法才有意义[9].当维数远远高于 10 维时,数据集的聚集情况变差,树状索引的效率随之下降.1998 年,Webber[3]证明了当维数大于 610 维时,任何基于聚类或分割的索引方法的查询效率都低于顺序扫描,称为维灾,基于顺序扫描的索引在此情况下是一种十分可行的检索策略.当数据维数小于 10 维时,由于数据的聚集性强,多数已知的树状索引已经证明了其索引性能的有效性.显然,低维的数据选择树状索引,高维的数据选择顺序扫描策略比较合适.但是,当数据集的维数处于中等规模(大于 10 维而小于 610 维)时,作出选择树状索引还是顺序扫描策略的判断并不容易.因为一方面,维数高的数据集的聚集性并非一定比维数低的要差;另一方面,同一固定维数的数据集中存在着不同类型的数据,即密集数据与稀疏数据,密集数据分布较密,由树状索引存储效率较高,稀疏数据分布较散,若用树状索引存储,索引效率会由于稀疏数据过滤能力较差而降低;最后,密集数据与稀疏数据之间、数据聚集程度与索引策略之间尚无有效的判别方法.

对于常见的介于几十到几百维的多媒体数据,如何选择索引策略是一个难题,需要考虑数据集的数据分布与索引策略之间的关系.树状索引能够有效过滤密集数据,而顺序扫描策略对检索稀疏数据更加有效.对于大量中等规模维数的多媒体数据,单独使用树状索引时,稀疏数据不可避免地加入到密集数据的聚类中,使得聚类平均半径过大,索引过滤能力减弱,随之而来的是数据过滤能力的下降,甚至低于顺序扫描的效率;而在单独使用顺序扫描策略时,由于数据维数并没有达到足够高,数据的分布仍不均匀,理论上,此时彻底抛弃树状索引也是不合适的.因此,对于大量中等规模维数的多媒体数据,固定地使用单一索引策略存储不同类型的数据,对于数据分布不均匀的实际数据集与维数可能变化的数据集缺乏自适应能力.

本文提出一种支持中等维数多媒体数据查询的混合索引方法,能够自适应地对实际分布不同的数据采用树状过滤技术或顺序扫描方法.树状索引与顺序扫描的结合是一种平滑的过渡,随着维数的增高,树状索引的成分逐渐减小并过渡到顺序扫描.由于实际数据的分布难以把握,我们提出一种构造性的方法,先根据数据实际分布建立树状索引,再根据数据分布对索引性能的影响自适应地对树的分支进行裁剪.为了得到数据的真实分布,首先对数据进行聚类分析,再使用聚类分解方法对各聚类内部数据按分布情况进一步划分.数据划分后,通过查询采样算法,以聚类环为单位,得到数据被访问的平均概率,据此分析数据实际分布对不同索引效率的贡献,并

从树状索引中裁剪稀疏数据直接存储到顺序文件中,用于顺序扫描.实验结果显示,本文提出的混合索引方法明显优于 iDistance 等度量空间索引,在维数达到 300 多维时查询效率仍高于顺序扫描,查询采样算法在采样数据量仅为$\sqrt{N}$(N 为数据量)的情况下即可获得满足索引需要的分布估计结果.

本文第 1 节介绍相关工作.第 2 节提出混合索引结构与查询采样算法.第 3 节介绍混合查询算法.第 4 节给出实验结果与分析.第 5 节得出本文的结论.

1 相关工作

树状索引与基于顺序扫描的索引都使用单一的索引策略,没有考虑不同的数据分布与索引策略之间的关系.另有一些研究在单一的索引结构中结合了树状索引与顺序扫描两种方法,根据数据的分布使用合适的索引策略.这些方法分为两类.一类是在顺序扫描方法的基础上使用树状索引,如 2002 年 Berchtold 提出的 IQ-tree[10],为 VA-file 的压缩文件建立索引,避免扫描全部压缩文件;2002 年,Guang 提出的 GC-tree[11],在为估计文件建立索引时,基于数据密度动态生成估计单元,对稀疏数据与密度数据分别索引,但判断数据是否为密集数据需要人为指定.另一类是在树状索引中引入顺序扫描方法,如 2000 年,Bohm[12]等人提出的动态最优化高维索引结构方法,采用一级目录的方法减少中间节点的访问代价;2001 年,Yu Cui 提出的 iDistance 索引结构[13],通过在 B^+-tree 节点之间加入双向链表实现局部的顺序扫描;2004 年,Edgar[14]在索引中使用了对部分数据顺序扫描的方法,提高了索引在高维上的承受力.2006 年,张军旗等人[8]提出了基于聚类分解的高维度量空间索引,并对聚类的最优分割进行了理论证明.在单一索引结构中,结合树状索引与顺序扫描的索引方法认识到,不同的数据分布应该采用不同的索引策略,但仍有以下缺点:1) 单一索引结构不能很好地适应数据分布不同的索引问题,限制了树状过滤技术与顺序扫描方法各自优势的发挥;2) 数据分布一般通过代价模型或体积计算来估计,不能自适应地反映数据的真实分布.

通过查询代价估计来调整索引策略是常用的索引优化方法.查询代价的估计一般采用两种方法,一种是代价模型,另一种是采样技术.基于向量空间或度量空间的各种代价模型可以预测索引结构性能,如 Ciaacia 等人[15]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的查询代价模型,并基于代价模型来调整 M-tree[2]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邻查询的 I/O 与 CPU 代价.

采样技术由于独立于数据维度并保存了数据聚类信息,成为商业数据库查询优化的标准方法.2001 年,Christian 等人[16]为了调整、优化索引结构,利用采样技术预测查询代价.该方法通过在内存中建立索引结构的缩影,模拟实际索引结构性能,提高内存利用率,进而提高预测索引结构性能的效率与效果.2006 年,Jayendra 等人[17]在如何选择参考点问题上使用采样方法,通过预先采样得到一个查询集合,在查询过程中统计所有参考点对其他数据的过滤能力,以此作为选择参考点的标准,因此是获取数据集分布信息和建立数据分布与查询效率之间关系的有效计算方法.采样技术的优势在于简单、有效,能够处理高维与非均匀分布的真实数据,保存数据的聚类信息;缺点在于预测精度与采样代价成正比,因此在大型数据库中使用采样技术,需要在预测精度与采样代价之间加以权衡;另一方面,可以结合代价模型与采样技术,通过代价模型的指导减小采样代价,利用采样技术进一步提高索引结构的性能.

2 混合索引结构

2.1 两阶段数据划分

首先通过 k-means 进行初始的全局聚类,k 为初始聚类个数,以对数据分布进行全局分析,获得初步数据分布信息.但实验中发现,在实际的高维图像数据库上进行 k-means 聚类时,初始聚类个数 k 越大,聚类的计算代价越高昂,甚至是不可接受的.因此,我们再使用聚类分解方法(参见文献[8])细分数据.与 iDistance[13]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(聚类环总数)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.通过最小化代价模型得到的最优聚类分解数目的理论估计方法、分配聚类环和聚类划分方法参见文献[8].

2.2 建立树状索引结构

对数据进行聚类划分后,首先使用 B^+树建立树状索引.利用距离是单维值的性质,把数据按照与参考点的距离与所属的聚类环编号进行排序,并索引到 B^+树中.通过 PCA 分析,在方差最大的 Principal Component 方向上选择最佳参考点[8].我们继承了 iDistance 方法[13]中建立 B^+树索引的 Index key 的计算公式.对于数据点 $p(x_1,x_2,\ldots,x_d)$,若 $0\le x_j\le 1,0\le j\le d$,则数据点 p 具有 index key:

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

其中,i 为数据点 p 所属的聚类环号,且 $0\le i\le m$,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,以避免 y 值出现重叠;$dist(p,O)$是数据点 p 与全局参考点 O 的距离.

首先建立一个空的 B^+树与一个空的顺序文件,以聚类环而非聚类为单位索引,把聚类环中的数据存储到 B^+树中.

2.3 边缘数据分析与混合索引结构(混合索引内容较少)

为了分析数据分布与索引性能的关系,本文建立了基于聚类分解方法的图像检索原型系统,如图 1 所示.其索引结构为 58 维 B^+树,初始聚类数为 100,聚类环数为 600.图 1 给出了系统界面,其中左上方柱状图统计各个聚类环被查询访问的情况,系统左下方显示每次查询需要的时间.本文针对两个随机查询 a 和 b 统计了各个聚类环被访问的情况,并对系统界面左上角相应显示的柱状图进行分析,如图 2 所示,每个柱子为一个聚类,每个聚类环按照与聚类中心的距离在柱子上从下往上升序排序,每个柱子代表的聚类按照与查询点的距离升序排序.图中显示,大多数聚类(柱子)的外环(柱子中靠上的格子)被访问(深色),而内环(柱子中靠下的格子)没有被访问(浅色).由此系统观察到以下现象:1) 查询时被访问的聚类环数量越多,查询速度越慢,甚至低于顺序扫描的性能;被访问的聚类环数量越少,查询速度越快.2) 聚类分裂确实避免了聚类因外环与查询区域相交而引起的对整个聚类的查询;3) 某些靠近柱子边缘的聚类环总是被访问,降低了索引结构的查询效率(本文称其为边缘聚类环).

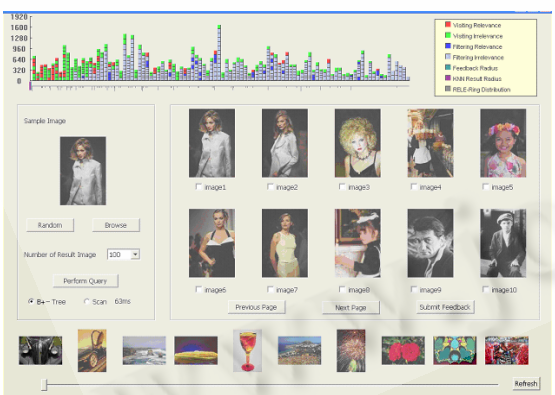
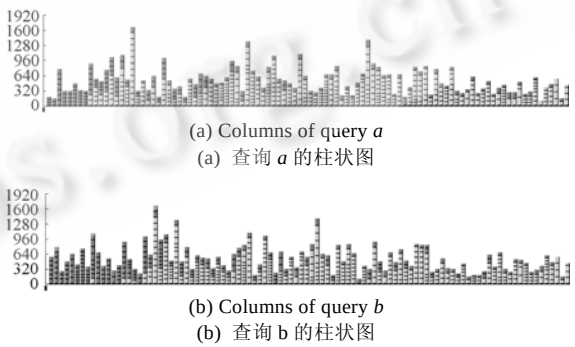


Fig.1 Image retrieval system

图 1 图像检索系统



(a) Columns of query a

(a) 查询 a 的柱状图

(b) Columns of query b

(b) 查询 b 的柱状图

Fig.2 Columns of two random queries

图 2 两个随机查询的柱状图

图 3 给出了对 10 000 次随机查询各个聚类环被访问频度的分布统计,从图中可以看出,各聚类环被访问的概率的分布非常不均匀,其中超过了 21%的数据被访问频率大于 80%,说明部分聚类环在 B^+树索引结构中被频繁访问,降低了索引结构的效率.

以上实验与分析表明,不同分布的数据对查询代价的贡献各不相同,边缘环在树状索引中被访问需要更多的定位时间,如果其查询代价大于在文件中顺序扫描的代价,则可以考虑把 B^+树中将经常被访问到的边缘聚类环摘除,并放到顺序的边缘数据文件中直接扫描,这要比放在 B^+树上检索效率更高.根据以上分析,本文首先在

基于聚类分解方法的基础上建立 B^+树索引结构,然后,将边缘数据环从 B^+树中摘除并存储到顺序扫描文件中.边缘数据环由本文提出的自适应的查询采样算法来确定.如图 4 所示,基于聚类分解的 B^+树索引结构建立之后,以聚类环为单位,根据本文提出的查询采样算法检测出边缘数据所在的聚类环(图 4 中第 3 个和第 i 个聚类环)并对 B^+树修剪,把边缘聚类环中的数据从 B^+树中摘除并顺序存储在边缘数据文件中.查询时,首先扫描边缘数据文件,然后在 B^+树索引结构中继续查询,最后得到精确的查询结果.

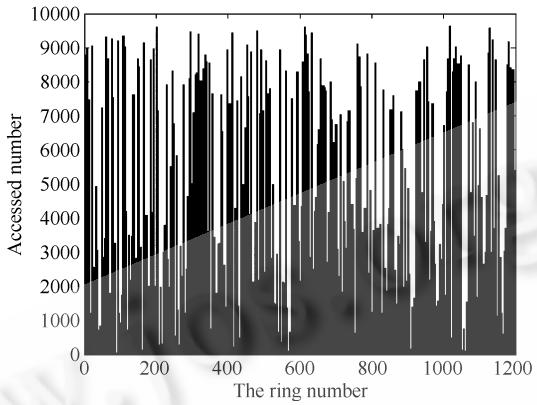


Fig.3 Frequency of rings being visited by queries

图 3 聚类环被查询访问的频度

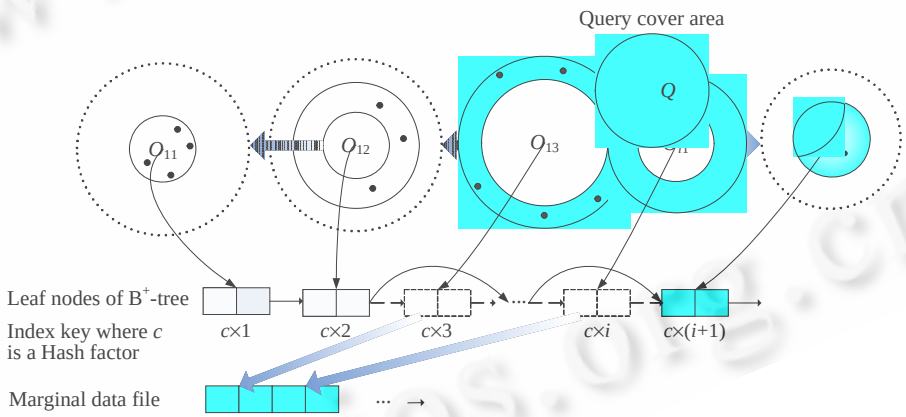


Fig.4 Hybrid index structure

图 4 混合索引结构

2.4 自适应的查询采样算法

数据在树中被访问的平均概率决定了该数据是否适合在树状索引中存储.当数据的平均被访问概率达到一定值时,数据在树中的平均被访问代价高于在顺序文件中扫描的代价,数据被从树中摘除并存储到顺序文件中.由于使用代价模型等理论方法估计数据在树中被访问的平均概率非常困难,并且精度不高,因此本文以聚类环为最小单位来判别数据是否为边缘数据,边缘数据所在的环为边缘聚类环,简称边缘环.如图 3 所示,各个聚类的边缘环也不一定都是外环.为了获得数据分布与索引策略的关系,本文提出一种查询采样算法,以聚类环为单位,得到数据被访问的平均概率,据此分析数据实际分布对不同索引效率的贡献,以便从树状索引中裁剪稀疏数据并直接存储到顺序文件中,用于顺序扫描.在实验中,我们比较了基于查询采样算法与直接将聚类最外环去掉(称为简单混合算法)两种方法的索引效率.

设 c_i 为第 i 个聚类环,$P(c_i)$为聚类环 i 被查询访问的概率函数,N_{ci} 为聚类环 i 中的数据个数,b 为顺序扫描此

聚类环时每个节点容纳的数据量(节点能力),u 为树中节点容纳的数据个数,且 u=0.69×b[8],树的中间节点高度为 H,顺序扫描聚类环 i 的代价为 $\frac{N_{ci}}{b}$,放在 B$^+$树中查询它的代价为 $P(ci)\left(H+\frac{N_{ci}}{u}\right)$.如定义 1 所述,聚类环的可索引能力就是使用两种索引策略代价的差,当聚类环的可索引能力大于 0 时,说明聚类环在 B$^+$树索引结构中的查询代价小于顺序扫描,反之,对此聚类环直接顺序扫描的查询效率更高.当聚类环的可索引能力等于 0 时,$\frac{N_{ci}}{b}=P_{0i}\left(H+\frac{N_{ci}}{u}\right)\Rightarrow P_{0i}=\frac{uN_{ci}}{Hub+bN_{ci}}$,此时,$P_{0i}$ 的值为聚类环是否为边缘环的概率阈值.

定义 1. 聚类环的可索引能力 IC(index capability)为

$$IC_i=\frac{N_{ci}}{b}-P(ci)\left(H+\frac{N_{ci}}{u}\right) \tag{2}$$

定义 2. 可索引能力小于等于 0 的聚类环为边缘聚类环(边缘环),边缘环内的数据为边缘数据.

定义 3. 聚类环是否为边缘环的概率阈值为 $\frac{uN_{ci}}{Hub+bN_{ci}}$.

查询采样算法在聚类分解与初步建立树状索引后,通过随机采样预先定义一个查询集合 Q,查询集合 Q 在 B$^+$树上进行检索并统计所有聚类环被访问的概率 P(此概率等于聚类环被查询的次数与查询总数的比).此概率越高,说明聚类环在 B$^+$树索引结构中的被过滤能力越差,反之,被过滤能力越强.显然,被过滤能力差的聚类环中的数据不适合存储在B$^+$树索引结构中.然后,结合各个聚类环的查询概率与可索引计算公式(2)计算聚类环 M 的可索引能力 IC,IC 值小于 0 的聚类环被判定为边缘聚类环,自适应地从 B$^+$树中摘除并放到边缘数据文件中.该方法由统计信息直接估计数据与查询效率的关系,减小了传统的利用代价模型和一系列假设进行估计的限制,得到了数据集更为真实的分布信息,提供了计算数据与查询效率关系的有效方法.然而对于大型数据库,过多的查询集合会引起过高的查询采样代价,因此需要在查询采样质量与查询采样效率之间进行权衡.本文根据中心极限定理,在用户需要的置信度下控制采样次数,并通过采样停止的加速条件进一步减小采样次数.

2.4.1 基于置信度的查询采样控制

各个聚类环被查询访问的期望是聚类环是否为边缘环的判别条件,是查询采样算法的采样目标.对于某个被查询访问的期望概率为 p 的聚类环,设每次此聚类环是否被访问为随机变量 X,将 η_n 看成是 n 个相互独立、服从同一(0-1)分布的诸随机变量 $X1,X2,\ldots,X_n$ 之和,即有 $\eta_n=\sum_{k=1}^{n}X_k$,其中 $X_k(k=1,2,\ldots,n)$的分布律为 $P\{X_k=i\}=p^i(1-p)^{1-i},i=0,1$. $E(X_k)=P,D(X_k)=p(1-p),k=1,2,...,n$,在 n 次采样中,此聚类环被访问到的频率 η_n 服从参数为 n,p 的二项分布.根据中心极限定理(德莫佛-拉普拉斯定理),对于任意 x,恒有 $\lim_{n\to\infty}P\left\{\frac{\eta_n-np}{\sqrt{np(1-p)}}\le x\right\}=\int_{-\infty}^{x}\frac{1}{\sqrt{2\pi}}e^{-\frac{t^2}{2}}dt$,即 $\frac{\eta_n-np}{\sqrt{np(1-p)}}$ 服从参数为$(0,1)$的标准正态分布.令聚类环的采样被访问概率 $\bar{P}=\frac{\eta_n}{n}$,有 $\frac{\bar{P}-p}{\sqrt{p(1-p)}/\sqrt{n}}$ 服从参数为(0,1)的标准正态分布,即聚类环的采样被访问概率 $\bar{P}$ 服从参数为$(p,\sqrt{p(1-p)}/\sqrt{n})$的正态分布.让$\mu$作为各个聚类环被访问概率的期望值 p,由于聚类环被访问概率 P 的分布未知,考虑到采样方差是方差的无偏估计,因此,$\frac{\bar{P}-\mu}{S/\sqrt{n}}\sim t_{a/2}(n-1)$,并且右边的分布 $t_{a/2}(n-1)$ 不依赖于任何未知参数.S 为聚类环被访问概率的采样标准差,$t_{a/2}(n-1)$,由查表得到,根据中心极限定理,均值落在置信区间内的置信度为(1–a)%时,置信区间与均值的偏差为 $\frac{S}{\sqrt{n}}t_{a/2}(n-1)$.当 $\frac{S}{\sqrt{n}}t_{a/2}(n-1)<\varepsilon$ 时,采样停止.用户给定置信度与误差值,置信度越高,误差值越小,采样次数就越多;反之则采样次数越少.误差值的大小对采样次数影响很大.

2.4.2 采样停止的加速条件

通过基于置信度的查询采样控制,用户可以通过调节置信度(1–a)%,取得采样精度与采样效率的权衡.由于

查询采样的目的是通过聚类环被访问的概率来估计聚类环可索引能力 IC 值的符号,据此对树状索引进行修剪.当采样得到的聚类环被访问概率在用户设定的置信度下的改变不再引起 IC 值符号的改变时,采样可以提前停止.因此,我们使用定理 1 来加速采样的停止,进一步减小采样数量.

定理 1. M 为聚类环集合,当查询采样算法中采样得到的聚类环 i 的被访问概率 $P(ci)$满足以下条件时,采样可以提前停止,并且不会改变聚类环 IC 值的符号:

$$P(ci)<\frac{uN_{ci}}{Hub+bN_{ci}}-\frac{S_i}{\sqrt{n}}t_{a/2}(n-1),\ \forall i\in M \tag{3}$$

或

$$P(ci)>\frac{uN_{ci}}{Hub+bN_{ci}}+\frac{S_i}{\sqrt{n}}t_{a/2}(n-1),\ \forall i\in M \tag{4}$$

证明:查询采样算法检测各个聚类环被访问概率的目的是为了计算聚类环的 IC 值是否小于 0,因此,若当前被检测聚类环的被访问概率 P_i 在置信区间内的移动不再引起聚类环的 IC 值符号改变,则查询采样可以停止.根据 IC 计算公式(2)可知,当 IC 值为 0 时,$\frac{N_{ci}}{b}=P_{0i}\left(H+\frac{N_{ci}}{u}\right)\Rightarrow P_{0i}=\frac{uN_{ci}}{Hub+bN_{ci}}$,即每个聚类环都存在一个被访问的概率阈值 P_{0i}.显然,当 $P(ci)<P_{0i}-\frac{S_i}{\sqrt{n}}t_{a/2}(n-1)$ 或 $P(ci)>P_{0i}+\frac{S_i}{\sqrt{n}}t_{a/2}(n-1)$ 时,用均值 u_i 代替 $P(ci)$,则 u_i 在置信区间内任意移动都不会改变聚类环 IC 值的符号.因此,继续采样没有必要,此时停止不会改变聚类环 IC 值的符号.

算法 1. 查询采样.

Input: N is dataset, a is the believe degree, Z_n is the constant to compute believe zone, M is the number of cluster circle, Q is the query set, querynum is current queries number, *flag* is the sample flag.

Output: S is marginal cluster circle set.

1. Q={}, *querynum*=30, *flag*=1, sample querynum queries and add these queries to Q

2. while (*flag*)

(a) sample 1 query and add it to Q

(b) *querynum*+=1

(c) *flag*=0

(d) $G[i]$=0, $1\le i\le M$, S=[] //$G[i]$ is the access times of the i-th cluster circle

(e) for each $q,i,\forall q\in Q,\forall i\in M$

(f) if (i is accessed by query q)

(g) $G[i]$++

(h) for each $q,i,\forall q\in Q,\forall i\in M$

(i) $u_i=\frac{G_i}{|Q|}$

(j) $error_i=\frac{S}{\sqrt{|Q|}}t_{a/2}(n-1)$

(k) If ($P_{0i}-error_i\le u_i\le P_{0i}+error_i$) &&($error_i>\varepsilon_0$)

flag=1

3. for each G_i

(a) $IC_i=\frac{N_{ci}}{b}-u_i\left(H+\frac{N_{ci}}{u}\right)$

(b) If (IC_i<0)

$S=S\cup\{i\}$

(c) for each $j \in D$

(d) if $j \in M(i)$

delete j from B^+-tree and put it into the marginal data file

3 混合索引的KNN查询算法

树状索引能够有效过滤聚集性强的数据,顺序扫描检索稀疏数据更加有效.若将稀疏数据存储在树状索引中,则存放稀疏数据的叶子节点经常被访问,势必引起过多的中间节点访问代价来定位稀疏数据.若将聚集性强的数据存储在顺序文件中,索引性能则会因没有有效过滤数据而下降.因此,我们将边缘聚类环中的数据存放在一个顺序的边缘数据文件中,将非边缘聚类环中的数据存储在B^+树索引结构中.每次查询时,首先扫描边缘数据文件,然后在 B^+树索引结构中继续查询.需要指出的是,首先查找边缘数据文件,将缩小在 B^+树索引结构中进行KNN 查询的初始查询半径,加快 KNN 查询半径的收敛速度,从而提高查询效率.当数据维度较高时,边缘数据量较大,边缘数据文件可以采用 VA-file 进一步提高索引效率.

本文采用最优KNN查询策略[2],在对边缘数据文件检索时,顺序扫描文件中的每个数据,更新查询半径与查询结果.在对 B^+树检索时,所有聚类环按照与查询点的距离 $d(Q,C)$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 $d(Q,C)=\max\{0,d(Q,O)-r\}$.依次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.

Algorithm 2. The KNN search algorithm.

Input: Q is the query point, K is a integer, M is the total number of cluster circles, $nn_{Q,k}$ is the distance from Q to the k-th nearest neighbor, MD is the marginal datafile, P is the reference point.

Output: The result set RL.

1. $RL[j]=[\_,\infty], j=1,...,K, nn_{Q,k}=\infty$
2. for $\forall O_j \in MD$

   if $d(Q,O_j) \geq nn_{Q,k}$ then

   RL is updated by $[O_j, d(Q,O_j)]$

   $nn_{Q,k}=d(Q,O_j)$
3. All cluster circles is sorted ascending by $d(Q,C_i)$ in the queue of PQ, $[C_i, d(Q,C_i)]$ is the element in $PQ, i=1,...,M$
4. While $PQ \neq \varnothing$ do

   (a) C_i is the first cluster circle in PQ

   (b) If $d(Q,C_i) \geq nn_{Q,k}$ then exit, else do

   for $\forall O_j \in C_i$

   if $|d(Q,P)-d(O_j,P)| < nn_{Q,k}$ then

   compute $d(O_j,Q)$

   if $d(O_j,Q) < nn_{Q,k}$ then

   RL is updated by $[O_j, d(Q,O_j)]$

4 实 验

实验所用硬件系统为 P4 2.8GHz CPU,768MB Memory 的 PC.实验中采用了 4 个数据集:

*D*1:32 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.建立 B^+树的页面大小为 4K,节点的存储能力 u 为 20,实验中初始聚

类数为 64,对本数据集最优的聚类分解总数为 381.

*D*2: 58 维接近 60 000 余幅图像数据集.建立 B^+树的页面大小为 4K,节点的存储能力 *u* 为 11,实验中初始聚类数为 600,对本数据集最优的聚类分解总数为 1 278.

*D*3: 336 维接近 60 000 余幅图像数据集.建立 B^+树的页面大小为 24K,节点的存储能力 *u* 为 12,实验中初始聚类数为 100,对本数据集最优的聚类分解总数为 447.

*D*4: 80 维 200 000 余副图像数据集.此数据集是通过网页抓取工具 LARBIN[18]获得并进行图像特征抽取得到的,建立 B^+树的页面大小为 24K,节点的存储能力 *u* 为 11,实验中初始聚类数为 300,对本数据集最优的聚类分解总数为 647.

为了验证本文提出的混合索引结构的有效性,我们安排了 3 部分实验.第 1 部分比较了混合索引与其他索引结构的查询效率;第 2 部分统计了 3 个数据集合中边缘数据被查询的概率分布,并给出了查询采样方法在不同维度与不同数据集大小情况下对整个索引结构效率的提升;第 3 部分测试了采样数量与查询效率的关系以及利用定理 1 进一步减小的采样数量.所有查询效率的统计均通过 10 000 次随机查询得到.

4.1 混合索引性能

图 5 比较了聚类分解方法以及基于查询采样的混合索引与其他几种索引结构的查询性能,如 M-tree, Omini[19],iDisance 以及顺序查找.图 5 显示了聚类环分解方法以及基于查询采样的混合索引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,而且计算 *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[13],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.实验显示,本文提出的混合索引方法在查询效率上是 iDistance 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.而 iDistance 搜索算法需要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,过小时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.而本文提出的混合索引可以通过聚类分解方法与最优 KNN 查询策略,使得查询算法自动收敛,不需要通过实验或经验确定查询参数.

4.2 查询采样方法与其他方法在性能上的比较

本部分实验在 *D*2,*D*3 与 *D*4 数据集上统计了边缘数据被查询的概率分布,并比较了初始聚类性能(*k*-means)、预测聚类分解(predicted cluster splitting)、简单查询采样算法(simple hybrid)、最优查询采样算法(sampling based hybrid)和顺序扫描(scan)五种方法的查询效率.初始聚类是简单使用 *k*-means 进行聚类得到的查询效率,预测聚类分解是使用聚类分解方法得到的查询效率,简单查询采样算法是直接将各个聚类的外环作为边缘环,最优查询采样算法是基于查询采样算法得到的混合索引.

图 6~图 8 在 *D*2,*D*3 和 *D*4 数据集上比较了初始聚类性能、预测聚类分解、简单查询采样算法、最优查询采样算法和顺序扫描五种方法的查询效率.实验显示,单纯使用 *k*-means 进行聚类仅在 *D*1 上查询效率高于顺序扫描,基于理论指导的聚类分解方法在 *D*1~*D*4 上都提高了索引的查询效率.基于简单查询采样算法的混合索引仅在 *D*2,*D*3 上提高了索引效率.由于维灾,大多数的高维索引结构在维数超过 15 维以后,性能都将低于顺序扫描.而基于查询采样算法的混合索引结构在 *D*1,*D*2,*D*3 和 *D*4 上都进一步提高了索引的查询效率,并且查询效率优于顺序扫描,使得混合索引结构在中高维数据集合上具有良好的高维承受能力.图 7 和图 8 显示,当维数达到 80 维和 336 维时,随着维数的增高,本文提出的混合索引结构顺序文件中存储的数据越来越多,B^+树中存储的数据越来越少,混合索引退化为顺序扫描,其性能趋向于顺序扫描,而不会像大多数索引结构那样,随着维数的升高,性能很快下降到远低于顺序扫描.

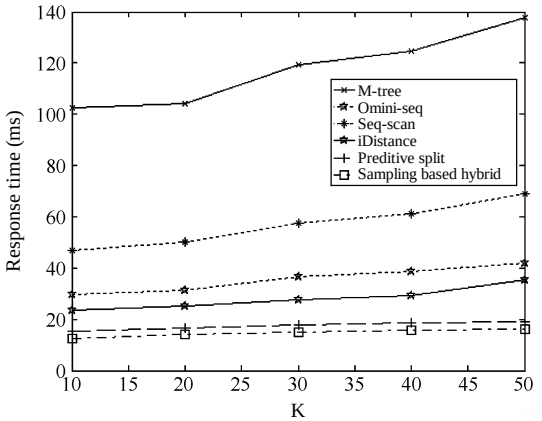


Fig.5 Comparison with other indexes on *D*1

图 5 在 *D*1 上与其他索引的比较

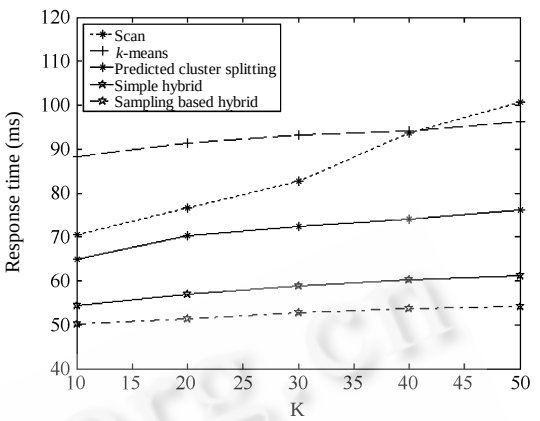


Fig.6 Comparison with other indexes on *D*2

图 6 在 *D*2 上与其他索引的比较

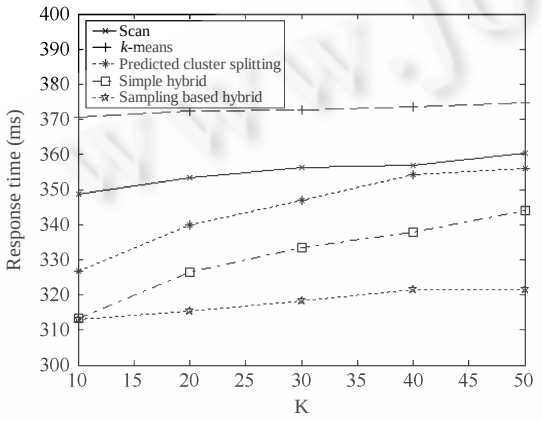


Fig.7 Comparison with other indexes on *D*3

图 7 在 *D*3 上与其他索引的比较

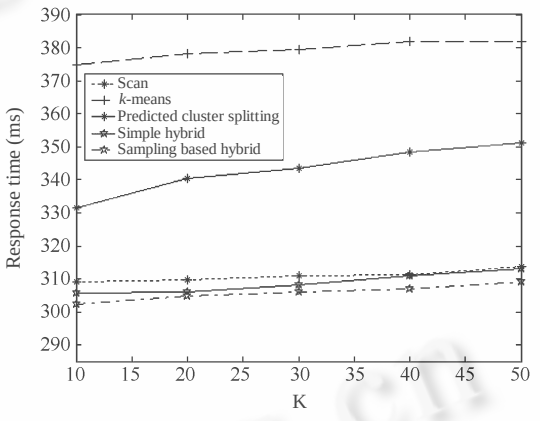


Fig.8 Comparison with other indexes on *D*4

图 8 在 *D*4 上与其他索引的比较

4.3 采样控制中采样次数与定理1停止条件的实验效果

我们在实验中令采样次数 n 满足以下条件: $\frac{1}{10}m\sqrt{N} \le n \le m\sqrt{N}$, m 为常数,N 为数据库的基数,由于 $S_i \le 0.25$,令 $\varepsilon_0 = \frac{0.25}{\sqrt{m\sqrt{N}}} t_{\alpha/2}(n-1)$,并设定它为方差停止条件,即每次采样先进行 $\frac{1}{10}m\sqrt{N}$ 次,然后按 $\frac{1}{10}m\sqrt{N}$ 递增,所有聚类环的查询采样都能在 $m\sqrt{N}$ 内停止.此条件作为查询采样停止条件.本部分实验在 *D*2,*D*3,*D*4 上采用置信度为 95%,比较了当 m 取值不同时取得的查询效率及所进行采样的次数.实验显示,当 m=1,即采样次数为 $\sqrt{N}$ 时,与经过大量采样得到的查询效率一致(如当 m=10 时,采样次数为 $10\sqrt{N}$ 的查询效率).图 9 比较了在 *D*2 上,当 m 取值不同时(使用加速停止条件 2)取得的查询效率,表 1 比较了在 *D*2 上,当 m 取值不同时所进行采样的次数.图 9 说明,当 m=1,即采样次数为 $\sqrt{N}$ 时,非常接近经过大量采样得到的查询效率(如当 m=10 时,采样次数为 $10\sqrt{N}$ 的查询效率).表 1 说明,使用加速停止条件 2 能够进一步减少采样次数,由于不会改变可索引能力 IC 的符号,因此不影响混合索引的查询效率.

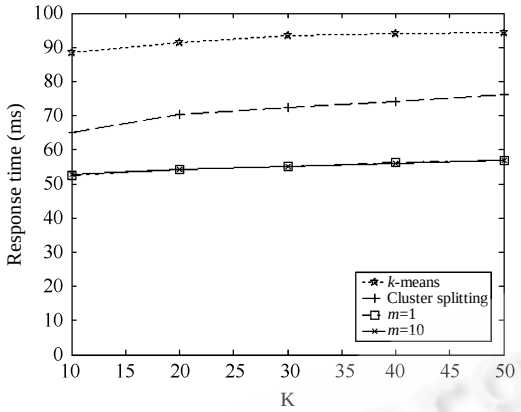


Fig.9 Rosponse time of Sampling on *D*2

图 9 在 *D*2 上查询采样的响应时间

Table 1 Total number of sampling on *D*2

表 1 在 *D*2 上的采样次数

m	1	3	5	7	9
No theory 2	245	735	1 224	1 714	2 203
Using theory 2	243	724	1 172	1 627	2 099

5 结 论

本文提出了一种混合索引方法,能够自适应地对实际分布不同的数据采用不同的索引方法.以查询代价最小为目标,初步建立 B^+树索引,再通过采样判断估计出边缘数据,自适应地对 B^+树进行修剪,摘除边缘环数据到存储边缘数据的顺序文件,对实际分布不同的数据结合 B^+树与顺序文件两种存储结构分别进行存储.实验结果显示,基于查询采样算法的混合索引在聚类分解的基础上进一步提高了索引效率,使得索引结构对高维的承受力加强,在 32 维数据集合上,查询效率高于其他索引结构,在 336 维数据集上的查询效率仍然高于顺序扫描,取得了查询效率在不同维度的伸缩与承受能力.实验结果还显示,此查询采样算法在采样数据量仅为 $\sqrt{N}$ (N 为数据量)的情况下即可获得满足索引需要的分布估计结果.

References:

[1] Guttman A. *R*-Trees: A dynamic index structure for spatial searching. In: Yormark B, ed. Proc. of the 1984 ACM SIGMOD Int'l Conf. on Management of Data. ACM Press, 1984. 47−57.

[2] 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.

[3] Webber R, Schek HJ, 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 Int'l Conf. on Very Large Data Bases. New York: Morgan Kaufmann Publishers, 1998. 194−205.

[4] Zhou XH, Li X, Xu HY, 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.pdf

[5] 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.pdf

[6] 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

[7] Ye HJ, Xu GY. Fast image search using vector quantization. Journal of Software, 2004,15(9):712−719 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/712.htm

[8] 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 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/1401.htm

[9] 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, Springer-Verlag, 1999. 217−235.

[10] Berchtold S, Böhm C, Jagadish HV, Kriegel HP, Sander J. Independent quantization: An index compression technique for high-dimensional data spaces. In: Proc. of the 16th Int'l Conf. on Data Engineering (ICDE 2000). New Orleans: IEEE Computer Science Society Press, 2000. 577−588.

[11] Cha GH, Chung CW. The GC-tree: A high dimensional index structure for similarity search in image databases. IEEE Trans. on Multimedia, 2002,4(2):235−247.

[12] Böhm C, Kriegel HP. Dynamically optimizing high-dimensional index structures. In: Youmark B, ed. Proc. of the 7th Int'l Conf. on Extending Database Technology (EDBT 2000). Konstanz: Springer-Verlag, 2000. 36−50.

[13] 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 Int'l Conf. on Very Large Data Bases. Rome: Morgan Kaufmann Publishers, 2001. 421−430.

[14] Chávez E, Herrera N, Reyes N. Spatial approximation+sequential scan=efficient metric indexing. In: Proc. of the XXIV Int'l Conf. of the Chilean (SCCC 2004). Computer Science Society, 2004. 121−128.

[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, 1998. 59−68.

[16] Lang CA, Singh AK. Modeling high-dimensional index structures using sampling. In: Proc. of the Special Interest Group on Management of Data. New York: ACM, 2001. 389−400.

[17] Venkateswaran J, Lachwani D, Kahveci T, Jermaine C. Reference-Based indexing of sequence databases. In: Proc. of the Int'l Conf. on Very Large Data Bases. 2006. 906−917.

[18] Ailleret S. GPL software. 2004. http://larbin.sourceforge.net/index-eng.html

[19] Filho, RFS, Traina A, Jr. 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. IEEE Computer Society, 2001. 623−630.

附中文参考文献:

[4] 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.pdf

[5] 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.pdf

[6] 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

[7] 叶航军,徐光祐.基于矢量量化的快速图像检索.软件学报,2004,15(9):712−719. http://www.jos.org.cn/1000-9825/15/712.htm

[8] 张军旗,周向东,王梅,施伯乐.基于聚类分解的高维度量空间索引 B^+-Tree.软件学报,2008,19(6):1401−1412. http://www.jos.org.cn/1000-9825/19/1401.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,智能计算.

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

周向东(1969－),男,博士,副教授,主要研究领域为数据库,信息检索.