一种障碍空间中的反 k 最近邻查询方法

于晓楠 谷 峪 张天成 于 戈

(东北大学信息科学与工程学院 沈阳 110819)

(医学影像计算教育部重点实验室(东北大学) 沈阳 110819)

摘 要 随着基于位置的服务(LBS)和物联网的快速发展,空间查询技术越来越重要,而空间查询中的最近邻查询及其各种变体有着广泛的应用.近几年,已有较多对于查询前 k 个反最近邻对象(RkNN)的研究,其中大部分针对的都是理想欧氏空间.而在真实的情况下,反 k 最近邻查询通常受障碍物影响.文中研究了障碍空间中反 k 最近邻查询算法,提出了一种基于障碍 Voronoi 图的高效的剪枝方法.根据 Voronoi 图和障碍距离的特性,大幅度减少了数据点处理个数.最后,作者使用真实的数据集和多种方式分布的模拟数据,验证了算法的高效性和准确性.

关键词 空间查询;反 k 最近邻(RkNN);障碍空间;Voronoi 图

中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2011.01917

A Method for Reverse k-Nearest-Neighbor Queries in Obstructed Spaces

YU Xiao-Nan GU Yu ZHANG Tian-Cheng YU Ge

(*College of Information Science and Engineering, Northeastern University, Shenyang* 110819)

(*Key Laboratory of Medical Image Computing of Ministry of Education (Northeastern University), Shenyang* 110819)

Abstract With the rapid development of location-based services (LBS) and the Internet of Things, technologies for spatial queries are becoming more and more important. Moreover, nearest neighbor queries and the variant are widely used spatial queries. Recently, there has been much work on reverse k nearest neighbors (RkNN) queries. However, these studies are proposed for the ideal Euclidean Space. Queries for reverse k nearest neighbors are influenced by obstacles in practice. In this paper, we study a method for reverse k nearest neighbors queries in obstructed spaces, and propose efficient pruning algorithms based on an obstructed Voronoi diagram. Furthermore, these pruning methods greatly reduce the number of searched points by properties of Voronoi diagrams and obstructed distance. Finally, our experiments based on real and synthetic data sets demonstrate the efficiency and accuracy of our proposed approach.

Keywords spatial query; Reverse kNN; obstructed space; Voronoi diagram

1 引 言

近年来,物联网和基于位置的服务[1]获得了越来越多的关注.空间查询作为重要的支撑技术,涉及多种查询类型及处理方法,如范围查询、k 近邻查询[2-4]、skyline 查询[5-6]和反 k 近邻查询[7-8]等.特别的,在这些查询中,反 k 最近邻查询用来获取那些 k 最近邻里面包含查询点 q 的空间数据点,在空间决策支持、资源分配和数据挖掘方面有着广

收稿日期:2011-07-07;最终修改稿收到日期:2011-08-17.本课题得到国家自然科学基金(61003058,60873009)、辽宁省博士启动基金(20091025)、中央高校基本科研业务费专项资金(090404013)资助.于晓楠,女,1987 年生,硕士,主要研究方向为 RFID、时空数据库等.E-mail: yuxiaonananshan@163.com. 谷 峪(通信作者),男,1981 年生,博士,副教授,主要研究方向为 RFID、空间数据管理、云计算.E-mail: guyu@ise.neu.edu.cn. 张天成,男,1965 年生,博士,副教授,主要研究方向为物联网. 于 戈,男,1962 年生,博士,教授,博士生导师,中国计算机学会(CCF)高级会员,主要研究领域为数据库理论和技术、分布与并行系统等.

泛的应用①. k 最近邻查询反映了查询点受到哪些空间数据点影响较大,而反 k 最近邻查询则反映了查询点对哪些空间数据点影响较大.此外,现有的空间查询工作大多考虑理想的欧式空间和路网空间.然而,地面或室内移动的物体一般都会受到地理条件的限制(例如建筑、湖泊等),之间的最短距离必须考虑障碍物的因素.例如,在灾区医疗和餐饮安置点通过对其它居住安置点的反 k 近邻查询,来获取合理的资源配置,由于其它废弃形成的障碍,需要考虑障碍空间内的反 k 近邻查询方法.

有关无障碍空间和路网上的 kNN 和 RkNN 查询以及障碍空间的 kNN 查询,都已经有较多的研究工作.

对于无障碍空间的 kNN 查询,近年来广泛使用的是基于 Voronoi 图处理的技术. Sharifzadeh 等人[9]给出了结合 Voronoi 图和 R-tree 的索引结构的方法,能够进行高效的 kNN 查询.无障碍空间中的 RkNN 查询是获取那些 k 近邻里面有查询点 q 的空间数据点.对于无障碍空间 RkNN 查询,文献[7-8,10]提出了无需预计算的 RkNN 查询方法. Stanoi 等人[7]给出了在查询点 q 处将空间划分成六等份(每份为 60°)的方法,可以证明在每一个区域离查询点 q 的第 k 个最近临的点以外的区域都可以被剪枝,得到这些未被剪枝的区域里的数据点就成为 RkNN 查询的候选点. Tao 等人[8]提出了使用中垂线性质剪枝搜索空间的方法,有效的剪枝那些处在 k 个以上中垂线的半区间的数据点. Wu 等人[10]提出一种新的剪枝方法——FINCH 算法,该方法并未使用文献[8]提出的中垂线,而是使用一个多边形近似未被剪枝区域.文献[11]提出了使用 Voronoi 图的方法解决无障碍的 RkNN 查询方法.路网上的 kNN 查询也有相关研究,文献[12]使用基于 Voronoi 图的方法查询路网上的 k 最近邻.尽管这些方法能高效地执行 kNN 和 RkNN 查询,但它们都是在理想的欧氏空间即无障碍物空间中查询,而在现实场景中这种假设是不合理的.

对于障碍空间中的查询,目前主要集中在 kNN 查询的研究.障碍空间中,kNN 查询是获取在障碍距离上 k 个距离查询点 q 最近的空间数据点. Zhang 等人[13]给出了在障碍空间中基于 R-tree 方法解决常见的空间查询,例如,范围查询、最近邻查询、e-距离连接查询、最近对查询. Xia 等人[14]提出了障碍最近邻查询(ONN 查询)方法,用增量的方式处理只与查询有关的数据点和障碍物,因此文献[14]过滤掉了大量数据点和障碍物. Gao 等人[15]通过有效地分割对象行进的路径,提出了障碍空间内连续 kNN 查询的优化方法.但是在障碍空间中,现有的空间查询仅仅局限于上述以 kNN 查询为主的几种查询上,并没有涉及 RkNN 查询.因此在障碍空间中,如何查询反 kNN 对象是如今亟待解决的问题.

本文给出一种基于障碍 Voronoi 图的高效搜索反 kNN 对象的方法.通过剪枝的方法,减少查询处理反 kNN 的代价.我们结合了划分空间六等份、障碍 Voronoi 图的性质等方法和 R-tree 索引结构,由 Voronoi 邻居增量地得到反 kNN 查询结果.

本文的主要贡献如下:

(1)我们形式化定义了障碍反 kNN 查询,在障碍空间中添加了一种新的查询.

(2)我们将空间划分成六等份的方法用于障碍空间,并将此方法加以改进.

(3)利用障碍 Voronoi 图的性质,我们设计了几种剪枝策略,基于过滤和求精的思想,大幅度地减少搜索障碍反 kNN 对象的时间和空间.

(4)我们通过实验验证了本文提出的算法在时间和空间上的优越性.

本文第 2 节形式化定义障碍空间反 kNN 查询,并给出相关定义;第 3 节详细描述提出的障碍空间中反 kNN 查询的高效处理方法,提出几种剪枝和求精的方法;第 4 节给出实验结果和数据分析;最后,第 5 节总结.

2 问题定义

在给定的空间中存在着数据点和障碍物,设数据点集合为 $P=\{p_1,p_2,\cdots,p_n\}$,障碍物集合为 $O=\{O_1,O_2,\cdots,O_m\}$ 以及一个查询点 q.为了不失一般性,本文假设障碍物为线段形状,且障碍物不相交于数据点,即 $O=\{O_1,O_2,\cdots,O_m \mid O_i=O_{i_1}O_{i_2},O_{i_1}O_{i_2}\cap p_j\neq\varnothing,\ i\in[1,m],j\in[1,n]\}$.

查询点和数据点之间可能存在着障碍物,因此数据点移动到障碍物时,不能穿过障碍物,需要绕着障碍物移动,如图 1 所示.数据点 p_1 和查询点 q 之间无障碍物,但是数据点 p_2 和 q 之间有障碍物 O_1,所以 p_2 移动到 q 需要绕过障碍物 O_1,可以经由 $p_2\rightarrow O_{1_1}\rightarrow q$ 路径到达 q 或者经 $p_2\rightarrow O_{1_2}\rightarrow q$ 到达 q.由此可知在障碍空间中的两点之间的距离有可能需要绕

① Lv Ying. The key technologies research on keyword search in the mobile environment(吕瑛:移动环境中关键词搜索的关键技术研究. http://www.jiahenglu.net/NSFC/spatial%20queries.html)

过障碍物计算，下面给出障碍距离的定义.

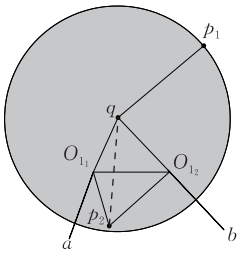

图 1　障碍距离

定义 1.　障碍距离. 在有障碍物的空间中，欧氏距离记为 d_{euc}，若两点 p 和 q 之间无障碍物，则障碍距离即两点间的欧氏距离，即 $d_{obs}(p,q)=d_{euc}(p,q)$；若 p 和 q 之间有障碍物 O_k，则障碍距离为这两点绕过障碍物的最短距离，即

$$d_{obs}(p,q)=\min\{d_{euc}(q,O_{k_1})+d_{euc}(O_{k_1},p_2),\ d_{euc}(q,O_{k_2})+d_{euc}(O_{k_2},p_2)\}.$$

由图 1 可知，$|O_{1_1}p_2|=|O_{1_1}a|$，$|O_{1_2}p_2|=|O_{1_2}b|$，因此点 p_2 绕过障碍物 O_1 到达点 q 的路径的长度为 $|qa|$ 和 $|qb|$，又 $|qa|<|qb|$，因此点 p_2 到查询点 q 的障碍距离为 $|qa|$.

根据数据点之间的障碍距离的定义，下面给出障碍 Voronoi 图的定义. 为了帮助理解障碍 Voronoi 图，我们先介绍障碍中垂线.

定义 2.　障碍中垂线. 在障碍空间中，两点 p 和 q 的障碍中垂线上的点到 p 和 q 的障碍距离相等，记为 $l_{bis_{obs}}(p,q)=\{p_x\,|\,d_{obs}(p,p_x)=d_{obs}(q,p_x)\}$.

引理 1.　障碍中垂线是由直线组成或由射线和曲线段共同组成.

证明.　当两点之间无障碍物时，障碍中垂线即为欧氏空间的中垂线；当两点之间存在障碍物时，在被障碍物遮挡的区域里的中垂线是双曲线的一部分.　　证毕.

如图 2 所示，点 q 和 p_2 的障碍中垂线是这样得到的：假设点 q 和 p_2 无障碍，得到中垂线 $l_{n_1n_5}$；将 q 和障碍物的边界连线与直线 $l_{n_1n_5}$ 相交于点 n_2 和 n_4，得到在 $l_{n_1n_5}$ 一侧查询点 q 不能移动到的区域 $O_{1_1}n_2n_4O_{1_2}$；在此区域外，射线 $l_{n_1n_2}$ 和 $l_{n_4n_5}$ 即为障碍中垂线的一段；在此区域内，取任意一点 x，使得 x 绕过障碍物 $O_{1_1}O_{1_2}$ 到 q 的欧氏距离等于 x 到 p_2 的欧氏距离，即 $|qO_{1_2}|+|O_{1_2}x|=|xp_2|$ 以及 $|qO_{1_1}|+|O_{1_1}x|=|xp_2|$，所以点 x 在以 O_{1_2}、p_2 为焦点、$|qO_{1_2}|$ 为焦距的双曲线的一支（弧线 $\widehat{n_3n_4}$）上以及以 O_{1_1}、p_2 为焦点、$|qO_{1_1}|$ 为焦距的双曲线的一支（弧线 $\widehat{n_2n_3}$）上. 因此，点 q 和 p_2 的障碍中垂线为由射线 n_2n_1、弧线 $\widehat{n_2n_3}$、弧线 $\widehat{n_3n_4}$ 和射线 n_4n_5 组成.

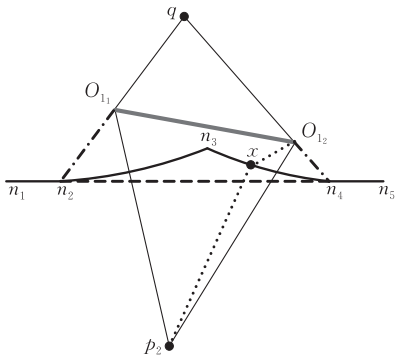

图 2　障碍中垂线

n 个数据点的无障碍物的空间中，Voronoi 图是由 n 个以数据点为中心的区域组成的，即 $V(P)=\{V(p_1),V(p_2),\cdots,V(p_n)\}$，其中每个区域的边界 $b(p_i)$ 是所有到中心点距离最小的点的集合，即 $V(p_i)=\{(p_i,b(p_i))\,|\,\forall p\in b(p_i),d_{euc}(p,p_i)\le d_{euc}(p,p_j),j\ne i,j\in[1,n]\}$. 结合前文的障碍距离，下面给出在障碍空间中的障碍 Voronoi 图的定义.

定义 3.　障碍 Voronoi 图. 障碍 Voronoi 图由 n 个以数据点为中心的障碍区域组成，即 $V_{obs}(P)=\{V_{obs}(p_1),V_{obs}(p_2),\cdots,V_{obs}(p_n)\}$. 其中每个障碍区域的边界是所有到中心点的障碍距离最小的点的集合，即 $V_{obs}(p_i)=\{(p_i,b_{obs}(p_i))\,|\,\forall p\in b_{obs}(p_i),d_{obs}(p,p_i)\le d_{obs}(p,p_j),j\ne i,j\in[1,n]\}$.

定理 1.　障碍 Voronoi 图的各个障碍区域 $V(p_i)$ 的边界 $b_{obs}(p_i)$ 在数据点 p_i 与其它各个数据点 p_j 的障碍中垂线上，即 $\forall p\in b_{obs}(p_i)$，$p\in l_{bis_{obs}}(p_i,p_j)$，$i,j\in[1,n]$，$j\ne i$.

证明.　由定义 3 得到，障碍区域的边界上的点到中心点的障碍距离比到其它数据点的障碍距离小，又由定义 2 可知，到中心点的障碍距离和到其它数据点的障碍距离相等的点落在障碍中垂线上，因此，障碍区域的边界也落在相应的障碍中垂线上.　　证毕.

我们假设障碍空间的数据点集为 $P=\{p_1,p_2,\cdots,p_{31}\}$，障碍物集为 $O=\{O_{1_1}O_{1_2},O_{2_1}O_{2_2},O_{3_1}O_{3_2}\}$，由定义 3 可以画出障碍 Voronoi 图如图 3 所示，例如障碍区域 $V(p_3)$ 的边界 $b_{obs}(p_3)$ 就是数据点 p_3 与 p_1，p_2，p_4，p_5，p_{31}，p_{29} 的障碍中垂线组成的.

首先定义障碍空间中的 kNN 查询如下.

定义 4.　障碍 kNN 查询. 在障碍空间中，k 最近邻查询返回 k 个距离查询点 q 最近的数据点，即 $k\mathrm{NN}_{obs}(q,k,P,O)=\{p\in P\,|\,x\in P\,|\,d_{obs}(p,q)\le d_{obs}(p,x)|<k\}$.

如果多个数据点有相同的距查询点 q 第 k 个远的距离，那么这些点也都包含在 kNN 查询结果集中.

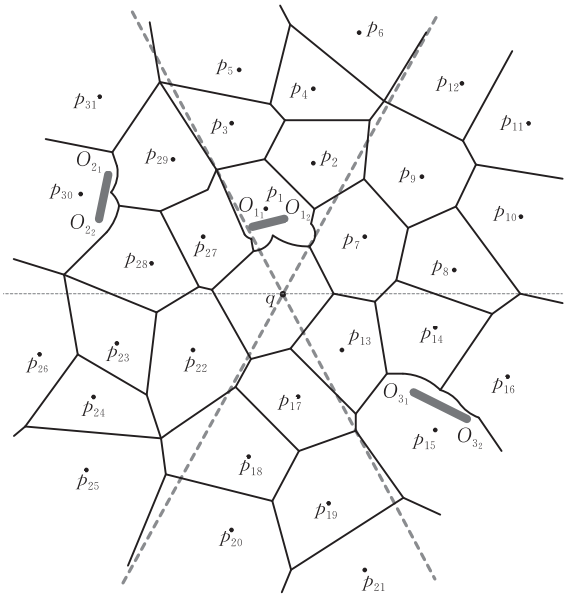

图 3 障碍 Voronoi 图

例如,图 3 中查询点 q 的障碍2NN为 $2NN_{obs}(q,2,P,O)=\{p_{13},p_7\}$.

定义 5. 障碍 RkNN 查询. 在障碍空间中,在数据集 P 中获取那些 k 个最近邻中包含查询点 q 的空间数据点,即 $RkNN_{obs}(q,k,P,O)=\{p\in P \mid q\in kNN_{obs}(p,k,P,O)\}$.

例如,图 3 中查询点 q 的障碍 R2NN 为 $R2NN(q)_{obs}(q,2,P,O)=\{p_{13},p_{17},p_{22}\}$.

3 障碍 RkNN 查询

在本节中,我们给出在障碍空间中对于一个查询点 q 的 RkNN 查询算法.

3.1 约减数据集

下面给出约减候选集的剪枝算法.

引理 2. 只有到查询点 q 经过的最少数据点个数小于或等于 k 的那些对象才能放入候选集.

证明. 假设 $numOfObj(p,q)$为对象 p 到查询点 q 需要经过最少的数据点的个数. 当 $k=1$ 时,即查询 q 的 RNN,而那些能成为 q 的 RNN 的数据点 p 一定是 q 的 Voronoi 邻居的,即点 p 满足 $numOfObj(p,q)\leq 1$;当$k\geq 1$ 时,假设$numOfObj(p_i,q)\leq i$, $0\leq i\leq k$,那么 $numOfObj(p_{k+1},q)\leq k+1$. 又由 Voronoi 图的定义可以得到, p_{k+1} 是 $p_i\in\{p_1,\cdots,p_k\}$的一个 Voronoi 邻居. 因此,可以得到 $numOfObj(p_{k+1},q)\leq \max(numOfObj(p,p_i))+1\leq k+1$,引理 2 得证. 证毕.

如图 4 所示,空间数据点集为$\{p_1,p_2,\cdots,p_6\}$,障碍物集为$\{O_{1_1}O_{1_2}\}$,查询点为 q,数据点 p_5不能放入候选集,因为 p_5需要最少经过 3 个数据点到 q,即 $p_5\rightarrow p_3\rightarrow p_1\rightarrow q$;同理,数据点 p_4,p_6也都不能放入候选集中;因此,此区域的候选集为阴影区域 $S_{cnd}(0)=\{p_3,p_2,p_1\}$.

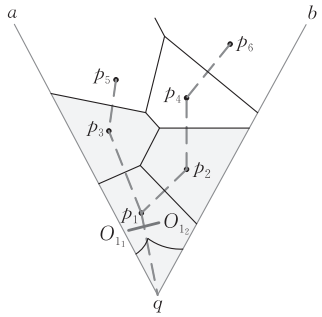

图 4 检查经过的对象个数方法($k=2$)

$GD(P,q,k)$方法:由引理 2,取到查询点 q 的点满足经过不大于 k 个的数据点. 如算法 1 所描述,首先构造一个最小堆,存放将要计算的数据点及其到查询点的距离. 由 $VN(q)$求出查询点 q 的邻居集合,计算这些邻居点到 q 点的障碍距离,再将这些邻居点放入最小堆. 对于每次从堆中取出的元素 e,若它到 q 点的经过的数据点个数不大于 k,就将其加入候选集 S_{cnd}中;再求出元素 e 的邻居集合,若这些邻居没被访问过的话,将其 $numOfObject$ 加 1,放入最小堆和 $Visited$ 集合中.

算法 1. $GD(P,q,k)$.

输入:查询点 q,查询对象集 P, RkNN 查询的 k 值

输出:约减的对象候选集 S_{cnd}

```
1.  Minheap H={}; Visited={};
2.  For each point p in VN(q) Do
3.     Add (p,1) into H; add p into Visited;
4.  While H≠∅ Do
5.     Deheap an entry e;
6.     If numOfObj(e,q)≤k then
7.        Add (e,d_obs(e,q)) into S_cnd;
8.        For each point p' in VN(e) Do
9.           If p'∉Visited then
10.             numOfObj(p',q)=numOfObj(e,q)+1;
11.             add (p',numOfObj(p',q)) into H;
12.             add p' into Visted;
13. Return S_cnd;
```

引理 3. 对象 p 与查询点 q 之间经过若干对象如 p_i,连接 p 与 p_i,若无障碍物,则计数 $countNonObs(p,q)$增加 1. 如果$countNonObs(p,q)\geq k$,则对象 p 被剪枝.

证明. 对象 p 与 p_i之间若无障碍物,说明点 p 到 p_i一定比到查询点 q 的欧氏距离近,若 p 到 q 之

间存在障碍物，那么 p 到 p_i 的障碍距离也一定比到 q 的近. 当 $k=1$ 时，这样的 p_i 就有可能是 p 的 NN，而 q 一定不能是 p 的 NN；若 $k\neq 1$ 时，只要这样的 p_i 有不小于 k 个，那么 p 的 kNN 一定不能包含查询点 q，即 $countNonObs(p,q)\geq k$ 时，p 一定不是候选集里的对象.　　证毕.

如图 5 所示，由于数据点 p_3 与查询点 q 之间经过 p_2 和 p_1 对象，而且 p_3p_1 无障碍物，p_3p_2 也无障碍物，所以 p_3 的 2NN 一定不包含点 q，因此点 p_3 被移除候选集，候选集约减为阴影区域 $S_{cnd}(0)=\{p_1,p_2\}$.

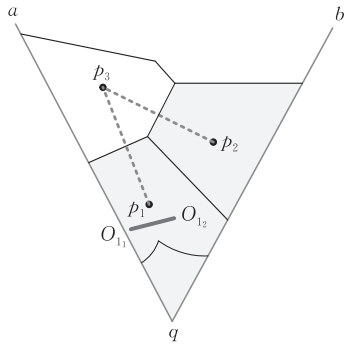

图 5　检查无障碍物个数方法($k=2$)

$CheckNumOfObs(S_{cnd}(i),q,O,k)$ 方法：由引理 3 查看候选集里的每一个对象 p 与它到的查询点 q，若 p 与它经过的对象之间有 k 个以上无障碍物的对象，则 p 被剪枝. 如算法 2 所示，对于候选集中的每个对象 p，以 p 为圆心，$|pq|$ 为半径做圆 $R(p)$，若在此候选集中的对象 e 也在这个圆里，就考查 p 与 e 之间是否存在障碍物，若对于一个 p，有多于 k 个无障碍物，则 p 就从候选集中去除.

算法 2.　$CheckNumOfObs(S_{cnd}(i),q,O,k)$.

输入：候选集 $S_{cnd}(i)$，查询点 q，障碍物集 O，RkNN 查询的 k 值

输出：约减的对象候选集 $S_{cnd}(i)$

```
1.  count=0;
2.  For each point p in S_cnd(i) Do
3.    For each point e in R(p)∩S_cnd(i) Do
4.      If NonObs(p,e)==true then
5.        count++;
6.    If count≥k then
7.      S_cnd(i)=S_cnd(i)-p;
8.  Return S_cnd(i);
```

引理 4.　若候选集中的点 p 的 Voronoi 邻居都被剪枝了，则此点 p 也被剪枝.

证明.　假设在候选集中取一点为 p，且点 p 的 Voronoi 邻居(如 v)都被剪枝. 由 Voronoi 图的性质知，p 到 q 一定经过 p 的 Voronoi 邻居(如 v)，而 $countNonObs(v,q)\geq k$，从而得到 $countNonObs(p,q)\geq k+1>k$，由引理 3 知，点 p 被剪枝.

$PruneByNeighbor(S_{cnd}(i),q)$ 方法. 由引理 4 对候选集对象进一步剪枝，如算法 3 描述，设置布尔变量 $nonPruneFlag$ 为 false 来标记此对象是需要被剪枝的. 对于候选集中每个对象(如 p)的每个邻居，若都被剪枝，则此对象 p 也被剪枝.

算法 3.　$PruneByNeighbor(S_{cnd}(i),q)$.

输入：候选集 $S_{cnd}(i)$，查询点 q

输出：约减的对象候选集 $S_{cnd}(i)$

```
1.  nonPruneFlag==false;
2.  For each point p in S_cnd(i) Do
3.    For each point e∈VN(p) Do
4.      If e∈S_cnd(i) then
5.        nonPruneFlag==true;
6.    If nonPruneFlag==false then
7.      S_cnd(i)=S_cnd(i)-p;
8.  Return S_cnd(i);
```

3.2　障碍 RkNN 查询

下面介绍在障碍空间中，给出查询点 q，数据点集 P，障碍物集为 O 的 RkNN 查询.

算法 4 描述了障碍空间的 RkNN 查询的过程.

算法 4.　$RkNN_{obs}(q,k,P,O)$.

输入：q：查询点

　　k：RkNN 查询的 k 值

　　P：对象点集合

　　O：障碍物集合

输出：RkNN 查询的结果 S_{cnd}

```
1.  For 1≤i≤6 Do S_cnd(i)=∅; S_cnd=∅;
2.  SixRegionPartition(P,O);
3.  DrawVG(P,O);
4.  For each S_cnd(i) Do
5.    S_cnd(i)=GD(P,q,k);
6.    S_cnd(i)=CheckNumOfObs(S_cnd(i),q,O,k);
7.    PruneByNeighbor(S_cnd(i),q);
8.    For each point p in S_cnd(i) Do
9.      p_k=k-th NN of p;
10.     If d_obs(p,q)>d_obs(p,p_k) Then
11.       S_cnd(i)=S_cnd-p;
12.   S_cnd=S_cnd∪S_cnd(i);
13. Return S_cnd;
```

首先将空间划分成 6 等份区域，再由空间的数据点和障碍物画障碍 Voronoi 图；对于 6 个等份的每个区域中，进行数据约减：只取那些到查询点 q 经过数据点个数不大于 k 的数据点放入候选集(算法 1 的 $GD(P,q,k)$)，再查看候选集中的点 p 与它到 q 之前的所有点之间，若有不少于 k 个点无障碍物，则这样的点 p 就从候选集中去除(算法 2

的 $CheckNumOfObs(S_{cnd}(i),q,O,k))$，若候选集中存在点 p，它的所有 Voronoi 邻居都被移除候选集，则此点 p 也被移除候选集(算法 3 的 $PruneByNeighbor(S_{cnd}(i),q))$；最后，对候选集中的每个对象 p，计算 p 的第 k 个近邻点 p_k，若查询点 q 到 p 的障碍距离 $d_{obs}(p,q)$ 大于 p_k 到 q 的障碍距离 $d_{obs}(p,p_k)$，则 q 的 RkNNs 里没有点 p，因此从候选集中移除点 p. 最后，把 6 个区域的候选集合并起来，即为所求结果. 设定空间中数据点的个数为 n，障碍物的个数为 m，则预处理(即构建障碍 Voronoi 图)的时间为 $O(n\log(n+m))$，设剪枝后剩余数据点的个数为 $n'(n'\ll n)$，则查询处理时间为 $n'\log n$.

如图 6 所示，障碍空间中，对象集为 $P=\{p_1,p_2,\cdots,p_{31}\}$，障碍物集为 $O=\{O_{1_1}O_{1_2},O_{2_1}O_{2_2},O_{3_1}O_{3_2}\}$，查询点为 q，由障碍 Voronoi 图的定义，画出如图 6 所示的空间障碍 Voronoi 图.

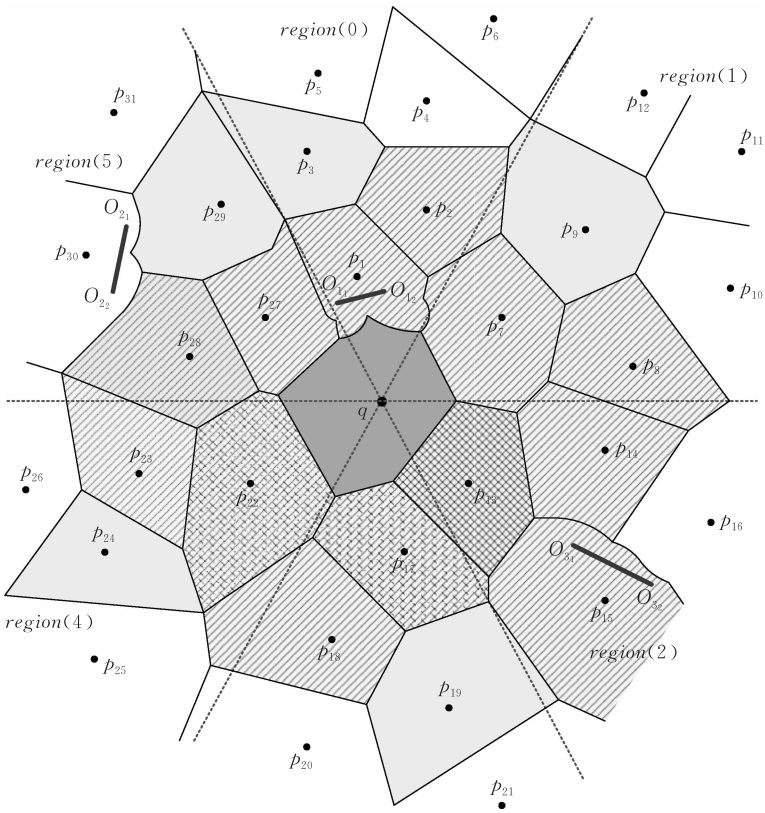

图 6 障碍 RkNN 查询($k=2$)

将空间分成 6 等份；在每一等份中分别对候选集进行约减，首先由算法 1 选择那些到查询点 q 经过的数据点个数不大于 k 的对象点，如 $region(0)$ 里的 p_3，p_4，p_5，它们到 q 需要至少经过 3 个数据点，因此被剪枝掉，得到图 6 中非空白区域中的对象构成候选集；接着由算法 2 去除那些到此对象之前的数据点有不少于 k 个无障碍物的对象，例如 $region(5)$ 中的点 p_{29} 到 p_{28} 和 p_{27} 两个数据点都无障碍物，因此点 p_{29} 被剪枝掉，得到图 6 中单斜线阴影区域中的对象构成的候选集；再由算法 3 剪枝掉那些它们的所有 Voronoi 邻居都被剪枝掉的数据点；最后，对于候选集中的每一个对象，分别计算它们的 R2NNs，若 q 被包含在这些 R2NNs 里，则此对象点放入结果集中，可以得到如图 6 所示的双斜线阴影区域的结果集 $R2NN(q)_{obs}(q,2,P,O)=\{p_{13},p_{17},p_{22}\}$.

4 实验结果与分析

在这一节，我们用实验评估本文所提出的算法 ORkNN 的处理时间和 I/O 代价，并与用基本方法(Basic 方法)查询障碍空间 RkNN 对象(即求出每个点的 kNN，再找出这些 kNNs 里面包含有查询点 q 的数据点)进行比较. 我们的实验使用 C++语言实现，运行环境为 Inter Core2 E8400 3.00 GHz CPU、3.00 GB 内存和运行 32 位 Windows 7 操作系统.

本文使用模拟数据集和真实数据集验证所提出的算法. 模拟数据集的大小和真实数据集的大小相同，数据点和障碍物的分布表示为(Distribute_Point，Distribution_Obstacle)，其中 Distribute _ Point、Distribution_Obstacle 取值为均匀分布(Uniform)、

真实分布(Real)、正态分布(Normal). 真实数据集为包含 76999MBRs 的地势数据 hypsogr(如图 7 所示)和包含 36334 MBRs 的铁路线数据 rrlines[①](如图 8 所示). 为不失一般性,假设障碍物为线段. 我们使用两个 R-tree 分别索引数据点(hypsogr 数据)和障碍物(rrlines 数据). R-tree 节点大小缺省值为 4 KB,k 值缺省为 10,查询点 q 随机从数据集中取出,障碍 RkNN 查询执行 100 次取平均值,实验结果显示出我们提出的算法处理的高效性.

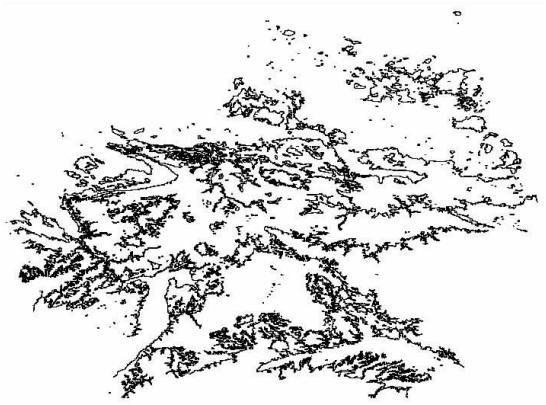

图 7　Hypsogr

图 8　Rrlines

k 值的影响. 图 9 和图 10 分别给出了 k 值的变化对 CPU 时间和磁盘 I/O 代价的影响.

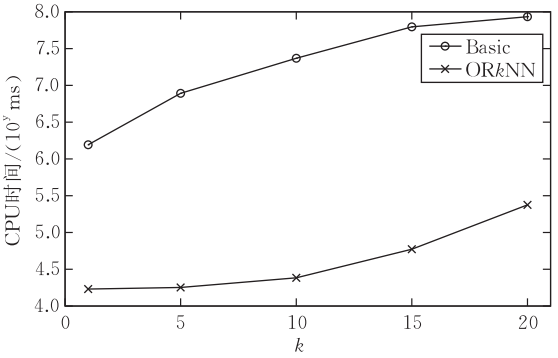

图 9　k 值对 CPU 时间的影响

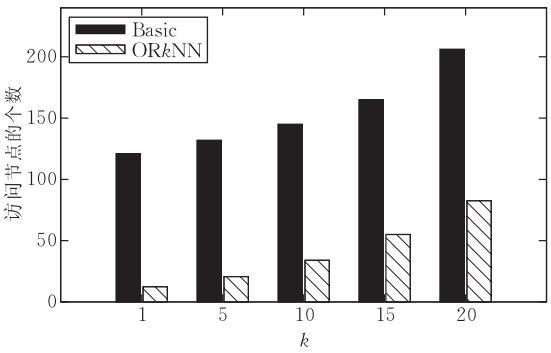

图 10　k 值对磁盘 I/O 代价的影响

由图 9 可以看出,k 值从 1 变化到 20 的过程中,CPU 时间随着 k 值的增加而增加. ORkNN 方法的 CPU 时间增长率快于传统方法,是因为前者剪枝过程中时间随 k 增长得比传统方法增长得要快,而且由于两种方法的 CPU 时间差距较大,因此用时间的对数表示. 实验结果表明本文提出的 ORkNN 方法明显好于传统 Basic 方法.

图 10 给出了随着 k 值的增加对磁盘 I/O 代价的影响. 两种方法的磁盘 I/O 代价都随着 k 值的增加而增加. 这是由于 k 值越大,需要访问的数据点越多,磁盘 I/O 越大,而且由于我们的 ORkNN 方法仅需要访问离查询点较近的数据点,因此磁盘 I/O 代价要小很多. 可以看出,ORkNN 方法要优于传统方法.

数据点个数的影响. 为了研究数据点的密度对 CPU 执行性能和获得数据点个数的影响,我们给出了如图 11 和图 12 所示的实验结果.

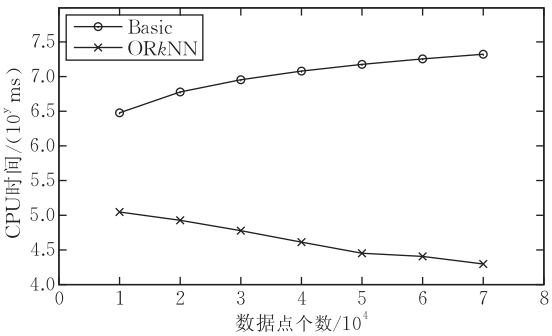

图 11　数据点的个数对 CPU 时间的影响

由图 11 可以得到,随着数据点数量的增加,我们的 ORkNN 方法的 CPU 用时下降,而传统的方法的 CPU 用时增加,而且 ORkNN 方法比传统方法的耗时少. 这是因为对于 ORkNN 方法,数据点越密集,查询点和它的障碍 Voronoi 邻居之间越近,计算

① R-tree Portal-Spatial (geographical) Datasets in 2D space. http://www.rtreeportal.org/index.php?option=com_content&task=view&id=29&Itemid=42

障碍距离的时候大多数只需要计算欧氏距离.而传统Basic方法需要计算每一个数据点的障碍反kNN,因此对于Basic方法,数据点越多,CPU用时越多.

图12反映了随着数据点的数量的增加,传统Basic方法的磁盘I/O代价随之增加,而我们的ORkNN方法的I/O代价下降.这是由于Basic方法需要计算的每一个数据点,因而数据点越多,计算的数据点越多.对于ORkNN方法,数据点越密集,需要访问的障碍物越少,因而在每个区域里剪枝掉的数据点越多,所以ORkNN方法的磁盘I/O代价随着k值的增加而减少.

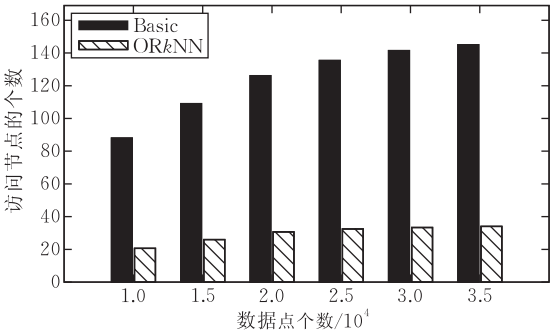

图12 数据点的个数对磁盘I/O代价的影响

障碍物个数的影响.我们通过图13和图14研究障碍物密度的变化对执行效率的影响.

由图13可知,障碍物密度越大,CPU执行时间越多,而且我们的ORkNN方法明显好于传统方法.这是由于障碍物个数越多,需要计算的障碍距离越多,CPU时间越大.而计算障碍距离要比计算欧氏距离用时更多,传统Basic方法需要计算每一个数据点的障碍距离,比ORkNN方法计算的障碍距离要多,因此它的CPU耗时更多.

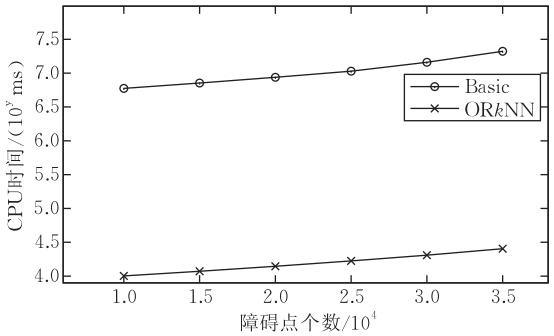

图13 障碍物的个数对CPU时间的影响

图14给出了不同数量的障碍物对磁盘I/O代价的影响.随着障碍物个数的增加,两种方法的I/O代价都随之增加,但是ORkNN方法由于有效的剪枝策略极大地减少了障碍物的访问数量,因此磁盘I/O代价明显比传统Basic方法的代价少很多.

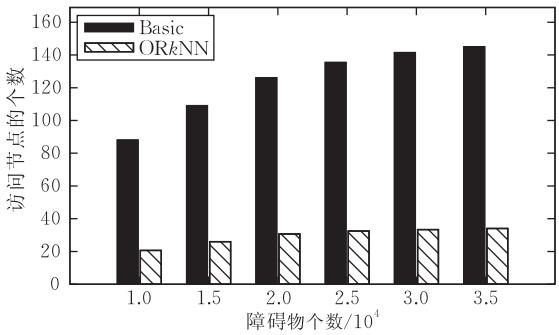

图14 障碍物的个数对磁盘I/O代价的影响

数据分布的影响.图15和图16分别给出了两种算法不同数据分布影响的实验结果.横坐标1,2,…,6分别代表数据点和障碍物分布(数据点分布,障碍物分布)的不同组合,分别对应为(U,R),(R,R),(N,R),(U,N),(R,N)和(N,N).例如,(U,N)代表数据点服从均匀分布,障碍物服从正态分布.

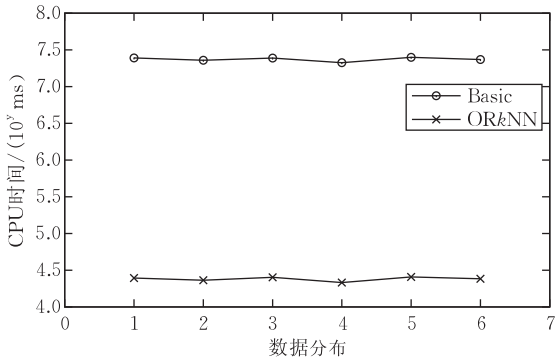

图15 不同数据分布对CPU时间的影响

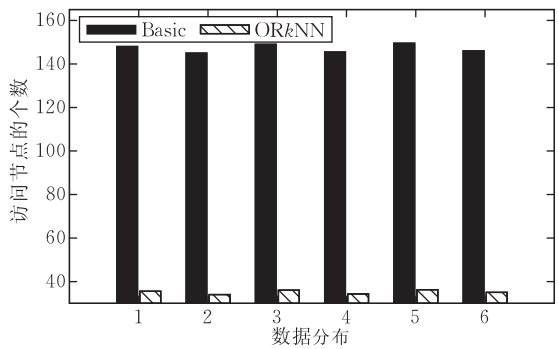

图16 不同数据分布对磁盘I/O代价的影响

由图15给出了数据点和障碍物服从于不同的数据分布对CPU执行时间的影响情况.可以看出,数据分布的变化对CPU执行时间的影响不大,但是我们的ORkNN方法由于访问数据比Basic方法少,因此在CPU时间上优于传统方法.

图16给出了不同的数据分布组合对数据点和障碍物在磁盘I/O代价上的影响.可以发现,数据分布的不同对磁盘I/O代价影响不大.ORkNN方法由于具有高效的剪枝方法,减少了大量访问数据

点和障碍物的个数，降低了磁盘 I/O 代价，因此 ORkNN 方法优于传统 Basic 方法.

5 结　论

本文研究障碍空间中反 k 最近邻查询算法. 我们形式化定义了障碍反 kNN 查询，提出了一种基于障碍 Voronoi 图的高效的剪枝方法. 将空间划分成 6 等份的方法用于障碍空间，并将此方法加以改进. 根据 Voronoi 图和障碍距离的特性，大幅度减少了处理数据点和障碍物的数量. 使用真实的数据集和多种方式分布的模拟数据的实验结果，表明了所提出算法的高效性和准确性.

参　考　文　献

[1] Zhang J, Zhu M, Papadias D, Tao Y. Location-based spatial queries//Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data. New York, USA, 2003: 443-454

[2] Roussopoulos N, Kelley S, Vincent F. Nearest neighbor queries//Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data. New York, USA, 1995: 71-79

[3] Berchtold S, Ertl B, Keim D A, Kriegel H P, Seidl T. Fast nearest neighbor search in high-dimensional space//Proceedings of the ICDE. Orlando, FL, USA, 1998: 209-218-

[4] Gao Y, Zheng B, Chen G et al. Continuous visible nearest neighbor query processing in spatial databases. VLDB Journal, 2010, 20(3): 371-396

[5] Sharifzadeh M, Shahabi C. The spatial skyline queries//Proceedings of the VLDB. Seoul, Korea, 2006: 751-762

[6] Lee K C, Lee W C, Zheng B et al. Z-SKY: An efficient skyline query processing framework based on Z-order. VLDB Journal, 2010, 19(3): 333-362

[7] Stanoi I, Agrawal D, Abbadi A E. Reverse nearest neighbor queries for dynamic databases//Proceedings of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. 2000: 44-53

[8] Tao Y, Papadias D, Lian X. Reverse kNN search in arbitrary dimensionality//Proceedings of the VLDB. Toronto, Canada, 2004: 744-755

[9] Sharifzadeh M, Shahabi C. VoR-Tree: R-trees with voronoi diagrams for efficient processing//Proceedings of the VLDB. Singapore, 2010: 1231-1242

[10] Wu W, Yang F, Chan C Y et al. Finch: Evaluating reverse k-nearest-neighbor queries on location data//Proceedings of the VLDB. Auckland, New Zealand, 2008: 1056-1067

[11] Achtert E, Böhm C, Kröger P et al. Efficient reverse k-nearest neighbor estimation. Informatik Forschung und Entwicklung, 2007, 21(3-4): 179-195

[12] Kolahdouzan M, Shahabi C. Voronoi-based k nearest neighbor search for spatial network databases//Proceedings of the VLDB. Toronto, Canada, 2004: 840-851

[13] Zhang J, Papadias D, Mouratidis K et al. Spatial queries in the presence of obstacles//Proceedings of the EDBT'04. LNCS 2992. Berlin: Springer, 2004: 366-384

[14] Xia C, Hsu D, Tung A K H. A fast filter for obstructed nearest neighbor queries//Proceedings of the BNCOD'04. LNCS 3112. Berlin: Springer, 2004: 203-215

[15] Gao Y, Zheng B. Continuous obstructed nearest neighbor queries in spatial databases//Proceedings of the 28th ACM SIGMOD International Conference of Management of Data (Sigmod'09). New York, USA, 2009: 577-590

YU Xiao-Nan, born in 1987, M. S.. Her major research interests include RFID and spatio-temporal data management.

GU Yu, born in 1981, Ph. D., associate professor. His major research interests include RFID data management, spatio data management and cloud computing.

ZHANG Tian-Cheng, born in 1969, Ph. D., associate professor. His major research interests include Internet of Things.

YU Ge, born in 1962, Ph. D., professor, Ph. D. supervisor. His major research interests include database theory and technology, distributed and parallel systems, etc.

Background

This paper studies the reverse k-nearest-neighbor queries by the existence of obstacles. With the rapid development of location-based services (LBS) and the Internet of Things, technologies for spatial queries are becoming more and more important. Moreover, nearest neighbor queries and the variant are widely used spatial queries. Recently, there has been much work on reverse k nearest neighbors (RkNN) queries. However, these studies are proposed for the ideal Euclidean Space. Queries for reverse k nearest neighbors are influenced by obstacles in practice.

In this paper, the authors study a method for reverse k nearest neighbors queries in obstructed spaces, and propose efficient pruning algorithms based on an obstructed Voronoi diagram. Furthermore, these pruning methods greatly reduce the number of searched points by properties of Voronoi diagrams and obstructed distance. Finally, the experiments based on real and synthetic data sets demonstrate the efficiency and accuracy of our proposed approach.

This work was supported by the National Natural Science Foundation of China under grant No. 61003058 and No. 60873009, Liaoning Province Doctor Startup Fund under grant No. 20091025 and Fundamental Research Funds for the Central University under grant No. 090404013.

The authors of this paper have been engaged in spatial-temporal query processing and uncertain data management. They have published some papers in this area.