

文章编号:1001-9081(2010)05-1156-03

## P2P网络下的KNN查询

刘丹<sup>1,2</sup>, 谢文君<sup>1</sup>

(1. 武汉大学 遥感信息工程学院, 武汉 430079; 2. 华中师范大学 信息技术系, 武汉 430079)

(rock\_fall@tom.com)

**摘要:** K最近邻(KNN)查询是相似性查询的一种,已有大部分KNN查询算法都是针对集中式计算环境的,因此很容易形成性能瓶颈。P2P这种新的分布式计算技术能够有效克服集中式计算环境中的性能瓶颈问题。提出了一种分组式P2P网络结构下基于iDistance索引的KNN查询方法,其主要思想是通过分布式簇索引裁剪搜索空间,降低网络通信开销,从而在P2P环境下执行KNN查询。最后通过仿真测试了该方法的有效性以及分组数量与数据分布对查询开销的影响。

**关键词:** 分组; 对等网; K最近邻(KNN)查询; 簇; iDistance

**中图分类号:** TP393 **文献标志码:** A

## KNN query on P2P networks

LIU Dan<sup>1,2</sup>, XIE Wen-jun<sup>1</sup>

(1. School of Remote Sensing and Information Engineering, Wuhan University, Wuhan Hubei 430079, China;

2. Department of Information Technology, Huazhong Normal University, Wuhan Hubei 430079, China)

**Abstract:** KNN (K-Nearest Neighbor) query is a kind of similarity query. A majority of KNN query algorithms are intended for concentrated computing environment, which will easily lead to performance bottle-neck. P2P, the new distributed computing technology, can effectively overcome the performance bottle-neck highlighted in the concentrated computing environment. A KNN query approach based on iDistance index in grouping P2P networks was put forward. In the precondition of distributed clustering index prune in searching of space, this approach effectively reduced the network communication overhead, thus the KNN query could be implemented in P2P. The simulation results prove its validity and influences on query spending from the group number and data distribution.

**Key words:** grouping; Peer-to Peer (P2P) network; K-Nearest Neighbor (KNN) query; cluster; iDistance

### 0 引言

K最近邻(K-Nearest Neighbor, KNN)查询即检索与某个给定对象 $q$ 的距离最近的 $k$ 个对象。很多索引技术如M-tree<sup>[1]</sup>、Sim-tree<sup>[2]</sup>、iDistance<sup>[3]</sup>等,就是针对相似性查询而提出的。由于范围查询涉及到大量的距离计算,而大部分查询算法都是针对集中式计算环境的,因此很容易形成性能瓶颈。本文在前期工作的基础上,提出了一种在分组式P2P网络下KNN查询的方法。

### 1 相关工作

P2P环境下的范围查询近年来得到很多关注,有很多P2P环境下的范围搜索方法被提出,比如P2PR-tree<sup>[4]</sup>、Distributed Quadtree<sup>[5]</sup>等,它们都能够用来解决空间范围搜索问题。在空间数据查询中,一般将这类范围搜索归纳为区域搜索,也就是搜索与特定区域(比如矩形)相交的对象。但是本文的工作主要是针对KNN查询,它也是公制空间下相似性搜索的一种。目前针对公制空间下相似性搜索的方法有MCAN<sup>[6]</sup>和M-Chord<sup>[7]</sup>。其中MCAN是利用一种基于pivot的技术将所有数据对象映射到向量空间中;M-Chord也是利用iDistance将数据对象映射到一维空间。但是这两种技术都是基于结构化的P2P技术,也就是说可能会出现由于分布式

哈希表(Distributed Hash Table, DHT)造成的空间数据的物理特性丢失。而本文的算法是基于一种混合P2P模式,在保持空间数据的物理特性的同时保证负载均衡特性。

### 2 分组式P2P网络

#### 2.1 分组式P2P网络结构

首先将系统划分为 $a \times b = m$ 个组,组的个数作为参数可以根据系统的规模进行设置。将整个数据空间按照一定的比例映射到这 $m$ 个组上,这样每个组就对应一个相应的物理空间范围(如图1所示)。对系统进行分组后,为每个组分配一个超级节点 $S_i (1 \leq i \leq m)$ 。每个超级节点维护给所有分组合配的物理坐标范围,定义为 $Area_i (1 \leq i \leq m)$ 。

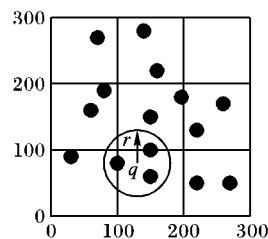


图1 分组示意图

#### 2.2 节点的加入和离开

每个非超级节点的加入之前,首先对本地数据执行簇算

收稿日期:2009-11-27;修回日期:2010-01-13。

作者简介:刘丹(1978-),男,湖北武汉人,讲师,博士研究生,主要研究方向:GIS、P2P; 谢文君(1981-),男,湖北武汉人,博士研究生,主要研究方向:GIS、P2P。

法,以求出本地数据簇中心,并加入管辖范围包含此中心坐标的分组。这么做的目的是尽可能缩小分组的簇范围,从而在同等条件下降低每个组被搜索的可能性。

另外,由于空间数据的非均匀分布特性,当某个超级节点管理的非超级节点个数超过预先设置的门限值的时候,会在空闲节点列表中选取一个节点作为超级节点共同管理某个分组,以避免出现某个超级节点的负载过大的情况。

### 3 KNN 查询

#### 3.1 簇的划分策略

在本文的分组式 P2P 网络中,采用基于空间的簇划分策略。为每个组里的每个节点都选取同样的固定的  $n$  个参考点。一种可选的方式为选取每个组的四条边界线的中心点为固定参考点。存储在每个节点的数据根据这个四个参考点以距离最近原则划分为四个簇。这样一来,一个组中的所有节点的簇就形成了很多以四个固定参考点为圆心,半径最大为  $(1/2) \cdot X$  ( $X$  为组的边界长度) 的同心圆。如图 2 所示。选择基于空间的簇划分策略,并且为每个节点选取同样的、固定的参考点是基于以下几点考虑:

1) 在 P2P 这种网络环境下,存储在节点中的数据在任意时刻都有可能发生变化,因此网络中的节点很难选择一个合适时刻对存储在本地的数据进行基于数据的动态簇划分;

2) 选择基于空间的簇划分策略,并且为每个节点选取同样的固定的参考点这种方式,不管节点中的数据如何变化,簇的参考点都不会发生变化,降低了在本地维护 B+ 树索引的开销。

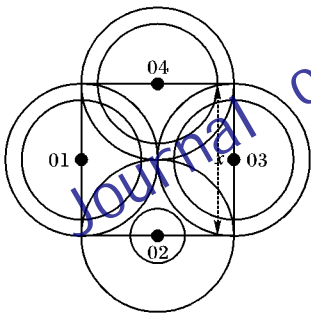


图2 节点簇的划分示意图

#### 3.2 节点簇的建立

现在假设某个节点为  $P$ , 属于某个特定的组。将存储在  $P$  上的所有数据对象根据前面提到的固定参考点划分为  $n$  个簇。将节点  $P$  的簇集合定义为  $CS = \{C_j(O_j, dist\_max_j, dist\_min_j) \mid 0 < j < n\}$ 。其中:  $O_j$  为固定的参考点;  $dist\_max_j$  为簇  $C_j$  中离参考点  $O_j$  最远的数据对象的距离值,即簇的半径;  $dist\_min_j$  为簇  $C_j$  中离参考点  $O_j$  最近的数据对象的距离值。然后把存储在  $P$  中的数据对象根据不同的  $O_j$  计算其  $iDistance$ , 并将这些值作为索引构造一个 B+ 树保存于  $P$  的主存中。除此以外,  $P$  将本地数据中距离本组管辖范围的中心点最远的距离值  $central\_max$  发送给本组的  $S_i$  节点。  $S_i$  节点可以根据本组所有节点的  $central\_max$  中的最大值来确定本组数据的簇范围(即以本组管辖范围的中心点为圆心,  $central\_max$  中的最大值为半径)。

#### 3.3 范围搜索

对于一个范围搜索  $Q(q, r)$ , 采取三个步骤来进行处理:

1) 先求出包含查询范围或与之相交的组集合,并向组集合中的每个  $S_i$  节点发送查询;2) 当某个  $S_i$  节点收到查询消息后,筛选出所有与查询相关的节点簇,并向相应的节点发送查询消息;3) 组中的节点收到查询消息后在本地执行查询,最后分别将查询结果返回给提出查询的节点。下面对第 2) 步进行详细描述。

当每个组中的超级节点收到查询请求后调用 Search 方法(见算法 1)。该方法首先计算查询范围与  $n$  个簇的相交情况,这里可能出现四种情况。如果查询范围与某个节点簇相交,则首先通过 LocateIn 方法定位初始叶节点,该方法以  $dist(O_j, q)$  为参数。如果没有找到与  $dist(O_j, q)$  匹配的叶节点,则返回两个节点并赋值给  $node[]$  数组。这两个节点分别为:第一个索引值大于  $dist(O_j, q)$  的节点(记作  $node[0]$ )和它的相邻左节点(记作  $node[1]$ )。然后分别以  $node[1]$ ,  $dist(O_j, q) - r$  和  $node[0]$ ,  $dist(O_j, q) + r$  为参数通过 searchInward(见算法 2)和 searchOutward(见算法 3)方法求出数据结果集  $RS$ , 否则不用对该簇进行检测,即不需要向管理该簇的节点转发查询消息。

算法 1 Search 算法。

```
Search( $q, r$ )
1) for  $j = 0$  to  $n$ 
2)    $dis = dist(q, O_j)$ 
3)   if ( $dis - r \leq dist\_max_j$ )
4)     if ( $|dis - r| > dist\_max_j$ )
5)       return all objects in  $C_j$ 
6)     end if
7)     if ( $dis + r \leq dist\_max_j$ )
8)        $node = LocateIn(btree, j * c + dis)$ 
9)       searchInward( $node[1], j * c + dis - r$ )
10)      searchOutward( $node[0], j * c + dis + r$ )
11)     else
12)        $node = LocateOut(btree, j * c + dist\_max_j)$ 
13)       searchInward( $node[0], j * c + dis - r$ )
14)     end if
15)   else return null
16) end if
17) end for
```

算法 2 searchInward 算法。

```
searchInward( $node, ivalue$ )
/*  $e_k$  is entry in  $node$  */
1) if (not end of Partion) and ( $e_1.key > ivalue$ )
2)   for each entry  $e_k$  in  $node$ 
3)     if ( $dist(e_k, q) < r$ )
4)        $RS = RS \cup e_k$ 
5)     end if
6)   end for
7) else return  $RS$ 
8) end if
9) searchInward( $node.leftnode, ivalue$ )
```

算法 3 searchOutward 算法。

```
searchOutward( $node, ivalue$ )
/*  $e_k$  is entry in  $node$  */
1) if (not end of Partion)
2)   for each entry  $e_k$  in  $node$ 
3)     if ( $e_k.dist\_mini < ivalue$ )
4)       if ( $dist(e_k, q) < r$ )
5)          $RS = RS \cup e_k$ 
6)       end if
```

```

7)   end if
8)   end for
9)   else return RS
10)  end if
11)  searchOutward (node, rightnode, ivalue)

```

### 3.4 KNN 查询

对于 KNN 查询,最关键的就是求出  $R_{knn}$ ,即在什么范围内可以找出离目标对象  $q$  最近的  $k$  个对象。对于集中式 KNN 查询,可以通过每次将查询范围增加一个很小的值,直至找到结果集为止。但对于 P2P,这种方式将导致极大的通信开销。本分组网络结构下的 KNN 查询算法如下所示。

步骤 1 提出查询的节点将查询转发给管辖范围包含目标对象  $q$  的超级节点。

步骤 2 该超级节点检查本组内所有节点管理的数据对象个数,如果大于等于  $k$ ,则对本组内所有节点进行广播,以计算本组所有数据对象和目标对象之间的距离值,并将这些距离值返回给超级节点。超级节点对这些值进行排序,将第  $k$  个距离值作为  $R_{knn}$ ,并利用 3.3 节所描述的范围搜索算法  $Search(q, R_{knn})$  进行查询。

步骤 3 如果本组内所有节点管理的数据对象个数小于  $k$ ,则以目标对象距离分组管辖范围距离最小为原则,依次向不同分组发送查询消息,直到分组集合所包含的数据对象个数大于等于  $k$ 。然后利用步骤 2 所描述的方法求出最终的结果集。

## 4 仿真与分析

首先随机生成 1000 个非超级节点和若干个超级节点。测试数据集为 10000 个均匀分布的随机点数据和均匀分布的随机点数据。测试分两个部分:1) 分别在两种数据集下对查询开销进行测试;2) 针对分组数量对查询开销进行测试。其中测试 1) 在 64 个超级节点(即 64 个分组)的情况下对  $k$  为 25, 50, 75, 100, 125, 150 进行测试。测试 2) 则分别在 49, 64, 91, 100, 121 个超级节点的下对  $k = 50$  进行测试。

从图 3 可以看出,在非均匀分布的簇数据集下,查询开销要小一些。这是因为簇数据可以使得节点簇范围更小,因此与均匀分布的数据相比更能够排除掉对结果集合没有贡献的节点。从图 4 可以看出随着分组数量的增加,查询开销在减少,但减少到一定的时候,查询开销有增加的趋势。这是因为在一个分组内能够确定  $R_{knn}$  的前提下,查询开销很大一部分来自于分组内的广播,因此分组内的节点数量越少,查询开销越小;但随着分组的进一步增加,单个分组不能确定  $R_{knn}$ ,需要多个分组来共同决定  $R_{knn}$  的时候,消息开销将有增大的趋势。

## 5 结语

P2P 技术近年来在数据共享和交换方面显现出了其广阔的应用前景,然而在传统的结构化 P2P 系统中一个明显的缺陷就是只支持单关键字查询和精确匹配,因此 P2P 系统下的复杂查询得到了越来越多的关注。针对空间数据的 KNN 查询是 GIS 中最典型的应用之一,本文提出一种基于分组式 P2P 网络结构的 KNN 查询方法,该方法在将计算分布到不同节点的前提下,能有效减少 P2P 结构下的通信开销。最后

通过仿真验证了其有效性。在今后的工作中考虑对本文提出的系统在两个方面进行完善:

1) 考虑如何在本系统中进行非超级节点的数据负载均衡;

2) 考虑如何进行有效的近似 KNN 查询。

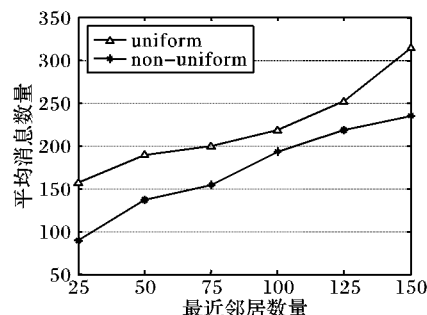


图3 两种数据集下的查询开销

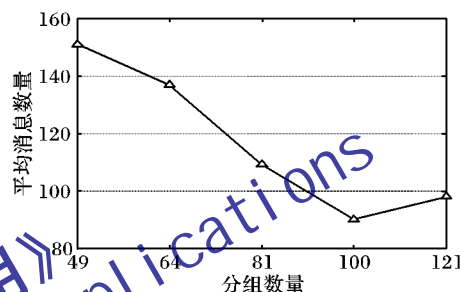


图4 不同分组下的查询开销

### 参考文献:

- [1] CIACCIA P, PATELLA M, ZEZULA P. M-tree: An efficient access method for similarity search in metric spaces [C]// Proceedings of the 23rd International Conference on Very Large Data Bases. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997: 426-435.
- [2] TRAINA C, Jr, TRAINA A, SEEGER B, et al. Slim-trees: High performance metric trees minimizing overlap between nodes [C]// EDBT 2000: International Conference on Extending Database Technology, LNCS 1777. Berlin: Springer-Verlag, 2000: 51-65.
- [3] JAGADISH H V, OOI B C, TAN K-L, et al. iDistance: An adaptive B+ tree based indexing method for nearest neighbor search [J]. ACM Transactions on Database Systems, 2005, 30(2): 364-397.
- [4] MONDAL A, YI LIFU, KITSUREGEWA M. P2PR-tree: An R-tree based spatial index for peer-to-peer environments [C]// EDBT 2004: Proceedings of the International Workshop on Current Trends in Database Technology, LNCS 3268. Berlin: Springer-Verlag, 2003: 516-525.
- [5] TANIN E, HARWOOD A, SAMET H, et al. A Serverless 3D World [C]// Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems. New York: ACM Press, 2004: 157-165.
- [6] FALCHI F, GENNARO C, ZEZULA P. A content-addressable network for similarity search in metric spaces [C]// DBISP2P05: Proceedings of Databases, Information Systems, and Peer-to-Peer Computing, LNCS 4125. Berlin: Springer-Verlag, 2005: 126-137.
- [7] NOVAK D, ZEZULA P. M-Chord: A scalable distributed similarity search structure [C]// InfoScale'06: Proceedings of the 1st International Conference on Scalable Information Systems. New York: ACM Press, 2006: 18.