空间相似查询中 MBR 边界区域关系研究①

Boundary Region Relation of MBR in Spacial Similarity Search

闫 实 (牡丹江医学院 计算机技术与信息中心 黑龙江 牡丹江 157011)
王学良 (黑龙江大学 计算机学院 黑龙江 哈尔滨 150086)

摘 要: 相似查询是基于向量空间的一种重要查询方法。点、线段、区域是向量空间对空间对象的三种基本表达。本文在不改变结点 MBR 区域前提下，通过区域扫描对 MBR 区域重叠面积进行计算。利用 R*树结点 MBR 允许重叠的特性，在不能消除区域重叠产生的死空间情况下，研究了更为精确的 MBR 边界的线段关系，并给出线段的最近邻查询算法和相似线段选取算法。实验结果表明该方法的 CPU 计算代价较低且显著提高了相似查询与更新的效率。

关键词: 相似查询

1 引言

空间数据库技术主要是提供数据库中对象的集合模型和针对空间数据的索引及查询的支持，其在地理信息系统(GIS)，计算机辅助设计与制造(CAD/CAM)，决策支持(DSS)，多媒体数据库等方面都有着广泛的应用。传统数据库用来处理预先定义好结构的数据对象，并通过相关对象建立索引来优化查询。这些查询通常是把对象看成多维空间中的点，使用点访问方法来进行查询[1]。

常见的查询有最近邻查询，反向最近邻查询，约束近邻查询，范围查询以及最近对查询等[2-4]。空间数据的结构大多是半结构或无结构化的。除空间数据之外还有一些如图像、视频、时间序列、DNA 序列等复杂的数据。这些数据通常既不能排序，也不进行精确的相等比较。对于这些数据，相似性查询是一个更适合的查询方法。这些数据对象通过一个特征集合来进行描述，称为特征向量。如找到与某条特定 DNA 序列相似的序列。这些应用通常可以概括为给定某个对象 q，以及查询半径(误差范围)r，对某个数据库 S 基于某种相似性度量找到一个子集 P，使得对于 P 中的每一个对象 p，有 d(p, q) ≤r，这种查询一般称为范围查询(range query)，范围查询有一种扩展称为 k 最近对查询(k-CPQ)：给定对象 q，找到和 q 最相似的 k 个对象对。对于范围查询和 k-CPQ 查询本文统称为相似查询。查询效率和准确性是衡量数据库系统性能的重要标准。在众多的空间索引结构中，以 R*树[1]应用最为广泛和深入。R*树作为一种多级平衡树，它是 B 树在多维空间的扩展。但是 R*树查询性能在很大程度上受节点覆盖和交叠两个因素影响。R*树的插入操作会引起结点分裂，因此有许多研究都涉及到结点最小外包矩形(MBR)。目前多数研究集中在改变其形状上[6]，而对 MBR 自身边界性质和其中抽象出的线段关系却研究较少，而这正是 MBR 重叠造成死空间最关键的问题所在。由于死空间的存在可能导致部分查询会在空白区域(即没有数据集的区域)进行无效查询。

本文针对 R*树的结点 MBR 结构，通过区域扫描计算 MBR 相关属性，以较小的时空代价解决了更新操作后由于插入新的数据点而出现更多的结点分裂，提高了 R*树路径选择准确率。为进一步提高相似查询效率和准确性，对 MBR 进行了扩展，当某些向量空间对象无法抽象为点的情况时，提出了线段的相似最近对查询，并给出实验结果与分析。

① 收稿时间:2008-07-14

2　预备知识

定义 1[1] R*树的结点结构描述如下：(obj,MBR)为叶子结点；(CP,MBR)为索引结点。其中(obj,MBR)也称为数据项，obj 为对象标识，MBR 为该对象在 k 维空间中的最小外包矩形；(CP,MBR)也称为索引项，CP 为指向子根结点指针，MBR 代表其子树索引空间。

其中 MBR 由左下角和右上角的坐标组成。对二维空间而言，MBR 可表示为(xmin,ymin)，(xmax,ymax)。除根结点外，每个结点的 MBR 数量为 m 和 M 之间。规定 $M/2 \le m \le M$ (m 为最小记录数,M 为最大记录数)。

定义 2 [3] 2d 空间对象的 MBR 之间的距离。如图 1 所示:

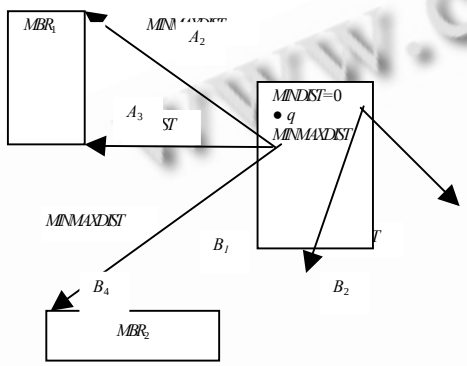

图 1　MBR 与 MBR 的距离定义

设 MBR_1 的四条边为 A_1,A_2,A_3,A_4; MBR_2 的四条边为 B_1,B_2,B_3,B_4。

$MINDIST(M_A,M_B) = \min_{i,j}$ (MINDIST(A_i, B_j));

$MINMAXDIST(M_A,M_B) = \min_{i,j}$ (MAXDIST(A_i, B_j));

定义 3[6] 对于任意的两个空间对象 obj_1 和 obj_2，它们称为ε-相似，当且仅当 $sim(obj_2,obj_1) < \varepsilon$。sim 是相似函数。其中 $obj_1 \times obj_2 \to R^+$。特别地，当 $\varepsilon=0$ 时，ε-相似转化为ε一致，即空间对象相同。

定义 4[6] 对于任意的两个空间 obj_1 和 obj_2，它们称之为 NN-相似，当且仅当 $\forall ob_j \in R^+$, $obj \neq obj_1$, $sim(obj_2,obj_1) \le sim(obj_2,obj)$。当考虑到 d 维数据空间时，$obj_x$ 的形式可表示为(1…)，obj_y 的形式可表示为(1…)。假定对象的属性单位标准化，即 $0 \le x_i \le 1$，$0 \le y_i \le 1$，(i=1 … d)，那么 $sim=(\sum_{i=1}^{d}|(x_i - y_i)^p|)^{1/p} \le \varepsilon$ $(1 \le p \le \infty)$。本文后续研究将采用 p=2，即欧氏空间。

定义 5[6] 已知一个空间查询对象 obj_s，找到空间中的所有与 obj_s ε相似的 obj，则称为ε-相似查询 $\{obj \in R^+ | sim(obj_s,obj) < \varepsilon\}$。类似地，NN-相似查询进一步缩小查询范围。已知一个空间查询对象 obj_s，找到空间中所有与 obj_sNN-相似的 obj_{NN}，则称为 NN- 相似查询 (Nearest Neighbor)，$\{obj \in {}_{NN}R^+ | \forall obj \in R^+, obj \neq obj_{NN}, sim(obj_s,obj_{NN}) \le sim(obj_s, obj)\}$，在本文中简称相似查询。

相似查询中，当插入一个空间对象，基于 R*树结构将从根结点开始采用深度优先策略，遵循最小覆盖面积优化原则找到索引项。具体情况如下：(1)包含新增对象，在非叶结点中找到 MBR 面积增量最小的索引项。(2)若增量相同，直接定位非叶结点 MBR 的面积最小的索引项。结点分裂有三种算法，分别是指数级，平方级和线性级[7]。三种分裂算法都追求最小化结点分裂产生的非叶结点 MBR 面积。其中指数级算法得到全局最优解，但是算法时间复杂性过高。而线性级算法通过找到距离最远的结点来创建两个集合，并通过递归得到次优解。

3　MBR 重叠区域的面积计算

当空间对象插入后，需要调整相应的 MBR。由于空间对象插入实质上并未改变矩形形状，因此只需对每个 MBR 的至多一对互相垂直边界进行调整。由于存在扫描区域的划分，重叠边界可以映射为一维数组，所以由空间对象插入导致的结点分裂，其 MBR 重叠边界的出现可通过区域扫描转化为二叉线段树中结点的插入来实现。因此二叉树线段可以描述空间重叠边界[8]。

首先考虑 1d 空间情况，将平行于 x 轴的 n 条线段设为 $L_1 \cdots L_n$。设 L 为线段的集合，那么 L 的所有元素的扫描区域的并集即为所求，其算法如下：

算法 1：扫描线段集合的有效长度 length(v)

输入：L 中 L_i 的横坐标；

输出：L 中元素的重叠长度；

begin

将 l_i 的端点横坐标 x_{i-1},x_i 存入数组;
$x_1 \leftarrow x_2$, length(v) $\leftarrow 0$,L$\leftarrow 0$; //初始化 L 并集
for i=1 to 2n do
if length(v) $\neq 0$ then L$\leftarrow$L+x_i-x_{i-1}
if x_i 是左端点 then length(v) $\leftarrow$length(v)+1;
else length(v) $\leftarrow$length(v)-1;
end

定理 1 算法 1 是正确的，可终止的，其时间复杂性为 O(nlogn)。

由算法 1 进而可得到计算 MBR 重叠区域面积的方法。其中 L*()代表结点处的垂直线段长度。计算重叠区域面积算法如下：

算法 2：计算重叠区域面积 area(L)
输入：重叠区域边界坐标；
输出：MBR 重叠区域的面积；
begin
x_{i-1}:=x_i;
L:=0;
area(L):=0;
if length(v) $\neq 0$ then L*(v):=L*(v)+y_i-y_{i-1} //求 MBR 的高
else if v 不是叶结 then L*(v):=L*($Child_L$(v)+$Child_R$(v));
else L*(v):=0;
for i=1 to 2n do
area(L):= area(L)+L*×(x_i-x_{i-1}); //求 MBR 的面积
if x_i 是左端点 then insert(L(v),R(v);v));
else 取消 insert(L(v),R(v);v));
end

定理 2 算法 2 是正确的，可终止的，其时间复杂性为 O(nlogn)。

4 MBR 边界关系

4.1 问题描述

在 NN-相似查询中，根据其定义可知需满足 sim(obj_2,obj_1) $\leq$sim(obj_2,obj)。当空间对象如图 3 所示沿一定倾角方向分布的时候，MBR 边界区域会出现两个问题：(1)MBR 的范围扩大，造成存储空间的浪费。特别当分散程度越大，周长等参数的增加也越大。容易证明，在同等面积下矩形长宽比例越大，周长越长。(2)根据 NN-查询的定义 NN-查询的相似函数 sim 要小于等于 ε-相似查询的相似函数 sim。文献[9]中采用聚类技术虽然可适用于密集型分布的空间对象，但是如果空间分布即包含密集型又包含分散型，且密集与分散成带状分布时则效果较差。例如，现代城市建筑的分布带有典型的密集型与分散型混合的特征。如图 2 所示，以 MBR1 为最小外包矩形虽然可以全部包含空间对象，但是边界附近的对象较少，造成空间浪费。在以 O2 为中心的 MBR2 中，空间对象分布较集中，分布呈区域性特征。因此 MBR 得到的结果不能保证得到 NN-相似查询，由此可能会出现查询结果不一致。非正交的重叠区域为形态往往是比较复杂的，并且难于计算其参数信息。针对这种情况我们提出了一种基于线段的相似查询方法。

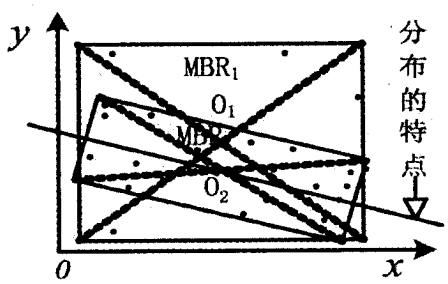

图 2　根据分布变化的 MBR

4.2 MBR 边界区域线段关系判定

定义 6[9] 已知线段 K 和线段 L。如果线段 K 的两个端点位于线段 L 的两侧，且线段 L 的两个端点也位于线段 K 的两侧，则线段 K 和线段 L 相交。

定理 3 已知线段 K 和线段 L，且线段 K 的两个端点 s,e$\in$area(L)，则 sim(K,L)= min(sim(s,L),sim(e,L))。

证明：线段上的任意一点到另一线段的相似距离是：如果该点到线段的垂足在线段上，则相似距离是该点与垂足之间 y 的距离; 如果该点到线段的垂足不在线段上，则相似距离需要另行确定。由已知得线段 K 上的任意一点到线段 L 的垂直线段都在 area(L)中(即垂足在线段上)。分两种情况讨论：

(1) K$/\!/$L

当 K$/\!/$L 时，线段 K 上任意一点到线段 L 的相似距离是一个常数，故成立。

(2) K$\not/\!/$L

K$\not/\!/$L 时，计算得到 sim(s,L)和 sim(e,L)的最小

值，则线段 K 上的任意一点到线段 L 的垂直距离一定大于或等于该最小值。故结论成立。证毕。

定义 7 当线段 K 和线段 L 不平行时，过线段 K 的一个端点做线段 L 的平行线。如果线段 K 的另一个端点与线段 L 位于平行线的同侧，称线段 K 相对于线段 L 内倾；如果线段 K 的另一个端点与线段 L 位于平行线的两侧，称线段 K 相对于线段 L 外倾。

定理 4 已知线段 K 和线段 L，线段 K 的两个端点为 m 和 n，m∈area(L)，n∉area(L)，且线段 L 的两个端点 s，e∉area(K)，则 sim(K,L)=min(sim(m,L), sim(s,n),sim(e,n))。

证明：根据已知条件，线段 K 和线段 L 一定不平行，分两种情况：

(1) 线段 K 相对于线段 L 内倾

如图 3 所示，线段 A 相对于线段 L 是内倾，线段 B 相对于线段 L 是外倾。

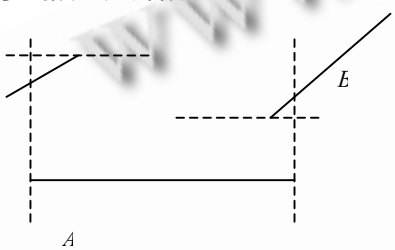

图 3　线段的内倾和外倾

线段 A^{s} 和线段 L，L^{e} 因为线段 L 的端点 s 和端点 e 都不在 area(A)中，故过端点 s 和端点 e 做线段 A 的垂直线，垂足必定位于线段 A 的延长线上

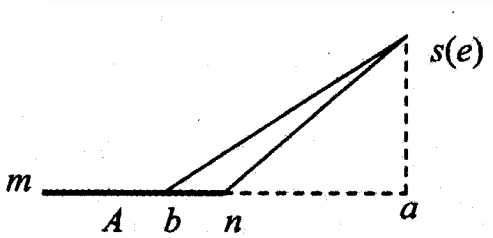

图 4　area(A)外一点到线段 A 的相似距离

如图 4 所示，sim(s,a)为点 s 到线段 A 的垂直距离，n 为线段 A 的端点，b 为线段 A 中的任意一点。假设 sim(s,b)< sim(s,n)，为线段 K 和线段 L 的最近邻。但 sim(a,n)必定小于或等于 sim(a,b)(b 为线段 A 中的任意一点)，所以 sim(s,n) ≤sim(s,b)，与假设矛盾。故 sim(s,n)或 sim(e,n)为点到线段的相似距离。

(2) 线段 K 相对于线段 L 外倾

因为线段 K 相对于线段 L 外倾，故线段 K 上的任意一点(除点 m)，都位于平行线的外侧，所以线段 K 中点 m 距离线段 L 最近，即 sim(m,L)。证毕。

定理 5 已知线段 K 和线段 L，线段 K 的端点 m∈area(L)，而另一个端点 n∈area(L)，且线段 L 的端点 m′∈area(K)，而另一个端点 n′∈area(K)。则 sim(K,L)= min(sim(m,L)，sim(m′,K),sim(n,n′))。

证明：过线段 K 在 area(L)中的端点 m，做线段 L 的平行线；过线段 L 在 area(K)中的端点 m′，做线段 K 的平行线。据此判断两条线段的内外倾关系。

分三种情况：

(1) 线段 K 相对于线段 L 内倾

线段 K 相对于线段 L 内倾，但不能确定线段 L 相对于线段 K 是内倾还是外倾。因此再分①②两种情况讨论。

① 线段 K 相对于线段 L 内倾，且线段 L 也相对于线段 K 内倾

由定理 4，这种情况 sim(K,L)等于两个不在影响区域中的端点之间的距离，即 sim(K,L)=sim(n,n′)。如图 5 中线段 A。

② 线段 K 相对于线段 L 内倾，但线段 L 相对于线段 K 外倾。

因为线段 L 相对于线段 K 外倾，由于 sim(K,L)= sim(L,K)，则 sim(K,L)=sim(m′,K)。如图 5 中线段 B。

(2) 线段 K 相对于线段 L 外倾

因线段 K 相对于线段 L 外倾，根据线段的平行线外的点，到该线段的距离一定大于两条平行线之间的距离，因此 sim(K,L)=sim(m′,L)，如图 5 中线段 C。

(3) K∥L

根据已知条件，当且仅当线段 K 的一个端点位于线段 L 的直接影响区域的边界上，两条线段才能平行，故两条线段的相似距离为 sim(m,L)。如图 5 中线段 D。故结论成立。证毕。

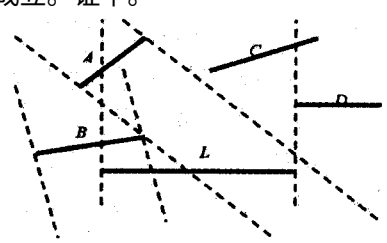

图 5　线段的内倾和外倾

定理 6 如果线段 K 的两个端点 s,e∉area(L)，且线段 L 的两个端点 s′，e′∉area(K)，则：

sim(K,L)=min(sim(s,s′),sim(s,e′),sim(e,s′),sim(e,e′))。

证明：因为两条线段相互都不在对方的直接影响区域内，所以，点到线段的垂足一定在线段的延长线上，即一定不是线段与线段的相似距离，故只需考虑点与点的距离。根据定理 4，端点之间的距离为相似距离，本定理中包括了所有端点之间的距离，故结论成立。证毕。

4.3 MBR 边界关系相关算法

根据以上分析，线段间相似查询算法如下：

```
算法 3：L-CPQ Search(Node R.root, L)
输入：R 树根结点 R_root，线段 L；
输出：线段间的最近对；
begin
  sim:=∞;stack.push(R_root);
  current:=stack.pop();
  //将 R 树根结点压栈,将栈顶结点赋于 current
 //顶结点出栈
  while stack 非空 do
    if current∉MidNode(R) then
      if MINMINDIST(current,MBR_L)>sim then
        if stack.empty then
        return K;    //返回 K
        else current:=stack.pop();
      else stack.push(current.child); //将子结点
  压栈
         current:=stack.pop();
       else if MINMINDIST(current,MBR_L)=0
  and
          intersect(silbling_current,L) then return
  sibling_current;  //返回当前兄弟结点
    else sign:=position(silbling_current,L); //判
  断两条线段的位置关系,赋值 sign
      if d_tem<sim then
       begin
         sim:=d_tem;
         K:=silbling_current;
         else if stack.empty then return K; //返回 K
        else current:=stack.pop(); //将当前结点
  置出栈
        end
      return (L,K);   //返回线段最近对
end
```

定理 7 算法 3 是正确的和可终止的，其算法时间复杂性为 O(n)。

5 实验结果与分析

平台配置：CPU Pentium D 3.0GHz，1G 内存，Windows XP，Borland C++ builder 6.0 编程实现。测试数据是 the R-tree Portal ① 提供的 SpatialDataGenerator 随机产生的 uniform 类型的 2d 空间数据。R*树结点大小设为 1KB，包含的数据最大值为 100，最小值为 50。页面初始大小设定为 1KB。采用 LRU 缓存策略，初始时仅保存根结点所在页面，缓存最多可加载 256 个页面。文献[10]给出了基于压缩 B+树的线段数据库查询方法。借鉴其存储方法我们对本文方法进行了验证。

实验分两部分：①将本文提出的基于线段的相似查询与基于 MBR 的最近对查询(即 CP-search)[3]在插入新的对象时的更新性能进行对比。其实验内容是改变插入对象数量时结点存取次数的对比。②改变 uniform 数据集中数据数量，给出 CPU 运算时间。

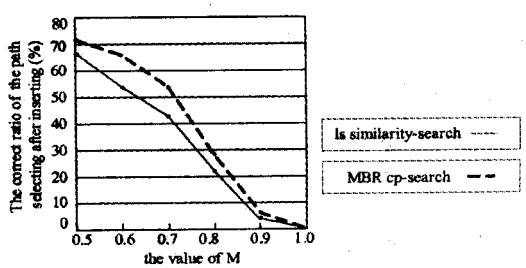

图 6 插入路径选择的准确率对比

如图 6 所示，在插入路径选择的对比中，随着 M 值的增大，两种方法正确率均下降，这是由于结点的饱和度增加造成 MBR 重叠区域增加，死空间的出现概率增大，线段相似查询的正确率低于 MBR 最近对查询。这是由于相似查询的是近似的结果并非是精确，因此在路径选择的正确率上不如精确的基于 MBR 的最近对查询。而这种正确性的降低带来的是查询效率的提高。如图 7 所示，当插入新的空间对象时，结点

分裂与重插需要访问大量的 R*树结点，线段相似查询的结点访问次数明显少于精确的基于 MBR 的最近对查询，提高在 50%以上。由此可见对 MBR 边界区域的相似查询基于线段的方式是有效的。为了避免随机性带来的影响，取 100 次运算的平均值作为实际指标。

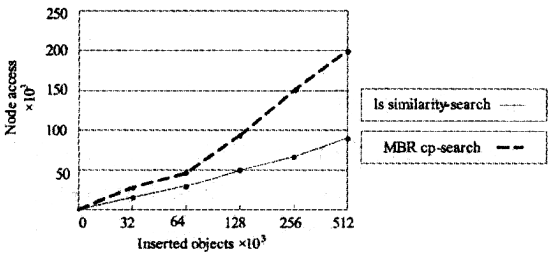

图 7　插入空间对象后的结点存取次数比较

最后，为了验证计算线段间最近距离是否具有良好的可伸缩性且运算时间在合理的范围内，实验对 5000~50,000 个线段中找到其中的线段最近对的距离计算时间进行了统计。同样为避免随机性带来的影响，循环执行计算 100 次，得到执行数据结果，如图 8 所示。实验表明每次执行时间最多不超过 10s。而 100 次执行时间也在可以接受的时间范围内且运算时间近似线性，因此算法具有较好的可伸缩性。

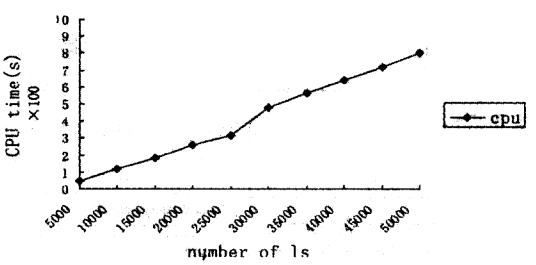

图 8　线段相似计算 CPU 时间

图 9 为实验 1 的初始数据，由 SpatialDataGenerator 产生的 100,000 个 2d 空间数据的 uniform 数据。

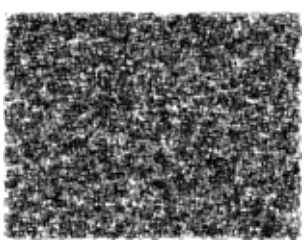

图 9　10,000 个 uniform 分布的 2d 对象

6　结论

本文研究了空间相似查询中 R*树的结点 MBR 边界关系，在此基础上计算了 MBR 重叠区域的面积，给出了基于线段关系的定义、定理，并提出其相似查询算法和相似线段选择算法。该算法扩展了结点 MBR 自身属性，使 MBR 不仅局限在矩形区域内，而是可以借助线段的关系来判断其边界关系。因此，在近似性和效率上取得较好的折衷。实验结果表明本文算法具有较好的性能。

参考文献

1 Gaede V,Gunther O. Multidimensional access methods. ACM Computer Survey, 1998, 20(6):170–231.

2 Yufei T,Dimitris P,Qiongmao S.Continuous Nearest Neighbor Search. Proceedings of the 28th VLDB Conference. Hong Kong, 2002:287–298.

3 Corral A, Manolopoulos Y, Theodoridis Y, Vassilakopoulos M. Algorithms for processing k-closest-pair queries in spatial databases.Data & Knowledge Engineering, 2004, 49(2):67–104.

4 Haibo H.,Dik Lun L. Range Nearest Neighbor Query. IEEE Transactions on Knowledge and Data Engineering,2006,18(1):78–91.

5 刘小峰,刘云生,肖迎元.空间数据库中约束 K 最接近对查询.计算机科学,2006,33(5):156–158,165.

6 Bohm C,Berchtold S, Keim D. Searching in High-dimensional Spaces-index Structures for Improving the Performance of Multimedia Databases. ACM Computing Surveys, 2001, 33(3):322–373.

7 Yongjian Fu, Juiche T, Subramanya S.R. Node splitting algorithms in tree structured high–dimensional indexes for similarity search. Proceedings of the ACM Symposium on Applied Computing, 2002:766–770.

8 周培德.计算几何.北京:清华大学出版社.2005.

9 黄添强,秦小麟,王金栋.多代表点特征树与空间聚类算法,计算机科学,2006,33(12):189–195.

10 Hungyi L.Efficient and Compact indexing structure for processing of spatial queries in line–based databases Data & Knowledge Engineering,2008, 64(1):365–380.