路网环境下访问序列受限的多标签路线查询算法

张金增 文 洁 孟小峰

(中国人民大学信息学院 北京 100872)

摘 要 随着移动互联网、地理定位技术和智能终端设备的迅速普及,产生了大量的位置信息和其对应的标签(tag)描述信息.路线搜索是人们出行时经常进行的活动,但面临多个任务需求时,寻找最佳路线是一项极为耗时的工作.此外空间对象本身的访问权限和用户指定的限制一定程度上制约了对象的访问次序.针对上述情况,文中提出了一种路网环境下访问序列受限的多标签路线(MTROC)查询,该查询的目标是找出一条从源点到目标点、经由与查询中给定的tag相匹配的空间对象且满足序列约束的最短线路.文中证明了MTROC查询问题是NP-hard,并基于增强的路线叠置-关联目录(EROAD)索引提出了3种近似算法.路线扩展RE-Greedy算法和路线渐增插入RII-Greedy算法通过局部更新获得满足需求的路线,而全局路线优化算法GROA为MTROC查询提供一个全局近似最优解.使用真实和合成数据集对文中提出的算法的有效性和可扩展性进行分析评估,实验结果表明3种算法都能有效地完成MTROC查询,其中GROA算法可扩展性最好,而RII-Greedy算法返回的路线质量最高.

关键词 路网;路线查询;序列约束;签名文件;标签

中图法分类号 TP311 **DOI号**: 10.3724/SP.J.1016.2012.02317

Multi-Tag Route Query Based on Order Constraints in Road Networks

ZHANG Jin-Zeng WEN Jie MENG Xiao-Feng

(*School of Information, Renmin University of China, Beijing* 100872)

Abstract With the increasing popularity of mobile Web, geo-positioning technologies and smart devices, it enables users to generate amounts of location information and corresponding descriptive tags. Route search is a frequent laborious task when users have multiple demands on the trip. In addition, users prefer to specifying the order by which some spatial objects should be visited before others. This paper proposes a new type of route search, multi-tag route query based on order constraints (MTROC) in road networks. We prove that the MTROC query is NP-hard and present three approximate algorithms based on enhanced route overlay and association directory index structure. The route extension greedy algorithm and the route incremental intersection algorithm build a route by greedily inserting a spatial object into some certain segments of the sequence. On contrast, the global route optimistic algorithm provides a globally approximate optimal solution for MTROC query. Extensive experiments on both synthetic and real-world datasets illustrate the efficiency and scalability of the three algorithms proposed in this paper.

Keywords road networks; route query; order constraints; signature file; tag

收稿日期:2012-06-05;最终修改稿收到日期:2012-07-27.本课题得到国家自然科学基金(60833005,61070055,91024032)、中国人民大学科学研究基金(10XNI018)、核高基重大专项(2010ZX01042-002-003)、高等学校博士学科点专项科研基金(200800020002)资助.**张金增**,女,1983年生,博士研究生,主要研究方向为移动对象管理、移动Web搜索、空间数据挖掘.E-mail: zajize@163.com.**文 洁**,女,1989年生,硕士研究生,主要研究方向为移动数据管理.**孟小峰**,男,1964年生,博士,教授,博士生导师,主要研究领域为Web数据集成、XML数据库系统、移动对象管理.

1 引 言

随着地理定位技术 GPS、WiFi 和智能移动设备的普及以及移动互联网的迅速发展，出现了大量的本地搜索、地理社交网络和在线地图服务，比如 Google Map、Bing Local Search、Foursquare、Flicker 等. 使用各种基于位置的服务，人们可以随时随地进行以下活动：记录位置和活动信息并将这些资源上传，标记在地图上；利用 tag 或者 tweet 对空间对象以及其它实体进行描述，为其提供丰富的语义内容；搜索满足用户查询需求的场所或者路线.

在日常生活中，大多数用户外出时都有多个任务，不管对该城市是否熟悉，根据这些任务需求手动地制定出行计划和行程路线，都是一项费时费力的工作. 因此为用户提供一条经由多个空间对象具有最小代价(距离或者时间花费最短)、满足多个需求的路线是非常必要的. 此外，空间对象会受到用户指定的访问或者权利限定(比如银行的营业时间为早 9:00～下午 5:00)的制约，因此在规划路线时，需要考虑制约因素对访问顺序的影响. 考虑图 1 所示的例子，假设从开会地点回酒店的路上，Mary 需要给汽车加油，去中国银行办理业务，到肯德基吃饭，买手机充电器. 此外，有两个次序的约束：(1) 吃饭前先去银行；(2) 买充电器前先去加油. Mary 想要找到一条满足上述需求和约束条件的最短路线(实线箭头显示). 因此本文提出了一种新的空间查询：路网环境下访问序列受限的多标签(tag)路线(MTROC)查询. 在该查询中，用户使用多个 tag 指定要访问的各个空间对象以及它们之间的访问序列约束，查询的源点由用户提供，或通过移动设备自带的 GPS 来识别，而终点由用户指定. 查询目标在于找到一条从源点到终点、经由与每一个查询 tag 匹配的空间对象并满足序列约束的最佳路线.

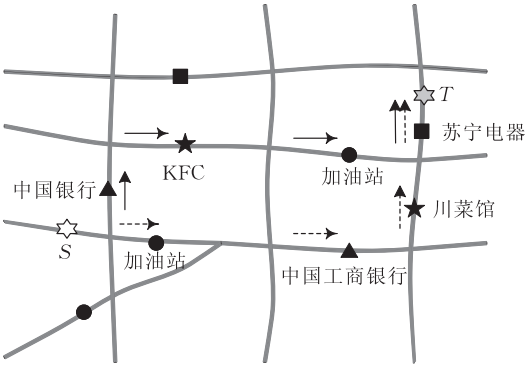


图 1 MTROC 查询例子

目前，与路线搜索相关的工作有以下两类. 首先，已有的在线 Web 地图服务(Google Map、Bing Map、百度地图)以及空间关键字查询[1-5]方面的研究工作不能很好地回答 MTROC 查询. 在线地图服务使用关键字或地址搜索出任意两个位置之间的线路，而空间关键字查询[1-5]是以一个位置和关键字集合作为参数，检索出离查询点最近并匹配查询关键字的对象. 如果利用上述的方法回答图 1 中的查询，用户首先需要查询附近的一些加油站和中国银行，定位出几个邻近的肯德基和卖充电器的商场，然后将找到的这些空间对象根据序列限制进行组合，最终找出一条满足用户需求的路线. 这种方法需要列举出所有候选路线，从中找到最短的路线，显然该方案效率低，费时费力. 其次，已有的路线搜索方法[6-9]根据空间对象所属的类别指定路线需求，如果采用此类方法解决 MTROC 查询，将得到图 1 虚线箭头所显示的路线，该路线中的对象“中国工商银行”和“川菜馆”虽然属于银行类和餐饮类，但并不满足用户的需求，不适合本文提出的查询. 因此亟需设计有效的查询策略回答 MTROC 查询.

构建查询路线存在以下挑战性问题：一方面 Web 上的空间对象数据快速增长，另一方面次序限制仅仅是对查询需求的部分约束，这就要求找出满足约束的所有全序序列，从得到的大量候选路线中挑出最短的路径，对于路线查询处理来说代价很高. 此外，旅行商(TSP)问题可以看作是路网环境下访问序列受限的多 tag 路线(MTROC)查询问题的特例，所以 MTROC 查询问题是 NP-hard(详细证明见第 4 节). 为了回答该查询，本文提出了 3 种近似算法. 本文的主要贡献在于：

(1) 定义了一种路网环境下访问序列受限的多 tag 路线(MTROC)查询，并证明 MTROC 查询问题是 NP-hard.

(2) 采用哈斯图表示序列约束集，利用拓扑排序将约束关系和查询算法相结合处理 MTROC 查询.

(3) 为了回答 MTROC 查询，首先提出两种局部算法，路线扩展 RE-Greedy 算法和路线渐增插入 RII-Greedy 算法. 由于局部算法在构建路线的过程中，仅考虑对其中某一段路线的影响，为此提出了一种全局路线优化算法 GROA.

(4) 采用真实和合成数据集对提出的算法进行充分的实验评估，实验结果表明 3 种算法都能有效地完成 MTROC 查询，其中算法 GROA 可扩展性最好，而 RII-Greedy 算法返回的路线质量最高.

本文第 2 节对 MTROC 查询进行形式化定义；第 3 节介绍一种增强的路线叠置-关联目录索引 EROAD；第 4 节详细阐述 3 种查询处理策略；第 5 节通过实验对算法进行评估，并对结果进行详细的对比分析；第 6 节对全文进行总结并对未来研究工作进行展望.

2　问题描述

形式上，空间数据集 D 是由路网中的空间对象组成. D 中的每一个对象 o 用 $\langle p,t\rangle$ 表示，p 代表对象 o 的位置；t 是 o 对应的 tag 描述集合. 路网用无向图 $G(N,E)$ 表示，N 是节点 n_i（交叉口）的集合；E 表示任意两节点 n_i、n_j 链接边的集合. 两节点之间的距离 $dist(n_i,n_j)$ 是 n_i 到 n_j 之间的最短路网距离，采用经典的 Dijkstra 算法计算. MTROC 查询采用一个四元组 $Q=\{s,e,\Psi,\Theta\}$ 表示，其中 s 和 e 表示用户给定源点和目标点；Ψ 表示查询 tag 集 $\{\varphi_1,\varphi_2,\cdots,\varphi_m\}$；$\varphi_i$ 表示第 i 个查询 tag；Θ 表示有效的序列约束集合.

定义 1.　序列约束. 一条序列约束是一个查询 tag 对 (φ_i,φ_j)，它表示用户必须在访问完一个匹配 φ_i 的对象之后才能到达匹配 φ_j 的对象，也就是 φ_i 是 φ_j 的直接前驱，φ_j 是 φ_i 的直接后继，记为 $\varphi_i\prec\varphi_j$.

为了找到满足需求的路径，将序列约束集合 Θ 表示成一个哈斯图 H_Θ[10]，图中的顶点表示查询 tag φ_i，有向边表示节点间的序列约束 $\langle\varphi_i,\varphi_j\rangle$. Θ 是有效的序列约束集合当且仅当 H_Θ 是一个有向无环图. 而如果 H_Θ 存在环，则不可能找出满足 Θ 的路线，因为两个查询 tag 是彼此的前序的情况是不成立的. 当存在一条哈密顿通路时，Θ 中的查询 tag 是全序关系.

定义 2.　候选路线. 给定查询 $Q\{s,e,\Psi,\Theta\}$，路径 $r=\{s,o_{x_1},o_{x_2},\cdots,o_{x_k},e\}(o_{x_i}\in D,k\leq m)$ 是一条候选路线当且仅当对于任意查询 tag φ_j，至少存在一个空间对象 $o_{x_i}\in D$ 满足 $\varphi_j\in o_{x_i}.t$，而且路线 r 中对象的访问顺序满足序列约束集合 Θ. 所有的候选路线集合记为 $\Re$.

可以看出路线 r 是任意两个连续的对象点通过最短路径链接起来的，因此路线 r 的长度 $L(r)=dist(s,o_{x_1})+\sum_{i=1}^{k-1}dist(o_{x_i},o_{x_{i+1}})+dist(o_{x_k},e)$.

定义 3.　MTROC 查询. 访问序列受限的多 tag 路线（MTROC）查询 $Q\{s,e,\Psi,\Theta\}$ 是从候选路线集合 $\Re$ 中找出一条最佳路线 r^*，使得长度最短，匹配查询 tag 集 Ψ，并满足序列约束 Θ，即 $r^*=\arg\min_{r\in\Re}L(r)$.

3　增强的路线叠置-关联目录索引

为了有效地回答 MTROC 查询，搜索出满足用户需求的最佳路线，本节构造一种增强的路线叠置-关联目录（EROAD）索引结构方便查询处理. 它基于可扩展区域子网分层结构[11]（如图 2 所示），将路网数据和空间对象分开存储，并对该结构进行必要的调整，使其适合本文提出的查询算法.

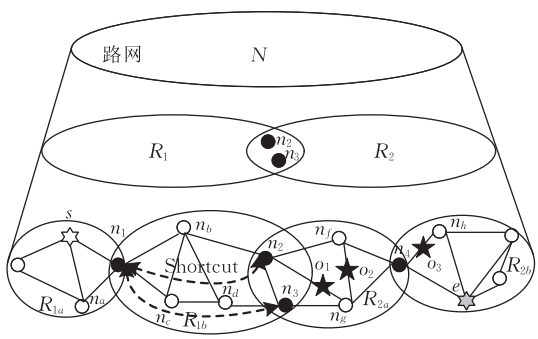


图 2　区域子网分层结构

EROAD 采用几何方法[12-13] 和 Kernighan-Lin 算法[14] 对整个路网按层次进行划分，每一层由相互连接的区域子网组成. 在图 2 中，首先路网空间 N 被划分为两个区域子网 R_1 和 R_2，每一个 R_i 又被划分为两个更小的区域子网 R_{ia} 和 R_{ib}，子网之间公有的节点（比如 n_1,n_2,n_3,n_4）为边界节点. 然后以自底向上的方式构造区域子网的对象摘要（Object abstract）和它们之间的捷径（Shortcut）. 对象摘要包括两部分：包含在区域子网中的对象集合和对应的 tag 概要信息. 上层子网的对象摘要是下层孩子摘要的并集，捷径是区域子网边界点之间的最短路径. 而对于那些不包含任何空间对象的区域子网，在遍历的过程中可以直接剪掉.

为了实现该结构，采取路线叠置（Route Overlay，RO）和关联目录（Association Directory，AD）分别索引存储路网数据和空间对象. 路线叠置利用 B^+ 树对路网节点进行索引，其键值为节点 ID，每一个叶子项指向一个路网节点以及对应的捷径树，如图 3 所示. 非边界节点（比如 s）的捷径树只有一项，存储该节点到其邻近节点的边以及长度（比如 (s,n_1)，(s,n_a)）；而边界节点（比如 n_2）的捷径树按照该点所在区域子网的层次进行构造，该树非叶子节点存储

同一层包含该节点的区域子网(比如 R_{1b},R_{2a})以及每个子网的捷径边(比如 $S(n_2,n_1)$,$S(n_2,n_4)$),叶子节点存储到它邻近节点的边(比如(n_2,n_b),(n_2,n_f),(n_2,n_g)).对于 MTROC 查询,需要不断地更新当前查询点 s,所以与 s 相邻的节点对应的捷径树需要更新,只需在该点捷径树的底层添加一条边即可.

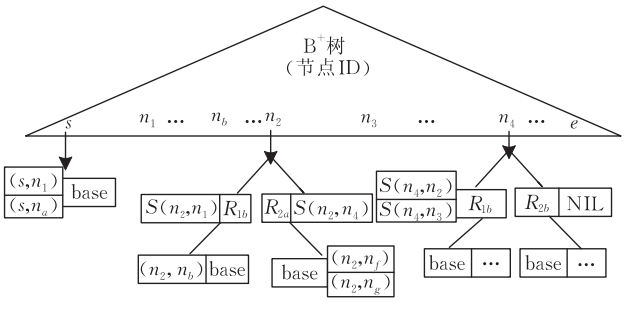


图 3 路线覆盖索引结构

关联目录作为对象查找结构,采用 B+ 树对空间对象进行索引,如图 4 所示.键值为节点的 ID,叶子节点包含路网节点和区域子网两类.若为路网节点,存储两项额外信息:(1)与该节点相邻的对象的距离信息;(2)与该节点相邻对象的 tag 描述信息,采用签名文件[15]存储.而区域子网节点存储其对象摘要,其 tag 概要信息是该区域子网包含的所有对象 tag 签名文件的叠加.如果查询 tag 的签名文件与对象或者区域子网对应的签名文件不匹配,可以直接剪掉该区域子空间.

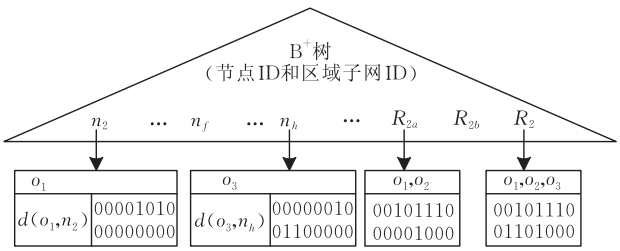


图 4 关联目录索引结果

4 查询处理算法

在分析和提出解决方案之前,给出以下结论.

定理 1. 访问序列受限的多标签路线查询问题是 NP-hard 问题.

证明. 经典的旅行商问题(TSP)是 NP-Complete 问题[16-17].为了证明访问序列受限的多 tag 路线查询问题是 NP-hard,需要构造一个多项式转化将 TSP 问题规约到本文提出的问题.其中查询 tag 集合为{φ_1,φ_2,…,φ_m},每一个 tag 都和一个空间对象集合相关联,假设路线中的每一个空间对象唯一地匹配查询集合中的 tag φ_i,那么 TSP 问题可以看作是访问序列受限的多 tag 路线查询问题中始点 s 等于终点 e,并且序列约束为空的情况.由此可以得出,访问序列受限的多 tag 路线查询问题是 NP-hard 问题. 证毕.

可以看出,不存在多项式时间的算法处理 MTROC 查询,因此本节提出了两种局部贪婪算法和一种全局优化算法.

4.1 局部贪婪算法

在这一部分,提出两种不同的局部算法回答 MTROC 查询.首先介绍算法中使用的一些符号:$qt(o)$表示对象 o 对应的查询 tag;$sig(t_i)$表示 tag t_i 的签名文件;$P_o(t_i)$表示包含 tag t_i 的空间对象集合;$P_{Rj}(t_i)$表示第 j 层匹配 t_i 的区域子网集合,$R(o_i)$表示包含对象 o_i 的区域子网,$P_o(t_i)$、$P_{Rj}(t_i)$和 $R(o_i)$通过遍历关联目录索引得到.

为了降低路线 r 中最短距离的计算代价,减少查询处理中精确路网距离的计算次数,本节利用路线长度下边界 $dist^-(r)$和路线长度上边界 $dist^+(r)$估计路线 r 的长度,仅当路线 r 变成候选结果时再计算其精确的路网距离,由于在索引结构中边界点之间的距离已知,路线准确距离只需计算出对象到所在区域子网的边界点距离即可得到.路线中任意两个对象之间距离下边界和上边界的定义如下.

定义 4. 距离边界 $dist^-$,$dist^+$. 给定路线 r 中任意两个点 o_i 和 o_j,假设包含 o_i 和 o_j 的最底层的区域子网分别为 $R_{ma}(n_{f1},n_{f2})$和 $R_{mb}(n_{g1},n_{g2})$,其中 m 表示区域子网所在的层次,n_{f1},n_{f2} 为 R_{ma} 的左边界节点和右边界节点,n_{g1},n_{g2} 为 R_{nb} 的左边界点和右边界点.因此下边界距离 $dist^-(o_i,o_j)=dist(n_{f2},n_{g1})$,上边界距离 $dist^+(o_i,o_j)=dist(n_{f1},n_{g2})$.

4.1.1 路线扩展 RE-Greedy 算法

路线扩展算法是一种发现最短路线的启发式算法.给定一个查询 $Q\{s,e,\Psi,\Theta\}$,该算法首先通过对哈斯图 H_Θ 进行拓扑排序[18],获得入度为 0 的 tag 集合 Δ_{zero}.然后渐增地向部分候选路径 $r(s,o_{x1},\cdots,o_{xl},e)(l<m)$的最后一段线段中添加对象 o,使得 o 的 tag 描述与 Δ_{zero} 中的某个查询 tag φ_i 相匹配,即 $sig(\varphi_i)\&sig(o)=sig(\varphi_i)$,并且插入对象 o 后线段 (o_{xl},e)长度变化最小,即 $o=\underset{o\in P_o(\Delta_{zero})}{\arg\min}\{dist(o_{x_l},o)+dist(o,t)\}$.由于向候选路径中每添加一个对象,都

需要$|P_o(\Delta_{zero})|$次线段长度的计算，为了降低计算代价，给出以下定理.

定理 2. 给定两个对象o_i和o_j，o_i和o_j均与Δ_{zero}中的某个 tag 匹配，如果

$$dist^-(o_{xl},o_i,e)\geq dist^+(o_{xl},o_j,e),$$

则o_i不可能被添加到候选路径C_r中.

证明. 由于$dist(o_{xl},o_j,e)<dist^+(o_{xl},o_j,e)$，$dist(o_{xl},o_i,e)>dist^-(o_{xl},o_i,e)$，根据条件$dist^-(o_{xl},o_i,e)\geq dist^+(o_{xl},o_j,e)$，显然，$dist(o_{xl},o_i,e)>dist(o_{xl},o_j,e)$，故$o_i$不可能添加至$C_r$中.

证毕.

算法 1. 路线扩展 RE-Greedy 算法.

输入：MTROC 查询$Q\{s,e,\Psi,\Theta\}$，路线叠置索引RO，关联目录索引AD

输出：满足序列约束Θ且匹配查询 tag 集合Ψ的最短路线r

1. $r\leftarrow\{s,e\}$; $cur\_q\leftarrow s$;
2. build Hasse Diagram H_Θ using Ψ and Θ;
3. $\Delta_{zero}\leftarrow$tags with indegree=0 in H_Θ;
4. while $\Delta_{zero}\neq\varnothing$ do
5. for each tag $t_i\in\Delta_{zero}$ do
6. search $AD(sig(t_i))$;
7. $P_o(t_i)\leftarrow\{o|sig(o)\&sig(t_i)=sig(t_i)\}$;
8. $P_R(t_i)\leftarrow\{R|sig(R)\&sig(t_i)=sig(t_i)\}$;
9. $P(\Delta_{zero})\leftarrow\bigcup_i P_o(t_i)$;
10. randomly select an object o_j;
11. U. enqueue$(o_j,dist(cur_q,o_j,t))$;
12. for each object o in $P(\Delta_{zero})$ do
13. if $(dist^-(cur\_q,o,t)<dist(cur\_q,o_j,t))<(dist^+(cur\_q,o,t)$
14. U. enqueue$(o,dist(cur_q,o,t))$;
15. else if $dist(cur\_q,o_j,t))>(dist^+(cur\_q,o,t)$
16. U. enqueue$(o,dist^-(cur_q,o,t))$;
17. $Cur_best=$U. dequeue(); $r\leftarrow r$. insert(Cur_best);
18. $Cur\_q=Cur\_best$; update RO;
19. remove t_i form $Q.\Psi$ and Δ_{zero};
20. update Δ_{zero};
21. return r.

算法 1 给出了路线扩展 RE-Greedy 算法的处理过程. 首先，路线r初始化为$\{s,e\}$，并设置当前的查询点Cur_q为s(第 1 行). 使用Ψ和Θ构造哈斯图H_Θ，并得到Δ_{zero}，将不在Θ中的 tag 项也添加到Δ_{zero}中(2～3 行)，通过遍历关联目录索引 AD 得到包含Δ_{zero}中 tag 的对象和区域子网(5～9 行). 维护一个优先队列U，对象按$dist^-(cur_q,o,e)$递增的顺序存储，不满足定理 1 条件的节点将其下边界距离值更新为精确值$dist(cur_q,o,e)$，上下边界距离值的计算通过访问路线叠置索引RO得到(12～16 行). 每次循环都可以得到一个最优的对象点cur_best，并将其插入到路线r中(第 17 行). 更新路线覆盖索引RO的当前查询点为cur_best以及其相邻节点对应的捷径树，同时采用拓扑排序的方法更新Δ_{zero}(18～20 行). 整个过程迭代执行，当Δ_{zero}为空时，满足用户需求的最短路线r返回.

4.1.2 路线渐增插入 RII-Greedy 算法

RE-Greedy 算法每次向路线r中添加新对象时，仅考虑对当前查询点到目标点线段长度的影响，造成路线快速达到目标点. 但在很多情况下，当前查询点到目标点那段路线可能并不是对象的最佳插入位置. 因此，本节提出一种路线渐增插入 RII-Greedy 算法，它允许将匹配Δ_{zero}中某个 tag 的对象插入到r中满足序列约束的任何线段中.

假设一条部分候选路线$r=\{s,o_{x1},\cdots,o_{xl},e\}$，共$l+1$段. RII-Greedy 算法每次处理一个查询 tag φ_i，从$P_o(\varphi_i)$中挑选出一个对象o，并将其插入到满足限制条件线段集合Seg_Θ的某一段中，使得添加o后对该线段长度影响最小，

$$(k,o)=\arg\min_{o\in P_o(\varphi_i)\wedge k\in Seg_\Theta}\{dist(o_{xk-1},o)+dist(o_{xk},o)\}.$$

为了确定每个φ_i匹配对象的插入位置，需要构造一张$P_o(\Psi)$前序表，表中的每一项$P_o(\varphi_i)$记录其对应的前序项，这些前序项通过遍历哈斯图H_Θ获得，假如φ_i的序列限制为$\{\varphi_1\prec\varphi_i,\varphi_2\prec\varphi_i\}$，那么$P_o(\varphi_i)$的前序项为$\{P_o(\varphi_1),P_o(\varphi_2)\}$. 当处理$\varphi_i$时，需要检查$P_o(\varphi_i)$对应的前序项，找出$o_{x1}\in P_o(\varphi_1)$和$o_{x2}\in P_o(\varphi_2)$在路线中位置索引最大的点，该点之后的那些线段即是$Seg_\Theta$. 当路线$r$成为完整路径时，整个迭代过程结束，最短路线返回. 与 RE-Greedy 方法相比，本节提出的算法向r中插入空间对象的处理方式不同，需要对算法 1 中 4～16 行的代码进行如下修改，由于降低计算路线精确距离的方法与算法 1 相同，所以不再重复给出.

替换算法 1 第 4～19 行的代码

create a preorder table POT by H_Θ;
for each tag φ_i in Δ_{zero} do
　search for preorder of $P_o(\varphi_i)$ in POT;
　find $Seg_\Theta(\varphi_i)$;
　for each object o in $P_o(\varphi_i)$ do
　　find a segment (o_{xk-1},o_{xk}) of $Seg_\Theta(\varphi_i)$ such that
　　$\min\{dist(o_{xk-1},o)+dist(o,o_{xk})\}$;
　insert o into (o_{xk-1},o_{xk});

4.2 全局路线优化算法 GROA

以上提出的两种局部算法在寻找最短路线过程

中只考虑对某一段路线的影响，是局部最优解. 本节提出一种全局路线优化算法 GROA，每一次都要考虑从源点经由满足需求的对象点到终点的路线长度.

GROA 算法首先找出一条满足查询 Q 相对较短的全局候选路线 r'，初始化路线长度的阈值 T_v 为 r' 的长度 $L(r')$，将那些到当前查询点距离大于 T_v 的对象点去掉，利用序列约束依次更新路线 r' 中经由的对象点，直到路线长度达到最短，并匹配给定的查询 tag 集合. 为了找出路线 r'，首先对每一个查询 tag φ_i，遍历关联目录索引找出 φ_i 对应的 $P_o(\varphi_i)$. 然后从对应的 $P_o(\varphi_i)$ 中检索出一个对象 o_{xi} 使得 $dist(s,o_{xi})+dist(o_{xi},e)$ 的值最小，得到对象集合 $O^*\{o_{x1},\cdots,o_{xk}\}$. 然后每次都从 O^* 入度为 0 的对象集合 $O^*(\Delta_{\text{zero}})$ 中挑选出离当前查询点最近的对象插入到 r' 中. 当 Δ_{zero} 为空时，就得到了路线 r'. 接下来介绍如何利用 r' 和 O^* 获得更短的路线 r.

由于 MTROC 查询访问顺序的限制，造成最短路线的长度随着已满足的查询 tag 集发生变化. 对路线 r 的每次更新，需要找出已匹配的查询 tag 集合 $\vartheta\text{-sat}_\Psi$ 和以 $\vartheta\text{-sat}_\Psi$ 为前缀的对象集合 OS，对于每一个对象 $o\in OS$，需要计算出从 o 经由尚未匹配的查询 tag 对象 $o_{xi}\in O^*$ 到目标点 e 的距离，记为 $dist\text{-}e(o)$，该距离通过遍历路线覆盖索引得到. 如果当前路线 r 的长度小于 T_v，则更新路线 r 和阈值 T_v. 为了找出以 $\vartheta\text{-sat}_\Psi$ 为前缀的对象集合，为 $P_o(\Psi)$ 中的对象构建一张前缀-距离邻接表 PDL，表中的每一项记录对象 $o\in P_o(\Psi)$ 的前缀查询 tag 序列和相应的 $dist\text{-}e(o)$ 值. 假设查询 tag 集合为 $\{\varphi_1,\varphi_2,\varphi_3,\varphi_4\}$，其序列约束为 $\{\varphi_1\prec\varphi_2,\varphi_1\prec\varphi_3,\varphi_2\prec\varphi_4,\varphi_3\prec\varphi_4\}$，相应的哈斯图和查询 tag 对象集如图 5 所示. 根据拓扑排序方法[18]，得到查询 tag 的全序集 $\Gamma(\{\varphi_1,\varphi_2,\varphi_3,\varphi_4\},\{\varphi_1,\varphi_3,\varphi_2,\varphi_4\})$. 其前缀-距离邻近表如表 1 所示，比如对象 o_{21} 的前缀分别为 $\{\varphi_1\}$ 和 $\{\varphi_1,\varphi_3\}$，从 o_{21} 到 t 路线有两条：(1) 经由顺序 $(\varphi_2,\varphi_3,\varphi_4)$ 的路线，距离为 $dist(o_{21},o_{31},o_{41},t)$；(2) 经由顺序 (φ_2,φ_4) 的路线，距离为 $dist(o_{21},o_{41},t)$.

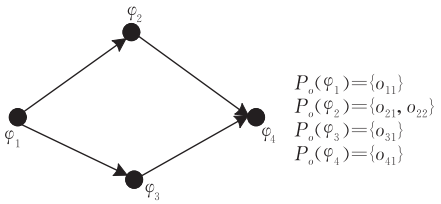


图 5 哈斯图及查询 tag 对象集

表 1 前序-距离邻近表 *PDL*

对象 o	前序	$dist\text{-}e(o)$
o_{21}	φ_1	$dist(o_{21},o_{31},o_{41},t)$
	φ_1,φ_3	$dist(o_{21},o_{41},t)$
o_{22}	φ_1	$dist(o_{22},o_{31},o_{41},t)$
	φ_1,φ_3	$dist(o_{22},o_{\varphi 4},t)$
o_{31}	φ_1	$dist(o_{31},o_{\varphi 21},o_{41},t)$
	φ_1,φ_2	$dist(o_{31},o_{41},t)$
o_{41}	$\varphi_1,\varphi_2,\varphi_3$	$dist(o_{41},t)$

算法 2 描述了全局路线优化算法 GROA 处理流程. 初始化线路 $r=r'=\{s,e\}$，构建哈斯图 H_Θ 并获得拓扑序列 Γ(1～3 行). 按照 Γ 找出每个对象对应的查询 tag 前缀序列，并通过遍历路线覆盖索引 RO 得到 $dist\text{-}e(o)$，从而得到前缀-距离邻近表 PDL(第 4 行). 首先找出全局最短路线 r'(5～14 行)，然后迭代更新获取新的最短路线 r(12～23 行). 在每一次循环遍历的过程中，从 PDL 中查找前缀为 $\vartheta\text{-sat}_\Psi$ 的链表项，得到相应的对象集合 OS，将 OS 中 $dist(Cur\_q,o)+dist\text{-}e(o)$ 值最小的对象 o 插入 r 中，如果当前路线 r 的长度 $L(r)<$阈值 T_v，r 就变成当前最短的路线(16～22 行). 当所有的查询 tag 均匹配时，最短路线 r 返回.

算法 2. 全局路线优化算法 GROA.

输入：MTROC 查询 $Q\{s,e,\Psi,\Theta\}$，$\Psi=\{\varphi_1,\cdots,\varphi_m\}$，路线叠置索引 RO，关联目录索引 AD

输出：满足序列约束 Θ 且匹配查询 tag 集合 Ψ 最短路线 r

1. $r=\{s,e\}$; $r'=\{s,e\}$; $Cur\_q=s$;
2. build Hasse Diagram H_Θ using Ψ and Θ;
3. $\Gamma\leftarrow TopologicalSort(H_\Theta)$;
4. create Prefix-Distance adjacency list PDL by Γ and RO
5. for i from 1 to m do
6. search B$^+$ tree(AD, $sig(\varphi_i)$);
7. $P_o(\varphi_i)\leftarrow\{o\,|\,sig(o)\&sig(\varphi_i)=sig(\varphi_i)\}$;
8. find an object o_{xi} in $P_o(\varphi_i)$ such that min($dist(s,o_{xi})+$dist(o_{xi},e));
9. $O^*\leftarrow\cup\{o_{xi}\}$;
10. pick an object o_{xj} in $O^*(\Delta_{\text{zero}})$ with smallest $dist(s,o_{xj})+dist(o_{xj},e)$;
11. insert o_{xj} into r and r';
12. while $\Delta_{\text{zero}}\neq\varnothing$ do
13. $o'=NN(o_{xj},O^*(\Delta_{\text{zero}}))$;
14. insert o' into r'; update Δ_{zero} and $O^*(\Delta_{\text{zero}})$;
15. $\vartheta\text{-sat}_\Psi\leftarrow\{qt(o_{xj})\}$; $Cur\_q=o_{xj}$; $T_v\leftarrow L(r')$;
16. while $\vartheta\text{-sat}_\Psi<>\Psi$ do
17. $OS=\{o\,|$ prefix of o is equal to $\vartheta\text{-sat}_\Psi$ in $PDL\}$;
18. find an object o in OS such that argmin$\{dist(Cur_q,o)+dist\text{-}t\{Cur_q\}\}$;
19. $L(r)=dist(s,Cur\_q)+dist(Cur\_q,o)+dist\text{-}e\{o\}$;
20. if $L(r)<T_v$ then
21. insert o into r; $Cur\_q=o$; $T_v\leftarrow L(r)$;
22. $\vartheta\text{-sat}_\Psi\leftarrow\cup\{qt(o)\}$;
23. return r.

5　实验结果与分析

5.1　实验设置

在本节中，通过大量的实验评估本文提出的 3 种算法 RE-Greedy、RII-Greddy 和 GROA 的有效性和可扩展性，并进行对比分析. 所有实验采用 Java 实现，运行环境为 Intel® Core™ 2 Duo CPU T7500@2.66 GHz、4 GB RAM 和运行 32 位的 Windows 7 操作系统.

本文使用合成和真实数据集对提出的算法进行评估. 首先从 DCW① 获取两个城市的真实路网数据，分别是 Oldenburg(OL)和 California(CA)，OL 包含 6150 个节点和 7035 条边，CA 包含 21 048 个节点和 22 830 条边. 为了产生合成数据集 OL，使用空间节点生成器随机产生 100 000 个对象点，将其均匀地分布在路网边上，然后从 del. ic. ious② 中抽取出 300 000 个 tag，与 OL 中的空间对象组合生成合成数据集. 对于真实数据集 CA，从 OpenStreetMap③ 获得 California(CA)的空间对象(POI)集合，包含 100 000 个 POI，每个 POI 对应有 2～5 个 tag 描述信息，并将 POI 数据集和路网数据合并.

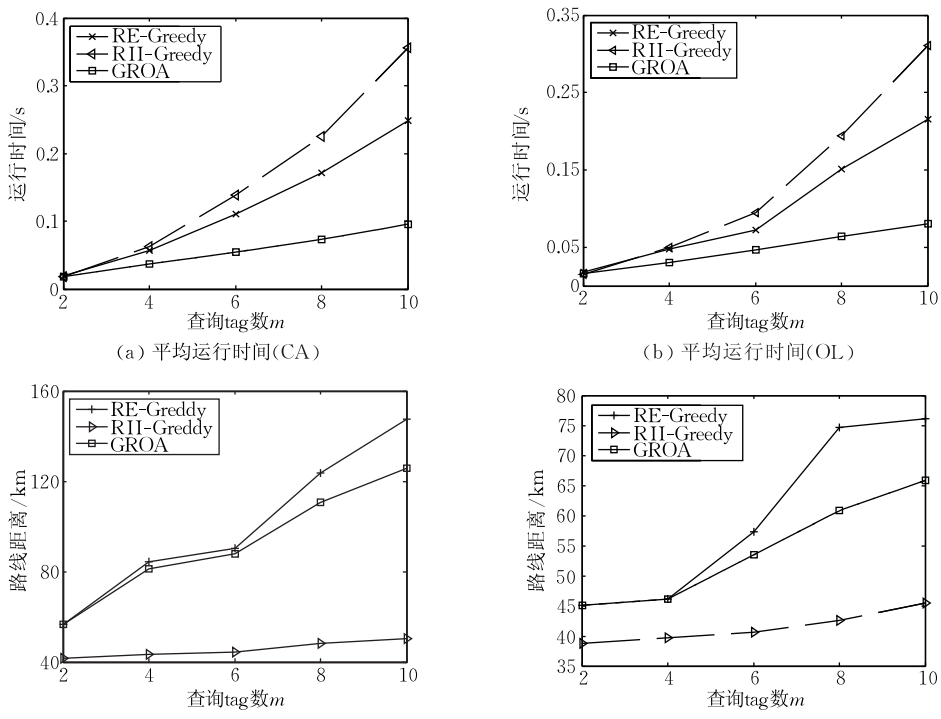
随机产生 5 个查询集，每个查询集包含 50 个查询，每个查询的源点和目标点从路网数据集的节点中随机选择，查询 tag 集合从 POI 数据集中对象对应的 tag 集中随机挑选，分别为 2，4，6，8，10，而序列约束集合由查询 tag 集合随机产生. 测试过程使用的参数如表 2 所示，在每一组实验中，仅变化一个参数的取值而其它参数取缺省值，其中受限查询 tag 比率 PCT 表示受到约束限制的查询 tag 数占总的查询 tag 数的比率.

表 2　实验参数

参数	取值	缺省值
查询 tag 数 m	2，4，6，8，10	4
受限 tag 比率 PCT	0.0，0.25，0.5，0.75，1.0	0.5
数据集大小$\|D\|$	100 000～1 000 000	100 000

5.2　查询性能分析

本节分别在真实数据集和合成数据集上测试了 RE-Greedy、RII-Greedy 和 GROA 算法的有效性，并对可扩展性进行评估，采用查询平均运行时间和路线质量作为评价指标，路线质量采用路线长度度量. 性能测试内容包括以下 3 个方面.



(a) 平均运行时间(CA)　(b) 平均运行时间(OL)

(c) 路线质量(CA)　(d) 路线质量(OL)

图 6　查询 tag 数的影响

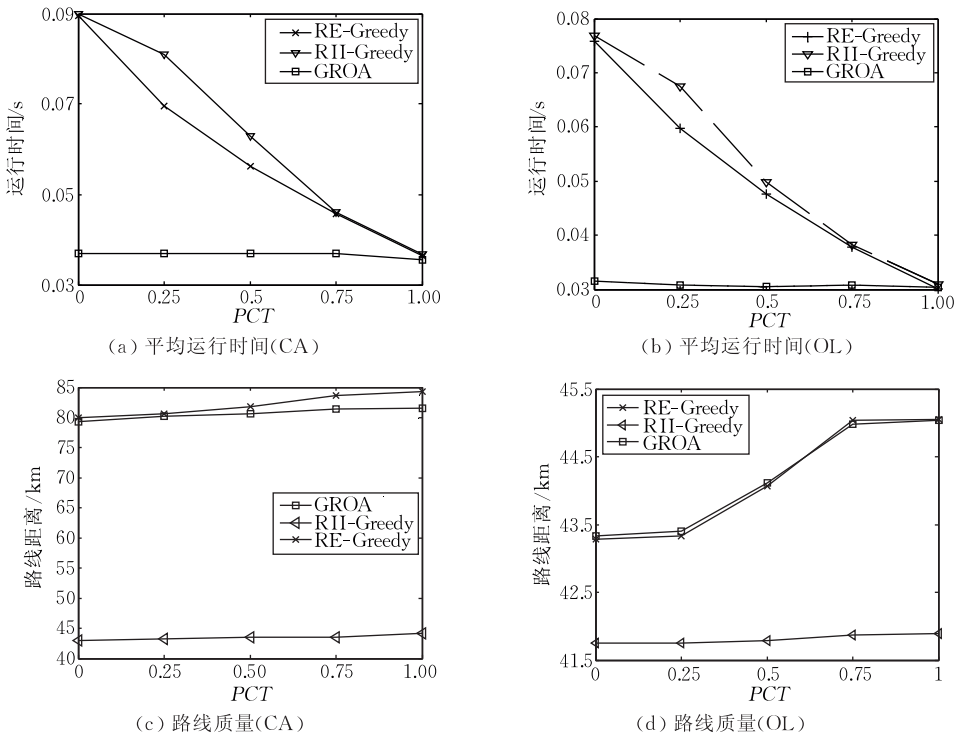
查询 tag 数对算法性能的影响. 该组实验测试了随着查询 tag 数的变化，3 种算法的平均运行时间和路线质量的变化情况，实验结果如图 6 所示.

① DCW(Digital Chart of the World)，http://www.maprom.psu.edu/dcw
② Del. ic. ious，http://www.delicious.com/
③ OpenStreetMap，http://www.openstreetmap.org/

图 6(a)和 6(b)给出了查询 tag 数的不同取值对算法平均运行时间的影响情况. 可以看出,3 种算法的平均运行时间随着查询 tag 数 m 的增大而增加. 因为当 m 变大时,为了回答 MTROC 查询所有的算法需要访问更多的空间对象和计算更多的路网距离. 其中 GROA 运行效率最高,RE-Greedy 算法次之,RII-Greedy 算法运行时间最长. 原因在于相对于 GROA 算法而言,RII-Greedy 算法不仅需要访问更多的空间对象,还需要寻找最佳的插入位置. 所有算法的运行时间不超过 0.36 s. 图 6(c)和 6(d)给出查询 tag 数的变化对路线质量的影响情况. 可以看出,随着查询 tag 数的增加,3 种算法的路线长度都随之增加,这是因为需要访问更多的空间对象. RII-Greedy 算法得到的路线质量最优,而 GROA 优于 RE-Greedy 算法.

受限查询 tag 比率 PCT 对算法性能的影响. 该组实验的目的是评估受限查询 tag 率变化时,查询算法的时间代价和路线质量. 如图 7 所示. 图 7(a)和(b)给出了受限查询 tag 比率 PCT 取不同值时,3 种算法的时间开销. 可以看出,随着 PCT 变大,算法运行时间减少. 这是因为在每次迭代循环过程中,一个更高的 PCT 值减少了遍历空间对象的数目,降低了搜索空间. 还注意到 GROA 算法的运行时间最小,并且变化不太明显,而算法 RE-Greedy 和 RII-Greedy 的运行时间急剧下降. 当 $PCT \geq 0.75$ 时,3 种算法的运行时间相当,都不大于 0.04 s.

图 7(c)和 7(d)给出了 PCT 不同取值对路线质量的影响情况. 可以看出,随着受限 tag 比率 PCT 的增加,路线长度增大. 这是因为 PCT 越大,序列约束限制就越多,从而造成产生的路线更长. 在两个数据集上的测试 RII-Greedy 算法的路线质量都要优于其它两种算法,且路线长度最短. 此外,可以看到 OL 数据集的路线长度均小于 CA 数据集的路线长度,这是因为 OL 数据集中的空间对象的密度较高,分布集中,而 CA 数据集的对象分布比较稀疏.



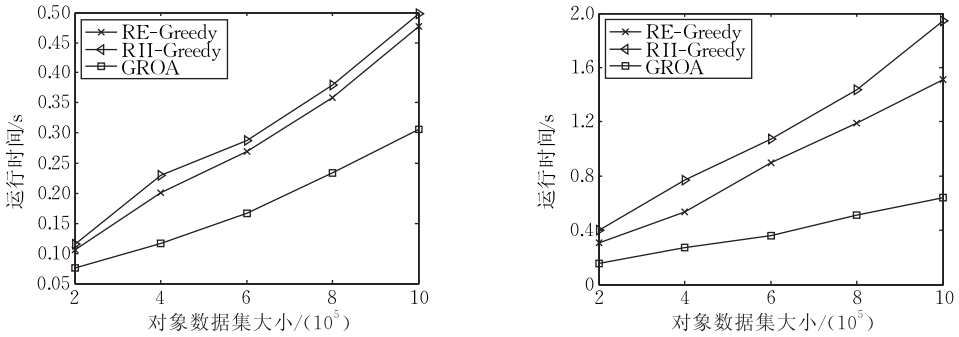
(a) 平均运行时间(CA)　(b) 平均运行时间(OL)

(c) 路线质量(CA)　(d) 路线质量(OL)

图 7　受限查询 tag 率的影响

总的来说,当 $PCT \geq 0.5$ 时,RII-Greedy 算法的时间代价略高,最长的运行时间为 0.089 s,但路线质量始终最好,因此该算法的整体性能较高,适用于查询序列约束较多的情况,而算法 GROA 适合于对查询响应时间要求较高的情况.

空间对象数量对算法的影响. 该组实验通过变化空间对象数据集的大小,在合成数据集 OL 上对本文提出的 3 种算法的可扩展性进行评估. 数据集大小的变化范围为 200 000～1 000 000,空间对象采用空间数据产生器随机产生. 实验结果如图 8 所示. 从图 8(a)和 8(b)可以看出,随着空间对象数量的增加,所有的算法的平均运行时间随之增加. 其中 GROA 的可扩展性明显优于 RE-Greedy 算法和 RII-Greedy 算法. 这是因为对象个数越多,需要计算的对象的空间距离就越多. 但总体上看,所有算法最长运行时间也不超过 2 s.



(a) 查询tag数=4　　(b) 查询tag数=8

图 8　空间对象数据集大小的影响

6　结束语

本文研究了路网环境下访问序列受限的多标签路线(MTROC)查询.形式化定义了 MTROC 查询问题,并依据增强的路线叠置-关联目录(EROAD)索引提出了 3 种查询处理算法:路线扩展 RE-Greedy 算法和路线渐增插入 RII-Greedy 算法.通过对路线局部更新获得满足用户需求的线路,而全局路线优化算法 GROA 提供了查询的全局最优近似解.本文使用真实和合成数据集对提出的 3 种算法性能进行分析评估,实验结果表明 GROA 算法的时间代价和可扩展性要优于 RE-Greedy 和 RII-Greedy 算法,但返回的路线质量相对较差.而 RII-Greedy 算法查询响应时间虽然偏长,但最坏情况也不超过 2 s,而且该算法可以返回高质量的最短路线.在未来的工作中,我们把工作扩展到动态路网中,在规划路线的过程中考虑实时交通信息(比如交通拥堵、访问时间等).此外,访问序列受限的多标签连续路线查询也是一个非常有挑战性问题.

参　考　文　献

[1] Hariharan R, Hore B, Li C, Mehrotra S. Processing spatial keyword (sk) queries in geographic information retrieval systems//Proceedings of the 19th International Conference on Scientific and Statistical Database Management. Banff, Canada, 2007: 16

[2] Felipe I D, Hristidis V, Rishe N. Keyword search on spatial databases//Proceeding of the 24th International Conference on Data Engineering. Cancún, México, 2008: 656-665

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

[4] Cao Xin, Cong Gao, Jensen C S, Ooi B C. Collective spatial keyword querying//Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data. Athens, Greece, 2011: 373-384

[5] Zhang Dong-Xiang, Chee Yeow-Meng, Mondal A, Tung Anthony K H. Keyword search in spatial databases: Towards searching by document//Proceedings of the 25th International Conference on Data Engineering. Shanghai, China, 2009: 688-699

[6] Li Fei-Fei, Cheng Di-Han, Hadjieleftheriou M, Kollios G, Teng Shang-Hua. On trip planning queries in spatial databases//Proceedings of the 9th International Conference on Advances in Spatial and Temporal Databases. Angra dos Reis, Brazil, 2005: 273-290

[7] Chen Hai-Quan, Ku Wei-Shinn, Sun Min-Te, Zimmermann R. The partial sequenced route query with traveling rules in road networks. Geoinformatica, 2011, 15(3): 541-569

[8] Kanza Y, Safra E, Sagiv Y, Doytsher Y. Heuristic algorithms for route-search queries over geographical data//Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. Irvine, CA, USA, 2008: 11

[9] Sharifzadeh M, Kolahdouzan M, Shahabi C. The optimal sequenced route query. The International Journal on Very Large Data Bases, 2008, 17(4): 765-787

[10] Baker KA, Fishburn P, Roberts FS. Partial orders of dimension 2. Networks, 1971, 2(1): 11-28

[11] Lee CK, Lee WC, Zheng Bai-Hua. Fast object search on road networks//Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology. Saint Petersburg, Russia, 2009: 1018-1029

[12] Jing Ning, Huang Yun-Wu, Rundensteiner EA. Hierarchical encoded path views for path query processing: An optimal model and its performance evaluation. IEEE Transactions Knowledge and Data Engineering, 1998, 10(3): 409-432

[13] Huang Yun-Wu, Jing Ning, Rundensteiner EA. Effective graph clustering for path queries in digital map databases//Proceedings of the 5th International Conference on Information and Knowledge Management. Rockville, Marvland, 1996: 215-222

[14] Kernighan BW, Lin S. An efficient heuristic procedure for partitioning graphs. Bell Systems Technical Journal, 1970: 49(2): 291-308

[15] Christos F, Stavros C. Signature files: An access method for documents and its analytical performance evaluation. ACM Transaction Information Systems, 1984, 2(4): 267-288

[16] Arora S. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. Journal of the ACM, 1998, 45(5): 753-782

[17] Arora S. Approximation schemes for NP-hard geometric optimization problems: A survey. Mathematical Programming, 2003, 97(1): 43-69

[18] Kahn AB. Topological sorting of large networks. Communications of the ACM, 1962, 5(11): 558-562

ZHANG Jin-Zeng, born in 1983, Ph. D. candidate. Her research interests include mobile object management, mobile Web search, and spatial data mining.

WEN Jie, born in 1989, M. S. candidate. Her research interest is mobile object management.

MENG Xiao-Feng, born in 1964, Ph. D., professor, Ph. D. supervisor. His research interests include Web data management, native XML databases, mobile data management, etc.

Background

The advancement of wireless location-acquisition technologies like GPS and Wi-Fi and mobile Web and the popularity of mobile devices, has contributed to a number of local search, online map services and geo-social networks service, such as Google Map, Bing Local Search, Foursquare, Flicker etc. Such services allow users to record location and activity information and mark them on the map. Meantime, the spatial objects are annotated using tags or tweets that provide rich semantic content. More importantly, users can search for spatial objects and route satisfying users' query needs.

In daily life, route planning query is a frequent laborious task which requires skilled interaction with a multiple of resources when users have multiple demands on the trip. Therefore, it is imperative for user to provide an optimal route with smallest travel cost and satisfying users' needs. In addition, visiting spatial objects is constrained by specifying users' preference and objects' permissions. This paper proposes a new type of route query, multi-tag route query based on order constraints (MTROC) in road networks. The goal is to find the shortest path that passes through at least one matching object per tag and satisfies order constraints. The research about spatial keywords query and route planning query is related to our work. Spatial keywords query takes a single location and a set of keywords as parameters and returns the spatial objects that match keywords. Existing route planning query method can find a route through multiple objects with respect to specify a set of spatial objects types. Furthermore, all above methods can not deal with constrained order of visit and cannot answer the MTROC query.

In this paper, the author proves that MTROC query is NP-hard and presents three approximate algorithms, the route extension greedy algorithm, the route incremental intersection algorithm and the global route optimistic algorithm. Extensive experiments on both synthetic and real-world datasets illustrate the efficiency and scalability of the proposed three algorithms.

This research was partially supported by the Natural Science Foundation of China (Nos. 60833005, 61070055, 91024032), the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China (No. 10XNI018), National Science and Technology Major Project of Key Electronic Devices, High-end General-purpose Chips and Fundamental Software Products (No. 2010ZX01042-002-003), Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 200800020002).