基于广义超曲面树的相似性搜索算法*

张兆功, 李建中

(哈尔滨工业大学 计算机科学与技术学院,黑龙江 哈尔滨 150001);
(黑龙江大学,黑龙江 哈尔滨 150080)
E-mail: lijz@banner.hl.cninfo.net; zhangzhaogong@0451.com
http://www.hit.edu.cn

摘要: 相似性搜索是数据挖掘的主要领域之一.它在数据库中检索出相似的数据,发现数据间的相似性.它可以应用于图像数据库、空间数据库和时间序列分析.对于欧氏空间(一种特殊的度量空间),相似性搜索算法中基于 R-tree 的方法,在低维时是高效的,当维数增加时,R-tree 的方法将退化为线性扫描.该现象被称为维数灾难(dimensionality curse),主要原因是存在数据重复.当数据量很大且维数很高时,距离计算和 I/O 操作将非常费时.提出了度量空间上新的空间分割方法和索引结构 rgh-tree,利用数据库的数据对象与很少几个固定参考对象的距离信息进行数据分割和分布,产生一个各节点没有数据重复的平衡树.另外,在 rgh-tree 的基础上提出了相应的相似性搜索算法,该算法具有较小的 I/O 代价和距离计算次数,平均复杂性近似为 $o(n^{0.58})$.解决了目前算法存在的一些问题.

关 键 词: 算法;相似性搜索;度量空间;数据库
中图法分类号: TP311 文献标识码: A

在许多数据库的应用中,存在着基于邻近关系(proximity)的查询问题.这些应用包括图像数据库[1]、遗传学、信息检索、数据挖掘中的时间序列分析[2~4]、文本匹配[5]、空间数据库[6]等等.例如,时间序列分析中的相似性搜索[2,4]是从时间序列数据库中搜索出彼此相似的所有时间序列对.又如,在图像数据库中,给定一个图像,找出所有与它相似的图像[1],其中两个图像的相似程度由对应像素灰度差的绝对值之和来描述.

一般化的相似性查询问题是给定一组数据对象,找出数据库中所有与它们相似的数据对象的集合.数据对象是指数据库中存放的客观对象的特征数据(或特征数据的一个子集).给定的一组数据对象也称为模板数据,特殊情况下可以只有一个模板数据.查询结果也可以是最相似的几个数据对象.模板数据还可以有一定的变化,例如时间序列数据可以伸缩、平移和上移,图像可以伸缩、平移和旋转等.

数据对象的相似性是通过应用领域中给出的距离函数计算得到的.由用户给定一个最大阈值,若数据对象之间的距离小于该最大阈值,则认为是相似的.由于距离计算和 I/O 操作比较费时,因此,研究主要集中在减小距离计算的次数和 I/O 的次数,通过各种技术和索引结构,依据三角不等式快速过滤掉距离较远的数据对象,避免费时的比较计算、距离计算及 I/O 操作.

人们已提出了多种相似性查询算法,这些算法采用的技术大致可分为如下 4 类:

(1) 基于距离变换算法.这类算法将数据对象映射到欧几里德空间中,变换保证数据对象之间的距离不变.再利用低维欧几里德空间中的高效查询技术[2].文献[2]利用傅里叶变换的算法,适合数据间有较强的相关性,但不具有通用性的情况.

* 收稿日期: 2001-10-15; 修改日期: 2002-04-22
基金项目: 国家自然科学基金资助项目(69873014);国家高技术研究发展计划资助项目(2001AA415410);国家重点基础研究发展规划 973 资助项目(G1999032704);国家教育部博士点基金资助项目(2000021303);黑龙江省自然科学基金资助项目(F00-11)

作者简介: 张兆功(1963－),男,山东青岛人,博士生,副教授,主要研究领域为数据挖掘技术;李建中(1950－),男,黑龙江哈尔滨人,教授,博士生导师,主要研究领域为并行数据库理论,数据仓库技术,数据挖掘技术.

(2) 基于空间分割算法.这类算法将整个数据空间分割为规则或不规则的子空间,如文献[3]中的ε-kDB 树、R-tree 等.ε-kDB 树方法可能产生树的偏斜,从而增加 I/O 操作,并且在使用时需临时建立树,建树的费用较高.R-tree 方法在低维时是高效的.当维数增加时,R-tree 的方法将退化为线性扫描[7].该现象被称为维数灾难(dimensionality curse),它主要是因为存在数据重复.

(3) 基于曲线填充算法[8].它是将高维空间分割为规则的 cell,对 cell 进行编码或索引,将其映射到一维空间上,其相似搜索结果存在着遗漏问题.

(4) 基于距离的索引算法,这类算法通过对数据对象之间的距离信息进行索引,获得查询的高效性.索引方法是研究的关键问题,如文献[1]中的 mvp-tree 和文献[9]中的 gh-tree.gh-tree 不是平衡树,需要平衡化.mpv-tree 是平衡树,但区域不对称.

当前的研究主要集中在如何减小距离计算和比较计算的次数[1],但都没有考虑 I/O 费用,而且多数限于解决二维和三维的几何问题[10].

针对这些问题,在今后的相似性搜索研究中我们需要进一步解决以下问题:

(1) 避免数据重复问题,保证树的平衡性;

(2) 如何减小距离计算次数;

(3) 如何减小 I/O 操作开销;

(4) 保证可扩展性.当数据量变大且维数增加时,系统效率不降低,解决维数灾难问题.

本文综合考虑减小距离计算的次数和 I/O 费用的问题,深入研究了距离索引方法,在 gh-tree 和 M-tree 的基础上,提出了新的数据存储结构 rgh-tree,解决了树的平衡性,既减小了距离计算的次数,也降低了 I/O 开销,而且还避免了 R-tree 和 M-tree 在高维情况下产生的数据重复问题.在 rgh-tree 的基础上,本文还提出了相似性搜索算法,理论分析和实验结果表明,该算法的性能高于目前同类算法.

本文第 1 节介绍了几个基本概念并对已有技术进行了回顾.第 2 节描述了 rgh-tree 的结构和生成算法.第 3 节给出了基于 rgh-tree 的相似搜索算法.第 4 节显示了实验结果.第 5 节总结全文.

1 基本概念及对已有技术的回顾

1.1 度量空间和相似性查询

首先我们给出几个基本概念.

定义 **1** (度量空间)[1]. 度量空间是指由一个数据对象集合 X 和这个数据对象集合上的一个距离函数 d 构成的二元组(X,d),其中距离函数 d 是二元函数并满足以下条件:

(i) 有 $0<d(x,y)<\infty, x\neq y, d(x,x)=0$(非负性).

(ii) $d(x,y)=d(y,x)$,对于任意的 x,y 属于 X 都成立(对称性).

(iii) $d(x,y)\leq d(x,z)+d(z,y)$,对于任意的 x,y,z 属于 X 都成立(三角不等式).

定义 **2**(相似性查询). 设(X,d)是一个度量空间,D 是该度量空间中有限的数据对象集合.单点相似性查询检索出所有属于 D,并且到给定点 Q 的距离小于 r 的数据对象,结果集合为

$$U(Q,r)=\{x_i|x_i\in D, d(x_i,Q)\leq r\}.$$

下面我们对度量空间中已有的相似检索技术进行回顾.

1.2 基于距离索引结构的方法

基于距离索引结构的具体方法如下:

方法 1. 采用 gh-tree(超曲面分解)结构[9],树的每个节点有两个参考点,每个参考点对应一个集合,其他点划归为离它较近的参考点所属的集合.如果参考点选得好,可以是均衡树,但均衡性不容易保证.

方法 2. 采用 GNAT(geometric near-neihbor access tree)结构[11],该方法类似于方法 1,不同之处在于它选取 k 个参考点.同样不容易保证均衡性.

方法 3. vp-tree[9]和 mvp-tree 结构[1],可以保证均衡性,但该方法的最大缺点是区域不对称.

方法 4. 采用 M-tree 结构[12],树的叶节点所含数据对象个数相同,内节点包含一个参考点,该点与父节点参考点的距离和覆盖半径.它类似于 R-tree 和 X-tree 方法,构造方法是自底向上动态生成的,这是它不同于以上方法的特点.该方法和 R-tree 方法一样,存在着边界数据的重复问题.

以上方法都存在不足,或者产生边界数据的重复问题,或者分割区域不对称,或者不易保证均衡性.我们提出的 rgh-tree 结构,克服了上述方法的缺点,在高维情况下仍然保持高效性.

2 广义超曲面树(rgh-tree)

2.1 rgh-tree的定义

我们在 gh-tree 和 GNAT 的基础上,引入一个广义超曲面的概念.

定义 **3**(广义超曲面). 设(X,d)是一个度量空间,且 x,y 属于 X,给定一个正的实数 R,称 R 为距离的比,则点集合

$$\text{R\_G\_Hypersurface}(x,y,R)=\{z|d(z,x)=R\times d(z,y),z\in X\},$$

被称为距 x 和 y 的距离比为 R 的广义超曲面.

定义 **4**(二叉广义超曲面树). 设(X,d)是一个度量空间,且 x,y 属于 X,给定一个正的实数 R,广义超曲面 R_G_Hypersurface(x,y,R)将空间分割为不相交的 3 部分,第 1 部分是曲面自身,第 2 部分是由距离比小于 R 的点组成,即$\{z|d(z,x)<R\times d(z,y),z\in X\}$.第 3 部分是由距离比大于 R 的点组成,即$\{z|d(z,x)>R\times d(z,y),z\in X\}$.后两部分分别对应 rgh-tree 的一个子节点.子节点利用新的广义超曲面继续分割,以此类推.

定义 **5**(二叉广义超曲面树的平衡性). 如果二叉广义超曲面树 rgh-tree 的所有左子节点所含的数据对象个数等于右节点所含的数据对象个数,或至多相差一个数据对象时,则称该二叉广义超曲面树具有平衡性.

定义 **6**(二叉广义超曲面树的区域近似对称性). 如果二叉广义超曲面树 rgh-tree 的所有广义超曲面距离比 R 接近于 1,则称该二叉广义超曲面树具有区域近似对称性,如果 R 始终为 1,则称具有区域对称性.

定义 **7**(数据集合的覆盖半径). 设(X,d)是一个度量空间,Y 是一个数据集合,P 为属于 X 的一个固定的点,所有 Y 中数据点与 P 点的最大距离值称为以 P 为中心的数据集合 Y 的覆盖半径,P 称为中心.通常取 P 为使半径最小的点作为中心,在欧氏空间中一般取 P 为 Y 中所有点的算术平均值.

gh-tree 的分割界面是由到两个参考点的距离相等的点组成,即距离比恒为 1,具有区域对称性.我们将 rgh-tree 分割界面的距离比定为使得两部分所含的数据对象个数相等的那个值,从而实现树的平衡,每个子节点利用递归调用生成完全树.显然,rgh-tree 具有平衡性,并且区域是近似对称的.更一般的情况是,用多个值产生多叉树.二叉 rgh-tree 的每个节点的数据结构如下:

P_l	左参考点	*Left*	指向左子树的指针
d_l	左集合的覆盖半径(P_l为中心)	O_l	P_l到父参考点的距离
R	分割界面的距离比		
P_r	右参考点	*Right*	指向右子树的指针
d_r	右集合的覆盖半径(P_r为中心)	O_r	P_r到父参考点的距离

树的结构如图 1 所示,图中的 $P_l^{(0)}$是根节点左参考点,$P_r^{(0)}$是根节点右参考点,上标(0)表示根节点,$d_l^{(0)}$是半径,即左子树中数据对象距离其参考点的最大值.节点(1)中的$O_l^{(1)}$表示该节点的左参考点$P_l^{(1)}$到其父参考点$P_l^{(0)}$之间的距离 $d(P_l^{(1)},P_l^{(0)})$,$O_r^{(1)}$表示该节点的右参考点 $P_r^{(1)}$到其父参考点 $P_l^{(0)}$之间的距离 $d(P_r^{(1)},P_l^{(0)})$.

2.2 rgh-tree的生成算法

rgh-tree 的生成程序是:给定一个含有 n 个数据对象的集合 $D=\{D_1,D_2,\ldots,D_n\}$,一个计算距离的度量函数 $d(D_i,D_j)$,一个二叉 rgh-tree 构造算法如下.

算法 **1**.

输入:D,阈值 d.

输出:rgh-tree 的根 *Root*.

Create_Tree (*Root*,*D*)

begin

(1) If (|D|小于阈值 d)

设 *Root* 为叶子节点,生成一个 *bucket*,用于存放 D,返回该节点.

(2) else

生成两个内节点 N_l 和 N_r,*Root* 是它们的父节点.选择 P_l,P_r 是 D 中的两个数据对象,称为左、右子树的参考点. /*选择 P_l,P_r 的策略参见注释 1*/

R=序列{$d(D_i,P_l)/d(D_i,P_r)|D_i\in D$}中的中位数

令 $C_l=\{D_i|d(D_i,P_l)/d(D_i,P_r)<R$,其中 $D_i\in D$ 并且 $D_i!=P_r\}$

$C_r=\{D_i|d(D_i,P_l)/d(D_i,P_r)>R$,其中 $D_i\in D$ 并且 $D_i!=P_r\}$

令 $a=|C_l|-|C_r|$,$b=|D|-|C_l|-|C_r|$.

令 $E_l=\{D_i|d(D_i,P_l)/d(D_i,P_r)=R$,其中 $D_i\in D$,取前$[(b+a+1)/2]$个数据对象}

$E_r=\{D_i|d(D_i,P_l)/d(D_i,P_r)=R$,其中 $D_i\in D$,取后$[(b-a)/2]$个数据对象} /*[]表示取整函数*/

且满足 $E_l\cup E_r=E,E_l\cap E_r=\varnothing$且$|E_l|-|E_r|=|C_r|-|C_l|$或$|E_l|-|E_r|=|C_r|-|C_l|+1$ /*此式可满足的相关证明见定理 1 和注释 2*/

$D_l=C_l\cup E_l,D_r=C_r\cup E_r$,满足 $D_l\cup D_r=D,D_l\cap D_r=\varnothing$且$|D_l|=|D_r|$或$|D_l|=|D_r|+1$ /*D_l 和 D_r 数据对象的个数相同,或至多相差一个的相关证明见定理 1*/

计算 D_l 和 D_r 的半径.

递归调用 Create_Tree(N_l,D_l)

递归调用 Create_Tree(N_r,D_r)

(3) end if

(4) 返回 *Root*,D_l 和 D_r 的参考点到 *Root* 的参考点的距离.

end

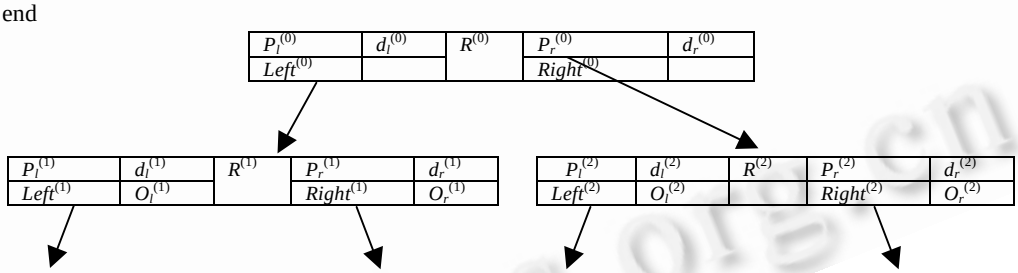


Fig.1 Structure of rgh-tree

图 1 rgh-tree 的结构

关于 rgh-tree 的平衡性有以下结论.

定理 **1**. rgh-tree 是一个平衡树.

证明:只需证明 D_l 和 D_r 包含的数据对象个数相同,或至多相差一个数据对象即可.

一个数列的中位数 R,如果数值的个数为奇数,则排序之后,选其正中间的数即可;如果数值的个数为偶数,则排序之后,选其正中间的两个数的平均数即可.如此得到的中位数 R 一定满足大于 R 的数值个数加上等于 R 的数值个数必然大于等于小于 R 的数值个数.

因为 R 是距离比序列的中位数,所以|{距离比大于 R 的数据对象}|+|{距离比等于 R 的数据对象}|≥|{距离比小于 R 的数据对象}|,即$|C_r|+|E_l|+|E_r|\geq|C_l|$,由 $a=|C_l|-|C_r|$,又由注释 2 知,$|E_l|+|E_r|=b$,$b=|D|-|C_l|-|C_r|$,则 $b\geq a$,即 E_r 的元素个数大于等于 0,同理可证 $b\geq -a$,即 E_1 的元素个数大于等于 0.又由注释 2 知,$|E_l|-|E_r|=|C_r|-|C_l|$ 或 $|E_l|-|E_r|=|C_r|-|C_l|+1$.再根据平衡性定义知,rgh-tree 是一个平衡的二叉树. □

由于 rgh-tree 具有平衡性,因而它可以很容易地分页存储到第 2 级存储中.树的构造阶段需要 $O(n\times\log n)$次距离计算,其中 n 为数据对象的个数.

注释 **1**. 参考点 P_l,P_r 的选取采用在随机子样中找到两个最大距离的一对点.设待抽样数据集合的大小为 M,子样的大小定为 M 的开方,即 $\sqrt{M}$.找到两个最大距离的一对点的时间复杂性为 $\sqrt{M}\times\sqrt{M}=M$.

注释 **2**. 容易看出$[(b+a+1)/2]+[(b-a)/2]=b$,并且$[(b+a+1)/2]-[(b-a)/2]=a$ 或 $a+1$.

3 rgh-tree 上的相似性查询算法

定理 **2**. 如果设 Q 是一个数据对象,r 是半径,P_l,P_r 是左右参考点,R 是正实数,d 是度量,我们有如下结论:

(1) 如果$(d(Q,P_l)-r)/(d(Q,P_r)+r)\geq R$,则对于 $x\in U(Q,r)$,有 $d(x,P_l)/d(x,P_r)>R$;

(2) 如果 $d(Q,P_r)>r$,且$(d(Q,P_l)+r)/(d(Q,P_r)-r)\leq R$,则对于 $x\in U(Q,r)$,有 $d(x,P_l)/d(x,P_r)<R$.

证明:对于 $x\in U(Q,r)$,则 $d(Q,x)<r$,由三角不等式得

$$d(x,P_l)\leq d(x,Q)+d(Q,P_l)<r+d(Q,P_l);$$

同理 $$d(x,P_r)\leq d(x,Q)+d(Q,P_r)<r+d(Q,P_r);$$

同样由三角不等式得 $$d(Q,P_l)\leq d(Q,x)+d(x,P_l)<r+d(x,P_l);$$

$$d(x,P_l)>d(Q,P_l)-r;$$

同理 $$d(x,P_r)>d(Q,P_r)-r;$$

总之,我们有 $$d(Q,P_l)-r<d(x,P_l)<r+d(Q,P_l);$$

$$d(Q,P_r)-r<d(x,P_r)<r+d(Q,P_r);$$

从而有 $$(d(Q,P_l)-r)/(d(Q,P_r)+r)<d(x,P_l)/d(x,P_r)<(d(Q,P_l)+r)/(d(Q,P_r)-r),$$

由上式可知结论成立. □

该定理告诉我们,只需搜索到两个判定点距离之比在$(d(Q,P_l)-r)/(d(Q,P_r)+r)$和$(d(Q,P_l)+r)/(d(Q,P_r)-r)$之间的子节点即可.

推论 **1**. 设条件如定理 2,R 是左右子树分割界面的距离比,则在 rgh-tree 的查找阶段,通过将两个判定点与 Q 的距离 $d(Q,P_l)$和 $d(Q,P_r)$,将得知被查询点所在分支

(1) 如果$(d(Q,P_l)+r)/(d(Q,P_r)-r)\leq R$,则右子树不含有查询结果;

(2) 如果$(d(Q,P_l)-r)/(d(Q,P_r)+r)\geq R$,则左子树不含有查询结果.

证明:(1) 由定理 2 可知,如果$(d(Q,P_l)+r)/(d(Q,P_r)-r)\leq R$,则对于 $x\in U(Q,r)$,有 $d(x,P_l)/d(x,P_r)<R$,则根据树的生成算法可知,右子树不含有查询结果.(2) 同(1)略,结论证毕. □

给定一个数据对象 Q,我们使用如下的搜索算法,查找与 Q 距离不超过 r 的数据对象集合.算法的主要思想是通过估计被查询对象与两个参考点的距离之比来确定查询范围.

搜索算法主程序:

输入:树的根节点 *Root*,查询对象 Q 和半径 r.

输出:与 Q 距离在 r 之内的所有数据对象.

Search_similar(Q,r,*Root*)

搜索算法递归子程序:

Search_similar(Q,r,V)

(a) If 节点为中间节点

(1) If $d(Q,P_l)\leq r$ 或 $d(Q,P_r)\leq r$ then P_l 或 P_r 放在结果集合中 End if /*P_l,P_r 是左、右子树的参考点*/

(2) If $(d(Q,P_l)-r)/(d(Q,P_r)+r)\geq R$ then

If $|d(Q,P_r)-d(P_r,V.right.P_l)|\leq V.right.d_l+r$ 或$|d(Q,P_r)-d(P_r,V.right.P_r)|\leq V.left.d_r+r$ then

If $d(Q, V.right.P_l)\leq V.right.d_l + r$ 或 $d(Q, V.right.P_r)\leq V.right.d_r + r$ then

Search_similar (Q,r,*V.right*)

End if

End if

End if

(3) If $(d(Q,P_l)+r)/(d(Q,P_r)-r)\le R$, then

If $|d(Q,P_l)-d(P_l,V.left.P_l)|\le V.left.d_l+r$ 或$|d(Q,P_l)-d(P_l,V.left.P_r)|\le V.left.d_r+r$ then

If $d(Q,V.left.P_l)\le V.left.d_l+r$ 或 $d(Q,V.left.P_r)\le V.left.d_r+r$ then

Search_similar (Q,r,$V.left$)

End if

End if

End if

(b) else 如果该节点为叶子节点,

对于它的所有数据对象 x 计算 $d(Q,x)$

If $d(Q,x)\le r$ 则将 x 放入返回结果集合中

End if

算法复杂性分析:

定理 3. 假设每个判断成立的概率为 p,如果 p=0.5,则 $T(n)=O(\log n)$,如果 p=0.6,则 $T(n)=\theta(n^{\log_2 2\times p})\approx\theta(n^{1/4})$. 如果 p=0.75,则 $T(n)=\theta(n^{\log_2 2\times p})\approx\theta(n^{1/2})$.

证明:由于是递归算法,则得到算法的时间复杂性满足递归方程:

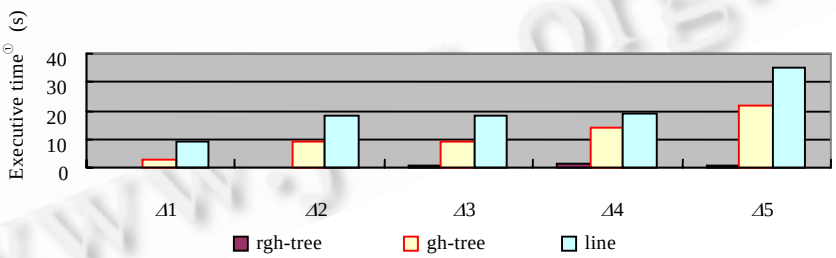
$$T(n)=2\times p\times T(n/2)+O(1).$$

由 Master 定理[13]可知,算法的时间复杂性为 $T(n)=\theta(n^{\log_2 2\times p})$.如果 p=0.5,则 $T(n)=O(\log n)$,如果 p=0.6,则 $T(n)=\theta(n^{\log_2 2\times p})\approx\theta(n^{1/4})$.如果 p=0.75,则 $T(n)=\theta(n^{\log_2 2\times p})\approx\theta(n^{1/2})$. □

4 实验结果

4.1 算法性能对比

我们用 VC++实现了算法,环境为 Windows2000、128M 内存、4G 硬盘的 PII-300MHz 平台.输入数据每维采用 0~1 的随机数,D 表示维数,R 表示记录个数,允许误差 r 为 0.5,y 轴表示 100 次查询的执行时间. $\Delta1=D60R12800$,$\Delta2=D60R25600$,$\Delta3=D120R12800$,$\Delta4=D1200R1280$,$\Delta5=D120R25600$.图 2 对比了该算法、gh-tree 算法和线性扫描算法的执行时间.当数据量增加时,执行时间呈线性增加(如图 3 所示),当维数增加时,执行时间呈线性增加(如图 4 所示).



①执行时间,②实验数据.

Fig. 2 Executive time of algorithms

图 2 算法执行时间

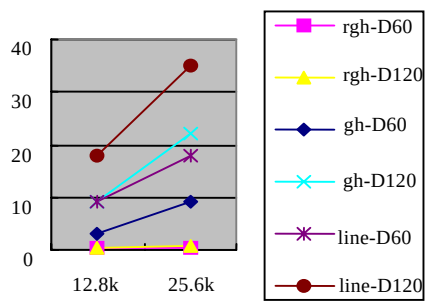


Fig.3 Executive time with datasize

图 3 数据量增加时执行时间的变化

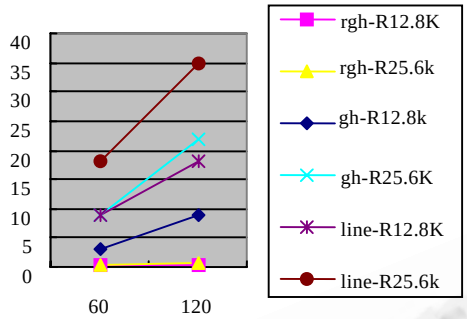


Fig.4 Executive time with dimensional

图 4 维数增加时执行时间的变化

5 结 论

相似性搜索是数据挖掘研究中的一个重要问题.本文分析了已有相似性搜索算法,针对它们存在的问题,提出了一种新的索引结构,并以该结构为基础给出度量空间上的空间分割方法.它利用数据对象与几个固定参考点的距离信息进行数据分割和分布,产生一个各节点区域对称且没有数据重复的均衡树,并给出了基于该索引结构的相似性搜索算法.理论分析与实验结果表明该方法优于已知算法.目前,我们正对该算法的并行化进行深入研究,并将它应用到实际问题中.

References:

[1] Bozkaya, T., Ozsoyoglu M. Indexing large metric spaces for similarity search queries. ACM Transactions on Database Systems, 1999,24(3):361~404.

[2] Agrawal, R., Lin, K-I., Sawhney, H. S., *et al*. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In: Dayal, U., Gray, P.M.D., Nishio, S., eds. Proceedings of the 21st VLDB Conference. Zurich: Morgan Kaufmann Publishers, Inc., 1995. 490~501.

[3] Shafer, J., Agrawal, R. Parallel algorithms for high-dimensional proximity joins. In: Jarke, M., Carey, M.J., Dittrich, K.R., *et al*., eds. Proceedings of the 23rd International Conference on Very Large Data Bases. Athens: Morgan Kaufmann Publishers, Inc., 1997. 176~185.

[4] Agrawal, R., Faloutsos, C., Swami, A. Efficient similarity search in sequence databases. In: Lomet, D.B., ed. Proceedings of the 4th International Conference, Foundations of Data Organization and Algorithms. Heidelberg: Springer-Verlag, 1993. 69~84.

[5] Baeza-Yates, R., Navarro, G. Block-Addressing indices for approximate text retrieval. In: Golshani, F., Makki, K., eds. Proceedings of the 6th International Conference on Information and Knowledge Management. New York: ACM Press, 1997. 1~8.

[6] Hjaltason, G. R., Samet, H. Distance browsing in spatial databases. ACM Transactions on Database Systems, 1999,24(2):265~ 318.

[7] Otterman, M. Approximate matching with high dimensionality R-trees [MS. Thesis]. Department of Computer Science, University of Maryland, College Park, 1992.

[8] Moon, B., Jagadish, H.V., Faloutsos, C., *et al*. Analysis of the clustering properties of the hilbert space-filling curve. IEEE Transactions on Knowledge and Data Engineering, 2001,13(1):124~141.

[9] Uhlmann, J.K. Satisfying general proximity/similarity queries with metric trees. Information Processing Letters, 1991,40(4):175~179.

[10] Patel, J.M., DeWitt, D.J. Partition based spatial merge join. In: Jagadish, H.V., Mumick, I.S., eds. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data. Montreal: ACM Press, 1996. 259~270.

[11] Brin, S. Near neighbor search in large metric space. In: Dayal, U., Gray, P.M.D., Nishioeds, S., eds. Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95). Zurich: Morgan Kaufmann Publishers, Inc., 1995. 574~584.

[12] Ciaccia, P., Patella, M., Zezula, P. M-Tree: an efficient access method for similarity search in metric spaces. In: Jarke, M., Carey, M.J., Dittrich, K.R., *et al* eds. Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97). Athens: Morgan Kaufmann Publishers, Inc., 1997. 426~435.

[13] Knuth, D.E. The Art of Computer Programming. 3rd ed. Baltimore: Addison-Wesley. 1997.

An Algorithm Based on RGH-Tree for Similarity Search Queries*

ZHANG Zhao-gong, LI Jian-zhong

(*College of Computer Science and Technology*, *Harbin Institute of Technology*, *Harbin* 150001, *China*);

(*Heilongjiang University*, *Harbin* 150080, *China*)

E-mail: lijz@banner.hl.cninfo.net; zhangzhaogong@0451.com

http://www.hit.edu.cn

Abstract: Similarity search is a very important problem in data mining. It retrieves similar objects in database and finds proximity between objects. It can be applied to image/picture databases, spatial databases, and time-series databases. For Euclid space (a special metric space), similarity search algorithms based on R-tree are efficient in low-dimensional space, but degenerate into linear scan for high-dimensional space. This phenomenon is called dimensionality curse. This paper presents a new partition and index method of metric space, rgh-tree which distributes and partitions objects by using distance information of objects with few fixed reference. It produces a balance tree with no data overlay. In addition, an algorithm based on rgh-tree, which is suitable for similarity search in metric space, is presented in this paper. The algorithm overcomes the shortcomings of the exiting algorithms, which has less I/O cost and times of computing distance, with average complexity $o(n^{0.58})$.

Key words: algorithm; similarity search query; metric space; database

* Received October 15, 2001; accepted April 22, 2002

Supported by the National Natural Science Foundation of China under Grant No.69873014; the National High-Tech Research and Development Plan of China under Grant No.2001AA415410; the National Grand Fundamental Research 973 Program of China under Grant No.G1999032704; the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.2000021303; the Natural Science Foundation of Heilongjiang Province of China under Grant No.F00-11