doi:10.3969/j.issn.1005-3026.2017.06.007

基于位置的偏好查询处理技术

李 森,谷 峪,于 戈
(东北大学 计算机科学与工程学院,辽宁 沈阳 110169)

摘 要:在线位置服务技术日益普及,用户能够很容易获得他们的地理位置信息.随之产生了各类有关空间关键字的查询,这些查询可以提供定位服务的基本查询功能.研究了基于位置的偏好查询处理技术,旨在为用户找到一个目的地,找到的结果应该满足指定的特性,并且靠近满足用户提出的偏好.同时,提出一种新颖的查询框架,该框架通过对IR-tree的节点扩展给出预计算信息表,根据扩展的IR-tree能够减少搜索空间并提出准确计算方法来有效地回答基于位置的偏好查询.在真实数据集上进行实验验证了提出方法的有效性.

关 键 词:偏好查询;IR-树;扩展IR-树;倒排文件;位置服务
中图分类号:TP 392 **文献标志码**:A **文章编号**:1005-3026(2017)06-0793-05

A Technique for Processing Location-aware Preference Queries

LI Miao, *GU Yu*, *YU Ge*
(School of Computer Science & Engineering, Northeastern University, Shenyang 110169, China. Corresponding author: YU Ge, E-mail: yuge@mail.neu.edu.cn)

Abstract: There has been increasing popularity of online location-based services. It gives prominence to various types of spatial-keyword queries, which are employed to provide fundamental querying functionality for location-based services. A technique for processing location-aware preference queries was studied that aimed to find a destination place for a user. The user wants to go to a place labeled with a specified category feature (e.g., hotel), and he/she has a location and a set of additional preferences. It was expected that the result place of the query belongs to the specified feature, and it was close to places satisfying the preferences of the user. A novel framework was developed for answering the queries, which was called augmented IR-tree. An augmented IR-tree could be obtained by adding the pre-computed information into an IR-tree. The augmented IR-tree could be used to reduce the search space and compute the exact query result. The proposed technique was verified by extensive experiments on one real dataset, and the technique is more efficient than baseline methods.

Key words: preference query; IR-tree; augmented IR-tree; inverted files; location-based service

随着地理位置技术(如GPS技术)的蓬勃发展,在线的基于位置服务系统(如Google Maps, Foursquare和Bing Maps)越来越流行.随着定位技术快速发展,产生了大量带有地理信息的对象(或者空间兴趣点),并且它们有着不同的分类特征(宾馆、度假村、商店、车站和旅游景点等).随之产生了许多有关空间关键字类型的查询[1],都是可以通过基于位置服务来提供给用户基本的查询服务.

本文提出一种基于位置的偏好查询(location-aware preference, LP).一个LP查询为一个用户返回空间中一个位置,该位置被标记为一个特殊的分类特征,并且该用户有自己的偏好属性集合.那么,查询返回的位置满足特定的分类特征,并且满足与用户提出的附加偏好位置要近.

LP查询的基本原理基于以下两个因素考虑:1)用户与目的地离他当前位置要近;2)用户也希望目的地与他提出的偏好在空间位置上要靠近.

收稿日期:2015-05-24
基金项目:国家自然科学基金资助项目(61472071, 61433008);国家重点基础研究发展计划项目(2012CB316201).
作者简介:李 森(1985-),女,辽宁鞍山人,东北大学博士研究生;谷 峪(1981-),男,辽宁鞍山人,东北大学教授;于 戈(1963-),男,辽宁沈阳人,东北大学教授,博士生导师.

例如,如果一个用户想去酒店,但提出的偏好特征是“地铁站”,那么他可能更希望查到的目的地要离地铁站近一些. 之前的文献中,还没有对 LP 查询进行研究. 最直接的方法处理 LP 查询是估计所有目的特征对象. 特别地,对于每一个这样的对象,都需要计算目的特征对象与用户之间的空间邻近性和对象与用户提出的与其最近偏好之间的空间邻近性. 然而,这种方法对于大型数据集会带来相当大的计算代价.

为了有效处理 LP 查询,本文需要处理如下挑战. 第一,对于每一个候选对象 o,都需要消耗大量时间去找到目的对象与用户提出每一个的最近偏好,并计算所有这些距离. 需要更有效的方式来解决这种问题. 第二,包含目的对象的数量可能非常巨大,将产生很大的代价去计算每一个目的对象的得分值. 需要一种有效的剪枝方法来减少搜索空间.

本文提出一种新颖的框架来处理 LP 查询. 并且对每一个 IR – tree[2] 的节点进行扩展,对空间地理对象的一些信息进行预计算,从而帮助剪枝搜索空间. 根据扩展的 IR – tree 策略,设计一种最好最快的搜索算法.

1 相关工作

最接近本文的问题是空间偏好查询. 空间偏好查询的工作[3-6]是基于对象 o 的空间邻居中存在查询偏好特征的关系给出地理对象 o 的得分. 例如,对象 o 的得分通过以下方式计算:1)聚合以对象 o 为中心的一个空间范围内包括的所有查询偏好特征的分数来计算 o 的得分,对每一个对象的得分都建立在距离函数或者给定的权重值;2)对象 o 的最近邻来计算得分;3)每个对象的影响分子来计算得分,通常影响分子被定义为该对象 o 到查询偏好的空间距离.

Yiu 等[5-6]首先研究空间偏好查询的问题. 他们把对象分成两类,分别叫做数据对象和特征对象,并且用分别来索引. 其中,数据对象被存储在一个 R – tree 中[7],而特征集中的没种特征对象被单独存在一个聚合 R – tree 中[8](aggregateR – tree,aRtree). 在 R – tree 和 aR – tree 的基础上提出三种算法来处理空间偏好查询.

为提高文献[5]中算法的有效性,Rocha-Junior 等[3]提出一种基于实体化的方法来处理空间偏好查询,能有效节省计算代价和 I/O 代价. 处理空间偏好查询的问题也在路网上得到了研究[9-10].

然而,空间偏好查询与本文提出的 LP 查询有很大的不同. 因为空间偏好查询没有一个查询的位置,空间偏好查询的研究中提出的算法不能应用到本文提出的 LP 查询.

2 问题定义

定义 1 基于位置的偏好(LP)查询. 一个 LP 查询 q 表示一个元组 $q=\langle l,f_d,\Psi\rangle$,其中 l 表示查询 q 的位置,f_d 表示目的地特征,$\Psi=\langle f_0,f_1,\cdots,f_n\rangle$ 表示一组偏好集. S 表示地理对象的集合. 一个 LP 查询返回 S 中的一个地理对象 o_r,并且对象 o_r 满足其分类标签与目的地特征 f_d 相同,o_r 有最高的位置偏好得分.

直观地,LP 查询 q 与对象 o 的位置偏好得分从以下两个方面进行考虑:

1)查询点 q 与对象 o 的空间邻近性;

2)对于 $q.\Psi$ 中的每一个 f_i,需要找到 o 的符合其偏好的最近邻,并计算偏好到 o 的距离. 两个方面的得分和被作为对象 o 的位置偏好得分.

特别地,利用函数 $S(q,o)$ 来表示查询点 q 与对象 o 的位置偏好得分,并且通过式(1)计算.

$$S(q,o)=\lambda\left(1-\frac{d(q,o)}{d_{\max}}\right)+(1-\lambda)\frac{1}{|q.\Psi|}\cdot\sum_{f_i\in q.\Psi}\left(1-\frac{\mathrm{minDist}(o,f_i)}{d_{\max}}\right). \tag{1}$$

其中:$d(q,o)$ 表示查询点 q 与对象 o 的欧式距离,$d_{\max}$ 表示数据集中两个点在空间中的最大合理距离,$\mathrm{minDist}(o,f_i)$ 表示对象 o 和包含有特征 f_i 的对象之间的距离,使用 $\lambda\in[0,1]$ 来平衡两个空间邻近性的得分.

3 基本算法

处理 LP 查询直接的方法是基于倒排文件和 R – tree 的结合而提出. 一个倒排文件包含一组词,并且每个词与一个信息记录表(postings list)联系在一起. 每个信息记录表由一系列记录(postings)组成,记录中包括一个分类标签为 $o.f$ 的对象 o 的标示符. R – tree 是空间查询中占有主导地位的索引,广泛应用在加速查找最近邻[11].

让 q 表示 LP 的一个查询,$f.d$ 表示 q 的目的地特征. 算法首先通过便利倒排文件来检索所有包含特征为 $f.d$ 的对象;然后对于每一个被检索的对象 o_i,需要在式(1)的基础上对 q 去计算. 特

别地，计算 $d(q, o_i)$ 是比较容易的，计算 $\text{minDist}(o_i, f_i)$ 则需要通过利用 R－tree 的最近邻搜索找到与对象 o_i 最近的对象，并且这些对象满足特征为 f_i. 那么，这样的基础算法的复杂度为 $O(N_{f_d}(1 + |N_F| \cdot N))$，也是在 R－tree 上进行最近邻搜索时的最坏情况的时间复杂度，其中，N_{f_d} 表示包含特征为 f_d 的对象的数量；N_F 表示在查询 q 中偏好特征的数量.

第二种方法基于 IR－tree[2]. IR－tree 是扩展 R－tree 上的每个节点，在相应的子树上增加每个节点的文本内容. 具体地，每个节点包含一个指针指向一个倒排文件，倒排文件描述了在根节点下的子树中的对象. 利用 IR－tree 来处理 LP 查询的过程如下，让 q 是一个 LP 查询，f_d 是 q 的目的特征. 从 IR－tree 的根节点出发，如果一个非叶子节点被访问，那么遍历这个节点的所有孩子并且计算 $\text{minDist}(R_i, f_i)$，得到 $S(q, o_i)$ 的最小值. 然后把该节点和这个最小值 $S(q, o_i)$ 压入到队列中. 如果被访问的节点是叶子节点，需要遍历该叶子节点中的所有对象，并且得到真实距离 $S(q, o_i)$. 然后，比较 $S(q, o_i)$ 和队列中的第一个元素. 如果得分 $S(q, o_i)$ 小于等于第一个元素值，把 $S(q, o_i)$ 放入结果集中. 否则，把这个对象和他的得分放入队列直到找到 top－k 个结果.

4　扩展的 IR－tree

通过预计算信息扩展 IR－tree[2]，使扩展 IR－tree 能够利用提出的算法有效减少搜索空间.

处理 LP 查询的基本算法要求计算每一个 $f_i \in S_F$ 并且对象 o_i 含有特征 f_d 的 $\text{minDist}(o_i, f_i)$ 值需要执行最近邻搜索，但是这样计算代价相当大.

一个直接的可以提高查询效率的方法是事先计算好并存储每个对象 o_i 和每个特征 f_i 的 $\text{minDist}(o_i, f_i)$ 值. 然而，该方法会导致很大的空间代价，空间复杂度为 $O(N^2)$（最坏情况是每一个对象都有一个与其他不同的特征），其中 N 表示对象的数量. 为进一步提高查询效率而又不导致高的空间代价，扩展的 IR－tree 将被使用组织地理对象.

扩展 IR－tree 是通过 IR－tree 的每个节点都增加一个指针指向一个预计算信息表（pre-computed information）扩展而得到的，其中，这个预计算信息表存储的是每个节点中的特征反向最近邻（feature-aware reverse nearest neighbors，简称为 f－RNN），并且它对应的 f－RNNs 的最小距离可以帮助剪枝搜索空间来减低空间代价. 首先介绍特征反向最近邻（f－RNN）的定义和 f－RNNs 的最小距离的定义.

定义 2　特征反向最近邻（f－RNN）. o 是一个地理对象，f 是一个偏好特征. 如果对象 o 是在所有特征为 f 的对象中 o' 的最近邻（即 $\forall o_i \in \{o_j \mid o_j.f = o.f \wedge o_j \neq o\}(d(o', o_i) > d(o', o))$），那么对象 o' 是 o 的一个特征反向最近邻.

特征反向最近邻的定义可以分组对于每一种特征有相同最近邻的对象，并且减少空间代价比基础算法要有实质性的提高.

定义 3　节点 R 中 f－RNNs 的最小距离（$\text{minDist}_f(R)$）. R 是 R－tree 上的一个节点，f_k 是一个特征，f_k－RNN(o) 是一组对象 o 的 f_k－RNNs 集合. 对象在 R 中的 f_k－RNNs 的最小距离表示为 $\text{minDist}_{f_k}(R)$，并且通过以下公式进行计算. 如果 R 不是根节点，存在：

$$\text{minDist}_{f_k}(R) = \min\{d(o_i, o_j) \mid o_i \in R \wedge o_j \in R.\text{parent} \wedge o_j \in f_k\text{－RNN}(o_i)\}.$$

如果 R 是根节点，那么存在：

$$\text{minDist}_{f_k}(R) = \min\{d(o_i, o_j) \mid o_i \in R \wedge o_j \in R \wedge o_j \in f_k\text{－RNN}(o_i)\}.$$

通过扩展 IR－tree 上的每个节点 R 提出被维护的预计算信息表. 关于每个特征 f_k 的预计算信息表包含以下内容：

1）Postings list of f_k，包含特征为 f_k 的对象集合.

2）RNN list，存储满足以下条件的节点 $\mathscr{R}$ 结合：(i) 对于每一个 $R' \in \mathscr{R}$，至少在节点 R 中有一个包含特征为 f_k 的对象是在节点 R' 中的特征反向最近邻；(ii) R' 和 R 有相同的父亲节点. 注意，R 和 R' 可能是同一个节点.

3）$\Delta\text{minDist}_{f_k}(R)$，记录 $\text{minDist}_{f_k}(R)$ 和 $\text{minDist}_{f_k}(R.\text{parent})$ 的差值. 如果 R 是根节点，那么有：$\Delta\text{minDist}_{f_k}(R) = \text{minDist}_{f_k}(R)$.

如果 R 不是根节点，存在：

$$\Delta\text{minDist}_{f_k}(R) = \text{minDist}_{f_k}(R) - \text{minDist}_{f_k}(R.\text{parent}).$$

图 1a 对于空间中的对象建立一个扩展 IR－tree，并且描述所有节点的预计算信息表（图 1b）.

下面分析扩展 IR－tree 的空间复杂度.

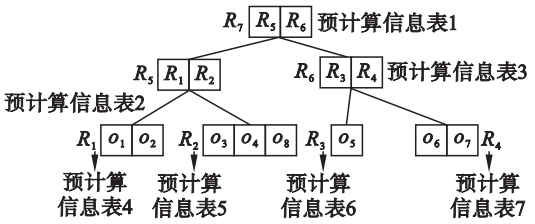
定理 1　给定一个查询 q，在最坏的情况下，扩展 IR－tree 的空间复杂度是 $O(NB \cdot \log_B N)$，其中 N 是空间中对象的总数，B 是每个预计算信息表中列 RNN list 中元素的数量.

证明　扩展 IR－tree 索引的空间复杂度取决

于四个方面:1) 空间中对象 N 的总数量;2) 每个节点中对象或者孩子节点的数量;3) 每个节点中包含特征的 F 的数量;4) 在每一个预计算信息表中的列 Node 中元素 B 的数量.

考虑扩展 IR – tree 索引中叶子节点层,每个叶子节点需要 $F \cdot B$ 的存储空间并且有 N/C 个节点在这一层,因此叶子节点层的空间复杂度为 $O(N/C \cdot F \cdot B)$.

相似地,叶子节点的父亲节点层的空间复杂度也为 $O(N/C \cdot F \cdot B)$. 当同时考虑叶子节点层和其父亲节点层,那么空间复杂度为 $O(N/C^2 \cdot F^2 B)$.



(a)

叶结点		R_1		R_2			R_3			R_4		
关键字	Postings list	RNN list	Δmin Dist	Postings list	RNN list	Δmin Dist	Postings list	RNN list	Δmin Dist	Postings list	RNN list	Δmin Dist
酒店	o_2	R_1	0	o_3	R_2	0						
咖啡厅	o_1	R_2	0	$o_4 o_8$						o_6	R_3	0
电影院							o_5		0			
酒吧										o_7	R_4	0

非叶结点	R_5			R_6		
关键字	Postings list	RNN list	Δmin Dist	Postings list	RNN list	Δmin Dist
酒店	$R_1 R_2$	R_1	0			
咖啡厅	$R_1 R_2$	R_2	0.286	R_4	R_6	0
电影院				R_3	$R_3 R_6$	0
酒吧				R_4	R_6	0

叶结点	R_7	
关键字	Postings list	min Dist
酒店	R_3	0.211
咖啡厅	$R_3 R_4$	0.321
电影院	R_6	0.278
酒吧	R_4	0.350

(b)

图 1 索引结构

Fig. 1 Structure of index

(a)—扩展 IR – tree; (b)—预计算信息表.

同样地,当考虑到从叶子节点层到第 i 层的空间复杂度为 $(N/C^i \cdot F^i B)$. 在最坏的情况下,假设每个节点中的特征都不同,并且在每个预计算信息表中的列 RNN list 中的元素 B 的最大数量等于每个节点 C 的数量. 那么,将得到叶子节点的空间复杂度为 $O(N \cdot B)$,叶子节点的父亲节点层的空间复杂度为 $O(N \cdot B)$,以此类推.

从给定条件可以得到扩展 IR – tree 的所有层的数量 $\log_B N$,这样可以计算得到扩展 IR – tree 索引的空间复杂度为 $O(NB \cdot \log_B N)$.

5 基于位置的偏好查询算法

给定一个 LP 查询,准确算法返回 k 个位置的偏好得分的最高值的候选地理对象.

处理 LP 查询的准确算法是在扩展 IR – tree 的最佳优先遍历算法(例如文献[12])的基础上提出的. 最佳优先遍历算法中,一个优先队列被使用,主要用来跟踪节点和未被访问的对象,并且 $S(q,o)$ 的值被作为对象 o 的主键,对于所有的对象 $o_i \subset R$ 的 $S(q,o)$ 的上界值(表示为 $S\hat{}(q,o)$)作为节点 R 的主键. 为了确定哪一个节点将在下一次被访问,算法将在所有已经被访问过的节点集中选择 $S\hat{}(q,o)$ 的最小值节点 R_i. 当有 k 个最好的对象(通过 $S(q,o)$ 排名)被选择出来后算法终止. 其中 $S\hat{}(q,o)$ 的计算在算法 1 中.

算法 1 给出准确算法的伪代码. 首先,需要维护一个优先队列,队列中每一个元素都有对象或者一个 R – tree 节点组成. 另外,在优先队列中需要为每一个元素 e 维护一个值:以 e 为根节点的子树上的所有对象 o_i 的 $S(q,o_i)$ 上界值. 算法流程如下.

算法 1:基于位置的偏好查询算法 LP(q,k)

```
1. PQueue←NewPriorityQueue();
2. Results←∅;
3. PQueue. Enqueue(Root, q);
4. while(PQueue is not empty) do
5.     e← PQueue. Dequeue();
6.     if(e is never retrieved from PQueue before)
         then
7.       Compute the exact value of S(q,o);
8.     if(PQueue is not empty and
         S(q,o) < PQueue. first(). Key) then
9.       PQueue. Enqueue(e,S^(q,o));
10.    else Results. add(e);
11.      if(|Results| = k)
12.            break;
13.    Traverse(q,e,PQueue).
```

6 实验结果

对提出的算法在真实数据集中进行评估. 所有实验的环境为 Intel(R) Xeon(R) CPU E5 – 16200@3.60 GHz,内存 8 GB,Windows XP 系统的 PC 机,算法使用 Java 语言实现. 真实数据集为 EURO. EURO 是一个欧洲范围内包含 179 506 个兴趣点的真实数据集. 每个兴趣点作为一个地理对象,并且每个点的位置和特征通过它们的分类表示(如: park, hospital, supermarket). 不同分类的特征数量为 68 052 个.

提出关于处理 LP 查询的扩展 IR – tree (augmented IR)算法和基本算法(RIF, IR 算法)的实验结果比较. 默认的参数为 $k = 3$,偏好数量为 2,$\lambda = 0.3$.

在这组实验中，将估计处理 LP 查询在两个数据集中的查询结果的数量(即 k)对于实验效果的影响. 从图 2 中可以看出算法的执行时间和 I/O 代价随着 k 值的增加而增加. 原因是 k 值越大，在查询过程中将带来更大的搜索区域，这将导致更多的页访问来遍历倒排文件和树的节点. 同时，从实验结果也能发现，算法 augmented IR 始终优于 RIF 和 IR，这是因为 augmented IR 具有一定的剪枝效果.

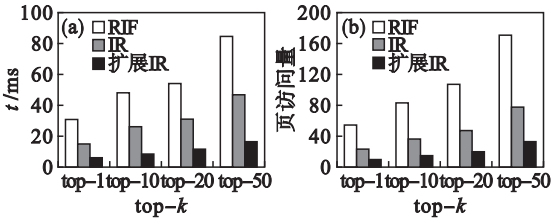


图 2　k 在 EURO 数据集上对 LP 查询的影响

Fig. 2　Varying k for LP query on EURO

(a)—执行时间；(b)—I/O 代价.

图 3 显示了当偏好数量发生变化时，不同算法的执行时间和 I/O 代价的变化. 从实验结果中可以看出，无论在执行时间还是在 I/O 代价，算法 augmented IR 都优于算法 RIF 和 IR. 在 EURO 数据集中，这两种方法的性能都随着偏好数量的增加而呈上升趋势. 原因是偏好数量的增加将带来更多的信息记录表被访问.

图 4 表明式(1)中参数 λ 的变化对实验效果的影响，λ 是用户用来平衡在 $d(q,o)$ 和 $\mathrm{minDist}(o,f_i)$ 之间的一个值. 从实验结果中可以看出 λ 值对于算法的执行时间和 I/O 代价的改变没有大的影响. 同时，实验结果也表明算法 augmented IR 始终要优于算法 RIF 和 IR.

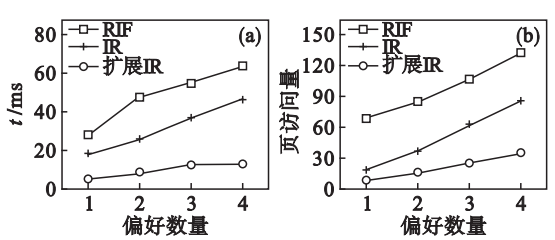


图 3　偏好数量的变化对 LP 查询的影响

Fig. 3　Varying the number of preference keywords

(a)—执行时间；(b)—I/O 代价.

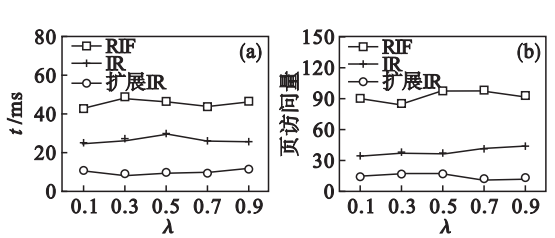


图 4　λ 在 EURO 数据集上对 LP 查询的影响

Fig. 4　Varying λ for LP query on EURO

(a)—执行时间；(b)—I/O 代价.

7　结　论

本文考虑如何处理位置的偏好(LP)查询，最终找到用户想去的目的地；用户在空间中有自己的位置并且会有自己的偏好集合. 预期的查询结果对象属于一种指定的特征，并且该特征点满足与用户提出的偏好在空间中位置的邻近.

本文提出一种框架来有效处理 LP 查询. 实际上，本文提出了一种扩展 IR－tree 的索引结构，可以帮助在处理 LP 查询时剪枝掉搜索空间. 利用扩展 IR－tree 开发出一种基于搜索算法的最快优先算法. 所提方法通过一个真实数据集得以验证，与基础算法比较，查询效率提高了 10 倍以上.

参考文献：

[1] Chen L, Cong G, Jensen C S, et al. Spatial keyword query processing: an experimental evaluation [J]. *Proceedings of the VLDB Endowment*, 2013, 6(3): 217－228.

[2] Cong G, Jensen C S, Wu D. Efficient retrieval of the top-k most relevant spatial web objects [J]. *Proceedings of the VLDB Endowment*, 2009, 2(1): 337－348.

[3] Rocha-Junior J B, Vlachou A, Doulkeridis C, et al. Efficient processing of top-k spatial preference queries [J]. *Proceedings of the VLDB Endowment*, 2010, 3(2): 93－104.

[4] Tsatsanifos G, Vlachou A. On processing top-k spatio-textual preference queries [C]// International Conference on Extending Database Technology. Brussels, 2015: 433－444.

[5] Yiu M L, Dai X, Mamoulis N, et al. Top-k spatial preference queries[C]// International Conference on Data Engineering. Istanbul, 2007: 1076－1085.

[6] Yiu M L, Lu H, Mamoulis N, et al. Ranking spatial data by quality preferences[J]. *Transactions on Knowledge and Data Engineering*, 2010, 23(3): 433－446.

[7] Guttman A. R-trees: a dynamic index structure for spatial searching [C]// ACM's Special Interest Group on Management of Data. Boston, 1984: 47－57.

[8] Attique M, Qamar R, Cho H, et al. A new approach to process top-k spatial preference queries in a directed road network[C]//Proceedings of the Third ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems. Dalls, 2014: 34－42.

[9] Cho H, Kwon S J, Chung T. ALPS: an efficient algorithm for top-k spatial preference search in road networks[J]. *Knowledge and Information Systems*, 2015, 42(3): 599－631.

[10] Roussopoulos N, Kelley S, Vincent F. Nearest neighbor queries [C]// ACM's Special Interest Group on Management of Data. San Jose, 1995: 71－79.

[11] Li Z, Lee K C, K, Zheng B, et al. Irtree: an efficient index for geographic document search[J]. *Transactions on Knowledge and Data Engineering*, 2011, 23(4): 585－599.

[12] Korn F, Muthukrishnan S. Influence sets based on reverse nearest neighbor queries [C]// ACM's Special Interest Group on Management of Data. Dallas, 2000: 201－212.