申请上海交通大学学士学位论文

相似轨迹查询方法设计与实现

论文作者 戚　文韬

学　　号 5130309593

导　　师 朱燕民教授

专　　业 计算机科学与技术专业

答辩日期 2017 年 5 月 15 日

上海交通大学
学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：某某

日　期：某年某月某日

上海交通大学
学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保　密□，在 _____ 年解密后适用本授权书。

本学位论文属于

不保密☑。

（请在以上方框内打✓）

学位论文作者签名：某某　　指导教师签名：某某

日　期：某年某月某日　　日　期：某年某月某日

相似轨迹查询方法设计与实现

摘　要

待续

关键词： 上海交大　饮水思源　爱国荣校

A Sample Document for LaTeX-basedd SJTU Thesis Template

ABSTRACT

TBC

KEY WORDS: SJTU, master thesis, XeTeX/LaTeX template

目 录

第一章　绪论

现代社会地理位置获取和移动计算科技进步，促使轨迹数据的大规模发展。这些轨迹数据体现了例如人类，车辆以及动物等移动物体的移动多样性。在过去十几年间，许多旨在处理、管理和挖掘轨迹数据的算法与技术许多应用中有着广泛而重要的应用价值。如今以轨迹数据挖掘为首的轨迹数据处理技术已经日趋系统且规范，从轨迹数据生成，到轨迹数据预处理，再到轨迹数据管理，最后到多样的数据挖掘任务（例如轨迹模式挖掘、轨迹异常检测、轨迹分类等等）。已有轨迹处理和轨迹挖掘的技术在相互应用中有着重要的联系与关联，轨迹数据转化成其他轨迹形式，例如图、矩阵和张量的方法也在越来越多的轨迹数据挖掘和机器学习领域有着常见的应用。

轨迹从概念上定义是一个移动物体的移动轨迹，轨迹数据可以用于许多领域的复杂分析。例如，公共交通系统可以应用过去时刻的轨迹数据分析交通流量模式并找出致使交通拥塞的原因；生物领域的动物长途迁移轨迹或是短途移动变化可以为人类提供宝贵的数据分析人类活动对生态环境的影响程度；还可以通过分析数据预测城乡车辆移动情况并及时提供符合公众出行的公共交通支持。其他应用领域也包括了路径优化设计，公共交通安全管理和基于兴趣点的用户个性化服务。

基于以上应用情景，轨迹数据挖掘在计算机科学、社会学和地理学领域都变得愈发重要。在轨技数据挖掘领域研究从深度和广度都已经取得了不错的成果，从图1–1可以看出当前轨迹数据挖掘与处理的基本研究步骤。本课题相似轨迹查询方法设计与实现主要基于其该范例中的轨迹预处理与轨迹数据索引与获取这两个领域中已存在的方法，并结合自己的理解和数据的格式实现改善和创新。

1.1　相似轨迹查询

大量空间轨迹数据为我们提供了分析移动物体移动方式的可能性，这种移动方式的分析可以体现出单个轨迹所包含的某种特定移动方式或是一组轨迹所共享的相似移动方式。通常情况下相似轨迹查询是基于时空关系的查询，除此之外有些情况下一些相似轨迹查询会增加特定的查询条件，例如最快速度、偏移方向或是在规定的时间段内经过特定地理区域等等条件。在相似轨迹查询中缺少时间维度参数 (时间戳或是时间段) 是可以接受的，加入时间参数的相似轨迹查询本文将他们视为其中的一种特殊情况处理。

1.1.1　轨迹查询概念简介

完成在轨迹数据库中复杂的轨迹查询操作是复杂且费时的操作，因为轨迹数据库的规模一般是非常庞大的。因此，轨迹数据库的一个重要点事支持高效的轨迹索引以加速轨迹插叙过程。通常情况下，时空数据的索引技术是空间数据索引辅以时间度量参数。轨迹查询既关注经过的地理位置的拓扑位置顺序，也关注空间物体之间的距离度量，从简单的欧式距离度量到复杂的轨迹之间相似性。从大体上说，如今的轨迹查询依照时空关系分为三类:1）$P\text{-}query$，查询满足特定轨迹段或者时空关系的兴趣点或者查询针对某些兴趣点满足时空关系的轨迹；2）$R\text{-}query$，根据给定的时空区域查询

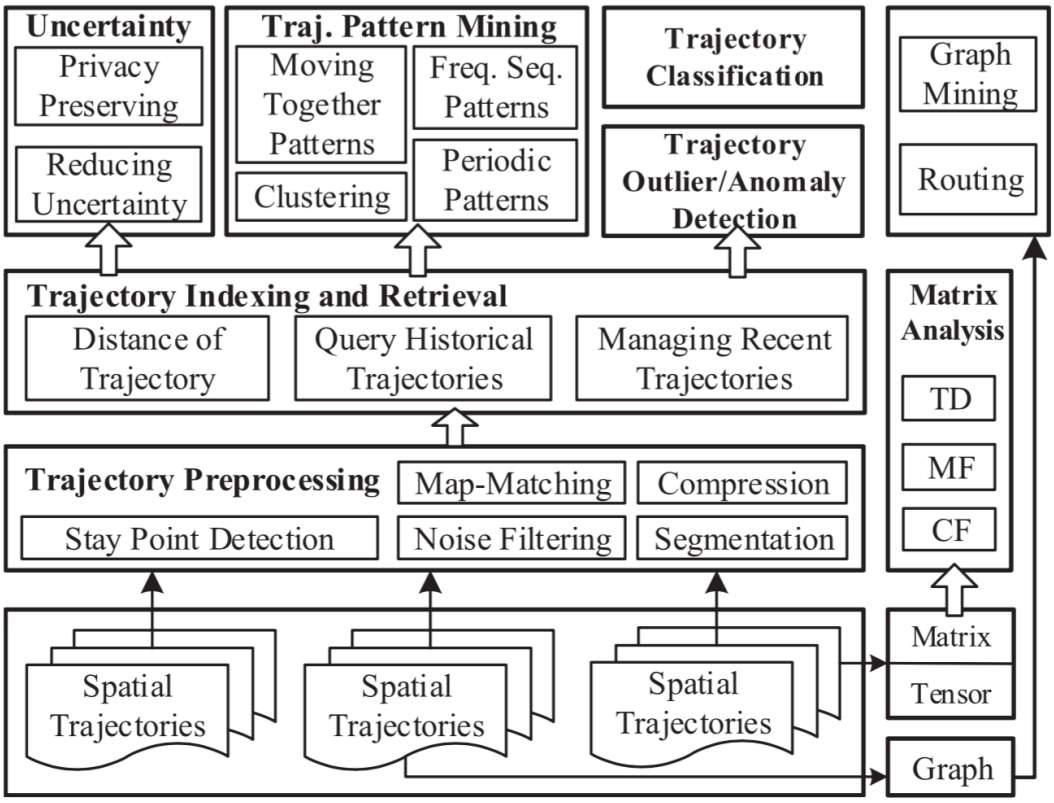

图 1–1 轨技数据挖掘范例

Fig 1–1 Paradigm of trajectory data mining

轨迹或者给定轨迹查询目的区域，3）T-$query$, 查询在一组轨迹数据集中查询相似轨迹或在给定的距离阈值内查询轨迹。

1.1.2 相似轨迹查询应用现状

相似轨迹查询主要是基于上述的轨迹查询方法中 P-$query$ 和 T-$query$ 展开的。

P-$query$ 主要应用在给定地理位置点后找到满足时空关系的轨迹或者轨迹段。单点轨迹查询找到针对某一给定地理位置点的最近轨迹。多点轨迹查询在给定一组地理位置点集后在轨迹数据里中找到能在地理位置意义上连接查询点集的多条轨迹。前者用以找到某一地理位置范围内的潜在轨迹。后者在制定行进轨迹路线中有着很好的应用。T-$query$ 通常通过聚类或分类轨迹在轨迹数据库中查询轨迹。轨迹分类和聚类算法在许多应用中有着广泛的应用，例如基于移动物体特征的轨迹测或是分析路网流量结构，在轨迹集合发现共同的子轨迹以及查询与目标轨迹在欧式距离上最接近的轨迹集合。

基于 P-$query$ 的查询主要是衡量点到轨迹的中最近点的距离。目前也常通过拓展这一思路当多点的 P-$query$ 查询以评价一条轨迹连接多个查询点的好坏。在 T-$query$ 这一查询类型方面则有很多

较为成熟的方法主要的不同在于他们各自的相似距离函数的定义，例如动态时间规划轨迹方法 (Dynamic Time Warping)[ref]、最长公共子序列方法 (Longest Common Subsequence)[ref]、基于编辑代价的方法 (Edit Distance With Real Penalty)[ref] 和基于序列编辑距离的方法 (Edit Distance on Real Sequences) 等等。这些方法在初期主要应用于时序相关的数据上，但是由于轨迹在某种意义上可以看成是多维度上的时序数据，上述的相似距离方法则可以应用上轨迹数据上。

1.1.3 相似轨迹查询方法设计概述

本文研究的相似轨迹查询方法主要基于位置定的查询，即查询主要是基于一组有序或是无序的地理位置点。查询的首相要目标是从轨迹数据库中找出连接查询位置点的 k 条最佳连接轨迹 (K Best-Connected Trajectories) 使得这 k 条轨迹能够在地理位置上连接给定的未指定。不同于传统形状或其他查询标准通过给定一条轨迹的相似轨迹查询，本文的相似轨迹查询主要针对于所查找到的轨迹对于给定的一组轨迹点连接性的优劣。

如图1–2所示，通过点击地图或图像地理解码给定一组地理位置点 (图中点注释)，我们可以从数据库中获取找一条能够连接给点地理位置点原始轨迹 (图中线注释)，该实例体现出本相似轨迹查询方法在能够在包括旅游路线规划等新兴应用中更好地服务用户。与此同时，这种相似轨迹查询能还能在以上场景有所应用：旅行社或自由行游客对出行经典的路径规划；动物园能调查出动物对到某些特定地点的最短路径；交通运输部门对本地市民城乡情况的规划。

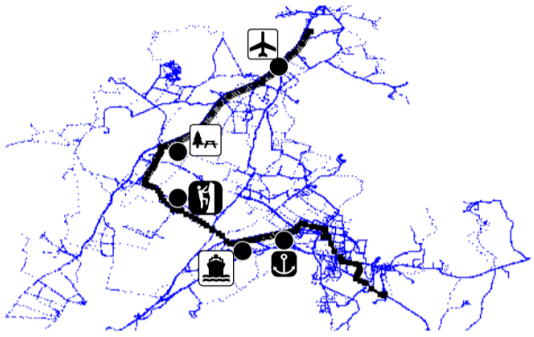

图 1–2 基于位置点的相似轨迹查询
Fig 1–2 Similar trajectory search by locations

大体上，k 条最佳连接轨迹查询基于的地理位置点需要具备必要的经纬度信息 $(latitude, longitude)$。这些经纬度位置地点可以是旅游景点，不明确的沙滩或是任何一处地理坐标。用户可以通过决定轨迹连接有序或是无序以决定查询结果。例如一个简单的查询包括三个地理位置点 A,B,C

$$A_{(37.2601,122.0941)}, B_{(37.2721,122.0648)}, C_{(37.3344,122.1538)}$$

其中 (37.2601, 122.0941) 代表 A 点的经纬度地理坐标。如果查询附带有序条件，则查询轨迹结果应保留轨迹点之间的相对顺序性，即 $A \rightarrow B \rightarrow C$。传统的相似轨迹查询方法显然无法解决查询

结果与查询条件之间有序一致性。另外，本文所提出的轨迹查询方法基于任意的地理位置点使得查询模式更加多样和灵活。从本质上而言，这种这种查询方法的设计基于传统的单点查询方法以寻找针对某一位置的最为临近轨迹。

为了实现这一相似轨迹查询方法，本文首先对一组给定地理位置点中的每一个点进行基于点的查询以从数据库中找到最近的轨迹点。如果查询结果存在，通过对每一个结果的汇总所得出的轨迹从理论上而言是最接近查询点的一条轨迹且对给定的地理位置点具有良好的连接性。从这个思路出发，本文拓展 K 最近邻 (k-NearestNeighbor algorithm) 算法并提出增长性 K 最近邻 ($Incremental\ k\text{-}NN\ algorithm$) 算法。该算法增长性获取每个查询位置点的最近轨迹并不断检查以查询到的轨迹。通过自定义的轨迹相似性上界与下界，借鉴备选和筛选的算法设计思路 (candidate-refinement) 来进行优化与剪枝。利用 R 树 (R-Tree) 作为地理位置点的索引结构，算法实现过程中根据具体计算机情况和性能需求选择性使用最好优先搜索 (best-first) 方法或是深度优先搜索 (depth-first) 方法进行查询。

1.2 论文大致结构

本毕业论文主要结构为：本章介绍轨迹查询大致概念与相似轨迹查询现状与方法大体设计；第二章介绍实现相似轨迹查询方法设计的相关工作；在第三章详细讨论算法实现细节与优化问题处理；第四章中讨论相似轨迹查询和分布式结合具体过程与算法实现；实验过程和结果会在第五章中予以描述并在第六章为本文做结论。

1.3 本章小结

1.1.1、1.1.2 和 1.1.3 三节内容已经初步介绍了相似轨迹查询这一概念和其相关背景，由于目前的在轨迹数据处理已经系统和规范的处理流程，轨迹数据挖掘这一领域的方法技术也已经较为成熟且丰富，通过学习传统的相似轨迹查询方法和他们各自的应用经验，本文所提出的方法在已有成果的基础上进行进一步的创新与优化，便可以使得相似轨迹查询方法与传统的相似轨迹查询有着较大的不同，且具有特定查询环境上的查询优势与性能优化。

本文通过拓展如今最为基本的 k 最近邻数据挖掘技术，以实现基础的相似轨迹查询方法为基本理论目的，移植单机运行代码至分布式环境系统为应用目标，开展毕业设计课题。

第二章　相关工作

2.1　相似方程定义

相似轨迹查询工作在某种意思上和时序数据相似查询共享一些方法定义。相似轨迹查询的首要步骤通过某种选定轨迹与轨迹点间的距离度量来定义相似度 (或称为距离) 方程，之后是设计高效的查询过程算法来解决从大规模数据库中找到符合要求的备选轨迹。定义相似度方法在过去有许多深度的讨论，之前的工作有通过利用离散傅里叶变化（Discrete Fourier Transform）[Ref bylocation2] 将轨迹数据转化为多维空间上的点，任何通过比较这些点在特征空间上的欧式距离来比较轨迹数据时间的相似性。之后有科研人员在此工作成果的基础上通过改善实现子轨迹的匹配查询，并验证了离散小波变换（Discrete Wavelet Transform）的可行性。切比雪夫多项式（Chebyshev polynomials）在轨迹近似和索引方面有被证明是可应用的。但是这些方法的前提调前是需要轨迹上时序上具有相同的长度，因此这些转变返程所提供的相似度方法不适用与本文所提出的相似轨迹查询。

Definition			
DTW(R,S)	=	0	if $m = n = 0$
		∞	if $m = 0$ or $n = 0$
		$dist(r_1, s_1) + min\{DTW(Rest(R), Rest(S)),$ $DTW(Rest(R), S), DTW(R, Rest(S))\}$	otherwise
ERP(R,S)	=	$\sum_1^n dist(s_i, g),\ \sum_1^m dist(r_i, g)$	if $m = 0$, if $n = 0$
		$min\{ERP(Rest(R), Rest(S)) + dist(r_1, s_1)$ $ERP(Rest(R), S) + dist(r_1, g),$ $ERP(R, Rest(S)) + dist(s_1, g)\}$	otherwise
LCSS(R,S)	=	0	if $m = 0$ or $n = 0$
		$LCSS(Rest(R), Rest(S)) + 1$	if $\forall d, \lvert r_{d,1} - s_{d,1} \rvert \leq \epsilon$
		$\max\{LCSS(Rest(R), S), LCSS(R, Rest(S))\}$	otherwise
EDR(R,S)	=	$n,\ m$	if $m = 0$, if $n = 0$
		$\min\{EDR(Rest(R), Rest(S))$+subcost, $EDR(Rest(R), S) + 1, EDR(R, Rest(S)) + 1\}$	otherwise

图 2–1 相似度函数定义[1]

Fig 2–1 Definition of distance functions

图2–1是典型且常用的相似度方程定义。这些方程根据自身特点与优势应用于不同的场景中，包括欧氏距离方程（Euclidean Distance），动态时间规整（Dynamic Time Warping），最长公共子序列算法（Longest Common Subsequence），基于编辑代价的方法（Edit Distance With Real Penalty）和基于序列编辑的距离方法（Edit Distance on Real Sequences）。动态时间规整（DTW）方法在比较轨迹之间相似性的过程中采用了时间偏移（time-shifting）来使得轨迹中的一些点可以尽可能多地重复出现以实

[1] $dist(ri, si)$ = L1 or L2 norm; $subcost = 0$ if $r1, t{-}s1, t$, else $subcost = 1$

现最好效果的校准。但这种方法在原有轨迹数据点出现误差（或称为噪声点）的时候会影响比较的准确性因为所有的点都需要被匹配。相比如动态时间规整方法（DTW），最长公共子序列（LCSS）选择忽略某些点以避免对他们的重排序过程，从结果上而言这种方法舍弃了偏离采样的误差点以提高了准确性，但需要人为预定距离阈值以判断什么数据属于误差点。基于编辑代价的距离方法（EDR）与 LCSS 方法类似，他们最初提出是为了解决字符串匹配问题，在轨迹数据匹配这一方面他们均采用一个阈值参数来判断两个点是否匹配，但 EDR 考虑了距离之间的衡量代价以决定是否将两个点进行匹配。在此基础上基于序列编辑的距离方法（ERP）结合 EDR 和 DTW 方法选择固定点进行距离计算。

相似度方法通常根据具体的应用进行具体的选取。但上述的相似度方法主要是基于轨迹与轨迹之前相似度的查询，在本文设计的相似轨迹查询方法上的应用度并不理想，本工作的查询条件是基于一组地理坐标点的查询，并且工作更关注与一条轨迹是否能够很好地连接上给定的一组查询点，从而提供基于轨迹点的相似轨迹结果。因此，在这样的情景下，我们需要定义一个新的相似度方程。

2.2 轨迹数据预处理

2.2.1 基于地理位置的轨迹简化方法

轨迹简化在本文中是比较重要的预处理过程。在保证轨迹数据可用性的同时也能压缩了原始轨迹数据的大小，可以在存储空间和搜索时间给予算法改善。在本文中，主要采用语义相关的轨迹简化算法，称为 $TS\ algorithm$（TS），它主要思路一条轨迹中重要的或具有特定语义的轨迹点从而进行压缩。TS 轨迹简化算法既保证了轨迹的大致路线也保留了重要的特定轨迹点。

该算法2–1主要分为四个过程：分段、段权值排序、点权值计算和点选择。由于本工作数据集只针对车载轨迹数据，则在第一步只用进行简单的分段操作而不用进行轨迹类型分类操作。而第二步根据设定参数求出每一段的权值并通过算法定义参数保留段轨迹语义。第三步计算每个数据点的权值并通过正则化决定每一段子轨迹内选择保留相对比例的数据点。最后选择符合算法条件的点作为简化结果返回。

算法 2–1 轨迹简化（Trajectory Simplification）算法

输入: 一条原始轨迹数据 $Traj$, 简化后轨迹点数目 m

输出: 一条简化后只有 m 个点的轨迹 $Traj'$

1: $Traj' \leftarrow \emptyset$
2: $Seg[] \leftarrow Segmentation(Traj)$ //将轨迹 Traj 分段
3: DistributePoints(Seg[], m); //求段权值并参数初始化
4: **for** Each Segment s in $Seg[]$ **do**
5: WeightPoints(s) //求点权值
6: $s' \leftarrow$ SelectPoints(s) //选择点组成新的轨迹段
7: $Traj' \leftarrow Traj \cup s'$ //合并新轨迹段组成结果
8: **end for**

2.3 轨迹数据索引与获取

空间数据结构对从一个大规模轨迹数据集中获取特定轨迹数据是十分重要的。效率问题是查询大规模数据库或数据集来获取信息的首要考虑因素。而查询效率十分依赖于合理的轨迹索引。轨迹数据根据数据特点的不同姓对索引技术也有着特殊的要求。目前主流的索引技术主要有三类：1）基于空间维度的索引，利用 R 树（R-tree）索引进行查询。通过 3DR 树（3D R-tree）或者 STR 树（STR-tree）进行带有时间维度的查询；2）利用多版本的数据结构，根据特定情况使用 MR 树（MR-tree）、HR 树（HR-tree）、MV3R 树（MV3R-tree）等等；3）将空间划分网格结构然后对应每个网格建立对应的空间索引，这类数据结构包括 MTSB 树（MTSB-tree）和 SETI。本文中我们使用 R 树这一最基本的数据结构，其满足我们对算法的实现需求。

R 树数据结构在空间数据库中应用广泛，许多轨迹索引结构大体上是基于 R 树进行拓展。R 树结构是一个平衡树结构，R 树中的每一个节点代表包含其所有子节点一个区域，这个区域通常被称为最小区域箱（Minimum Bounding Box）。节点中的每一个数据体指向对应的子节点的最小区域箱信息。R 树搜索的关键字是最小区域箱中的每一个节点。如图2–2所示的是 R 树数据结构的 2 种表现形式。在2–2(b) 中我们看到树结构而图2–2(a) 描述了数据和最小边界箱是如何分布在空间中的。

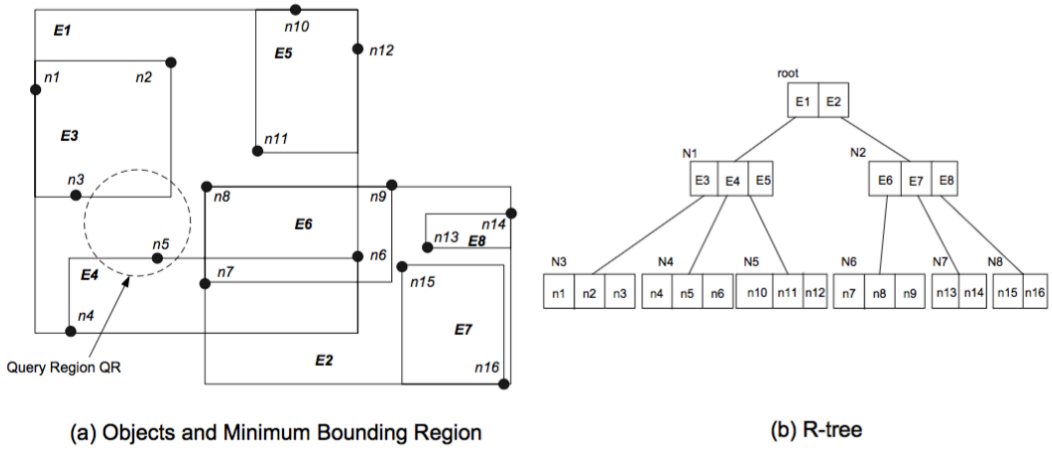

图 2–2 R 树数据结构举例

Fig 2–2 Two views of an R-tree example

在图2–2中，根节点有两个数据体 $E1$，$E2$，分别对应子节点 $N1$，$N2$。$N1$ 代表的最小边界箱包含了其子节点 $N3$、$N4$、$N5$ 以及数据体 $E1$ 所具有的最小边界箱信息。值得注意的是空间点的体现只有在 R 树的叶节点上。R 树可以应用在范围查询和近邻查询中。本文主要使用 R 树近邻查询这一属性。给定一个查询点，R 树可以通过最优优先搜索（best-first）和深度优先搜索（depth-first）两种树遍历策略找到在数据集中最接近查询点的数据。在两种搜索策略中，查询点和每一个最小边界箱的距离被定义为变量 $mindist$。之后的搜索过程基本遵从两种搜索策略各自算法。

在 R 树中插入一个新的节点大致需要以下几个步骤。当有新的轨迹数据需要被添加到已有的 R

树种，首先为待插入的轨迹数据点找到一个合适插入的子节点中。再寻找叶结点的过程中我们会选择符合最小边界范围且对 R 树扩展度最小的一个叶结点。然后若找到 R 树叶结点数据溢出，那么我们需要对叶子结点进行分裂操作；若没有溢出，则可以将待添加的轨迹数据加入到当前已经找到的叶子节点中。最后对 R 树进行变换向上传递并对树高进行增高以完成插入操作。删除操作近似于插入过程的逆过程，在此不予以赘述。

2.4 本章小结

本章节中，本文通过图标和伪代码讨论了相似轨迹查询方法的设计与实现的基本相关工作，对本文所应用的基本定义、处理思路和数据结构有了一个初步的了解。根据上述本文工作相关工作描述，我们可以得出实现本文算法的先决条件目前都是基于只有已有的成熟工作。在下一章节中，本文将开始对相似轨迹查询方法进行理论讨论。

致　谢

感谢所有测试和使用交大学位论文 LaTeX 模板的同学!
感谢那位最先制作出博士学位论文 LaTeX 模板的交大物理系同学!
感谢 William Wang 同学对模板移植做出的巨大贡献!