

一种多维向量并行查询算法

黄 海, 侯 颖, 朱圣平

(解放军信息工程大学, 河南 郑州 450002)

摘 要: 为了解决多维向量数据快速查询问题, 在查询范围上限已知的条件下, 通过对数据集采用最近邻准则进行空间划分, 构造一种多叉扩展平衡索引树, 并设计了索引树的串行和并行查询算法. 最后, 对并行查询算法的性能进行了分析, 测试结果验证了该方法的有效性.

关键词: 多维索引结构; 并行计算; 相似查询

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2010)08-0017-03

A Multi-dimensional Vector Parallel Search Algorithms

HUANG Hai, HOU Ying, ZHU Sheng-ping

(PLA Information Engineering University, Zhengzhou 450002, China)

Abstract: To solve the quick search problem of multi-dimensional vector data, this paper constructed a multi-branch extended and balanced index tree, by using the nearest neighbor criterion to realize the space division of the data sets, under the condition that the upper limit of the query scope is known. Then, the serial and parallel search algorithms of the index tree were designed. At last, the capability of the parallel search algorithms was analysed and the test results validated the validity of this method.

Key words: multi-dimensional search structure; parallel computing; similar query

1 引言

在基于内容的多媒体信息查询中, 一般是根据信息特征的空间距离来判断信息的相似程度. 图像、音视频等多媒体信息特征往往是以高维向量的形式存在, 高维向量的空间距离计算代价巨大, 当查询的信息数据集很大时, 使用顺序搜索方法难以满足实时应用的需求.

针对多维空间的相似度查询, 常规的解决方法是通过构造支持快速检索的索引结构, 通过三角不等式剪枝, 尽可能减少距离计算次数, 实现对多维数据的快速检索. 常用的高维数据的检索结构包括: R-Tree、R*-Tree 和 ESR-Tree^[1-3]等等. 但以上结构在恶劣情况下, 如查询子树包络的边界区域时, 需要遍历整个索引树^[4], 导致查询的整体性能下降. 文献[5]提出一种支持快速相似检索的二叉树结构—— η

树, 在查询阈值上限确定的条件下, 查询只在 η 树的一条路径上进行. 但是以上算法的并行性不是太好, 在对超大规模数据进行检索时, 尤其是需要进行复杂的距离计算时, 存在检索时间长、响应不及时的特点. 本文提出一种适合并行计算的多叉树索引结构, 并给出基于该索引结构的并行搜索算法, 试验结果表明, 该算法能够有效提高相似查询的速度.

2 多叉索引树

2.1 基本定义

设 D 为包含 n 个对象的数据集 $D = (d_1, d_2, \dots, d_n)$, 其中 d_i 为高维空间的一个点, $d(x, y)$ 为空间距离函数, 其满足三角不等式.

定义 1 数据集 D 的 m 阶 Voronoi 划分定义为: 根据数据空间中的 m 个代表点 $c_1, c_2, \dots, c_m$, 将数据集 D 分为 m 个子集 $D_1, D_2, \dots, D_m$, 如果对 $\forall x$

$\in D_i$, 均满足 $d(x, c_i) \leq (d, c_j), j \neq i$, 则该划分称为数据集 D 的 m 阶 Voronoi 划分。

定义2 数据集 D 的 m 阶平衡 Voronoi 划分定义为: 如果 $D_1, D_2, \dots, D_m$ 为数据集 D 的 m 阶 Voronoi 划分, 并且满足 $\max(|D_i| - |D_j|)$ 最小, 则称该划分为数据集 D 的 m 阶平衡 Voronoi 划分。

定义3 数据集 D 的 m 阶扩展平衡 Voronoi 划分定义为: 如果 $D_1, D_2, \dots, D_m$ 为数据集 D 的 m 阶平衡 Voronoi 划分, 对于相似查询上限 t 满足 $\bigcup_{i \neq j} D_{ij}$ 包含元素最少, 则称 $D'_1, D'_2, \dots, D'_m$ 为数据集 D 的 m 阶扩展平衡 Voronoi 划分。其中

$$D'_i = D_i \cup D_{i1} \cup D_{i2} \dots \cup D_{im}$$

$$D_{ij} = \{x \mid \text{abs}(d(x, c_i) - d(x, c_j)) \leq 2t \wedge x \in D_j\}$$

图1描述了一个2维空间的3阶扩展平衡 Voronoi 划分。

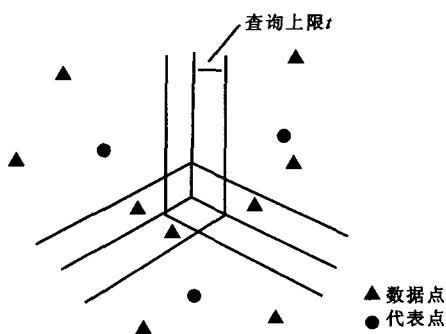


图1 3阶扩展平衡 Voronoi 划分示意图

2.2 多叉索引树的定义和构造

对数据集 D 进行 m 阶扩展平衡 Voronoi 划分, 可以得到数据集 D 的 m 阶多叉索引树。该索引树由叶子节点和内部节点组成。

叶子节点 L 结构为: $(l_1, l_2, \dots, l_k)$, 其中 l_i 为数据集 D 中的一个点。

内部节点 N 结构为: $(N_1, N_2, \dots, N_k), N_1, N_2, \dots, N_k$ 为节点 N 所包含数据集的 m 阶扩展平衡 Voronoi 划分, $k \leq m$, 其中 $N_i = (c, \text{childpoint})$, c 为 N_i 包含子树的数据集合的代表点, childpoint 为指向子树的指针。

根据多叉树的定义, 设计高维空间数据集 D 的 m 阶多叉树的构造算法如下:

(1) 通过最优化方法将集合 D 进行 m 阶扩展平衡 Voronoi 划分, 得到 m 个数据集 $D_1, D_2, \dots, D_m$;

(2) 得到根节点的 m 棵子树 $TD_1, TD_2, \dots,$

TD_m ;

(3) 如果 TD_i 包含元素数量小于 m , 则 TD_i 为叶子节点。

(4) 否则, 对每个子树重复以上步骤, 进行递归处理, 直到所有节点均为叶子节点。

形成的多叉索引树如图2所示。

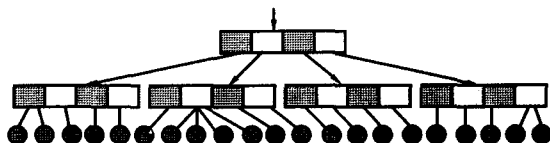


图2 多叉索引树

2.3 多叉索引树的相似查询

对于相似查询^[4] $q(d, t)$, d 为查询点, t 为相似度阈值, 根据多叉索引树的定义, 使得 $d(d, c_k) = \min(d(d, c_i))$ 成立的代表点 c_k 所在子树为查询结果所在子树。

因此多叉索引树相似查询算法如下:

(1) 计算查询点 d 与内部节点中 k 个代表点的距离, 取使得 $d(d, c_k) = \min(d(d, c_i))$ 成立的代表点 c_k 所在子树指针 childpoint 。

(2) 如果 childpoint 所指向的节点为内部节点, 重复(1)的处理。

(3) 如果 childpoint 所指向的节点为叶子节点, 计算查询点 d 与叶子节点中 k 个数据点的距离, 如果 $d(d, l_k) < t$, 则将 l_k 放入为查询结果集合中, 否则丢弃该数据点。 k 个数据点计算完毕后, 查询结束。

3 基于多叉索引树的并行查询算法

3.1 算法描述

考察多叉索引树^[5] 的结构, 我们提出多叉索引树的并行查询算法。其核心思想是: 将每个代表点或数据点与查询点的距离计算利用多个从节点机并行处理, 然后主节点机将计算结果汇总判决, 选取后续子树递归处理, 直到得到查询结果。算法中节点机个数为 $p+1$, 其中 p 为从节点机个数, 并且 $0 < p \leq m$ 。

具体算法描述如下:

(1) 主节点机处理过程

输入: 查询点 d , 相似范围 t

输出: 查询结果

① 将内部节点中 p 个代表点和查询点 d 发送到从节点机进行距离计算;

② 等待接收从节点机的距离计算结果;

③ 收到从节点机的距离计算结果, 如果该节点中还有代表点未进行计算, 将该代表点发送到从节

点机进行计算,并返回①,否则转到④执行;

④ 取使得 $d(d, c_k) = \min(d(d, c_i))$ 成立的代表点 c_k 所在子树指针 childpoint, 如果 childpoint 所指向的节点为内部节点返回①执行, 否则转到⑤执行;

⑤ 将叶子节点中 p 个数据点发送到从节点机进行距离计算;

⑥ 等待接收从节点机的距离计算结果;

⑦ 收到从节点机的距离计算结果, 如果距离小于相似范围 t , 将该数据点写入查询结果集; 如果该叶子节点中还有数据点未进行计算, 将该数据点发送到从节点机进行计算, 并返回⑤, 否则程序结束;

(2) 从节点机处理过程

① 接收主节点机传来的待计算数据点;

② 按照距离公式计算两个数据点之间的距离;

③ 将计算结果发送给主节点机;

④ 返回①执行.

3.2 算法分析和测试

下面采用 LogP 模型对并行查询算法进行性能分析.

设数据集中数据数目个数为 N , 索引树的扇出度为 m , 数据维度为 d , 数据点空间距离计算时间为 t_d , 主子节点机之间传送单位数据的时间为 o , 从节点机数为 p . 为简化分析, 本文忽略扩展区域点的数量, 并只对距离计算时间进行评估.

串行查询算法计算时间为

$$T_s \approx t_d m \log_m N;$$

并行查询算法并行计算的时间为

$$T_{\text{comp}} \approx t_d (m/p) \log_m N;$$

并行查询算法通信时间包括主节点机向子节点机发送数据点的时间和接收子节点机回送的距离结果的时间, 表示为

$$T_{\text{comm}} \approx o * m * d * \log_m N + o * m * \log_m N;$$

并行算法总时间为

$$T_p = T_{\text{comp}} + T_{\text{comm}}$$

$$\approx (t_d(m/p) + o * m * (d + 1)) \log_m N;$$

算法计算加速比为

$$S = T_s / T_p$$

$$= t_d m / (t_d(m/p) + o * m * (d + 1))$$

$$= p * t_d / (t_d + o * p * (d + 1)).$$

我们对不同并发节点机情况下, 多叉索引树的处理时间进行了试验, 并行节点机数量从1到16, 试验数据集为500 000个均匀分布的100维向量, 索引

树扇出度 m 取为16, 随机生成查询数据点. 分别取距离计算代价 t_d 与传送代价 $(o * d)$ 的比值为1, 10, 100进行测试, 试验结果如图3所示.

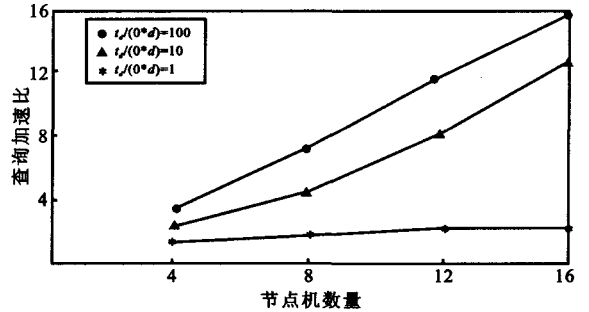


图3 测试结果示意图

测试结果表明, 当距离计算代价 t_d 与传送代价 $(o * d)$ 的比值增大时, 查询加速比趋近 $O(p)$. 而当距离计算代价与传送代价相当时, 加速比近似于1. 在实际应用中, 由于距离计算代价大大超过传送代价, 因此该算法能较大提高处理效率.

4 结束语

针对超大规模多维数据集, 串行查询算法难以满足实时性的需求, 在查询范围上限已知的条件下, 采用 m 阶扩展平衡 Voronoi 划分构造多叉索引树, 并提出基于多叉索引树的并行查询算法. 通过对并行算法的分析表明, 当数据点空间距离计算时间 t_d 远大于主子节点机传送数据点的时间时, 算法的加速比趋近于节点机数 p , 显著提高了查询的速度.

参考文献:

- [1] Guttman A. R-tree: a dynamic index structure for spatial searching[C]// Proc. of the ACM SIGMOD Intl Conf on Management of Data. Berkeley, 1984:47-54.
- [2] Beckmann N, Kriegel H, Schneider R, et al. The R*-tree: an efficient and robust access methods for points and rectangles[C]// Proc. ACM SIGMOD Int. Conf. On Management of Data. Atlantic, 1990:322-331.
- [3] 徐焕, 林坤辉. ESR-Tree: 一种多维对象的动态索引方法[J]. 计算机应用, 2005, 25(12):2872-2873.
- [4] 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[J]. 软件学报, 2002, 13(8):1678-1685.
- [5] 古毅, 吴忠福, 魏丽, 等. 高维空间数据索引结构分析研究[J]. 计算机科学, 2006, 33(5):142-145.

作者简介:

黄海 男, (1975-), 硕士, 副教授. 研究方向为网络安全.