网络受限移动对象不确定性轨迹的索引*)

丁治明 余 波 李 曼 韩京宇
(中国科学院软件研究所 北京 100080)

摘 要 近年来,人们对于如何表示和处理移动对象的不确定性进行了研究,提出了一些较为有效的模型和算法。但是,在如何索引移动对象的不确定时空轨迹方面,相关的研究工作十分有限。为了解决上述问题,本文提出了一种网络受限移动对象不确定轨迹的索引结构(UTR-Tree),并给出了相关的索引更新及查询算法。在该索引结构的支持下,移动对象数据库不仅可以快速地处理对移动对象过去可能位置的查询,而且能够对其现在及将来的可能位置进行高效的查询处理。

关键词 移动对象数据库,不确定性,轨迹,索引

Indexing the Uncertain Trajectories of Network-constrained Moving Objects

DING Zhi-Ming YU Bo LI Man HAN Jing-Yu
(Institute of Software, Chinese Academy of Sciences, Beijing 100080)

Abstract The uncertainty management problem for moving objects databases has been studied recently, with many models and algorithms proposed. However, very limited work has dealt with the index of uncertain trajectories for a running moving objects database. In this paper, we propose a framework, UTR-Tree, for indexing the uncertain trajectories of network-constrained moving objects. By the support of UTR-Tree, the moving objects database can efficiently track and query not only the historical, but also the current and even near future locations of moving objects with uncertainty considered.

Keywords Moving objects database, Uncertainty, Trajectory, Index

1 引言

不确定性处理是移动对象数据库的一项重要研究内容。在移动对象数据库中,移动对象(如汽车、飞机、轮船、行人等)安装了全球定位系统(GPS),并通过无线通讯接口向数据库服务器报告自己的位置、速度、方向等信息(我们称该过程为"位置更新")。通过位置更新,数据库服务器能够动态地跟踪移动对象的位置,并处理相关的位置查询。由于位置更新操作是间断地进行的,在两次位置更新之间,服务器并不能准确地知道移动对象的确切位置,而只能通过位置更新消息对其可能位置进行推断。上述处理方式使得不确定性成为了移动对象数据库的一个无法避免的内在属性[1,2]。

近年来,人们对移动对象的不确定性处理进行了较为广泛的研究,并提出了一些有效的模型和算法。文[1]对移动对象不确定性的各种成因进行了分类与分析,在此基础上提出了一种不确定性数据的表示及建模框架;文[3]针对不确切的时态数据、空间数据和时空数据,提出了一种不确定性和模糊数据表示方法;文[2,4]讨论了在DOMINO移动对象数据库系统中的不确定性处理方法。通过在时间维和空间维上的扩充,移动对象的时空轨迹曲线被表示成一种管状结构,同时在查询操作中也引入了不确定性语义,如"possibly"、"sometimes"、"definitely"、"always"等;文[5]从抽象数据类型的角度,定义了一系列的数据类型及查询操作,用以对不确定性时空数据对象进行表示和处理。然而,所有上述工作均是直接基于$X\times Y\times T$欧氏空间的,没有针对交通网络的结构对不确定性处理进行优化。此外,这些工作针对的主要是历史轨迹数据,没有考虑移动对象动态变化的当前位置信息,因此降低了相关模型和算法的实用性。

在移动对象数据库中,移动对象的位置建模方式能够极大地影响其不确定性。在早期的直接基于欧氏空间的移动对象数据库模型中,移动对象在任一时刻的不确定位置是$X\times Y$平面中的一个区域。近年来,基于绝大部分移动对象是在相对固定的交通网络中运行的事实,人们逐渐认识到了网络受限移动对象模型的重要性,并对网络受限的移动对象不确定性处理进行了研究。文[6]对基于交通网络的移动对象不确定性轨迹进行了分析,通过合理的位置模型及位置更新操作,移动对象在任一时刻的可能位置被缩减成了一个线状的路段,从而极大地降低了移动对象的不确定性;文[7,8]提出了通过将GPS采样数据与道路网络进行预匹配来降低和校正GPS采样误差和不确定性的方法;文[9]在文[6]的基础上,进一步分析了网络受限移动对象的不确定性,并定义了适合于表示历史不确定轨迹的数据类型及查询操作。

在移动对象时空轨迹的索引方面,大部分研究工作均是基于欧氏空间的[10,11]。近年来,人们针对网络受限移动对象的索引进行了研究。文[12]提出了一种基于固定路网的移动对象索引FNR-Tree;文[13]对上述方法进行了改进,并提出了一种MON-Tree结构。但是所有上述方法均是基于历史轨迹的。由于历史轨迹属于静态数据,上述索引结构无法支

*)本文研究工作得到国家自然科学基金(项目编号:60573164)及教育部留学回国人员科研启动基金的资助。丁治明 博士,研究员,主要研究方向为数据库与知识库系统、移动计算;余 波、李 曼、韩京宇 博士,主要研究方向为数据库与知识库系统。

持对现在及将来位置的查询。文[14]对网络受限移动对象的未来轨迹索引进行了讨论，但没有讨论不确定性轨迹的索引问题。

通过上述分析可以看出，一方面，目前针对网络受限移动对象不确定性处理的绝大部分工作均集中在数据的表示上；而另一方面，在移动对象轨迹的索引方面，基本上没有考虑不确定性。在这两者的交集，即网络受限移动对象不确定轨迹的索引方面，目前尚无有效的方法。为了解决这一问题，我们在本文中提出一种针对网络受限移动对象数据库的不确定性轨迹 R 树索引结构(Uncertain Trajectory R-Tree，简称 UTR-Tree)，并给出相关的索引维护及查询算法。UTR-Tree 不仅可以支持移动对象过去可能位置的查询，而且可以支持对现在及将来可能位置的查询。

本文后续部分的结构如下：第 2 节给出相关的数据模型以及不确定性时空轨迹的采集方法，第 3 节给出 UTR-Tree 的结构及相关算法，第 4 节对系统实现进行阐述并给出相关结论。

2 网络受限移动对象模型及不确定性轨迹的采集

在本节中，我们首先给出交通网络及网络受限移动对象的数据库表示模型，然后阐述移动对象的不确定性轨迹及其采集方法，最后对移动对象的不确定轨迹进行分析。

定义 1(交通网络) 交通网络 G 定义为如下形式：

$G=(R, J)$

其中，R 是道路的集合，J 是交叉路口的集合。

定义 2(道路) 道路 r 定义为如下形式：

$r=(rid, geo, len, (jid_j)_{j=1}^{m})$

其中，rid 是道路标识；len 是道路的长度；$(jid_j)_{j=1}^{m}$ 是该条道路所包含的各个交叉路口的标识；geo 是该道路的地理几何形状，用一条折线(Polyline) pl 表示。pl 是由 $X\times Y$ 平面中的一组点所组成的序列，即

$pl=(x_i, y_i)_{i=1}^{n}(n\geqslant 2)$

其中，(x_1, y_1)为 pl 的起点，或称“0-端点”；(x_n, y_n) 为 pl 的终点，或称“1-端点”。

定义 3(交叉路口) 交叉路口 j 定义为如下形式：

$j=(jid, loc, ((rid_i, pos_i))_{i=1}^{n}, m)$

其中 jid 是交叉路口的标识；loc 是 j 的地理位置，用其(x, y)坐标表示；(rid_i, pos_i) $(1\leqslant i\leqslant n)$描述 j 所连接的第 i 条道路，其中 rid_i 是道路标识，$pos_i\in[0, 1]$是 j 在该条道路中的相对位置；m 是 j 的连接矩阵，用以描述该交叉路口的各交通流之间的连通关系[15]。

定义 4(网络位置) 交通网络 G 中的任意一个位置 $gpos$ 可用其所在的道路标识及其在该道路中的相对位置表示，即

$gpos=(rid, pos)$

其中，rid 是道路的标识，$pos\in[0, 1]$是 $gpos$ 在该道路中的相对位置。由于每条道路的几何形状都以折线的方式存放在数据库中(见定义 2)，上述表示方法可以很容易地转换为(x, y)的形式。

定义 5(移动对象的运行矢量) 移动对象 mo 在 t 时刻的运行矢量 mv 定义为如下形式：

$mv=(t, rid, pos, \vec{v})$

其中，t 是采集该运行矢量的时间，(rid, pos)是移动对象在 t 时刻的网络位置，$\vec{v}$ 是一个带符号的速度，其符号表示移动对象的行驶方向(由 0-端向 1-端行驶符号为“+”，反之为“-”)，其值表示移动对象的速度。

当移动对象在交通网络中行驶时，需要不断地将当前运行参数(如位置、速度、方向等)与上一次位置更新时所提交的运行矢量 $mv_n=(t_n, rid_n, pos_n, \vec{v}_n)$进行比较。一旦某些预先定义的位置更新条件满足时，就需要触发一次位置更新过程，将其最新的运行参数发送给服务器。

在网络受限的移动对象数据库中，文[15]定义了三种类型的位置更新，即道路 ID 触发的位置更新(IDTLU)、距离阈值触发的位置更新(DTTLU)以及速度阈值触发的位置更新(STTLU)。这三种位置更新同时作用，共同完成数据采集过程。

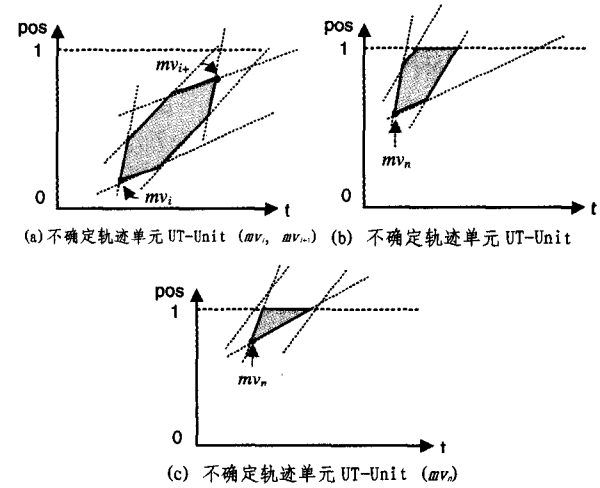
在上述位置更新过程中，DTTLU 和 STTLU 分别在移动对象超出距离阈值 ξ 和速度阈值 ψ 时进行触发，并分别产生一个运行矢量；IDTLU 在移动对象从一条道路 r_1 转换到另一条道路 r_2 时进行触发，并将产生三个运行矢量 mv_1，mv_2，mv_3，其中 mv_1 和 mv_2 分别对应于移动对象在通过交叉路口时的状态(分别以交叉路口在 r_1 和 r_2 中的位置表示)，mv_3 对应于移动对象在触发 IDTLU 时的状态。

定义 6(移动对象的时空轨迹) 移动对象 mo 的时空轨迹 $trajectory$ 是 mo 在行驶过程中通过位置更新操作所提交的一组运行矢量的序列，定义为如下形式：

$trajectory=(mv_i)_{i=1}^{n}=((t_i, rid_i, pos_i, \vec{v}_i))_{i=1}^{n}$

其中，mv_n 是移动对象最后一次所提交的运行矢量，我们称之为活动运行矢量。

如前所述，通过时空轨迹，我们仅能判断移动对象在数据采集点时刻(即 $t_i(1\leqslant i\leqslant n)$时刻)的准确位置。而在任意两个数据采集点之间或者最后一次数据采集点之后，移动对象的位置是不确定的，我们仅能根据相应的运行矢量计算出移动对象的可能位置。从这个意义上说，移动对象的时空轨迹所反映的是“不确定位置”，因此我们又称之为“不确定性时空轨迹”。为了叙述方便，本文中的时空轨迹均指不确定性轨迹。



(a)不确定轨迹单元 UT-Unit (mv_i, mv_{i+1}) (b) 不确定轨迹单元 UT-Unit

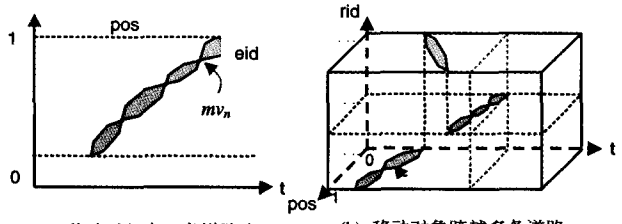
(c) 不确定轨迹单元 UT-Unit (mv_n)

图 1 移动对象的不确定性轨迹单元

在文[6, 9]中，Z. Ding 和 V. Almeida 等对通过移动对象的不确定性轨迹所导出的可能位置进行了分析。在任意两个连续的运行矢量 mv_i、$mv_{i+1}(1\leqslant i\leqslant n-1)$之间，根据触发位置更新的距离阈值 ξ、速度阈值 ψ，以及移动对象在 t_i 时刻的速度 $\vec{v}_i$，可以计算出移动对象的可能位置为 $POS\times T$ 平面中的一个六边形[6, 9]；在最后一个运行矢量(即活动运行矢量)

mv_n 之后，我们可以沿着 $\vec{v}_n$ 所指定的方向一直预测到道路端点(0-端点或1-端点)，这样移动对象的可能位置为一个五边形(也可能为三角形和四边形，取决于 mv_n 的值、距离阈值 ξ 以及速度阈值 ψ)。我们称上述代表移动对象可能位置的多边形结构为"不确定轨迹单元"(*Uncertain Trajectory Unit*，简称为 *UT-Unit*)，分别记为 UT-Unit(mv_i，mv_{i+1})和 *UT-Unit*(mv_n)，如图1所示。

从所代表的不确定性位置(即移动对象的可能位置)来看，移动对象的不确定性轨迹是一组不确定轨迹单元所组成的连续序列，如图2所示。



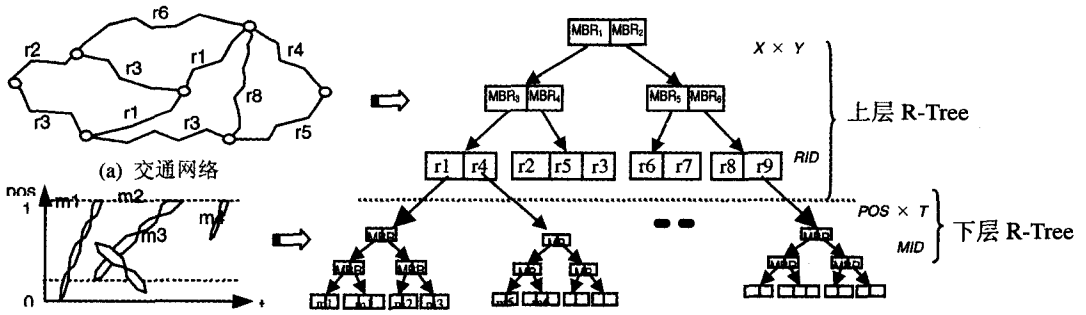
(a) 移动对象在一条道路内的不确定轨迹　(b) 移动对象跨越多条道路时的不确定轨迹迹

图2　移动对象的不确定性轨迹

3　移动对象不确定轨迹的索引

在本节，我们将提出一种能够对移动对象的不确定性轨迹(包括移动对象的过去、现在以及将来可能位置)进行索引的不确定轨迹R树结构(Uncertain Trajectory Based R-Tree，简称 UTR-Tree)，并给出相关的算法。UTR-Tree 的结构分为上下两层。其中上层为一个R树，用于对交通网络的道路数据进行索引；下层为一组独立的R树，每个R树与一条道路相对应，用于对各个移动对象在该条道路上提交的不确定性时空轨迹单元进行索引。图3给出了 UTR-Tree 的结构。

如图3所示，UTR-Tree 的上层R树是一个以道路为基本索引单位的标准R树结构。上层R树中间结点中的记录项为 $X\times Y$ 平面中的最小包容矩形(Minimum Bounding Rectangle，简称 MBR)，其叶子结点中的每个记录与一条具体的道路相对应，同时还含有一个指向下层R树的指针，该下层R树用于对移动对象在该条道路上提交的不确定轨迹单元进行索引。由于道路网络的拓扑结构相对固定，因此上层R树基本上属于静态结构。下面让我们来考察下层R树的结构。下层R树(设与之对应的道路为 r(标识为 rid))的基本索引单位是在 r 上行驶的移动对象在位置更新时所生成的不确定轨迹单元(UT-Unit)。下层R树中间结点中的记录项为 $POS\times T$ 平面中的 MBR，其叶子结点中记录项的结构为$\langle mid, mv_s, mv_e, mbr\rangle$，其中 mid 为移动对象的标识，$mv_s=(t_s, rid, pos_s, \vec{v}_s)$ 和 $mv_e=(t_e, rid, pos_e, \vec{v}_e)$ 分别为不确定轨迹单元对应的两个连续的运行矢量(对于活动运行矢量对应的轨迹单元 UT-Unit(mv_n)而言，mv_e 为空)，mbr 则是包含该轨迹单元的最小包容矩形。



(a) 交通网络　(b) 道路r1上各移动对象提交的轨迹片段　(c) 对应的 UTR-Tree 索引结构

图3　UTR-Tree 的结构

对于 UT-Unit(mv_s，mv_e)，根据网络受限移动对象的位置更新协议[15]可知，移动对象在 t_s，t_e 之间是沿着道路 r 单向地从 pos_s 向 pos_e 行驶的，即对于任一给定时刻 $t\in[t_s, t_e]$，移动对象的位置 $pos[t]$ 必然满足：$pos[t]\in[pos_s, pos_e]$。因此，UT-Unit(mv_s，mv_e)的 *MBR* 可以确定为 $mbr(mv_s, mv_e)=\langle pos_s, pos_e, t_s, t_e\rangle$，如图4(a)所示。

对于活动运行矢量 mv_n 对应的轨迹单元 UT-Unit(mv_n)，通过分析可知，移动对象在 t_q($t_n\leqslant t_q\leqslant t_{now}$)时刻的最大和最小可能位置 $pos_{q\,min}$ 和 $pos_{q\,max}$ 分别为[6](设 $\vec{v}_n>0$，即移动对象向1-端行驶。通过扩充，本文的方法能够处理 $\vec{v}_n<0$ 的情况)：

$$pos_{q\,min}=\max(pos_q^{\langle\rangle}-\xi,\quad pos_n+(\vec{v}_n-\psi)\times(t_q-t_n)) \tag{1}$$

$$pos_{q\,max}=\min(pos_q^{\langle\rangle}+\xi,\quad pos_n+(\vec{v}_n+\psi)\times(t_q-t_n)) \tag{2}$$

其中，$pos_q^{\langle\rangle}=pos_n+\vec{v}_n\times(t_q-t_n)$。

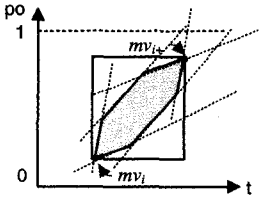
根据移动对象的位置更新协议可知，移动对象在最后一次位置更新之后是单向行驶的，因此必然有 $pos_{q\,min}>pos_n$。而对于 $pos_{q\,max}$，通过分析可知，它必然处于上述公式(3)和公式(4)所对应的直线 l_3 和 l_4 之上的区域(见图4(b)、(c))。

我们可以计算出 l_3、l_4 与直线 $pos=1$ 之间的交点 $\rho_\xi=(1, t_\xi)$和 $\rho_\psi=(1, t_\psi)$。因此 UT-Unit(mv_n)的 MBR 可以确定为

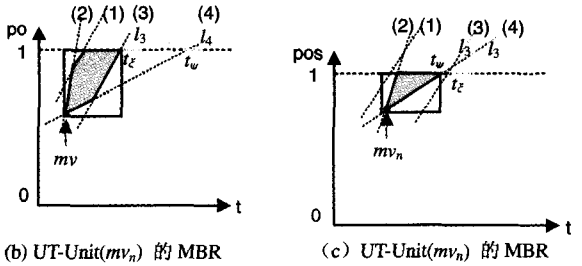
$$\mathrm{mbr}(mv_n)=\langle pos_n, 1, t_n, \min(t_\xi, t_\psi)\rangle$$

在移动对象数据库运行的过程中，系统需要首先扫描交通网络数据并建立上层R树索引。此时，上层R树叶子结点记录中指向下层R树的指针均为空。在移动对象运行并向服务器发送位置更新消息的过程中，系统将不断地向对应的下层R树插入所生成的 UT-Unit。由于活动运行矢量所对应的 UT-Unit 实际上包含了预测信息，因此在下一次位置更新时，需要以实际的 UT-Unit 和新的预测 UT-Unit 替换原来的预测信息。首先让我们来考察发生 DTTLU 和 STTLU 的情况。由网络受限移动对象的位置更新协议可知，在发生 DTTLU 和 STTLU 时，移动对象仍然在原先的道路上行进。

当服务器接收到一个新的位置更新消息 $mv_a=(t_a, rid_a, pos_a, \vec{v}_a)$时，如果 mv_a 是 rid_a 所对应的下层 R 树中该移动对象的第一个运行矢量，则直接将 UT-Unit(mv_a)插入该下层 R 树即可；否则需要对下层 R 树进行如下处理：(1)删除上一次位置更新时产生的 UT-Unit(mv_n)；(2)插入 UT-Unit(mv_n, mv_a)；(3)插入 Unit(mv_a)，如图 5 所示。

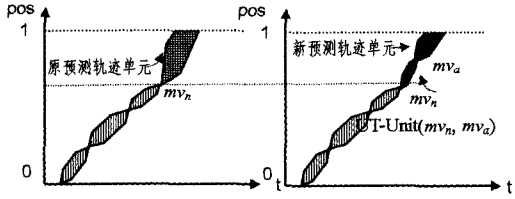


(a) UT-Unit (mv_s, mv_e) 的 MBR



(b) UT-Unit(mv_n) 的 MBR　　(c) UT-Unit(mv_n) 的 MBR

图 4　UTR-Tree MBR 的确定



(a) t_n时刻 UTR-Tree 中轨迹　　(b) t_s时刻 UTR-Tree 新轨迹

图 5　UTR-Tree 在发生 DTTLU 和 STTLU 时对下层 R 树的维护

当发生 IDTLU 时，处理过程要相对复杂一些。设移动对象从道路 $r1$ 转换到 $r2$，此时将产生 3 条位置更新消息：mv_{a1}, mv_{a2}, mv_{a3}。其中，mv_{a1} 和 mv_n 在 $r1$，mv_{a2} 和 mv_{a3} 在 $r2$，且 mv_{a1} 和 mv_{a2} 对应于移动对象在交叉路口时的状态。UTR-Tree 需要进行的操作如下：(1)在 $r1$ 对应的下层 R 树中删除 UT-Unit(mv_n)，并插入 UT-Unit(mv_n, mv_{a1})；(2)在 $r2$ 对应的下层 R 树中插入 UT-Unit(mv_{a2}, mv_{a3})；(3) 在 $r2$ 对应的下层 R 树中插入 UT-Unit(mv_{a3})。

下面给出 UTR-Tree 的建立及维护算法(见 Algorithm 1)。

Algorithm 1　UTR-Tree 的建立及维护算法

变量说明：G = (R, J)；//交通网络
1. 扫描交通网络中的道路集合 R，建立 UTR-Tree 的上层 R 树；
2. 将上层 R 树叶子结点记录中的指针全置为空；
3. While(MOD is running)Do
4. 接收位置更新消息 LUM（设发送消息的移动对象标识为 mid)；
5. If(LUM 仅包含 1 个运行矢量 $mv_a=(t_a, rid_a, pos_a, \vec{v}_a)$) Then
//LUM 是 DTTLU 或 STTLU 产生的
6. $RTree_{low}$：=rid_a 对应的下层 R 树；
7. If ($RTree_{low}$=NULL) Then $RTree_{low}$：=创建一个新的下层 R 树；Endif；
8. mv_n：=$RTree_{low}$中移动对象 mid 的活动运行矢量；
9. If (mv_n=NULL) Then
10. 将(mid, UT-Unit(mv_n),mbr(mv_n)) 插入 $RTree_{low}$；
11. Else
12. 删除 $RTree_{low}$中的 (mid, UT-Unit(mv_n), mbr(mv_n))
13. 将(mid, UT-Unit (mv_n, mv_a), mbr(mv_n, mv_a)) 插入 $RTree_{low}$；
14. 将(mid, UT-Unit(mv_a), mbr(mv_a)) 插入 $RTree_{low}$；
15. Endif；
16. Else If(LUM 包含 3 个运行矢量 mv_{a1}, mv_{a2}, mv_{a3}) Then//LUM 是 IDTLU 产生的
17. $RTree_{low1}$：=rid_{a1}对应的下层 R 树；
18. If ($RTree_{low1}$=NULL) Then
$RTree_{low1}$：=创建一个新的下层 R 树；Endif；
19. mv_n：=$RTree_{low1}$中移动对象 mid 的活动运行矢量；
20. 删除 $RTree_{low1}$中的 (mid, UT-Unit(mv_n), mbr(mv_n))；
21. 将(mid, UT-Unit(mv_n, mv_{a1}), mbr(mv_n, mv_{a1})) 插入 $RTree_{low1}$；
22. $RTree_{low2}$：=rid_{a2}对应的下层 R 树；
23. If($RTree_{low2}$=NULL)Then
$RTree_{low2}$：=创建一个新的下层 R 树；Endif；
24. 将(mid, UT-Unit(mv_{a2}, mv_{a3}), mbr(mv_{a2}, mv_{a3}))插入 $RTree_{low2}$；
25. 将(mid, UT-Unit(mv_{a3}), mbr(mv_{a3}))插入 $RTree_{low2}$；
26. Endif；
27. Endif；
28. Endwhile.

对 UTR-Tree 的查询分为两步进行。在移动对象数据库中，最常用的不确定性查询操作，如 possibly-inside ($trajectory$, $\Delta x\times\Delta y\times\Delta t$)，possibly-intersect($trajectory$, $\Delta x\times\Delta y\times\Delta t$)等，均属于区域查询($Range\ Query$)，即查询的输入为 $X\times Y\times T$ 空间中的一个立方体区域。在对区域查询进行处理时，系统将首先根据 $\Delta x\times\Delta y$ 查询 UTR-Tree 的上层 R 树，得到一组($rid\times period$)($period\in[0, 1]$且可有多个元素)偶对；然后对每一个($rid\times period$)偶对，在对应的下层 R 树中查询与 $period\times\Delta t$ 相交的轨迹单元，并输出相应的移动对象标识。图 6 给出了对 UTR-Tree 查询的过程，完整的查询算法如 Algorithm 2 所示。

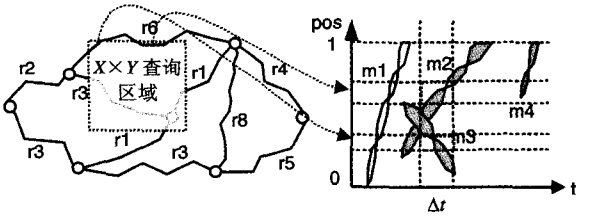


图 6　UTR-Tree 的区域查询过程

Algorithm 2　UTR-Tree 的区域查询算法

输入：查询区域 $\Delta x\times\Delta y\times\Delta t$；
输出：$Result$；//mid 的集合
1. 根据 $\Delta x\times\Delta y$ 查询上层 R 树，得到一组偶对 $P=(rid_i, period_i)_{i=1}^{n}$；
2. For(i:=1 to n) Do
3. $RTree_{low}$：=rid_i 对应的下层 R 树；
4. 根据 $period_i\times\Delta t$ 查询 $RTree_{low}$中的 UT-Unit 单元；
5. If (UT-Unit$\cap$($period_i\times\Delta t$)$\neq\varnothing$)Then
6. $Result$←UT-Unit 单元对应的 mid；
7. Endif；
8. Endfor；
9. Return Result.

4　系统实现及结论

不确定性处理是移动对象数据库的关键技术之一，近年来得到了较为广泛的研究。然而，由于缺乏有效的不确定性轨迹索引方法，现有的移动对象不确定性处理基本上是在操作一级实现的。由于要在查询处理的过程中逐个对所有的移动对象进行计算，因而极大地影响了查询处理的效率，降低了各种移动对象不确定性模型的实用性。

本文提出的 UTR-Tree 是一种适用于网络受限移动对象的不确定轨迹索引结构，能够支持各种不确定性的查询操作，如 possibly_inside，possibly_intersect，definitely_inside，definitely_ intersect 等。目前，我们已经研发了基于动态交通网络的移动对象数据库(Net-MOD)原型系统，并在 Net-MOD 的基础上对 UTR-Tree 进行了初步实验。实验结果表明，UTR-Tree 能够获得与普通轨迹索引方法相当的查询性能

(略逊于 MON-Tree,但优于 FNR-Tree)。

将来的研究工作包括对 UTR-Tree 做进一步的优化,并将在 Net-MOD 原型系统的基础上进行更为详细的实验比较与分析。此外,我们还将设计完整的不确定性数据类型和查询操作,并给出相应操作在 UTR-Tree 支持下的实现算法。

参 考 文 献

1 Pfoser D, Jensen C S. Capturing the Uncertainty of Moving Object Representations. In:Proc. of SSD'99, Hong Kong, China, July 1999

2 Trajcevski G, Wolfson O, Chamberlain S, et al. The Geometry of Uncertainty in Moving Objects Databases. In: Proc. of EDBT'02, Prague, Czech Republic, 2002

3 Pfoser D, Tryfona N. Capturing Fuzziness and Uncertainty of Spatiotemporal Objects. In: ADBIS 2001, Vilnius, Lithuania, Sept. 2001

4 Trajcevski G, Wolfson O, Cao H, et al. Managing Uncertain Trajectories of Moving Objects with Domino. In: ICEIS'02, Spain, April 2002

5 T∅ssebro E, Nygärd M. Uncertainty in Spatiotemporal Databases. In:Proc. 2nd Biennial Int Conf. on Advances in Information Systems (ADVIS), Turkey, Oct. 2002

6 Ding Z, Güting R H. Uncertainty Management for Network Constrained Moving Objects. In: Proc. of DEXA'2004, Zaragoza, Spain, August 2004

7 Gowrisankar H, Nitte S. Reducing Uncertainty in Location Prediction of Moving Objects in Road Networks. In: Proc. of GIScience 2002,Colorado, 2002

8 Meratnia N, Kainz W, et al. Spatio-temporal Methods to Reduce Data Uncertainty in Restricted Movement on a Road Network. In: Proc. of joint ISPRS, IGU and CIG Symp on Geospatial theory, Processing and Applications, Canada, 2002

9 Almeida V T, Güting R H. Supporting Uncertainty in Moving Objects in Network Databases. In: Proc. of the 13th Intl Workshop on Geographic Information Systems (ACM-GIS), Bremen, Germany, 2005

10 Pfoser D, Jensen C S, Theodoridis Y. Novel Approach to the Indexing of Moving Object Trajectories. In: Proc. of the 26th VLDB,Cairo, Egypt, 2000

11 Saltenis S, Jensen C S, Leutenegger S T, et al. Indexing the Position of Continuously Moving Objects. In: Proc. of ACM SIGMOD 2000, TX, USA, 2000

12 Frentzos E. Indexing objects moving on fixed networks. In: Proc the 8th Int Symp Spatial and Temporal Databases, Santorini Island, Greece, 2003

13 Almeida V T, Güting R H. Indexing the Trajectories of Moving Objects in Networks. GeoInformatica,2005,9(1)

14 Chen J*, Meng X. Indexing Future Trajectories of Moving Objects in a Constrained Network. Journal of Computer Science and Technology,2007,22(2): 245～251

15 Ding Z, Güting R H. Managing Moving Objects on Dynamic Transportation Networks. In: Proc. of the 16th International Conference on Science and Statistical Database Management (SSDBM'2004), Santorini, Greece, June 2004

(上接第 67 页)

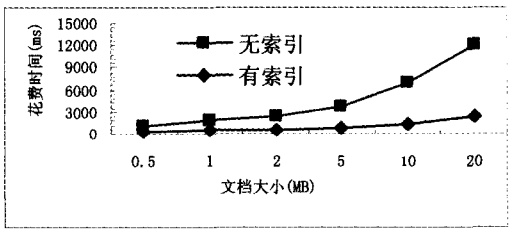


图 8 混合查询 Q3 在有无索引时花费时间对比

文档中地理数据涵盖了道路、河流、湖泊、城市、省区、水域、排水等,涉及到的几何对象包括 Point、LineString、LinearRing、Box、Polygon、MultiLine、MultiPolygon 等。图 6、图 7 和图 8 分别给出了这三种查询在无索引和有索引时所花费的时间对比。

从图 6～8 可以看出,非空间查询 Q1 由于不涉及空间运算,查询耗时最短,与无索引时相比,建立 B^+-树索引后,查询效率有一定程度的提高;空间查询 Q2 在建立 R 树索引后,查询效率有显著提高;混合查询 Q3 先经过非空间条件过滤后,查询时间明显比 Q2 有所下降,查询效率也比无索引时有显著提高。

结束语 本文提出一种基于区间编码的 GML 文档索引与查询方法,将 GML 文档树中结点按前序遍历中第一次和最后一次被访问时的次序进行编码。对元素、属性、文本结点以 B^+-树方式来组织索引,以提高值查询和结构查询的查询速度;对几何体结点按 R-树方式组织索引,以便提高空间查询和分析效率。查询语言采用基于 XQuery 的 GML 查询语言 GQL,并对三种查询(非空间查询、空间查询及混合查询)进行了分析与比较。实验证明,本文所提出的 GML 文档编码方案和索引机制是可行的,能够有效地处理在 GML 文档上进行的值查询和空间分析操作。

参 考 文 献

1 Cox S, Daisey P, Lake R, Portele C, Whiteside A. Geography Markup Language (GML) Implementation Specification version 3.0 [S], OpenGIS Consortium, 2003

2 Córcoles J E, González P. Analysis of Different Approaches for Storing GML Documents [C]. In: Proceedings of ACM GIS'02, 2002. 11～16

3 Sripada L N, Lu C, Wu W. Evaluating GML Support for Spatial Databases. In:COMPSAC Workshops, 2004. 74～77

4 Zhu F, Guan J, Zhou J, Zhou S. Storing and Querying GML in Object-Relational Databases [C]. In: Proceedings of ACM GIS'06,2006. 107～114

5 Corcoles J E, Gonzalez P. A Specification of a Spatial Query Language over GML [C]. In: Proceedings of ACM GIS'' 01, 2001. 112～117

6 Vatsavai R R. GML-QL: A Spatial Query Language Specification for GML. http://www. cobblestoneconcepts. com/ucgis2summer2002/vatsavai /vatsavai. htm, 2002

7 Guan J, Zhu F, Zhou J, Niu L. GQL: Extending XQuery to query GML documents [J]. Geo-Spatial Information Science, 2006,9(2):118～126

8 Dietz P F. Maintaining Order in a Linked List [C]. In: Proceedings of the Annual ACM Symposium on Theory of Computing, 1982. 122～127

9 Li Q, Moon B. Indexing and Querying XML Data for Regular Path Expressions [C]. In: Proceedings of VLDB'01, 2001. 361～370

10 Zhang C, Naughton J, DeWitt D, Luo Q, Lohman G. On Supporting Containment Queries in Relational Database Management Systems [C]. In: Proceedings of SIGMOD'01, 2001. 426～437