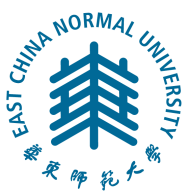
2018 届博士学位论文

分类号：_____ 学校代码：10269

密　级：_____ 学　　号：52141500013



East China Normal University

博 士 学 位 论 文

DOCTORAL DISSERTATION

论文题目：分布式 k 近邻轨迹查询

院　　系：	数据科学与工程学院
专 业 名 称：	软件工程
研 究 方 向：	基于位置的服务
指 导 教 师：	周傲英 教授
学位申请人：	章志刚

2018 年 05 月

Dissertation for doctor degree in 2018 University Code: 10269
Student ID: 52141500013

EAST CHINA NORMAL UNIVERSITY

k Nearest Neighbour Query over Distributed Trajectories

Department: School of Data Science and Engineering
Major: Software Engineering
Research direction: Location Based Service
Supervisor: Prof. Aoying Zhou
Candidate: Zhigang Zhang

2017.05

华东师范大学学位论文原创性声明

郑重声明：本人呈交的学位论文《基于矩阵分解的个性化推荐系统》，是在华东师范大学攻读硕士/博士（请勾选）学位期间，在导师的指导下进行的研究工作及取得的研究成果。除文中已经注明引用的内容外，本论文不包含其他个人已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中作了明确说明并表示谢意。

作者签名:_____ 日期: 年 月 日

华东师范大学学位论文著作权使用声明

《基于矩阵分解的个性化推荐系统》系本人在华东师范大学攻读学位期间在导师指导下完成的硕士/博士（请勾选）学位论文，本论文的研究成果归华东师范大学所有。本人同意华东师范大学根据相关规定保留和使用此学位论文，并向主管部门和相关机构如国家图书馆、中信所和“知网”送交学位论文的印刷版和电子版；允许学位论文进入华东师范大学图书馆及数据库被查阅、借阅；同意学校将学位论文加入全国博士、硕士学位论文共建单位数据库进行检索，将学位论文的标题和摘要汇编出版，采用影印、缩印或者其它方式合理复制学位论文。

本学位论文属于（请勾选）

（）1. 经华东师范大学相关部门审查核定的“内部”或“涉密”学位论文 *，于年月日解密，解密后适用上述授权。

（）2. 不保密，适用上述授权。

导师签名:_____ 本人签名:_____

年 月 日

* “涉密”学位论文应是已经华东师范大学学位评定委员会办公室或保密委员会审定过的学位论文（需附获批的《华东师范大学研究生申请学位论文“涉密”审批表》方为有效），未经上述部门审定的学位论文均为公开学位论文。此声明栏不填写的，默认为公开学位论文，均适用上述授权）。

章志刚 博士学位论文答辩委员会成员名单

姓名	职称	单位	备注
吴悦	教授	上海大学	**主席**
阚海斌	教授	复旦大学	
张娅	研究员	上海交通大学	
郑凯	特聘教授	苏州大学	
何晓丰	研究员	华东师范大学	

摘　要

近年来，随着手机等移动定位设备的大量使用，人们在生产生活中获得了飞速增长的轨迹大数据。该类大数据包含了车、人、动物甚至商品的移动行为。由于数据采集来源广、数据量大、数据增长快等原因，轨迹大数据往往以分布式形式存储。海量的轨迹数据不仅刻画了移动对象个体和群体的时空动态性，还蕴含着移动对象的行为信息，对交通导航、城市规划、物流调度等应用具有重要的价值。为充分挖掘轨迹数据的价值，学术界和工业界对轨迹数分析问题开展了大量研究工作，包括轨迹数据索引、轨迹聚类和分类、轨迹异常检测和移动预测等问题。

本文针对海量分布式轨迹数据，研究了 k 近邻查询，即从分布在不同数据结点的轨迹数据中找出与查询目标距离最近的 k 条轨迹。该查询广泛存在于基于位置服务的应用中，同时也是许多轨迹挖掘问题的子任务。尽管这一查询可以通过直接将待查询轨迹数据发送到所有远程结点，并在其上进行两两距离计算以找出结果集。但该算法的通信复杂度过高，达到 $O(n * M)$，其中 n 表示带查询轨迹的长度，M 表示远程结点的数量。由于有限的网络带宽资源，无法实际应用。此外，由于需要查询轨迹跟所有轨迹进行距离值计算，导致其时间复杂度较高。因此，该查询的重要挑战就是如何降低分布式算法的通信开销的同时快速给出查询结果。为此，本文提出通过使用概要数据进行距离值估算并利用估计值进行剪枝的策略，从而达到节约通信开销的目的。根据这一策略，设计了两种查询框架以满足各种距离度量准则的要求。第一种针对能使用概要数据同时估计出距离上、下界的场景。第二种针对那些只能根据概要数据估计出下界的场景。接着，我们介绍了如何将具体的距离度量准则应用到这两种框架中。本文主要贡献包括如下三方面：

1. **设计了两种查询框架 FTB 和 FLB 以处理分布式 k 近邻轨迹查询。** 在这两个框架中，我们将查询轨迹的多粒度概要数据，由粗到细发送到远程结点中。远程结点根据获取的概要数据进行距离范围估计，并利用估计值进行剪枝。随着所获概要数据粒度的增加，所估计的距离范围也越来越紧，剪枝后所剩候选也越来越少，直到只剩下 k 个候选。由于概要数据的大小远小于查询轨迹本身，因此使用概要数据进行剪枝的方式能显著减少通信开销。此外，使用概要数据计算估计值的时间也远小于计算真实距离的时间。从而使得这两个框架的效率也高。最后，这两个框架的主要区别在于 FTB 框架要求使用概要数据能同时估计出距离的上界和下界。而 FLB 框架仅要求能估计出下界。任一距离度量方式只要设计出能够满足距离估计的概要数据，就可以应用到

对应框架中。

2. **设计了 ED-FTB 算法以处理基于欧式距离的分布式 k 近邻轨迹查询**欧式距离因其简单易算等特性，被广泛应用于时间序列数据分析。为此，本文研究了基于该距离准则的查询。我们首先使用哈尔小波对轨迹数据进行变换，得到多粒度的哈尔小波系数并使用其作为概要数据。接着，设计了基于部分哈尔小波系数的欧式距离上界和下界，并证明了随着系数数据的增加所得到的距离上、下界同时会越来越紧。基于以上结论，设计了欧式距离和 FTB 框架相结合的 ED-FTB 算法。在该算法中，引入了在计算上、下界过程中进行剪枝的策略以提高执行效率。最后，理论分析和实验结果相结合，展示了算法相比于基准算法的优越性。

3. **设计了 DTW-FLB 算法以处理基于动态时间卷曲距离的分布式 k 近邻轨迹查询**动态时间卷曲距离因允许两轨迹变化的速度不一样同样也被广泛应用时间序列数据的分析中。比如，在寻找相似的运动轨迹时，只要路径一样，哪怕速度或步行时间不一致也被认为是极相似的轨迹。为此，本文研究了基于动态时间卷曲距离的查询。首先，设计了满足动态时间卷曲约束的包围信封。在此基础之上，设计了多粒度包围信封。接着，设计了基于包围信封的动态时间卷曲距离下界，并证明了随着包围信封粒度的增加所得到的距离的下界会越来越紧。基于以上分析，设计了动态时间卷曲距离和 FLB 框架箱结合的 DTW-FLB 算法。同样在该算法中引入了在计算下界过程中进行剪枝的策略以提高执行效率。最后，我们在真实轨迹数据集上验证了算法的有效性和可扩展性。

关键词: 轨迹大数据，分布式 k 近邻查询，通信开销，时间效率，距离上、下界.

ABSTRACT

Recently, big trajectory data is generated in an explosive way with the popularity of smartphones and other location-acquisition devices. These devices, monitoring the motion of vehicles, people, animals and goods, are producing massive distributed trajectories rapidly. These data not only reflect the spatio-temporal mobility of individuals and groups, but may also contain the behavior information of moving objects. They are invaluable for applications such as route planning, urban planning and logistics scheduling . To make full use of such data, tremendous efforts have been made to support effective trajectory data management analysis, including trajectory indexing, trajectory clustering, trajectory classification, anomaly detection and behavior prediction.

In this paper, we focus on the k nearest neighbor query over the distributed trajectory data, which finds k trajectories that are most similar to the given reference trajectory. This kind of query is a basic function of many LBS applications and also plays an important role in trajectory pattern mining tasks. Although this query can be solved by sending the query trajectory to all the remote sites, in which the pairwise distances are computed precisely. However, the overall communication cost, $O(n \cdot M)$, is huge when n or m is huge, where n is the size of query trajectory and M is the number of remote sites. Besides, the procedure of pairwise distances computation for all trajectories is also time consuming. So, the key challenge in this query is how to reduce the communication cost due to the limited network bandwidth resource and meanwhile give the query result quickly. Fortunately, there are some communication-saving ways to estimate pairwise distance by using sketch data, which allow filtering some trajectories in advance without precise computation. In order to overcome the above challenge in this query, we devise two general query processing frameworks, into which concrete distance measures can be plugged. The former one uses both the upper and lower bound of distance metric, while the latter one only uses the lower bound. Then, we introduce detailed implementations of these frameworks by embedding

representative distance measures. The research questions and technical contributions in this thesis can be summarized as follows:

1. **Two basic processing frameworks, FTB(Framework with Two Bounds) and FLB(Framework with Lower Bound), are proposed to solve the knn query over distributed trajectories.** In both of the frameworks, we send more and more fine-grained sketch data of the reference trajectories to get more and more tight distance bounds for each candidate. Then, we use these bounds to prune candidates. As the data size of sketch data is much smaller than the original trajectory, a great amount of communication are saved in them. Besides, the computation cost for distance bounds is also smaller than the exact distance value. So, these framework is also very efficient in running time. The main difference between these two frameworks is that: FTB framework requires both the upper and lower bound of each candidate, while FLB only uses the lower bound. Any distance metric can be plugged into these two frameworks provided that proper sketch data is designed and corresponding distance bounds can be inferred.

2. **ED-FTB algorithm is proposed to process Euclidean distance based query by implementing the FTB framework.** Euclidean distance is one of the most popular distance metrics for time series data due to its simplicity. This thesis designs ED-FTB algorithm to embed Euclidean distance into our query. We first exploit the Haar wavelet to transform the reference trajectory and adopts the Haar coefficients as the sketch data. Then, we design both upper and lower distance bounds for Euclidean distance by using only partial of the coefficients, and we ensure that these bounds are tightened when more coefficients are used. Consequently, we combine Euclidean distance with FTB framework and design the ED-FTB algorithm to process the query. Besides, early-abandoning policy is adopted to improve the query efficiency. Theoretical analysis and extensive experimental results show that ED-FTB outperforms the state-of-the-art algorithm.

3. **DTW-FLB algorithm is proposed to process DTW (Dynamic Time Warping) distance based query by implementing the FLB framework.** DTW distance is another popular metric as it allows two time series vary in speed. For instance, similarities in walking could be detected using DTW, even if one person was walking faster than the other. This thesis designs DTW-FLB algorithm to embed DTW distance into our query. We compute bounding envelopes of different granularities for the reference firstly. Then, we design a lower bound for DTW distance by using bounding envelope. We give the proof that finer-grained bounding envelopes lead to tighter lower bounds. Thus, we combine DTW distance with FLB framework and devise the DTW-FLB algorithm to process the query. Early-abandoning policy is also adopted to the query efficiency. Extensive experimental results show that DTW-FLB algorithm is efficient and scalable.

Keywords: *Big Trajectory Data; distributed knn query; Communication Cost; Time Efficiency; Distance Bounds.*

目录

插图

表格

第一章　绪论

轨迹大数据是移动物联网时代的产物，记录了移动对象的行为信息。通过轨迹数据分析技术可以挖掘人类活动规律与行为特征、城市车辆的移动模式、大气环境变化规律等信息。这些挖掘成果为研究城市发展、社会变迁和自然环境演变提供重要的参考价值。本文重点研究了分布式 k 近邻轨迹查询问题。本章中，章节 1.1阐述了分布式轨迹数据相似性查询的研究现状；章节 1.2详述了本文的具体研究内容以及所遇到的挑战；章节 1.3简述了本文的主要研究方法和研究贡献。

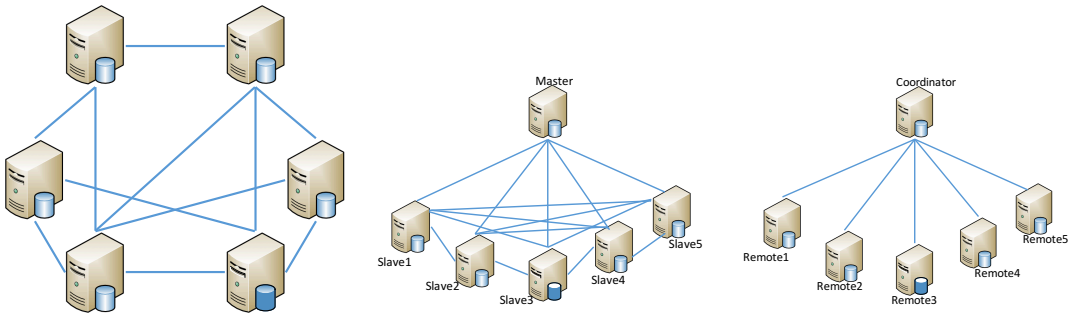
1.1　研究背景

随着卫星定位导航、无线通信和普适计算等高新技术的不断发展，带有定位功能的移动智能设备已被广泛应用，并成为社会生产、生活的重要组成部分。移动对象（含人、动物和车辆等）在主动或被动使用这些设备的同时，产生了大量记录其移动历史行为的轨迹数据（Trajectory Data）。例如：滴滴出行，全球最大的一站式多元化出行平台，在 2017 年 10 月发布的《第三季度全国重点城市交通出行报告》中称其每天新增的定位轨迹数据超过 70TB[1]。轨迹数据是地理空间在时间轴上形成的多维空间中的一条曲线，表示了移动对象在一段时间内时空信息等移动行为的变化。每条轨迹可看作一条时空采样点构成的序列，其中每个采样点记录了时间、位置、速度、方向等信息。从微观角度将，轨迹数据蕴含了移动对象的移动模式与规律，例如从轨迹数据种我们可以发现市民的居住地、工作地和消费娱乐场所。从宏观角度来讲，海量的轨迹大数据蕴含了群体的移动迁徙和社会发展的变化，如城市的发展、交通的演化以及社会的变迁。通过轨迹分析等手段进行知识发现，并将它们运用在各种交通和服务应用系统中，包括交通导航、交通智能指挥、车辆监控、物流配送、城市规划、军事调度等 [3]。

[1]http://www.xiaojukeji.com/index/index

海量的轨迹数据具有重要的社会和应用价值，不仅为解决拥堵、改善交通服务、缓解能源紧缺和降低大气污染等社会问题提供了新的机遇，而且对认知人民的社会活动、优化公共资源配置、为建立新型共享经济有着特殊的意义。2017 年 12 月 8 日，习近平总书记在中共中央政治局第二次集体学习时强调：实施国家大数据战略，加快建设数字中国。因此，轨迹大数据称为政府和企业的重要资源财富并得到广泛重视。在此背景下，轨迹大数据的分析和挖掘已经被学术界和工业界大量研究并成为数据挖掘领域的重要的新兴分支。在工业界，百度地图能根据实时轨迹数据进行路径规划。摩拜进行了基于自行车骑行目的地预测的研究。美团和饿了么等外卖公司设计了基于轨迹数据的智能派单系统。上海电信根据手机信令数据研究了人口流动分析。学术界业也出现一些针对轨迹数挖掘的热点研究工作，包括轨迹查询系统研究 [4–7] 查询可伸缩的快速轨迹聚类 [8–11]、轨迹流的连续查询 [12]、路径规划及路径发现 [13]、汇集模式发现 [14]、旅伴模式发现 [15, 16] 以及实时共享乘车 [17] 等。

轨迹相似性研究是轨迹挖掘和分析的核心内容之一。现有的轨迹数据间的相似性通常通过它们之间的距离来度量，即两条轨迹数据之间的距离越小，则认为它们之间越相似。根据这一准则，研究者们已经将广泛应用于时间序列分析的距离度量函数应用到轨迹数据中，这些度量准则包括：欧式距离（Euclidean Distance, ED)[18]、动态时间卷曲（Dynamic Time Warping, DTW)[19]、最长公共子序列（Longest Common Subsequence, LCSS)[20, 21]、实值序列上的编辑距离（Edit Distance on Real sequence，EDR）[22] 和带有惩罚项的编辑距离（Edit Distance with Real Penalty, ERP）[23]。此外，部分研究者针对轨迹数据特性设计了霍斯托夫距离 (Hausdorff Distance，HD)[11]、弗雷歇距离（Fréchet Distance, FD）[24, 25]、一路距离（One Way Distance，OWD）[26]、带投影的编辑距离（Edit Distance with Projections, EDwP）等。此外，还有部分学者针对轨迹的语义特性设计了用于计算轨迹语义相似度的距离 [27–29]。尽管这些距离度量函数可以较好地捕捉到轨迹之间的相似度，但它们的计算复杂度高，要想应用到大规模轨迹数据集上需要进一步研究如何提高计



(a) P2P (b) 主从式 (c) 协调者-远程结点式

图 1.1: 3 种常见分布式架构

算效率。此外，如何针对特定问题选择合适的相似度度量函数也是重要的研究内容 [30, 31]。目前，仍然缺乏统一的处理框架以支持多种距离度量准则的应用。

此外，轨迹数据由于来源广、数据量大等特性，往往以分布式形式采集和存储在各结点中。这类结点既可以是管理着大量轨迹数据的高性能服务器也可以是管理单一移动对象的的个人智能手机。此外，结点间可能相互独立，不能互相访问。因此，如何对分布在具有不同存储和计算能力的各个结点上的分布式轨迹数据进行统一的管理、分析和挖掘服务是合理利用迹数据的首要问题。现有的分布式解决方案可分为以下三类：（i）基于对等系统（Peer-To-Peer，P2P）架构的处理方案。该架构中各个结点能互相访问且各个结点都有原始数据的完整备份，因此不适用与轨迹大数据。（ii）基于主-从式（Master-Slave）的分布式集群架构的处理方案。该架构中由一个主（Master）结点和多个从（Slave）结点构成，其中从结点存放具体数据并负责任务的执行，主结点存放数据的元数据并负责任务的分发和调度。基于该类架构典型的处理系统有 Hadoop 和 Spark。该架构往往要求所有物理结点在同一个集群内部，结点间通过高速网络连通。（iii）基于协调者-远程结点的（Coordinator-Remote Site）的分布式架构。该架构中存在一个协调者和多个远程节点，其中协调者结点不存储任何数据只负责跟远程结点通信，而远程结点负责存放本地数据，且各远程结点间互不通信。第三种方案可看做第二种方案的推广，首先它允许所有结点物理上互相远离，不要求它们位于同一集群内部。其次，远程结点间做到完全独立且互不知晓。最后，协调者结点不需要知道远程结点数据的

任何信息。以上三点能有效地保证数据拥有者的隐私和数据存储、计算的独立性。因而，本文研究的分布式场景选择第三种处理框架。

目前，分布式轨迹相似度研究已经得到广泛的重视。文献 [32, 33] 研究了主-从式架构下轨迹数据的 join 查询，其关注重点是如何降低从节点间数据交换量过高的问题。文献 [**?**] 研究了基于协调者-远程节点架构下的 k 近邻轨迹查询研究。其研究的是用户手机连接基站的轨迹数据并将每个基站看做一个远程节点。由于手机在用户移动过程中会连接到不同的基站，故轨迹数据被切割成若干数据片段存放在不同的基站中。在该研究中，其研究目标是如何提高查询的效率而未关注如何降低通信开销。类似的，Smart Trace[21] 和 Smart Trace$^+$ [20] 在相同的系统架构中研究了相同的问题，但在其研究内容中，将每个智能手机看做远程结点且每个结点仅保存一条轨迹数据。在这两项工作中，计算效率和通信开销同时得到了考虑。与以上不同的是，本文研究了更加抽象的分布式场景，即每条轨迹完整的保存在一台远程结点上，且每个远程结点可存储多条轨迹数据。此外，文献 [7] 提出了基于 Spark 的 k 近邻轨迹查询系统。该系统由于不满足应用中远程结点间相互独立、互不知晓的要求，因而无法推广到本文所提查询中。

1.2 研究内容与挑战

本文选用基于协调者-远程节点的分布式架构进行管理和分析轨迹大数据，并对轨迹大数据上的 k 近邻轨迹查询进行了研究。该系统架构中存在一个协调者结点和 M 个远程结点。协调者节点和所有远程结点连通，但不知道远程结点的数据信息。远程结点间彼此独立且互不通信。在实现查询过程中，协调者结点扮演查询引擎的角色，其接受用户提交的查询，并将查询相关数据发送给远程结点。每个远程结点根据接收到的数据返回响应的查询结果。基于此框架，本文所研究查询如图1.2所示。协调者节点负责接收用户提交的查询轨迹，每个远程结点存储一部分轨迹数据。协调者节点的目标是从所有远程结点中找出与查询对象最相似的 k 条轨迹。一个直观的解决方案就是用户将 $\mathcal{Q}$ 提交给协调者结点后，该结点直接将 $\mathcal{Q}$

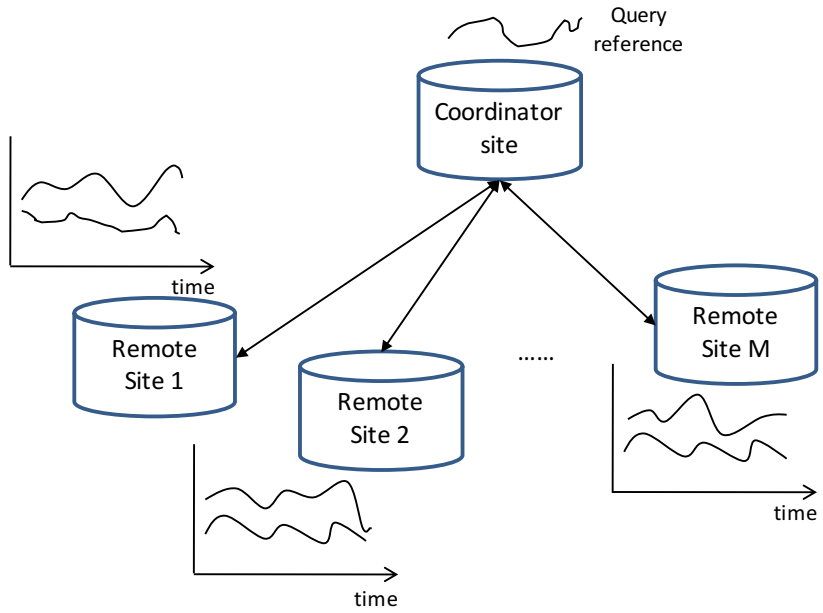


图 1.2: 查询展示

发送给所有的远程结点。远程结点接收到查询对象后，计算其与本地存储的所有轨迹间的距离并将相似度值返回给协调者结点。最终，协调者结点从接收到的距离值中选择最小的 k 个作为结果返回给用户。这样的解决方案，设计简单且易于实现，但也存在着通信量过大的问题，尤其是当待查询轨迹数据量较大或远程结点较多时。因此，如何降低通信开销是我们需要解决的首要问题。此外，该方案中由于需要对查询轨迹和远程结点间的轨迹进行距离值计算，导致其计算开销也较大，我们需要考虑如何提高查询的执行效率。最后，如上节所介绍，轨迹间的距离计算准则很多，现有工作只针对某一具体距离来进行通信或效率优化。因而，缺乏统一的框架或方法来处理。综上所述，本文研究面临的挑战主要表现为以下 3 点：

- **首先，针对时间序列上已有的 k 近邻轨迹查询技术往往只针对某一距离准则设计，缺乏通用性。**距离度量选择准则是轨迹数据挖掘分析的核心内容之一，选用不同的度量函数，导致的挖掘结果以及对结果的理解往往大不相同。现有的工作都是针对某一研究问题，精心挑选或自定义一距离函数以满足研究的需要。因此，所提出的解决方案具有较高的局限性，很难推广到其他的问题中。为此，需要提出通用或对一类问题适用的解决方案。

- **其次，现有分布式 k 近邻查询算法中仍然存在着通信开销过大的问题。**直接将原始查询轨迹发送到所有远程结点的方式，其通信开销随轨迹长度和远程结点的个数的增加而快速增大。而现有的数据降维方案如 Douglas-Peuker 算法、奇异值分解（Singular Value Decomposition, SVD）、离散傅里叶变换（Discrete Fourier Transform, DFT）、哈尔小波变换（Haaar Wavelet Transform, HWT）、分段聚合近似（Piecewise Aggregate Approximation, PAA）、自适应分段常量近似（Adaptive Piecewise Constant Approximation, APCA）等方法虽然降低了数据的维度，但也造成了数据信息损失，使得数据不再直观且不能保证结果的准确性。此外，这些方法常用于处理一维时间序列数据，而轨迹是天然的多维时间序列数据。为此，需要提出新的轨迹数据降维方法，在保证查询结果正确性的同时，降低通信开销。

- **最后，现有的分布式 top-k 查询时间效率需要得到提升。**轨迹距离的计算除欧式距离是一次的计算复杂度，其他往往需要二次的计算复杂度。在每个远程结点进行查询轨迹与局部保存的轨迹进行两两相似度计算的方案不可行。虽然适用传统索引技术是加快查询速度有效工具，但索引技术只能针对原始轨迹数据，无法适用于降维后的数据。为此，需要设计新的方案来提高查询效率。

1.3 主要贡献

本文围绕分布式 k 近邻轨迹查询这一问题，系统性的提出了解决方案。首先，针对轨迹距离度量的多样性，提出了两种查询实现框架以覆盖所有的距离函数。这两个框架分别针对不同类别的距离度量准则，在保证查询结果正确性的同时能有效地降低通信开销。接着，我们将两个常用相似度距离函数分别嵌入到两个框架中，并提出了对应的算法。最后，使用真实轨迹数据集验证了算法的性能。因此，本文的主要贡献分为以下 3 点：

- **针对通信开销较高问题，提出了两个通用查询实现框架 FTB（Framework**

with Two Bounds）和 FLB（Framework with Lower Bound）。FTB 框架中要求能根据降维后的概要数据计算出相似度距离的上下界，并能保证概当细粒度的概要数据获取后，所计算出来的上下界越来越紧并最终能收敛。FLB 框架中仅要求能根据降维后的概要数据计算出相似度距离的下界，且能保证当细粒度的概要数据获取后，所计算出来的下界越来越紧。这两个框架由于仅需要传递查询轨迹的概要数据，因此能大大降低通信开销。

- **针对 FTB 框架，提出了基于欧式距离的 FTB-ED 算法**。为验证 FTB 框架的有效性，本文研究了如何将具体欧式距离嵌入到该框架中。为此，本文首先利用 Haar 小波变换以得到不同粒度的轨迹概要数据，并证明了全部概要数据的欧式距离等于原始轨迹数据的欧式距离。接着，利用部分概要数据，本文提出了基于欧式距离的轨迹相似度上、下界，并理论证明了该相似度的正确性。然后，我们将该上、下界应用到 FTB 框架中，提出了 FTB-ED 算法。在该算法中，我们引入了性能优化策略以提高查询效率。最后，我们通过大量实验验证了 FTB-ED 算法的有效性和可扩展性。

- **针对 FLB 框架，提出了基于动态时间卷曲距离的 FLB-DTW 算法**。为验证 FLB 框架的有效性，本文研究了如何将具体动态时间卷曲距离嵌入到该框架中。为此，本文首先针对查询轨迹使用不同粒度的包围信封（Bounding Envelope）来表示轨迹的概要数据。接着提出了基于动态时间卷曲距离的下界，并理论证明了该下界的正确性。最后将该下界应用到 FLB-DTW 框架中，并提出了 FLB-DTW 算法。在该算法中我们引入了多种机制以提高查询效率。最后，我们通过大量实验验证了 FLB-DTW 算法的有效性和可扩展性。

1.4 章节安排

本文一共分为六章，章节安排如图 1.3所示：

- 第二章从轨迹数据存储、轨迹降维以及轨迹数据相似性查询三个方面介绍了

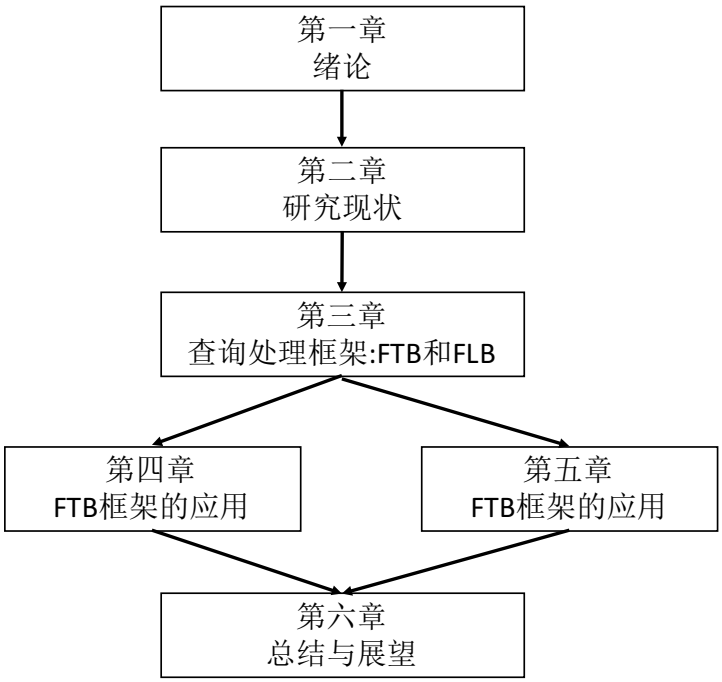


图 1.3: 本文的组织结构

研究背景知识和研究现状。

- 第三章介绍了两个通用查询处理框架 FTB 和 FLB 以分别处理能同时提供上、下界的和仅能提供下界的距离准则。
- 第四章介绍了如何将欧式距离嵌入到 FTB 框架中，并提供高效的查询结果。
- 第五章介绍了如何将 DTW 距离嵌入到 FLB 框架中，并提供高效的查询结果。
- 第六章对上述已有工作进行了总结，并展望了未来的研究方向和内容。

第二章　问题定义及研究现状

本章第一节首先对数据模型和要解决的查询问题进行了定义，然后从以下方面介绍了相关的研究工作：轨迹索引、轨迹降维、轨迹相似度度量、时间序列数据top-k查询，最后对本章内容进行小结。

2.1　问题定义

本文的研究目标是给定一条查询轨迹，从存储在若干远程结点中的轨迹中找出k条距离最短或相似度最高的轨迹。为此，本节首先介绍了轨迹数据模型，接着介绍了查询的定义。

2.1.1　轨迹数据

轨迹是描述移动对象移动行为的数据。通常来说，一条轨迹T可看做包含n个元素（即轨迹点）的有序序列。每个轨迹点$\boldsymbol{p}$包含了时间、位置等维度的信息。因此轨迹可被形式化定义为如下:

定义 2.1.1 (轨迹). *轨迹形式化表示为:*$T = \{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{n-1}\}$。

$$T = \{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{n-1}\} \tag{2.1}$$

其中$|T| = n$*代表轨迹所包含的点数，即轨迹的长度。每个轨迹点*$\boldsymbol{p}$*包含时间*(t)*、位置*(l)*等维度的信息。因而*$|\boldsymbol{p}| = d$*称为轨迹的维度。此外轨迹中的点严格按时间升序排列，即*$\forall i, j, 0 \leq i \leq j < n$*则*$\boldsymbol{p}_i.t \leq \boldsymbol{p}_j.t$。

轨迹数据的来源多样且复杂。根据移动对象的划分可分为如下几类:

- 人类活动轨迹数据: 该类数据分为主动式和被动式。主动式数据是人们主动利用移动定位设备分享或汇报自己的位置等信息。典型的有社交网络中的数

据，用户提交位置获得服务的数据。被动式数据是人们无意间使用各种服务时所产生的轨迹数据。典型的有公交刷卡轨迹和手机的信令轨迹数据。

- **交通工具轨迹数据**: 这类数据主要是交通工具使用车载 GPS 设备所产生的移动轨迹数据。例如, 出租车、公交车的活动轨迹数据。
- **动物活动轨迹数据**: 这类数据是为了研究动物生活、迁徙等行为和习惯而捕获的数据。
- **自然现象活动轨迹数据**: 这类数据典型的有台风、冰山、海洋事件等的轨迹数据，用以探索自然现象的活动规律。

轨迹数据符合大数据时代的 3V 特征, 即量大、实时、多样。轨迹数据采样由于受设备、采样频率等因素影响，数据质量较低且各个轨迹的采样间隔差异显著。这些问题导致原始轨迹数据的可用性较低。因此，我们在进行轨迹数据分析前往往需要经过数据清理 (data cleaning)、地图匹配 (map mathching)、轨迹分段 (trajectory segmentation) 等预处理方式化为校准轨迹。校准轨迹数据能够通过数据管理技术进行轨迹索引以便有效地存取。因此，本文所处理的轨迹数据为预处理后的校准轨迹数据。这样的数据有如下特点：（i）采样频率一致；（ii）长度一致；（iii）位置精度高。这为我们挖掘轨迹模式从而提炼有价值的知识提供了可靠保障。

2.1.2 分布式 k 近邻轨迹查询

轨迹数据往往是分布式采集并存储的。为此假设有 M 个远程结点，每个远程结点 i 包含轨迹数据集 $\mathcal{D}_i$。那么整个分布式轨迹数据集 $\mathcal{D} = \bigcup_{i=1}^{M} \mathcal{D}_i$。我们的目标是给定查询轨迹，从分布式存储的 $\mathcal{D}$ 数据集中，找出与其距离最近的 k 条轨迹。下面我们将给出查询的形式化定义:

定义 2.1.2 (分布式 k 近邻轨迹查询)**.** *该查询形式为 query*$(\mathcal{Q}, \mathcal{D}, DM, k)$*，其中 $\mathcal{Q}$ 为给定查询轨迹，$\mathcal{D}$ 为分布式轨迹数据集，DM 为距离度量准则以及 k 为返回结果*

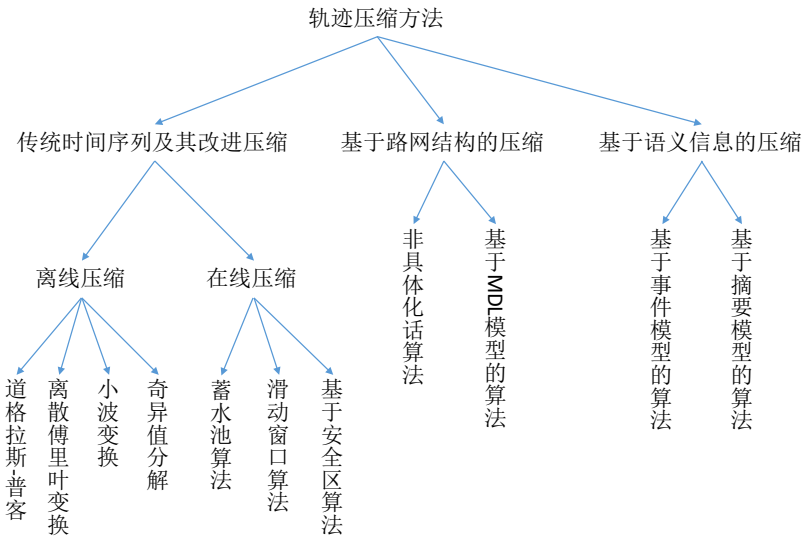


图 2.1: 基于安全区的压缩算法

集大小。查询的目标是返回满足如下条件的轨迹集 $\mathcal{S}$: (1) $\mathcal{S} \subseteq \mathcal{D}$; (2) $|\mathcal{S}| = k$; (3) $\forall \mathcal{C} \in \mathcal{S}, \mathcal{C}' \in \mathcal{D} - \mathcal{S}$, $DM(\mathcal{Q}, \mathcal{C}) \leq DM(\mathcal{Q}, \mathcal{C}')$。

传统的集中式环境下 k 近邻轨迹查询相比，分布式场景下的查询不仅注重查询效率，而且尤其注重通信开销。这是由于分布式场景中，远程结点和协调者结点的带宽资源往往是有限的。高的通信开销，意味着用户可能要花费更多的金钱。因此，用户允许多花一点时间以达到降低通信开销的目的。

2.2 轨迹压缩

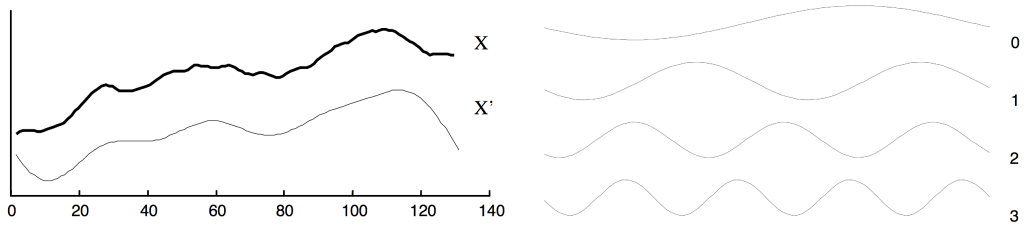
将查询轨迹数据压缩是降低通信开销的有效方式。如图2.1所示，现有轨迹压缩的方法根据处理数据类型的不同可分为 3 类 [34]：第一类是传统的时间序列压缩算法，其根据应用场景又可以分为离线和在线压缩两类算法。第二类是基于路网结构的轨迹压缩算法，该类算法依赖于所给路网数据。第三类是语义压缩，其目标是将原始数值型轨迹数据转换为人们能理解的语义信息。常用轨迹压缩算法

2.2.1 传统离线时间序列压缩算法

离线轨迹压缩算法主要的应用场景是，给定一静态历史轨迹数据库，对其中的轨迹进行压缩。该类压缩算法不考虑数据集的增加、删除和修改。该类算法的提出主要是为了解决时间序列数据的降维（降低序列的长度）。由于轨迹本质上也是时间序列数据，因此该类方法也可用于轨迹的压缩。这是由于轨迹数据的各属性维度上数据一般互相独立，因此只需将降维方法应用到各个维度上分别进行压缩即可。本节主要介绍几种典型的离线时间序列（含轨迹）压缩算法。

道格拉斯-普克算法：道格拉斯-普克 (Douglas-Peucker, DP) 算法将原始时间序列看作一条多维曲线，其目标是找出一条与原始曲线相似但使用更少点构成的简化曲线。此外，构成简化曲线的点必须来自原来曲线。该算法中原始轨迹与简化轨迹间的“不相似性”是通过两条曲线间的豪斯多夫（Hausdorff）距离表示。简化曲线中的点即为压缩后的数据。道格拉斯-普克算法的过程由如下 5 个步骤构成：（1）在曲线首尾两点 A，B 之间连接一条直线 AB；（2）找出曲线上离该直线段距离最大的点 C，计算其与 AB 的距离；（3）比较该距离与预先给定的阈值的大小，如果小于阈值，则该直线段作为曲线的近似，该段曲线处理完毕；（4）如果距离大于阈值，则用 C 将曲线分为两段 AC 和 BC，并分别对两段取信进行 1 至 3 步的处理；（5）当所有曲线都处理完毕时，保留依次收集分割点，得到压缩后的点集。DP 算法的思想简单且性能较好，在各个领域都有广泛的应用。但其缺点就是算法复杂度较高达到 $O(n^2)$，n 是轨迹长度。文献 [35] 提出了借助外部存储结构的改进算法，把时间复杂度降低为 $O(n \log n)$。此外，文献 [36] 和 [37] 分别提出了同时时间和空间维度的改进距离函数（Time-Distance Ratio, TDR）和 (Top-Down Time Ratio, TD-TR)，克服了 DP 仅考虑空间维度的缺点。

离散傅里叶变换：离散傅里叶变换 (Discrete Fourier Transform, DFT)[38] 是时间序列数据降维的常用方法，并且已有许多扩展和改进方案被提出 [39–41]。其基本思想是任何一个信号，不管其多复杂，都可以分解为有限个正弦和余弦曲线的叠加。每条曲线可以由一称为傅里叶系数的复数表示 [42]。此时，我们时间序列由



(a) 压缩前后数据　　(b) DFT 系数

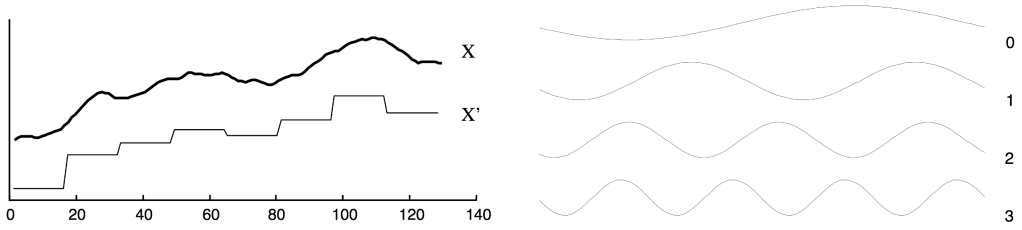
图 2.2: 离散傅里叶压缩数据 [1]

时域变换到频域了。使用频域的好处有很多，其最重要的就是能进行数据压缩。一条长度为 n 的信号（即时间序列）可以被分解为 n 个正/余弦曲线，且这 n 个曲线能重新组合成原来的信号序列。但由于 n 个曲线中存在着许多振幅较低的，这些低振幅曲线对重新构建原来的信号的贡献较低。因此，其对应的低振幅系数可以被丢弃掉，而保留那些振幅较高的系数。这些保留的系数重新组合所构成的信号数据与原始信号数据相比，并没有太多的信息丢失。因而，达到了使用少量数据来表示原始信息的目的。

此外，文献 [42] 中提出原始时域内信号数据间的欧式距离等于变换后频域内系数间的欧式距离。这一结果称为帕斯瓦尔定理（Parseval's law）。因此，若经离散傅里叶变换且将低振幅的系数丢掉后，使用剩下的系数计算距离，所得到的结果必然是原始欧式距离的下界（丢失的数据都是正数）。文献 [1] 提出了基于该下界的剪枝方法。

为将以长度为 n 的时间序列降低到 N 维特征。我们调用离散傅里叶变换并保留了 $N/2$ 个系数。保留 $N/2$ 而不是 N 的原因是每个系数是一个复数，我们需要同时保留实部和虚部的值。图2.2介绍了使用离散傅里叶变换进行压缩数据的思想。首先对左图原始信号 X 进行傅里叶变换。然后只保留了如右图振幅较高的 4 个曲线。接着，我们利用保留的 4 个系数恢复出原始轨迹的近似轨迹 X'。在我们的分布式查询中，可以通过发送压缩后的系数数据以达到降低通信开销的目的。

小波变换：小波变换（Wavelet Transform）将信号分解为一系列基/母小波函数的和与差的组合，这一点与离散傅里叶变换类似。但它们之间也有许多的不同



(a) 压缩前后数据　　(b) DFT 系数

图 2.3: 哈尔小波压缩数据 [1]

点。其中一个重要的区别是，小波在时间上是局部的，即小波系数只代表原始数据中的一小部分子段。这也是“小波”这一名词的由来，“小波”就是小区域、长度有限、均值为 0 的波形。而傅里叶系数总是代表着数据的全局信息。对于一些需要研究数据局部信息的应用来说，具有多解析度特性的小波变换更为适用。

当原始信号/时间序列数据被小波分解之后，会得到一个小波系数序列。该序列的前几个系数包含了数据的总体信息，是对原始信号数据的粗粒度估计。而其余的系数则包含着数据局部的细节信息。因此使用小波变换压缩数据，就是通过只保留前几个系数，以大体上逼近原始的数据，达到降维的目的。其代价是损失一些局部的细节信息。这一点跟傅里叶变换进行压缩一样，两者都是有损压缩。近年来，小波变换由于其同时保持了时域和频域的特征（傅里叶只包含了频域特性），在数据压缩、过滤、分析等领域得到广泛应用。

哈尔小波是表达和计算都最简单的一种小波。它的基函数是均值和差值函数。Chan 等人证明了原始时域内信号数据间的欧式距离等于变换后哈尔小波系数间的欧式距离 [43]。图2.3展示了经过哈尔小波变换压缩数据的过程。对于给定的时间序列 X，经过哈尔小波变换后，我们只保留了右图所示前 8 个系数。利用这 8 个系数，我们可以计算出原始时间序列的近似值 X'。在我们的分布式查询中，可以通过发送这些前面的哈尔小波系数以达到降低通信开销的目的。

奇异值分解： 尽管奇异值分解 (Singular Value Decomposition, SVD) 方法已经成功应用于图像和多媒体数据的分析中。现有的大量工作表面，其也可以被应用于时间序列数据的索引和压缩中。

前面介绍的几种方法都是对单一轨迹进行压缩，各个轨迹之间相互独立。奇异值分部街不是针对单个时间序列对象进行压缩，而是对全局数据进行整体变换，以得到整体数据的压缩结果。其分解过程如下，首先将整个数据集进行变换，在原始的数据空间中顺序地找一组相互正交的坐标轴，其中沿第一个坐标轴方向的方差最大，而在与第一个坐标轴正交的平面上，沿第二个坐标轴方向的方差最大，在于第一个坐标轴以及第二个坐标轴正交的平面中，沿第三个坐标轴的方向方差最大，依次类推。对于长度为 n 的时间序列，可以通过奇异值分解找到 n 个满足条件的坐标轴。为了达到降维的目的，取前 N 个坐标轴构成一个 N 维的空间来近似原来的 n 维空间，这样就将一个 n 维空间变换成了 N 维的近似空间，且使用上述方法选择的 N 个坐标轴所张成的空间中，数据的损失最小。

2.2.2 传统在线时间序列压缩算法

离线的轨迹压缩算法可以充分根据轨迹的所有信息来获取较好的压缩效果。但对于实时在线场景，我们无法事先得到完整的轨迹数据，因而需要在线算法来进行实时压缩。相比于离线压缩算法，降低计算开销是在线算法的重要内容，它需要快速确定是否要保留刚刚到来的轨迹点。为此，我们将介绍几种典型的在线算法。

蓄水池算法：蓄水池算法（Reservior Sampling Algorithm）[44] 的思想是为每个时间序列维持一个大小为 R 的缓冲区，并用这 R 个点来近似原始轨迹。由于轨迹数据是持续增加的，无法知道轨迹的最终长度。因而，该算法所要解决的核心问题就是无放回替换，即当一个点从缓冲区删除掉后，不能再将它取回。它的做法是，首先将前 R 个轨迹点放入缓冲区，当新的轨迹点到来时，判断该点是否需要放入缓冲区。具体地，当第 k 个点到来时 ($k > R$)，该算法以 R/k 的概率判断改点是否需要加入缓冲区。如果判断结果正确，则从现有的轨迹点中随机删除一个点，并将新点加入。该算法的时间复杂度为 $O(R(1 + \log N/R))$，因此效率较高。但它没有考虑轨迹的时间或空间属性，因而近似轨迹无法保留轨迹的原始特征。

滑动窗口算法：与蓄水池算法不同的是，滑动窗口算法 [45, 46] 的思想是维护

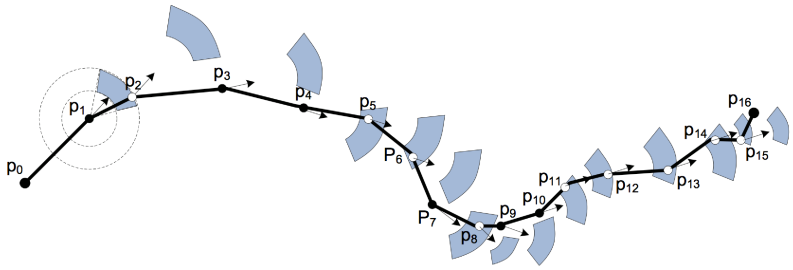


图 2.4: 基于安全区的压缩算法 [2]

一个不断增大的缓冲区，并用缓冲区中的轨迹来近似原始轨迹。它的算法过程是，首先选取第一个点作为锚点，用 $\boldsymbol{p}_a$ 来标识，然后将下一个点加入滑动窗口中。当新点 vp_i 加入后使用 $\overline{\boldsymbol{p}_a\boldsymbol{p}_i}$ 来拟合窗口中所有点组成的字轨迹。若 $\overline{\boldsymbol{p}_a\boldsymbol{p}_i}$ 拟合误差小于给定的阈值，则继续加入新点。否则，将 $\boldsymbol{p}_a$ 与 vp_{i-1} 之间的点从缓冲区删除，并使用 vp_{i-1} 作为新的锚点。该算法能有效保留轨迹的原始时空特征。

基于安全区的算法： 以上压缩算法没有考虑轨迹中所蕴含的速度和方向等特征，为此 Potamias[47] 等认为，一个轨迹点只要能表示轨迹发生了改变，就应该包含到最终的压缩轨迹中。而如果一个点可以从之前的运动中通过插值或位置预测等方法计算出来，那么就可以将其舍弃。因此，他们认为在轨迹压缩过程中应该考虑速度和方向这些特征，因为这些可以用来预测轨迹的位置。因而中提出了一种阈值导向的采样算法来减少轨迹中的冗余数据点。基本思路是使用最后两个轨迹点以及速度及方向预设的阈值来计算一个安全区域，以此判断一个新采集到的轨迹点是否含有重要的信息。如果该点位于安全区域之内，则认为它是多余的，可以被舍弃; 如果该点位于安全区域之外，则将其保留在压缩轨迹中，因为此时运动方向或者速度已经发生了改变。

图2.4举例介绍了本算法。假设 $\boldsymbol{p}_0$ 和 $\boldsymbol{p}_1$ 包含在压缩轨迹中，由于各轨迹点本身包含了角度和速度信息。当采集到轨迹点 $\boldsymbol{p}_2$ 时，根据物体在 $\boldsymbol{p}_1$ 处的运动速度、运动方向、速度和方向的误差阈值，以及 $\boldsymbol{p}_1$ 与 $\boldsymbol{p}_2$ 之间的时间间隔，可以构建出一个扇形区域，即物体在采集到 $\boldsymbol{p}_2$ 点的可能位置。图中 $\boldsymbol{p}_2$ 在该扇形区域之内，因此它并没有蕴含太多信息，可以将其舍弃。接着，当采集到轨迹点 $\boldsymbol{p}_3$ 时，又可以构

造出该时刻的安全区域。由于 $\boldsymbol{p}_3$ 不在这个安全区域之内，因将它保留。依次类推，最终保留的压缩轨迹为 $\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_3, \boldsymbol{p}_4, \boldsymbol{p}_7, \boldsymbol{p}_9, \boldsymbol{p}_{10}, \boldsymbol{p}_{16}$。

2.2.3 基于路网结构的轨迹压缩算法

上述传统时间序列算法受限于基于欧式空间，而人、车辆等运动轨迹受限于路网结构。简单使用欧式距离的度量方式，不能刻画这些移动对象的行为。随着城市路网结构数据的完善，结合路网数据的轨迹压缩方法不断被提出。这些算法往往需要将轨迹数据进行地图匹配预处理 (Map Matching)，以将原始轨迹点映射到路网中。这样做的好处有：1. 轨迹点落在路网结构上更能直观表达位置信息。2. 连续变动的多个轨迹点可能对应同一个路段，因而使用路段表达能有效的降低数据存储开销。此外，一个城市中的路网线段是有限的，而采集的轨迹点时无限的。使用路网数据表达原始位置，能有效降低数据表示的复杂度。

Non-materialized 算法文献 [48] 提出了 Non-materialized 算法来压缩数据，他首先将原始轨迹变成路网序列数据，接着将上述结果转换成路口的序列数据，从而压缩轨迹。文献 [49] 提出了基于最短路径和链接相结合的压缩算法。所谓最短路径就是，如果一条移动轨迹与其起始和终点的最短路径相一致，那么可以用头尾两条路段来表示整条轨迹。链接的思想上，假设司机在两个路段间的行驶轨迹都是沿着最小转角的路段行驶，并将相邻两路段的头尾连接起来拼凑成更长的轨迹。

基于 MDL 模型的算法文献 [50, 51] 把轨迹压缩问题转换成一个最优化的问题，其目标是寻找一条近似轨迹，使得压缩率尽可能高，且该轨迹与原始轨迹相似度尽可能高。文章引入信息论中的最小描述长度模型 (Minimal Description Length，MDL)，来表示最优化压缩率和相似度组成的目标函数。MDL 模型是信息论中的常用模型，有两个部分组成，分别是 $L(H)$ 和 $L(D|H)$。$L(H)$ 为假设条件；$L(D|H)$ 为在此假设条件下，数据的描述情况。在对应到轨迹数据压缩中，$L(H)$ 表示压缩后的轨迹；$L(D|H)$ 表示近似轨迹与原始轨迹之间的误差. 根据 MDL 原则，要找到一条路径，使得 $L(H) + L(D|H)$ 最小，那么这条路径就是满足目标的路径。这两

部分的定义如下：

$$MDL(D,H) = L(H) + L(D|H) \tag{2.2}$$

$$L(H) = \log_2(\delta * \frac{|T_{MMTC}|}{T'}) \quad ; \quad L(D|H) = \log_2(\delta * D_{total}(T_{MMTC}, T')) \tag{2.3}$$

其中，$|T_{MMTC}|$ 代表压缩后的轨迹的长度，$|T'|$ 是原始轨迹经地图匹配后的轨迹，$D_{total}(T_{MMTC}, T')$ 是指 T_{MMTC} 和 T' 之间的距离。整个算法过程如下：从 T' 的起点 $\boldsymbol{p}_1$ 开始，把后面的点都遍历一遍，找到 $\boldsymbol{p}_1$ 与 $\boldsymbol{p}_i$ 之间的最短路径 SP_{1i}，计算 T'_{1i} 和 SP_{1i} 之间的 MDL_{1i}，找出使 MDL_{1i} 最小的 i 值。用 SP_{1i} 来近似替代 T'_{1i}。然后从 $\boldsymbol{p}_i$ 开始，重复上述步骤，直到轨迹的终点。把所有找到的 SP_{ij} 连在一起便是最终的 T_{MMTC}。该算法在高频率的采样数据集和稀疏的路网结构下有较好的效果。此外针对估计分布不均的情况，[52] 利用哈夫曼思想编码思想的轨迹压缩方法，即出现频率越高的轨迹段编码越长。

2.2.4 语义压缩算法

原始轨迹和路网轨迹尽管能很好地记录移动物体的运动踪迹，但是对于人们理解轨迹的含义却帮助不大。因为人们阅读轨迹时，无法直观了解经纬度坐标代表的意义。因此出现了利用 POI、标志物、路口和路段来表示车辆的行驶过程，人们阅读语义轨迹时，能明确知道轨迹的起点、终点以及行驶经过的路段。

基于事件的压缩算法文献 [53, 54] 提出了仅保存有意义的状态和事件的语义轨迹压缩 (Semantic Trajectory Compres sion，STC)。它把一条轨迹拆分为若干连续的事件。事件包括行驶的路段，如从边 A 直走到达边 B，以及方向的改变，如在 B 路口左转到达边 C。并且这种描述方法，会用一条边来表示若干个点，以而达到压缩的目的。但这种方法的压缩和解压缩时间会比较长，且丢失原始的经纬度信息，仅保留的物体的一部分运行状态。因此，其支持的 LBS 应用也是有限的。但优也很明显，压缩后的语义便于轨迹便于阅读和理解。

基于摘要的压缩算法文献 [55] 针对现有语义压缩方法未考虑速度、方向等问

题，提出了利用轨迹摘要数据的压缩框架。在文章中，他们考虑了道路等级、路宽、速度、停车次数和 U 形转弯次数这六个特征表示摘要数据。其框架包含 Partition 和 Summarization 两个部分。在 Partition 阶段，根据之前提取的 6 个特征，把轨迹切分为几段子轨迹以使得每段子轨迹内部具有相似的特征，同时子轨迹之间的特征差异较大。在每段内，选取最具代表性的特征来描述这段的移动过程。在 Summarization 阶段，使用每段的特征来描述轨迹形式过程。该通过提取轨迹概要数据，不仅压缩了数据量，而且也便于人们理解轨迹的行为。文献 [56] 介绍了该算法的演示系统，要求输入一条原始轨迹序列，并输出一段描述性文字，大体描述了轨迹的行驶特征以及经过的重要位置。语义压缩后得到的轨迹，便于阅读理解且显著地减少了空间开销。但缺点是丢失了具体的经纬度信息，从而不能支持具体的点查询。

2.3 轨迹距离度量

轨迹距离的定义是轨迹挖掘中的核心内容之一。其度量方式可以分为 3 大类：传统时间序列距离、针对轨迹时空特性专门设计的距离（简称为时空轨迹距离）、以及针对轨迹语义属性设计的距离（简称为语义轨迹距离）。下面我们将介绍每一类的经典度量方式及其特点。

2.3.1 传统时间序列距离

传统时间序列距离广泛应用于时间序列数据分析上如：语音识别、股票期货等价格分析。其典型距离有 L_p 距离、动态时间卷曲距离和最长公共子串距离。

L_p 距离: L_p（闵可夫斯基距离）是 L_1(曼哈顿距离)、L_2（欧式距离）以及 L^∞（切比雪夫距离）的总称。使用这一类距离的假设是，两条轨迹具有相同的长度。欧式距离是最常用的一类 L_p 距离，其计算方式是对应轨迹点间欧式距离的累加和。特别的，给定两条轨迹分别为 $\mathcal{Q} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_n\}$ 和 $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_n\}$，其对应的任意一对点 $\boldsymbol{q}_i$ 和 $\boldsymbol{p}_i$ 间的欧式距离定义如下:

$$ED(\boldsymbol{q}_i, \boldsymbol{p}_i) = ||\boldsymbol{q}_i - \boldsymbol{q}_i||_2 \tag{2.4}$$

$||\boldsymbol{q}_i - \boldsymbol{q}_i||_2$ 为向量的二范数。在此基础上，两条轨迹间的欧式距离如下。

$$ED(\mathcal{Q}, \mathcal{P}) = \sum_{i=1}^{n} ED(\boldsymbol{q}_i, \boldsymbol{p}_i) \tag{2.5}$$

轨迹间欧式距离似对应点欧式距离的累加和。拓展到 L_p 距离，其距离为对应点的 p 范数距离的累加。以欧式距离为代表的 L_p 距离具有计算简单的有点。其缺点是结果易受噪声数据的影响。

动态时间卷曲距离: 动态时间卷曲（DTW）距离不要求两条轨迹具有相同的长度。因此，给定两条轨迹分别为 $\mathcal{Q} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_n\}$ 和 $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_m\}$ 动态时间卷曲距离 (Dynamic Time Warping, DTW) 由如下递归表达式定义2.6。它尝试将将两条轨迹的点以最小的距离和对应起来。

$$DTW(\mathcal{Q}_{1,n}, \mathcal{P}_{1,m}) = ||\boldsymbol{q}_n - \boldsymbol{p}_n|| + min \begin{cases} DTW(\mathcal{Q}_{1,n-1}, \mathcal{P}_{1,m-1}) \\ DTW(\mathcal{Q}_{1,n-1}, \mathcal{P}_{1,m}) \\ DTW(\mathcal{Q}_{1,n}, \mathcal{P}_{1,m-1}) \end{cases} \tag{2.6}$$

其中 $\mathcal{Q}_{1,n-1}$ 代表轨迹 $\mathcal{Q}$ 从第 1 到第 $n-1$ 个点构成的子轨迹。动态时间卷曲距离不满足三角不等式，因此它不是一个度量准则。

最长公共子串距离: 最长公共子串距离也不要两条轨迹具有相同的长度。其定义为如下递归表达式。

$$LCSS(\mathcal{Q}_{1,n}, \mathcal{P}_{1,m}) = \begin{cases} 0 & if\ n = 0 \vee m = 0 \\ 1 + LCSS(\mathcal{Q}_{1,n-1}, \mathcal{P}_{1,m-1}) & if\ ||\mathcal{Q}_n - \mathcal{P}_m|| < \epsilon \\ \max\{LCSS(\mathcal{Q}_{1,n-1}, \mathcal{P}_{1,m}), & \\ LCSS(\mathcal{Q}_{1,n}, \mathcal{P}_{1,m-1})\} & otherwise \end{cases} \tag{2.7}$$

跟以上距离相比，最长公共子串距离对于数据含噪声的情况处理结果更加鲁棒。但它需要额外参数 ϵ 来约束两个点空间的相似性。最近提出的 EDR 和 EDP 距离同样

也是设计出来处理带噪声的数据，其处理噪声的效果比最长公共子串距离还好。

2.3.2 时空轨迹距离

相比传统时间序列数据，轨迹数据独有空间属性。为刻画空间属性的影响，许多工作提出了新的距离。其中典型的有基于最小包围盒的距离、霍斯多夫距离和弗雷歇距离。

基于最小包围盒距离： 基于最小包围盒距离利用给定轨迹线段间的最小包围盒来快速计算轨迹线段间的距离。假设 B_1 和 B_2 分别代表轨迹线段 L_1 和 L_2 间的最小包围盒。$D_{min}(B_1, B_2)$ 距离代表 B_1 内任意一轨迹线段与 B_2 内任一轨迹线段间距离的下界。其计算方式如公式2.8所示。

$$D_{min}(B_1, B_2) = \sqrt{(\Delta(B_1.[x_l, x_u], B_2.[x_l, x_u]))^2 + (\Delta(B_1.[y_l, y_u], B_2.[y_l, y_u]))^2} \quad (2.8)$$

其中两个区间的距离定义如下：

$$\Delta([l_1, u_1], [l_2, u_2]) = \begin{cases} 0 & if\ [l_1, u_1] \cap [l_2, u_2] \neq 0 \\ l_2 - u_1 & if\ u_1 < l_2 \\ l_1 - u_2 & if\ u_2 < u_1 \end{cases} \quad (2.9)$$

霍斯多夫距离： 霍斯多夫距离是用来衡量两个轨迹段的相似度。给定轨迹段 L_1 和 L_2，霍斯多夫距离定义如公式2.10所示，表示为 3 个带权重距离的累加和。其中 $d_\perp$ 表示两个轨迹段的垂直距离，$d_{||}$ 代表轨迹段间的平行距离，d_θ 代表轨迹段间的夹角距离。这些几何意义如图2.5所示。而 $w_\perp$、$d_\perp$ 和 w_θ 为上述三个距离在距离的权重，这些权重值可以随着应用场景的不同而调整值。

$$Hausdorff(L_1, L_2) = w_\perp \cdot d_\perp + w_{||} \cdot d_{||} + w_\theta \cdot d_\theta$$
$$d_\perp = \frac{d_{\perp,a}^2 + d_{\perp,b}^2}{d_{\perp,a} + d_{\perp,b}}, \qquad d_{||} = \min\{d_{||,a}, d_{||,b}\}, \qquad d_\theta = ||L_2|| * \sin\theta \quad (2.10)$$

在上述公式中 $d_{\perp,a}$ 和 $d_{\perp,b}$ 分别表示 L_1 和 L_2 间的垂直距离，$d_{||,a}$ 和 $d_{||,b}$ 分别表示

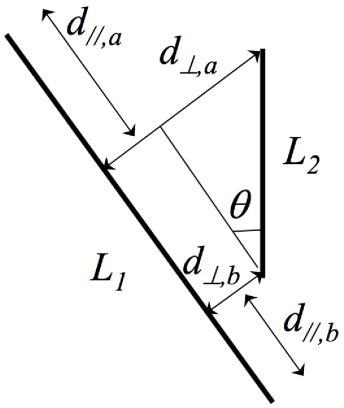


图 2.5: 霍斯多夫距离

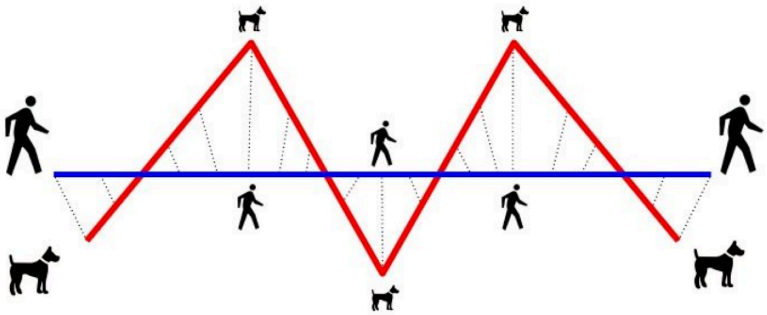


图 2.6: 霍斯多夫距离

L_1 和 L_2 间的平行距离，θ 代表两线段间的夹角。

弗雷歇距离: 弗雷歇距离通俗的讲就是狗绳距离。如图2.6所示人和狗的轨迹，两者之间有一条狗绳约束。主人走路径 A，狗走路径 B，各自走完这两条路径过程中所需要的最短狗绳长度就是弗雷歇距离。

2.3.3 语义轨迹距离

语义轨迹的特点是将原始轨迹中的坐标映射到一个语义信息上，如给点标注为“学校”。从而使得原始的坐标序列变为语义信息的序列。现有语义轨迹度量方式的思想就是从语义和空间两个角度来考虑相似性，但它们的实现方式各不相同。

文献 [57] 提出了基于位置分类属性的语义轨迹分析方法。如图2.7所示，该方法首先将 POI 分成博物馆、医院、学校等若干个类别，然后给原始轨迹的每个点赋予类别属性。最后使用类别属性的序列进行相似度计算。

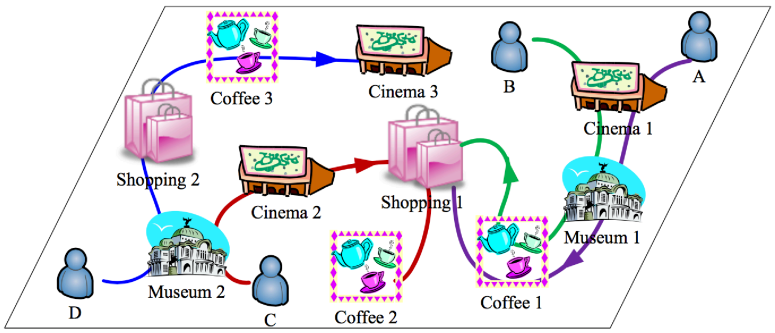


图 2.7: 基于分类属性的语义轨迹

文献 [27] 利用轨迹空间距离和语义距离的比值来作为最终两个轨迹的相似度。其计算方式如公式2.11所示，其中 $geoDist$ 代表了两条轨迹的空间距离，$semRatio$ 代表了两条轨迹间的语义距离。其空间距离考虑了轨迹中心点间的距离、轨迹长度的差别和轨迹间的角度。其语义距离使用最长公共子串距离除以短轨迹长度来表示。α 用于调节着空间和语义距离的比重。

$$totalDist(t_1, t_2) = geoDist(t_1, t_2) \cdot \frac{1}{1 + \alpha \cdot semRatio(t_1, t_2)} \tag{2.11}$$

文献 [29, 58, 59] 给空间距离和语义距离以不同权重，并用它们的权重和作为轨迹或轨迹点的相似度。其定义形式如公式2.12所示。

$$totalDist(t_1, t_2) = \alpha geoDist(t_1, t_2) + (1 - \alpha) semRatio(t_1, t_2) \tag{2.12}$$

其中文献 [58] 中要求被查询轨迹要完全包含待查询轨迹的所有语义信息。文献 [59] 不是给每个点一个标签属性，而是给每条轨迹一个标签信息。而文献 [29] 给出的是近似查询结果。

2.4 k 近邻轨迹查询

关于 k 近邻轨迹查询的研究工作有很多，我们将这些工作分为两大类：集中式环境下查询和分布式环境下查询，并分别作介绍。

2.4.1 集中式环境下查询

轨迹数据上的 k 近邻查询，已经得到广泛研究。我们将这些工作分为：历史数据上的查询，轨迹流上的连续查询以及不确定轨迹上的查询。

历史数据上的查询

目前已经有一些针对历史数据的轨迹管理系统以支持轨迹查询 [60–62]。Botea 等 [60] 首先构建了针对轨迹数据的管理系统 PIST。该系统将轨迹数据以点为基本单位构建空间索引。它首先统计轨迹数据在空间维度的分布，然后构建空间索引以对数据进行划分。接着对每个划分内部的数据根据时间维度构建索引。Chakka 等 [61] 提出了以子轨迹为基本单元的管理系统 SETI。该系统同样首先根据统计数据构建空间索引。然后根据空间索引划分轨迹数据，并将每个划分内同一移动对象的子轨迹作为基本处理单元构建时空索引。该工作表明，基于子轨迹的索引策略比基于点的策略查询效率要高。进一步地，Philippe 等 [62] 提出了基于 I/O 开销模型的轨迹数据管理系统 TrajStore 以支持动态轨迹划分。TrajStore 对每个划分内的子轨迹进行聚类并用簇中心来代表簇内所有轨迹，从而降低数据的存储开销。此外，它还使用了多种数据压缩技术以进一步降低存储开销。Wang 等提出针对大内存服务器的轨迹数据系统 SharkDB[63–65]。该系统提出了 3 种存储框架用以降低数据存储开销同时增大内存命中率。此外引入了分层索引以提高查询效率。上述轨迹系统需进行功能扩展以实现近邻轨迹查询。

此外集中式邻轨迹查询已经得到广泛研究。Agrawal 等 [66] 首先使用离散傅里叶变换来转换轨迹数据并保留最开始的几个频率作为轨迹特征。原始轨迹上的 k 近邻查询，可通过先在特征值上的查询结果来进行剪枝，以提高查询效率。Refiei 等 [67] 使用傅里叶描述子来表示轨迹形状的边界，接着对每个形状计算出多维指纹信息并对指纹数据构建 R 树索引。最后使用指纹间的距离来近似原始轨迹间的距离。该方法计算出来的距离不受位置，方向和轨迹起始点的影响。Pelekis 等 [68] 针对仅考虑空间维度和同时考虑时空维度两个方面定义了轨迹相似度，并分别提出了处理方法。此外，还提出了考虑速度、加速度和方向的方法。

Frentzos 等 [69] 提出了基于 R 树索引的查询算法用以处理近邻轨迹查询，并比较了当使用不同类型 R 树索引对查询效率的影响。Xu 等 [70] 提出了基于剪枝-提炼的解决方案。在剪枝阶段其同样利用 R 树索引来进行剪枝。具体地，其为树的每个叶子节点设计一个时间独立且收敛的覆盖函数，并在查询过程中通过计算查询对象与节点包围盒的距离更新该函数值。最终返回满足查询条件的轨迹段。在提炼阶段，将上一步获得的轨迹段按照到达时间进行排序并合成完整的轨迹。文献 [71] 提出了最近邻轨迹预测问题，其目标是预测接下来一段时间内，与给定查询轨迹最相似的轨迹。其使用 kd 树和 B+ 树来构建对偶索引以提高预测速度。Ranu 等针对轨迹数据的异频采样和无固定采样频率的问题，设计了新的轨迹距离计算方法 EDwP，并根据该距离设计了新的索引策略用于剪枝查询结果。Chen 等 [72] 提出了 k 最好连接轨迹轨迹查询问题。其目标是针对给定的一组有序或无序位置信息，找出 k 条其某条采样子轨迹与给定查询最匹配的轨迹。该算法首先给出了满足查询条件的轨迹距离定义方式，然后基于点最优匹配的查询算法。该算法针对索引树，提出了基于查询开销的最优优先和深度优先两种查询策略。最后比较验证了两种策略的性能。

此外，Skoumas 等 [73] 使用豪斯多夫距离来计算轨迹段的相似度，并用轨迹段的累加值作为最终轨迹间的相似度。因此，其首先提出了一个基于优先队列的剪枝算法 $ExactKNN$。该算法通过前几个时间戳上的数据来构建相似度上、下界，并维护一个上界的优先堆（大堆），当堆顶元素下界大于次小元素的上界时，则该元素加入最终的结果集并从堆中移除。否则，不段对堆顶元素增加更多时间戳内地数据以更新上、下界。该算法直到找出 k 个结果才会停止。该方法，虽然能找出精确的结果但也存在着迭代次数多的问题。为此，提出了两个近似算法以提高查询效率。第一个是基于简化轨迹的算法。其首先使用 [74] 所提技术简化轨迹，简化后的轨迹近保留有限个时间戳上的数据。然后对简化轨迹调用 $ExactKNN$ 算法。该方法虽然会导致部分结果不正确，但能大大降低计算开销。第二个是基于先验概率的加速算法。其首先计算出轨迹段的分布概率，接着在计算轨迹段距离上、下

界的过程中引入该分布。使得包含先验概率高轨迹段的轨迹得到优先扩展。

目前，还有一些针对语义轨迹的 k 近邻查询研究 [57, 58, 75]。Xiao 等 [57] 提出了基于语义轨迹的 k 近邻查询。其首先将原始的位置序列轨迹通过停留区查找算法变成停留区序列 (即语义轨迹)。接着对语义轨迹提出了匹配的定义并基于次给出两条语义轨迹的距离度量。最后，将语义轨迹间距离度量的问题转化为图的构建问题。Zheng 等 [58] 提出了 k 近邻活动轨迹的查询研究。其目标是给定一组位置序列，每个位置带有多种活动标签（这样的序列称为活动轨迹）。其目标是从用户活动轨迹数据集中找出与给定序列距匹配度最高的 k 条。为此，其分别定义了基于点和轨迹的匹配度量方式，接着给出了匹配距离的下界并证明了通过下界能快速剪枝候选。最后给出了两种算法以分别处理查询点集合有序和无序的场景。该工作所提查询要求返回的轨迹必须完全包含所有待查询活动标签的类别，只要有一个标签不包含，则认为返回的轨迹与查询不相似（相似度为 0）。Wang 等 [75] 等在此基础上进行了改进，认为一条轨迹包含待查询轨迹内容标签越多，则相似度越高。此外，该工作还给每个位置上的标签赋予一定的权重以满足不同用户的需求。为解决所提查询，他们首先提出了相似度上、下界的增量式算法。但该方法存在着迭代次数多的问题，为此提出了动态扩展的策略以给每个查询点以不同的扩展范围。进一步地，为解决每次扩展中需要对索引树进行深度遍历的问题，提出了两层阈值算法，以保证一次搜索就能找出满足条件的候选。

轨迹流上的连续查询

此外，Sacharidis 等 [76] 还研究了轨迹数据流上的 k 近邻轨迹查询。该工作使用滑动窗口来处理轨迹数据流，并使用窗口内每个时段点的距离的聚集值（最大、最小、累加、均值等）作为轨迹相似度。其首先提出了基于事情模型驱动的通用处理方法 BSL。该方法关注如下三类事件：（1）待查询移动对象的位置变化（Query location updates,$QUpd$），（2）被查询移动对象位置的变化 (Object location updates, $OUpd$) 含新位置到达和旧位置过期,（3）移动对象与被查询移动对象位置过期 (Object distance expiration, $OExp$)。BSL 为查询对象保留最后时刻的位置，为

其他移动对象保持最后的位置和窗口内的距离序列（每个时间戳对应一个其余查询对象的距离）。此外还维护了一个事件列表以便及时删除过期的位置信息。当事件某一类型事件到达时，按照其设计的方案更新轨迹间的距离。接着，提出了改进算法 XTR 用以去出无用点（不会影响到最终轨迹间距离值的点）以减少遍历时间。最后，提出了基于移动对象最大以速度改进算法 HRZ。该方法利用最大移动速度约束，以剪枝候选。XTR 和 HRZ 这两个改进方法只适用于使用最大和最小距离作为轨迹距离的情况。

Bakalov 等 [77] 研究了轨迹数据流上的连续 join 查询。其目标是针对两个不断变化的轨迹数据集，数据集间的轨迹进行两两相似度匹配并返回最相似的，且相似度有轨迹空间邻近度决定。其首先使用轨迹近似技术来降低数据的维度并对近似轨迹构建空间索引。接着对近似轨迹设计了距离下界用以进行剪枝。针对连续查询，设计了两阶段处理方法。第一阶段根据当前窗口内的数据初始化查询结果。在该阶段使用索引来降低匹配的个数并利用下界得到近似结果。第二阶段针对数据的变换，持续检查结果集并作出更新。

不确定轨迹上的查询

首先，隐藏时间序列数据和不确定数据流上的概率相似度查询 [78, 79] 上的查询已经被广泛研究。这类工作首先提出概率距离的计算方式，其需要设置两个阈值参数。一个是距离阈值 γ 另一个是概率阈值 τ。其查询目标是返回与查询轨迹距离小于 γ 的概率超过 τ 的轨迹。因此，查询结果对这两个阈值比较敏感。阈值作轻微改变可能查询结果的大小和内容发生较大变化。

此外，一些专门针对不确定轨迹数据上的查询也被提出 [80, 81] 文献 [80, 82] 研究了给定不确定轨迹数据集上的连续近邻查询。其假设查询与被查询的轨迹都是不确定的，但轨迹在给定范围内满足一定的概率帆布。其目标是不断返回与待查询轨迹概率距离最高的 k 条轨迹，并找出结果集内容或顺序发生变化的时间点。其主要思想是构建一棵几何对偶的树索引以剪枝候选。Ma 等 [81] 研究了不确定轨迹数据上的 k 近邻轨迹查询。其与文献 [80] 的研究内容差别是其给定的查询轨迹

是确定的。该工作假设所处理的轨迹数据位置具有一定的误差，且误差满足一定的概率分布。为度量两条不确定轨迹的相似度，其首先提出了 p 距离。给定轨迹 x 与查询 $\mathcal{Q}$ 的 p 距离由其他比 x 离 $\mathcal{Q}$ 更近的轨迹的概率距离和表示。即 x 离 $\mathcal{Q}$ 越近，越少的轨迹比 x 更近。接着提出了基于空间网格划分的 $UTgrid$ 索引，并对每个网格内的轨迹构根据时间维度构建 1 维 R 树索引。最后设计了针对不确定轨迹的查询算法，该算法利用 $UTgrid$ 进行剪枝以达到提高查询效率。与以上工作不同地是，Li 等 [83] 研究了基于路网的不确定轨迹查询。该工作中假设移动对象在路网中的移动速度未知。因此如何预测哪些轨迹会成为将来的候选集成为难点。

2.4.2 分布式环境下查询

为处理轨迹大数据，分布式环境下的 k 近邻轨迹查询得到越来越多的研究。我们将从分布式轨迹数据查询系统和分布式近邻轨迹查询两方面进行介绍。

分布式轨迹数据查询系统近 10 年来，随着 Hadoop[1]、OpenStack[2]和 Spark 等大数据处理开源系统的普及，在这些系统上构建轨迹轨迹数据管理系统成为新的方向。目前，已经出现了大量基于这些开源分布式平台的轨迹数据管理系统以支持轨迹查询。

Lu 等 [84] 设计了并行的空间数据 DBMS 系统 Parallel-Secondo。该系统在每台节点上使用传统的空间 DBMS 来实现数据存储和查询实现，仅使用 Hadoop 作为结点间任务调度器。Ma 等 [85, 86] 首先提出了基于 Hadoop 的轨迹数据管理系统。为提高查询效率，其构建了两类索引。第一类是基于空间的索引，用于将数据按空间维度划分。此类索引能加速带有空间约束的查询，但如果查询某个移动对象的轨迹，则需要遍历许多数据分区。为此设计了第二类基于移动对象的倒排索引，该索引中存储了每个移动对象的轨迹数据是存储在哪些数据分区上。Clost[5] 通过建立一个多层索引用以数据划分。同时，其支持数据的批量增加。SpatialHaddop[87] 系统是基于 Hadoop 的典型空间数据管理系统。该系统从以下四个方面对 Hadoop

[1]http://hadoop.apache.org/

[2]https://www.openstack.org/

进行了改进以满足空间数据查询的需求。（1）提供了基于多种索引树的存储策略。首先根据全局索引用以完成数据划分，其次在每个数据块内支持块内局部空间索引。这种全局-局部的索引方式能大大减少 I/O 开销；(2) 设计了新的文件读写方式，以支持对块内索引数据的读写；（3）设计了多种查询接口，以方便开发这进行查询调用；（4）设计了基于 Pig Latin 的查询语言，以方便普通用户实现查询功能。但 SpatialHadoop 只能用来管理静态数据，当新的数据到达时，需要将整个数据集重新划分，因而开销较大。AQWA[88] 是其改进版本，允许数据动态添加。AQWA 建立了新的数据划分模型，同时考虑了用户查询和数据分布对划分的影响。以上系统最终将查询需求，变成定制的 Map/Reduce[89, 90] 任务。

此外，还有一些基于 Hadoop 组件或类 Hadoop 系统的时空数据管理系统。Ablimit 等 [91] 等提出了基于 Hive[3]的轨迹管理仓库 Hadoop-Gis，其通过网格索引以提高范围和 join 查询的效率。MD-HBase[92] 和 R-HBase[93] 是两个基于 HBase[4]设计的实时时空数据管理系统。MD-HBase 使用 Z-Curve 空间填充曲线来划分数据空间。接着对所用划分子空间使用四叉树或 k-d 树索引来构建空间索引。R-HBase 使用了局部性更好的希尔伯特空间填充曲线来划分数据空间，并使用 R 树来构建空间索引。这两个系统借助了 HBase 的特性可支持数据的实时插入和修改。Geomesa[94] 与以上两个系统类似，可以快速部署到类似 HBase 的键值对存储系统中，且其从语言层定义了空间查询操作接口以方便用户调用。相较于以上系统，Elite[95] 专门设计用于处理不确定轨迹数据并提供了丰富的不确定查询接口。

尽管基于 Hadoop 生态圈内系统构建的分布式轨迹管理系统能满足轨迹查询的需求，但也面临着由于 I/O 开销较大，导致无法提供实时、高吞吐率的查询服务。为此，出现了许多基于 Spark 等分布式内存的时空数据管理系统，以提供实时和高吞吐率的查询服务。SpatialSpark[96] 和 GeoSpark[97] 是最早的两个基于 Spark 设计的专门用来处理空间数据查询的系统。其中 SpatialSpark 专门用来出列空间 join 查询。GeoSpark 设计了一个新的数据结构 SRDD 用来表示分布式空间数据。

[3]https://hive.apache.org/

[4]https://hbase.apache.org/

SRDD 使用全局空间索引来划分数据，且支持每个分区内部构建局部索引。进一步地，LocationSpark[98] 提出了使用布隆过滤器来解决数据分布倾斜问题。最新地，Dong 等人提出了新的基于 SparkSQL 的空间数据管理系统 Simba [99]。相比现有基于 Spark 的系统，Simba 从查询解析到查询执行都作了优化。其中最明显的是在其查询执行计划设计方面使用了空间索引等其技术来提高查询效率。此外，它还从语言层提供了用户查询接口。相比以上系统，TrajSpark[6] 是专门设计用来处理轨迹数据的系统。其实现了 3 类典型的轨迹查询接口，并利用全局-局部索引策略以提高查询效率。此外，目前还存在着专门用于对时空数据进行复杂分析的系统 [100, 101]。STORM[101] 设计用来处理交互式近似查询。相比于已有的设计用来精确查找的系统，该系统能在查询提出后，通过逐渐增加采样数据以不断以一定置信度给出查询结果。用户根据系统不断精细的结果随时可以停止查询。OceanST[100] 同样适用采样技术来获得查询近似结果，且能处理数据不断增加的场景（STORM 仅能处理静态历史数据）。此外，它还提供了丰富的查询接口用以满足各种查询需求。

分布式近邻轨迹查询

目前，已经出现了基于 Map/Reduce 并行计算框架下时间序列相似性问题的研究工作 [33, 33]。文献 [33] 研究了基于 Map/Reduce 框架的集合数据的 join 查询问题，其目标是减少数据节点间的通信问题。文献 [33] 针对 Map/Reduce 框架下数据集的 top-$k$join 查询问题进行了研究。其首先提出了基于分治法和分支限界法的两种实现策略。接着提出了全部配对划分和有限配对划分的数据划分策略以减少 map 和 reduce 任务间的通信开销。此外，还有一些使用多通用计算图形处理器（GPGPUs）并行计算框架的工作 [102–104]。其中文献 [102] 研究了范围查询即找出与查询轨迹距离小于给定阈值的轨迹。其提出了新的适用于 GPU 的索引结构有一替代传统的 R 树索引。给定一个查询集合，它提出了多个方法以将查询分成若干互相独立的批次以得到更好的内容命中率和降低计算的开销。文献 [103] 构建了针对轨迹数据的管理系统并研究了基于霍斯托夫距离的轨迹相似度 join 查询。其主要贡献在于

设计了四层轨迹数据表现方式并设计了一个 3 层所以架构用以提高查询效率。Leal 等 [104] 首先提出了基于霍斯托夫距离的 k 近邻轨迹查询算法 TKSIMGPU，其主要研究目标是使得各个计算线程负载均衡。Top-KaBT[105] 是 TKSIMGPU 的改进版本，其提出了轨迹间霍斯托夫距离上、下界用以减少需要计算精确距离的候选数。

针对协调者-远程结点分布式架构，一些轨迹序列 top-k 查询算法已经被提出 [20, 21, 106]。文献 [106] 研究针对手机信令轨迹数据进行了 k 近邻研究。其将每个手机基站看作远程结点，用户移动的过程中会经过不同的基站，因而其轨迹数据会分成若干段并保留在不同的节点上。为处理针对这一数据的查询，其提出的算法在每个远程结点保存子轨迹距离的上、下界以提高查询的效率。Smart Trace[21] 和 Smart Trace+[20] 两个系统是针对众包场景下的查询。其将每个用户的手机看作远程结点，因而每个远程结点只有一条轨迹数据。针对这一场景，Smart Trace 提出了基于 LCSS 距离的上界用以查询时间和能量消耗。进一步地，Smart Trace+ 还提出了新的下界并提出了结合上、下界的处理算法。以上工作均是将提高查询效率作为首要目标，并未深入所提算法在通信问题上的开销。

为解决协调者-远程结点架构下，k 近邻轨迹查询中的通信开销过大的问题。基于多粒度概要数据的方法已经被提出 [18, 19, 19, 107, 108]。Papadopoulos 等 [107] 首先对分布式时间序列进行了 k 近邻查询研究，并提出了 4 种基于协调者-远程结点的方案以求在提高时间效率的同时降低通信开销。但该方法仅能降低从远程结点返回给协调者结点的数据开销，协调者结点仍需要将待查询时间序列发送给所有远程结点，而该开销是所有算法通信开销的主要部分。因此，其并未彻底解决通信开销过大的问题。Yeh 等 [108] 提出了用于降低通信开销的算法 LEEWAVE。其使用小波变换的数据压缩放，将原始时间序列数据变换成多粒度的概要数据。LEEWAVE 算法中协调者结点将概要数据由粗到细发送给远程结点，远程结点根据概要数据计算候选轨迹与查询轨迹欧式距离范围的必要参数，并将参数发送给协调者结点用以计算距离上、下界并进行剪枝。其根据多粒度数据不断缩小距离范围

的方法能大大降低通信开销。文献 [18, 109] 提出了更紧的下界，因而能打到更好的剪枝效果。文献 [18, 108, 109] 都是基于哈尔小波变换的压缩方式，而该方式仅适用于距离度量方式是欧式距离的场景。此外，文献 [19] 提出了针对 DTW 距离的算法，该算法引入多粒度包围信封以作为概要数据，并根据该概要数据计算下界。此外，使用级联下界的方式以进一步降低计算开销。而文献 [19] 在其基础上提出了新的上界，并引入了同时根据上、下界进行剪枝的算法 MDTK。MDTK 算法引入了边剪枝边过滤的思想用以加快下界的计算速度。以上工作都是针对某一具体距离度量方法提出相应的基于概要数据的距离上、下界，并未考虑如何扩展到其他轨迹距离的度量方式上。为此，亟需提出通用的方法，以解决分布式 k 近邻轨迹查询。

第三章　查询处理框架

本章主要介绍两个查询处理框架以分别处理不同应用场景。首先，章节3.1阐述了本章的研究思路。其次，章节3.2介绍了协调者节点和远程结点上的通信接口函数。然后，章节3.3介绍了同时利用距离函数上、下界的 FTB 框架，并给出了实现方案。接着，章节3.4介绍了仅利用距离函数下界的 FLB 框架，并给出了实现方案。最后，小结本章的研究内容。

3.1　引言

本文的主要研究目标就是降低 top-k 查询时的网络开销，直接将查询轨迹发送到所有远程结点的方式虽然能保证结果的正确性，但网络传输开销消耗较大。为降低这一开销，一个直观的思路是：协调者结点选用合适的降维策略首先对轨迹数据进行降维处理，得到一个概要数据并将该概要数据发送给所有远程结点。远程节点在获取到概要数据后，我们希望根据概要数据能够对查询轨迹和局部轨迹能计算出距离的范围以进行剪枝。此外，我们希望通过降维得到的概要数据具有多粒度特性，即粒度越细，所包含的信息量也越多。同时，所计算出来的距离范围也越精确。通过不断精确的距离范围，我们能进一步剪枝不相关的候选。整个剪枝过程如图3.1所示，随着概要数据从粒度 1 到粒度 n 的不断传输，整个系统会过滤掉越来越多的候选。这里有两点需要注意：(i) 粒度越高，对应的概要数据所能表示的信息量也越接近原始轨迹数据，（iii）在由粗到细剪枝过程中，可能不需要到最细粒度，结果已经找出。

从以上思路出发，本章从所得到的距离范围出发考虑了两种情况：（i）根据概要数据能同时计算出所给距离函数的上界和下界,（ii）根据概要数据仅能计算出所给距离函数的下界。至于仅能计算出上界的情况不作考虑，这是因为我们需要找的是距离最近的 k 个轨迹。为此，我们首先给出了上、下界特征定义。

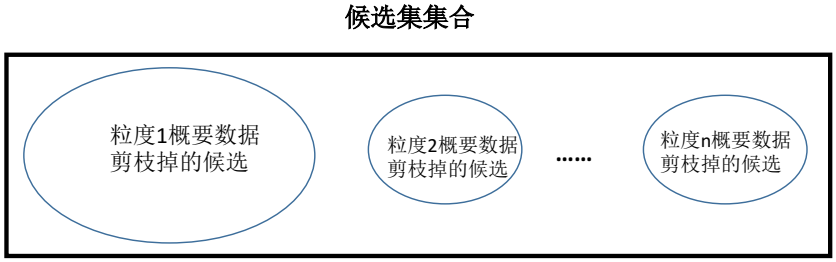


图 3.1: 逐步剪枝策略

定义 3.1.1. *界特征（Bound Feature,BF）. 界特征，$\langle id, ub, lb\rangle$，记录了一条轨迹的标识（$id$），以及它与待查询轨迹的距离上界（$ub$）和下界（$lb$）。*

对任意一条轨迹其默认的距离下界为 0，上界为正无穷。我们的目标是不断提高界特征的下界或降低界特征的上界。此外，根据前面所述，我们有时不能同时获得一个较紧的上界和下界。此时，只需使用其保留下界信息。在本章，我们将提出两个查询处理框架。其中前一个框架中将同时使用距离函数的上界和下界。而后一个用来处理仅有下界存在的情况。

3.2 接口函数

在介绍如何进行查询处理框架之前，本节将首先介绍，查询处理所用到的基本通信和处理接口函数。这些接口均为抽象函数，所以在具体应用中，用户可以添加具体的内容（我们再接下来两章将会介绍这些接口函数在使用欧式和动态时间卷曲距离下的具体实现）。我们将这些接口函数按照运行所处位置分为两类。一类是协调者结点上接口函数，另一类是远程结点接口函数。

协调者结点函数接口：

- coordinatorInit($\mathcal{Q}, \mathcal{R}$). 协调者结点运行该函数以实现查询初始化，涉及的内容包含对查询轨迹 $\mathcal{Q}$ 进行概要数据计算以及协调者结点跟跟所有远程结点的信息传递。

- generateInfo(). 协调者结点生成所需要发送的数据。

- sendToRemoteSites($\mathcal{R}, \S$). 协调者节点将数据 x 发给所有在集合 R 中的远程节点。

- getFromRemoteSites($\mathcal{R}$). 协调者结点从集合 $\mathcal{R}$ 中的每个远程结点获取信息。

协调者结点函数接口：

- remoteInit(TS_r, S_r). 远程结点运行该函数以实现初始化，主要工作其为保存在集合 TS_r 中的每条轨迹初始化界特征，并将这些特征值放在集合 S_r 中。此外，该函数还可能涉及到对每条轨迹数据进行概要数据计算。

- getFromCoordinator(). 该函数接受协调者结点运行 sendToRemoteSites 函数时所发数据。

- sendToCoordinator(x). 远程结点将信息 x 发送给协调者结点。协调者节点则通过 getFromRemoteSites 函数接受所发消息。

- updateBounds(S_r, x). 远程结点根据获取到的信息 x 更新界特征集合中值。

以上接口函数，为协调者结点与远程结点间的通信提供了保障。接下来的章节，我们将介绍如何利用这些接口函数所传递的信息来进行剪枝。

3.3 FTB 框架

本节将介绍 FTB 框架的设计原理及实现方案。

3.3.1 FTB 框架设计原理

本章第一节提到过，我们假设能对查询对象找到一个数据存储空间较小的概要数据，并利用该概要数据同时计算出距离的上、下界。FTB 框架核心思想就是从这点假设出发，具体的是：使用由粗到细粒度的概要数据，以不断获取更加紧凑的距离上、下界，并以此来进行剪枝过滤。图3.2给出了使用多粒度概要数据不断逼近与某一候选轨迹真实距离的过程。图中，横坐标为概要数据的粒度，纵坐标表示

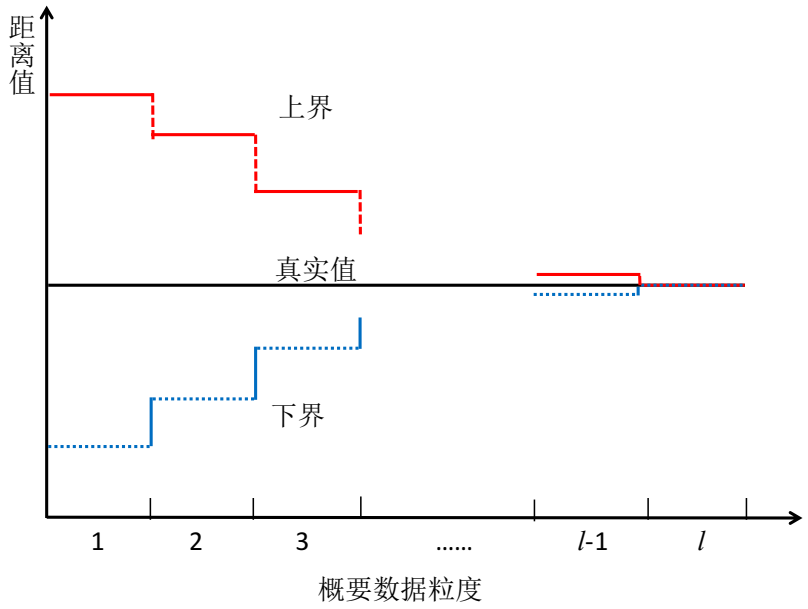


图 3.2: 多粒度概要数据逼近与某候选轨迹的真实距离

相似度距离的值。当概要数据粒度，不断增加的过程中，我们所计算出的相似度上界（红色实线）和下界（蓝色虚线）不断逼近相似度的真实值（黑色实线），并最终等于真实值。当获得了所有轨迹的界特征（上、下界）后，我们就可以使用它们进行剪枝。

引理 3.3.1 (剪枝原理). 给定一界特征集合 S，若某界特征的下界大于集合中第 k 小的上界，则该界特征所对应轨迹不会进入最终结果集，即可以被剪枝掉。

证明. 对 S 中的界特征按上界值由小到大排序，并记排序后的第 k 个界特征为 $S[k]$。假设某轨迹的特征为 $S[c]$（$c > k$），且 $S[c].lb > S[k].ub$。由于 $S[c]$ 所对应的真实值大于 $S[c].lb$，所以有 $S[c]$ 的真实值大于 $S[k].ub$。此外，又由于 $S[k].ub$ 大于前 $k-1$ 个界特征的上界，则 $S[k].ub$ 大于前 $k-1$ 个界特征所对应轨迹的真实距离值。因此，$S[c]$ 所对应的轨迹真实距离值大于前 $k-1$ 个界特征所对应轨迹的真实距离值。故 $S[c]$ 可以被剪枝掉。 □

图3.3展示介绍了剪枝原理。首先，将界特征按上界有小到大拍完序后。对于第 c 个轨迹（$c > k$），若其下界大于第 k 个的上界。则该轨迹的真实值必然大于前

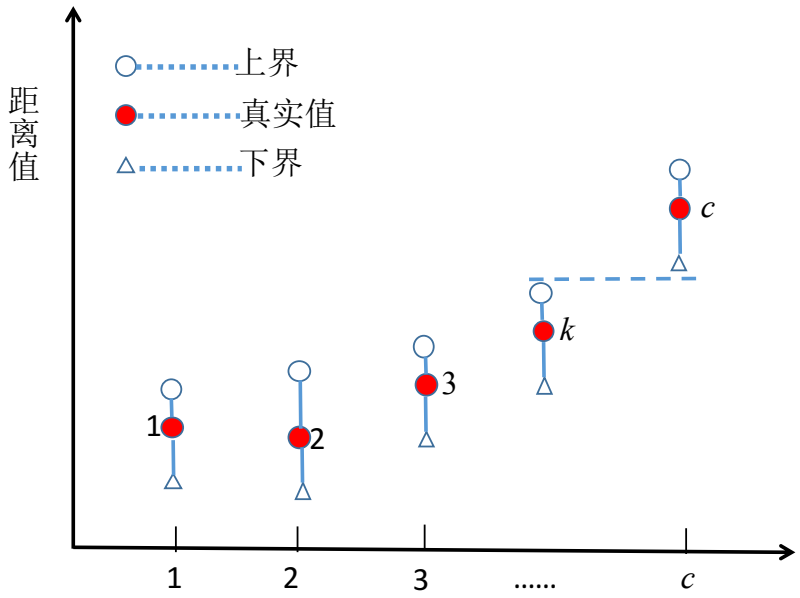


图 3.3: 剪枝示例

k 个轨迹的真实值。由于我们的差值只找距离最小的 k 条轨迹，所以第 c 个轨迹不可能存在于结果集中。此时，根据对其计算出来的上、下界就可以将其从候选集合中移除。

3.3.2 FTB 框架实现

本节将首先介绍 FTB（Framework with Two Bounds）框架，以处理那些能同时得到距离上、下界的情况。此外，需要保证当使用最细粒度的概要数据计算上、下界时，上界等于下界。即此时计算出来的上、下界等于距离的真实值。该框架分为两个部分，一部分是运行在协调者结点的算法 1，另一部分是运行在所有远程结点的算法 2。在框架运行过程中，这两个部分互相通信并协调工作。整个过程，可以分为 3 个阶段：初始化阶段，迭代交互式剪枝阶段，最终结果获取阶段。下面将对这三个部分分别进行介绍:

初始化阶段: 协调者结点在此阶段执行多种操作，如获取远程结点的列表，概要数据计算以及可能预发送一些信息。在此过程中，我们使用 $\mathcal{R}$ 来表示远程结点的集合（算法 1:1-2 行）。与之对应的，远程结点在该阶段的初始化主要工作是为

算法 1 FTB 之协调者结点

输入: 查询轨迹 $\mathcal{Q}$, 结果集大小 k;
输出: k 条最相似轨迹的 ID;
1: $\mathcal{R} \leftarrow$ 远程结点集合;
2: coordinatorInit($\mathcal{Q}, \mathcal{R}$);
3: **while true do**
4: /* 生成全局第 k 小上界 */
5: $Info \leftarrow$ generateInfo(); # 准备概要数据
6: sendToRemoteSites($\mathcal{R}$, $Info$);
7: $GUBS \leftarrow$ getFromRemoteSites($\mathcal{R}$);
8: $gkub \leftarrow argmin_{\tau}(|x \in GUBS, x < \tau| \geq k)$;
9: sendToRemoteSites($\mathcal{R}$, $gkub$);
10: /* 获取候选集大小 $\mathcal{R}$*/
11: $CSS \leftarrow$ getFromRemoteSites($\mathcal{R}$);
12: $\mathcal{R} \leftarrow \{x.r | x \in CSS, x.|S_r| > 0\}$ # 剪枝远程结点
13: $sum \leftarrow \sum_{x \in CSS} x.|S_r|$; # 候选集大小
14: **if** $sum == k$ **then**
15: sendToRemoteSites($\mathcal{R}$, ***finish***);
16: **break**;
17: $ids \leftarrow$ getFromRemoteSites($\mathcal{R}$);
18: **return** ids;

保存在本地数据集合 TS_r 中轨迹初始化界特征并将结果存在局部候选集 S_r 中。此外，它也会接受协调者结点发过来的数据（算法 2:1-2行）。一开始，所有轨迹都是候选。我们把包含候选的远程结点称为候选（远程）结点。

迭代交互式阶段: 在此阶段, 协调者结点与候选远程结点交互通信直到剪枝完毕。首先，远程结点为准好粗粒度的概要数据并发送给所有候选结点（算法 1:5-6行）。在接收到概要数据属后，候选远程结点利用该概要数据进行计算距离上、下界并更新界特征。需要注意的是：随着迭代次数的增加，所生成的概要数据粒度越细。这使得我们计算出来的上、下界越来越紧。根据最新的界特征，每个候选结点找出局部最小的 k 个上界并将它们发送给协调者结点 (算法 2:5-10行)。当协调者结点获取到所有候选远程结点发送来的上界后，它从中选择第 k 小的上界，记为 $gkub$，并将其值发送给那些候选远程结点 (算法 1 :7-9行)。候选结点在接受到 $gkub$ 后，就可以剪枝掉局部的一些候选轨迹。具体剪枝做法是，每个局部候选的下界与 $gkub$ 进行比较。如下界大于 $gkub$，则将该轨迹从局部候选中删除。剪枝后，候

算法 2 FTB 之远程结点

输入: 局部轨迹集合 TS_r;;
输出: 属于 top-k 结果集的轨迹 ID;

```
1:  S_r ← 局部轨迹界特征集;
2:  remoteInit(TS_r, S_r);
3:  while true do
4:      m ← getFromCoordinator();                          # 从协调者结点获取信息
5:      if m is Info then
6:          S_r ← UpdateBounds(S_r, m);
7:          /* 产生局部最小 k 个上界 */
8:          ub̂ ← argmin_τ(|x ∈ S_r, x.ub < τ| ≥ k);
9:          S' ← {α.ub | α ∈ S_r, α.ub ≤ ub̂} ;
10:         SendToCoordinator(S');                  # 将最小的 k 个上界返回给协调者结点
11:     else if m is gkub then
12:         S_r ← {β | β ∈ S_r, β.lb ≤ m}                                  # 局部剪枝
13:         SendToCoordinator(⟨r, |S_r|⟩);
14:         if |S_r| = 0 then
15:             return ;                          # 当远程结点无候选，则停止运行
16:     else
17:         /* m is finish*/
18:         ids ← {a.id | a ∈ S_r};
19:         SendToCoordinator(ids);
20:         return ;
```

选结点将其所剩候选的个数发给协调者结点。若候选结点的所有轨迹都被剪枝掉，则该结点运行结束（算法 2:11-15行）。在此之后，协调者结点收集候选远程结点所发送的局部候选的个数，并求和计算总的个数。若发现某结点不再包含候选，则将其从候选结点集中移除。此外，若发现当前所剩候选总数正好为 k 个，则迭代终止。协调者将向剩下的候选结点发送终止信号 ***finish***。若所剩个数大于 k，则迭代继续（算法 1:11-16 行）。

最终结果获取阶段: 经过上一阶段，最终会剩余 k 个候选。这些候选即为最终的结果。所剩的候选结点在收到上一阶段所发的结束信号后，会将本地所剩候选的 ID 发送给协调者节点。发送成功后，自身会结束查询 (算法2: 18-20行)。而协调者节点则会收集所有候选结点发送过来的 ID，并返回给用户（算法1: 17-18行)。

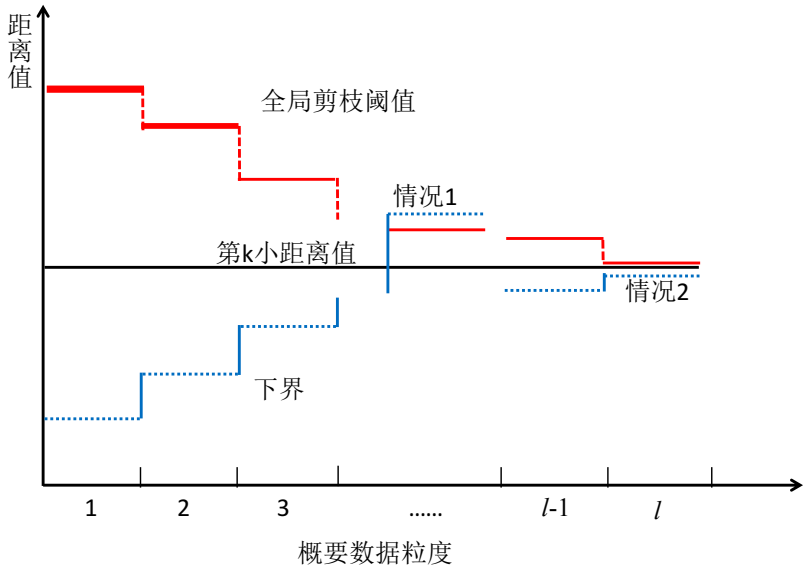


图 3.4: 利用下界和全局阈值的剪枝过程

3.4 FLB 框架

本节将介绍 FLB 框架的设计原理及实现方案。与 FTB 框架的不同的是，本框架主要为那些仅能根据概要数据得到距离下界的距离准则而设计。

3.4.1 FLB 框架设计原理

针对那些不能同时获得上界的距离度量准则，FTB 在使用由粗到细粒度的概要数据以不断获取更加紧凑的距离下界的同时，还不断计算一个越来越紧的全局阈值以剪枝数据。假设与待查询轨迹距离第 k 小的值为 gk，若某轨迹的界特征的下界大于该值，则该轨迹不会成为最终结果。但 gk 值很难一开始就计算出来。为此，本文的做法是以最小的代价（通信和时间开销）计算 gk 的上界 θ，并不断将 θ 逼近 gk。然后，在每轮迭代中，用 θ 来剪枝候选。θ 就是我们用来剪枝的全局阈值。需要区别的是，在 FTB 框架中，是利用界特征的上界进行剪枝。

图3.4示例介绍了，FTB 框架对某条轨迹的剪枝过程。图中，中间的黑色直线表示查询轨迹与所有轨迹距离中第 k 小的距离值（gk 值）。蓝色分段虚线介绍了轨迹距离下界随着概要数据粒度增加，该下界值越来越逼近其真实距离的过程。红

色分段实线介绍了全局阈值不断逼近 gk 值。在两者的不断逼近过程中会出现两种情况：第一种情况就是不断逼近过程中，该轨迹的下界大于阈值 θ。此时，我们可以将轨迹剪枝掉。第二种情况就是当最细粒度的概要数据后，轨迹的下界仍小于阈值 θ。此时，我们将对该轨迹继续保留为候选以待进一步分析。最后，当根据最细粒度的概要数据进行剪枝后，若所保留的候选数仍超过 k。此时，FTB 框架将向包含这些候选的远程结点发送原始轨迹，远程结点则可以计算出真实的距离值并找出最终的 k 个距离最近的轨迹。

算法 3 FLB 之协调者结点

输入: 查询轨迹 $\mathcal{Q}$, 结果集大小 k;
输出: k 条最相似轨迹的 ID;

1: $\mathcal{R} \leftarrow$ 远程结点集合，$\mathcal{R}' \leftarrow null$;
2: coordinatorInit($\mathcal{Q}$,$\mathcal{R}$);
3: **while true do**
4: $Info \leftarrow$ generateInfo(); # 生成待发送概要数据
5: **if** $Info \neq null$ **then**
6: sendToRemoteSites($\mathcal{R}$, $Info$); # /*(6-13行): 更新全局阈值 θ*/
7: $GLBS \leftarrow$ getFromRemoteSites($\mathcal{R}$); # 从远程结点获取局部 top-k 下界
8: $gklbs \leftarrow argmin_{\tau}(|x \in GLBS, x.lb < \tau| \geq k)$; # 选取下界最小的 k 条轨迹
9: $\mathcal{R}'' \leftarrow$ the remote sites that contain $gklbs$; # 找出包含上述轨迹的远程结点
10: sendToRemoteSites($\mathcal{R}'' - \mathcal{R}'$, $\langle \mathcal{Q}, gklbs \rangle$); # 对未收到 $\mathcal{Q}$ 的结点进行处理
11: sendToRemoteSites($\mathcal{R}'' \cap \mathcal{R}'$, $\langle null, gklbs \rangle$); # 对已收到 $\mathcal{Q}$ 的结点进行处理
12: $sv \leftarrow sv \cup$ getFromRemoteSites($\mathcal{R}''$);
13: $\theta \leftarrow argmin_{\tau}(|x \in sv, x < \tau| \geq k)$; # 选取第 k 小相似度值
14: sendToRemoteSites($\mathcal{R}$, θ);
15: $\mathcal{R}' = \mathcal{R}' \cup \mathcal{R}''$;
16: /*Step 2: 剪枝候选远程结点 */
17: $CSS \leftarrow$ getFromRemoteSites($\mathcal{R}$); # /*(17-22行): 剪枝并反馈 */
18: $\mathcal{R} \leftarrow \{x.r | x \in CSS, x.|S_r| > 0\}$;
19: $sum \leftarrow \sum_{x \in CSS} x.|S_r|$;
20: **if** $sum == k$ **then**
21: sendToRemoteSites(CS, ***finish***); # top-k 结果已经找到，发结束信号
22: **return** getFromRemoteSites($\mathcal{R}$);
23: **else**
24: sendToRemoteSites($\mathcal{R} - \mathcal{R}'$, $\langle null, \mathcal{Q} \rangle$); # /*(24-28行): 结果提炼 */
25: sendToRemoteSites($\mathcal{R} \cap \mathcal{R}'$, $\langle null, null \rangle$);
26: $CSet \leftarrow$ getFromRemoteSites($\mathcal{R}$);
27: $gkSim \leftarrow argmin_{\tau}(|x \in CSet, x.dis < \tau| \geq k)$;
28: **return** $\{x.id \,|\, x \in CSet, x.dis \leq gkSim\}$;

3.4.2 FLB 框架实现

本节将首先介绍 FLB（Framework with Two Bounds）框架，以处理那些仅能获取到下界的情况。该框架分为两个部分，一部分是运行在协调者结点的算法 3，另一部分是运行在所有远程结点的算法 4。在框架运行过程中，这两个部分互相通信并协调工作。整个过程，可以分为 4 个阶段：初始化阶段，全局阈值计算阶段，剪枝阶段和结果提炼阶段。下面将对这四个阶段分别进行介绍:

初始化阶段: 协调者结点在此阶段执行多种操作，如获取远程结点的列表，概要数据计算以及可能预发送一些信息。在此过程中，我们使用 $\mathcal{R}$ 来表示远程结点的集合, $\mathcal{R}'$ 表示那些已经接受了查询轨迹的结点（算法 3:1-2 行）。与之对应的，远程结点在该阶段的初始化主要工作是为保存在本地数据集合 TS_r 中轨迹初始化界特征并将结果存在局部候选集 S_r 中。此外，它也会接受协调者结点发过来的数据（算法 2:1-2行）。

全局阈值计算阶段: 该阶段，我们的主要目标是通过迭代通信交互，逐步获取更加精确的全局阈值。在每次迭代中，协调者结点首先发送概要数据给所有候选远程结点（算法 3 6行）。候选远程结点在接受到概要数据后，更新每个候选轨迹界特征值（注：在框架中，界特征只保留 ID 和下界，无需保持上界）。接着选取局部下界最小的 k 个界特征返回给协调者节点（算法 4：6-9行）。其次，协调者选取包含最小的 k 个下界的界特征集合 $gklbs$，及包含 $gklbs$ 的远程结点集合 $\mathcal{R}''$（算法 3：7-8行）。然后，它将待查询轨迹 $\mathcal{Q}$ 发送给 $\mathcal{R}''$ 中的远程结点。在发送前，他将 $\mathcal{R}''$ 中的结点分为两类：第一类是接收过 $\mathcal{Q}$ 的结点，此时我们需将 $\mathcal{Q}$ 以及 $gklbs$ 中属于该结点的轨迹发送给对应的结点。第二类是已接收过 $\mathcal{Q}$ 的结点，此时，仅需将对应结点的轨迹返回（算法 3：10-11）。当候选远程结点接收到信息这样的成对信息后，针对本地出现在 $gklbs$ 中的轨迹计算出距离值并返回给用户 (算法 4：11-12 行)。需要注意的是，对于接受成对信息的结点，我们并没有对本地所有候选轨迹计算真实值。其原因是尽管使用全部候选计算出来的距离能有更好的逼近第 k 小距离值，但这样做可能导致时间开销较大，尤其是当数据集中在这些节点上时。远

算法 4 FLB 之远程结点

输入: 局部轨迹集合 TS_r;;
输出: 属于 top-k 结果集的轨迹 ID;

```
1:  S_r ← 局部轨迹界特征集;
2:  remoteInit(TS_r, S_r);
3:  while true do
4:      m ← getFromCoordinator();                          # 从协调者结点获取信息
5:      if m is Info then
6:          S_r ← UpdateLowerBound(S_r, m);
7:          lb̂ ← argmin_τ(|x ∈ S_r, x.lb < τ| ≥ k);
8:          S' ← {x | x ∈ S_r, x.lb ≤ lb̂};
9:          SendToCoordinator(⟨r, S'⟩);
10:     else if m = ⟨Q/null, gklbs⟩ then
11:         sv ← {distance(Q, x)|x ∈ gklbs};
12:         SendToCoordinator(sv);
13:     else if m = θ then
14:         S_r ← {x | x ∈ S_r, x.lb ≤ θ};                  # 局部剪枝
15:         SendToCoordinator(⟨r, |S_r|⟩);
16:         if |S_r| = 0 then
17:             return ;                                   # 结点不包含候选，则结束
18:     else if m = finish then
19:         ids ← {x.id | x ∈ S_r};          # 收到结束信号，则返回候选 ID 并结束
20:         SendToCoordinator(ids);
21:         return;
22:     else
23:         SendToCoordinator({⟨x.id, distance(x, Q)⟩ | x.id ∈ S_r});
24:         return ;
```

程结点在接收到那些包含 $gklbs$ 的结点发送过来的距离值后，选择最小的值作为全局阈值，并将该值发送给候选结点（算法 3：13-14行）。

剪枝阶段: 该阶段主要利用 θ 和下界进行剪枝。当候选远程结点接收到 θ 后，将会将下界小于所接收的阈值的轨迹从候选集中删除。剪枝完毕后，该结点会将所剩候选数发送给协调者结点。此外，当该结点不再包含任何候选时，则停止运行（算法 4：13-17行）。协调者结点在接收到所有候选远程结点发送过来的候选数后，计算出总的所剩候选数。若所剩候选总数正好为 k 个，则说明所有候选已经找出。协调结点向候选远程结点发结束信号并等待接收最终结果的 ID（算法 3：21-22行）。候选远程在接收到结束信号后，则将本地候选的 ID 发送给协调者结点

（算法 4：18-21行）。若所剩候选总数仍超过 k，则迭代执行第二和本阶段任务直到 k 个候选被找出或概要数据发送完毕。

结果提炼阶段: 由于界下界最终不一定能逼近到原始距离值，且全局阈值不一定能逼近到第 k 小距离值。所以存在着当概要数据发送完毕，所剩候选数仍超过 k 的情况。此时，需要对剩下的候选进行甄别。FLB 框架的做法是将查询轨迹 $\mathcal{Q}$ 发送到剩下的候选结点中，并对所有剩余候选计算真实的距离值以找出最终的 k 个结果（算法 4 ： 24-28行）。需要注意的是在发送 $\mathcal{Q}$ 时，若某候选已经接收过 Q 则无需再次发送。

3.5 本章小结

本章首先介绍了使用概要数据计算距离界特征，并利用特征剪枝的思想。然后介绍了 FTB 框架以处理那些能根据概要数据同时计算出上、下界的距离函数。进一步的介绍了 FLB 框架以处理仅能根据概要数据计算出下界的距离函数。对比 FTB 和 FLB 两个框架的介绍，我们可以发现两者的共同点都是计算一个全局值来和轨迹下界来剪枝。它们的区别是 FTB 框架计算出的全局值是从候选轨迹的上界中选取出来的，而 FLB 框架是对某些轨迹计算真实的距离值，并从这些距离值中选取出来的。所以，尽管 FLB 框架也可以应用于能同时获取上、下界的距离函数，由于其计算全局阈值需要额外的计算和通信开销。故对于能同时根据概要数据计算上、下界的距离函数，我们使用 FTB 框架来处理。

第四章　FTB 框架的应用

本章主要介绍如何使用 FTB 框架处理基于欧式距离的查询。首先，章节4.1介绍了利用哈尔小波为欧式距离提供概要数据。然后，章节4.2介绍了基于小波系数的欧式距离上、下界。其次，章节4.3介绍了算法 ED-FTB，以处理当使用欧式距离作为距离度量准则时的查询。接着，章节4.3展示了 ED-FTB 算法的有效性和可扩展性。再其次，章节 4.5小结本章的研究内容。最后，本章涉及到的证明内容放在附件部分。

4.1　基于欧式距离的概要数据

4.1.1　基于欧式距离的轨迹相似度度量

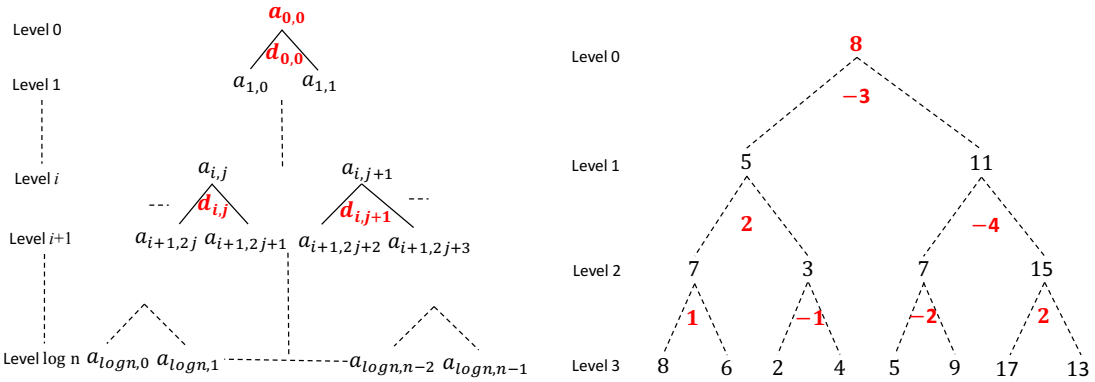
欧式距离用于处理两条长度相同的轨迹。假设待查询轨迹 $\mathcal{Q}$ 表示为 $\mathcal{Q} = \{\boldsymbol{q}_0, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_{n-1}\}$，一条候选轨迹 $\mathcal{C}$ 表示为 $\mathcal{C} = \{\boldsymbol{c}_0, \boldsymbol{c}_1, \cdots, \boldsymbol{c}_{n-1}\}$，其中 n 为轨迹长度，每个轨迹点来自 d 维空间，即 $\boldsymbol{q}_i, \boldsymbol{c}_i \in R^d$。我们首先给出点之间的距离定义，不失一般性的，我们使用欧式距离来度量。

$$ED(\boldsymbol{q}_i, \boldsymbol{c}_i) \quad = \quad ||\boldsymbol{q}_i - \boldsymbol{c}_i|| = \sqrt{||\boldsymbol{q}_i||^2 + ||\boldsymbol{c}_i||^2 - 2\boldsymbol{q}_i \cdot \boldsymbol{c}_i} \tag{4.1}$$

其中 $||\boldsymbol{q}_i||$ (或 $||\boldsymbol{c}_i||$) 表示 $\boldsymbol{q}_i$ (或 $||\boldsymbol{c}_i||$) 的二范数的值，$\boldsymbol{q}_i \cdot \boldsymbol{c}_i$ 表示向量 $\boldsymbol{q}_i$ 和 $\boldsymbol{c}_i$ 之间的点 (内) 积。在此基础上，我们给出轨迹间的欧式距离定义。

$$ED(\mathcal{Q}, \mathcal{C}) \quad = \quad \sqrt{\sum_{i=0}^{n-1} ED(\boldsymbol{q}_i, \boldsymbol{c}_i)^2} \tag{4.2}$$

为简化计算和表示，我们使用欧式距离的平方（Squared Euclidean Distance, SED）来替换距离，即 $SED(\boldsymbol{q}_i, \boldsymbol{c}_i) = ED(\boldsymbol{q}_i, \boldsymbol{c}_i)^2$ and $SED(\mathcal{Q}, \mathcal{C}) = ED(\mathcal{Q}, \mathcal{C})^2$。此时，轨迹距离的欧式距离可以看作是点距离的累加和。



(a) 哈尔小波变换过程　　(b) 哈尔小波示例

图 4.1: 哈尔小波变换

4.1.2 基于哈尔小波的轨迹概要数据抽取

哈尔小波变换（Haar Wavelet Transform,HWT）是降维时间序列、图像数据等的有效方法。它将原始数据从多个解析度来展示，每个解析度代表了不同频域下的信息。其变换过程又可以被抽象成如图4.1(a)所示的自底向上构建一颗二叉树 (称为误差树，Error-tree) 的过程。在该图中最底层（叶子层）自左往右是时间序列原始数据，非叶子节点保留两个值 a_i^j 和 d_i^j。a_i^j 记录了该结点两个孩子结点的均值的正则化均值，即 $a_{i,j} = (a_{i+1,2j} + a_{i+1,2j+1})/nf$。$d_i^j$ 记录了该结点两个孩子结点均值的正则化差值信息，即 $d_{i,j} = (a_{i+1,2j} + a_{i+1,2j+1})/nf$。$nf$ 称为正则化因子（normalization factor），其取值为 $1/2$ 或 $1/\sqrt{2}$，具体是应用场景而定。由于最底层（叶子节点层）结点仅包含原始数据，故倒数第二层结点直接从原始值计算出来。在自底向上构建的过程中，每层结点个数减半，且直到某层仅包含一个节点为止（第 0 层）。

在误差树结构中，每层的均值序列可以看做对原始时间序列的一层概要数据，自上而下粒度越来越细。然而，哈尔小波并没有直接保存这些均值，它依次保留了最终的均值及由上到下每层的差值，并将保留的值称为（哈尔小波）系数。我们能根据系数值能恢复出每层的均值。具体的，如图4.1(b)所示，给定时间序列 $f(t) = \{8, 6, 2, 4, 5, 9, 17, 13\}$ 且 $nf = 1/2$ 时，经过哈尔小波变换后的系数为 $H(f(t)) = \{8, -3, 2, -4, 1, -1, -2, 2\}$。给定第 0 层的系数 $8, -3$，我们能计算出第 1 层的均值分别为 $8 + (-3) = 5$,$8 - (-3) = 11$。此时，我们仅使用了 2 个值就表达了与 3 个

值（第 0 和第 1 层均值）同等量的信息。此外，若继续给出第 1 层的系数，我们使用相同的方法能恢复出第 2 层的均值。值得注意的是，为方便解释，示例中正则化因子为 $1/2$，本章接下来的应用中将使用 $1/\sqrt{2}$ 来进行变换。

上面部分介绍了哈尔小波变换以及如何使用它处理一维时间序列数据。但轨迹数据天然是多维时间序列数据，能否使用哈尔小波变换对轨迹数据进行多粒度解析呢? 为此，本章重点讲介绍如何使用小波的多粒度特性来对轨迹数据进行概要数据抽取。首先，给定查询轨迹 $\mathcal{Q}$ 和候选候选轨迹 $\mathcal{C}$。它们的长度均为 n，且假设 n 是 2 的正整数次方。由于长度为 n 的时间序列，其对应误差树的深度为 $L+1$，$L = \log_2 n$。对于 $\mathcal{Q}$ 和 $\mathcal{C}$ 的哈尔小波系数分别为 $H\mathcal{Q} = \{\boldsymbol{a}^{\mathcal{Q}}_{0,0}, \boldsymbol{d}^{\mathcal{Q}}_{0,0}, \boldsymbol{d}^{\mathcal{Q}}_{1,0}, \cdots, \boldsymbol{d}^{\mathcal{Q}}_{L-1,n/2-1}\}$ 和 $H\mathcal{C} = \{\boldsymbol{a}^{\mathcal{C}}_{0,0}, \boldsymbol{d}^{\mathcal{C}}_{0,0}, \boldsymbol{d}^{\mathcal{C}}_{1,0}, \cdots, \boldsymbol{d}^{\mathcal{C}}_{L-1,n/2-1}\}$, 其中 $\boldsymbol{a}^{\mathcal{Q}}_{0,0}$ 和 $\boldsymbol{a}^{\mathcal{C}}_{0,0}$ 分别是 $H(\mathcal{Q})$ 和 $H(\mathcal{C})$ 变换后的最终的正则化的均值，$\boldsymbol{d}^{\mathcal{Q}}_{i,j}$ 和 $\boldsymbol{d}^{\mathcal{C}}_{i,j}$ 是正则化后的差值，且 $\boldsymbol{a}^{\mathcal{Q}}_{i,j}, \boldsymbol{a}^{\mathcal{C}}_{i,j}, \boldsymbol{d}^{\mathcal{Q}}_{i,j}, \boldsymbol{d}^{\mathcal{C}}_{i,j} \in R^d$。根据正则化小波变换过程，$Q$ 的误差树中第 i 层第 j 个非叶子节点的内容 $\boldsymbol{a}^j_i$ 和 $\boldsymbol{d}^j_i$ 可以由如下公式计算。

$$\boldsymbol{a}^{\mathcal{Q}}_{i,j} = \frac{\boldsymbol{a}^{\mathcal{Q}}_{i+1,2j} + \boldsymbol{a}^{\mathcal{Q}}_{i+1,2j+1}}{\sqrt{2}}, \qquad \boldsymbol{d}^{\mathcal{Q}}_{i,j} = \frac{\boldsymbol{a}^{\mathcal{Q}}_{i+1,2j} - \boldsymbol{a}^{\mathcal{Q}}_{i+1,2j+1}}{\sqrt{2}} \tag{4.3}$$

4.2 轨迹欧式距离上下界

上节介绍了如何对轨迹数据使用哈尔小波进行轨迹变换，本节将介绍如何使用哈尔小波系数构建欧式距离的上下界。

4.2.1 基于哈尔小波的欧式距离表示

首先，对于 $\mathcal{Q}$ 和 $\mathcal{C}$ 的误差树中对应相邻节点对:$\{\boldsymbol{a}^{\mathcal{Q}}_{i+1,2j}, \boldsymbol{a}^{\mathcal{Q}}_{i+1,2j+1}\}$ 和 $\{\boldsymbol{a}^{\mathcal{C}}_{i+1,2j}, \boldsymbol{a}^{\mathcal{C}}_{i+1,2j+1}\}$，则节点对相应元素的欧式距离的和可以如下表示:

$$\begin{aligned} & SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,2j}, \boldsymbol{a}^{\mathcal{C}}_{i+1,2j}) + SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,2j+1}, \boldsymbol{a}^{\mathcal{C}}_{i+1,2j+1}) \\ = \;& SED(\frac{\boldsymbol{a}^{\mathcal{Q}}_{i,j} + \boldsymbol{d}^{\mathcal{Q}}_{i,j}}{\sqrt{2}}, \frac{\boldsymbol{a}^{\mathcal{C}}_{i,j} + \boldsymbol{d}^{\mathcal{C}}_{i,j}}{\sqrt{2}}) + SED(\frac{\boldsymbol{a}^{\mathcal{Q}}_{i,j} - \boldsymbol{d}^{\mathcal{Q}}_{i,j}}{\sqrt{2}}, \frac{\boldsymbol{a}^{\mathcal{C}}_{i,j} - \boldsymbol{d}^{\mathcal{C}}_{i,j}}{\sqrt{2}}) \end{aligned} \tag{4.4}$$

$$
\begin{aligned}
= \; & ||\frac{\boldsymbol{a}_{i,j}^{\mathcal{Q}}+\boldsymbol{d}_{i,j}^{\mathcal{Q}}}{\sqrt{2}}||^2+||\frac{\boldsymbol{a}_{i,j}^{\mathcal{C}}+\boldsymbol{d}_{i,j}^{\mathcal{C}}}{\sqrt{2}}||^2-2\frac{\boldsymbol{a}_{i,j}^{\mathcal{Q}}+\boldsymbol{d}_{i,j}^{\mathcal{Q}}}{\sqrt{2}}\cdot\frac{\boldsymbol{a}_{i,j}^{\mathcal{C}}+\boldsymbol{d}_{i,j}^{\mathcal{C}}}{\sqrt{2}} \\
& +||\frac{\boldsymbol{a}_{i,j}^{\mathcal{Q}}-\boldsymbol{d}_{i,j}^{\mathcal{Q}}}{\sqrt{2}}||^2+||\frac{\boldsymbol{a}_{i,j}^{\mathcal{C}}-\boldsymbol{d}_{i,j}^{\mathcal{C}}}{\sqrt{2}}||^2-2\frac{\boldsymbol{a}_{i,j}^{\mathcal{Q}}-\boldsymbol{d}_{i,j}^{\mathcal{Q}}}{\sqrt{2}}\cdot\frac{\boldsymbol{a}_{i,j}^{\mathcal{C}}-\boldsymbol{d}_{i,j}^{\mathcal{C}}}{\sqrt{2}} \\
= \; & SED(\boldsymbol{a}_{i,j}^{\mathcal{Q}},\boldsymbol{a}_{i,j}^{\mathcal{C}})+SED(\boldsymbol{d}_{i,j}^{\mathcal{Q}},\boldsymbol{d}_{i,j}^{\mathcal{C}})
\end{aligned}
$$

公式4.4说明，细粒度的概要数据间的距离可以由粗粒度的数据计算出来。为方便进一步表示，我们用 $S_i(\mathcal{Q},\mathcal{C})$ 表示 $\mathcal{Q}$ 和 $\mathcal{C}$ 误差树的第 i 层的对应均值间的欧式距离的平方和，用 $SED_i(\mathcal{Q},\mathcal{C})$ 表示 $\mathcal{Q}$ 和 $\mathcal{C}$ 误差树的第 i 层的对应差间的欧式距离的平方和。$S_i(\mathcal{Q},\mathcal{C})$ 和 $SED_i(\mathcal{Q},\mathcal{C})$ 的定义如下所示：

$$S_i(\mathcal{Q},\mathcal{C}) = \sum_{j=0}^{2^i-1} SED(\boldsymbol{a}_{i,j}^{\mathcal{Q}},\boldsymbol{a}_{i,j}^{\mathcal{C}}) \tag{4.5}$$

$$SED_i(\mathcal{Q},\mathcal{C}) = \sum_{j=0}^{2^i-1} SED(\boldsymbol{d}_{i,j}^{\mathcal{Q}},\boldsymbol{d}_{i,j}^{\mathcal{C}}) \tag{4.6}$$

此外，由于原始轨迹间的距离可以根据两个误差树的叶子计算出来，即 $SED_L(\mathcal{Q},\mathcal{C}) = SED(\mathcal{Q},\mathcal{C})$。根据这一结论和以上定义，我们得到如下定理。

引理 4.2.1. *给定两条轨迹 $\mathcal{Q}$ 和 $\mathcal{C}$，$H\mathcal{Q}$ 和 $H\mathcal{C}$ 分别表示 $\mathcal{Q}$ 和 $\mathcal{C}$ 经过哈尔小波变换后的系数序列。我们有如下结论：$SED(\mathcal{Q},\mathcal{C}) = SED(H\mathcal{Q},H\mathcal{C})$。*

引理4.2.1既说明了原始轨迹间的欧式距离等于哈尔小波系数间距离（证明见本章附件），又说明了通过对查询轨迹哈尔变换后的系数可以替换原始轨迹用于距离计算。此外，对于轨迹长度不是 2 的次方的情况，我们可以通过将轨迹切分成若干个字轨迹，只需保证每个轨迹是 2 的次方。然后，可以对子轨迹进行哈尔小波变换，并将所有子轨迹系数的欧式距离累加等到整体轨迹间的距离。因此，哈尔小波可以被用来分解任意长度的轨迹。

4.2.2 基于哈尔小波的欧式距离上下界

上一节介绍了，利用完整的哈尔小波系数，即全部的概要数据，能够用来计算原始的轨迹间欧式距离。但全部概要数据的数据量于原始轨迹数据量相同，不

能达到降低通信开销的目的。为此本节将介绍如何利用部分概要来计算轨迹的上、下界。首先根据引理4.2.1, 我们有:

$$\begin{aligned} SED(\mathcal{Q},\mathcal{C}) &= \sum_{i=0}^{L-1} SED_i(\mathcal{Q},\mathcal{C}) \\ &= \sum_{i=0}^{l} SED_i(\mathcal{Q},\mathcal{C}) + \sum_{i=l+1}^{L-1} SED_i(\mathcal{Q},\mathcal{C}) \end{aligned} \tag{4.7}$$

假设远程结点由粗到细已经获取了查询轨迹 $\mathcal{Q}$ 的前 $l+1$ 层概要数据，我们的目标就是根据这些已有的数据计算出 $\mathcal{Q}$ 与任一候选 $\mathcal{C}$ 的欧式距离上、下界。公式4.7有半部分分为两部分，前半部分可以根据已有概要数据计算出来，后半部分无法直接计算。为此，我们将后半部分继续展开，得到如下:

$$\sum_{i=l+1}^{L-1} SED_i(\mathcal{Q},B) = \sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}(||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2 + ||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2 - 2\boldsymbol{d}_{i,j}^{\mathcal{Q}} \bullet \boldsymbol{d}_{i,j}^{\mathcal{C}}) \tag{4.8}$$

公式4.8右半部分含有 3 个元素，第一个元素为查询轨迹 $\mathcal{Q}$ 剩下概要数据的和，该值可根据已有概要数据计算出来，计算方法为 $SSQ - \sum_{i=0}^{l}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2$，其中 SSQ 为 $\mathcal{Q}$ 所有概要数据的累加和。远程结点只需一开始就把 SSQ 发给所有远程结点即可。同理，第二个元素各远程结点可以计算出来。难点在于对第三个关于系数内积累加和的计算。本文的做法是使用两次柯西—施瓦茨不等式估计其区间。第一次使用具体过程如下:

$$-2\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}|| \cdot ||\boldsymbol{d}_{i,j}^{\mathcal{C}}|| \le \sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1} -2\boldsymbol{d}_{i,j}^{\mathcal{Q}} \bullet \boldsymbol{d}_{i,j}^{\mathcal{C}} \le 2\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}|| \cdot ||\boldsymbol{d}_{i,j}^{\mathcal{C}}|| \tag{4.9}$$

接着我们对 $\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}|| \cdot ||\boldsymbol{d}_{i,j}^{\mathcal{C}}||$ 进行再次使用柯西-施瓦茨不等式放缩:

$$\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}|| \cdot ||\boldsymbol{d}_{i,j}^{\mathcal{C}}|| \le \sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2} \cdot \sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2} \tag{4.10}$$

结合公式4.9和4.10, 我们得到如下完整的对 $\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1} -2\boldsymbol{d}_{i,j}^{\mathcal{Q}} \bullet \boldsymbol{d}_{i,j}^{\mathcal{C}}$ 计算出如下上、

下界。

$$-2\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2}\cdot\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2}$$
$$\leq\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}-2\boldsymbol{d}_{i,j}^{\mathcal{Q}}\cdot\boldsymbol{d}_{i,j}^{\mathcal{C}}\leq 2\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2}\cdot\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2} \quad (4.11)$$

最终，我们对欧式距离获得如下上、下界:

$$HLB_l(\mathcal{Q},\mathcal{C})=\sum_{i=0}^{l}SED_i(\mathcal{Q},\mathcal{C})+\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}(||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2+||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2)+S_0(\mathcal{Q},\mathcal{C}) \quad (4.12)$$
$$-2\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2}\cdot\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2}$$
$$HUB_l(\mathcal{Q},\mathcal{C})=\sum_{i=0}^{l}SED_i(\mathcal{Q},\mathcal{C})+\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}(||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2+||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2)+S_0(\mathcal{Q},\mathcal{C})$$
$$+2\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2}\cdot\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2} \quad (4.13)$$

进一步的我们提出两个性质，这两个性质表明我们的上、下界会随着概要数据的增加而越来越紧。它们的证明过程见本章附件。

性质 4.2.2. *HLB 会随着粒度的概要数据粒度的增加而逐渐上升，即 $HLB_l \leq HLB_{l+1}$。*

性质 4.2.3. *HUB 会随着粒度的概要数据粒度的增加而逐渐下降，即 $HUB_l \geq HUB_{l+1}$。*

4.3 基于欧式距离的查询算法:ED-FTB

4.3.1 ED-FTB 算法实现

在上一节，我们已经利用概要数据提出了欧氏距离的上、下界。这使得在 FTB

算法 5 *ED-FTB*在协调者结点

/* *ED-FTB* 协调者结点函数接口实现 */

coordinatorInit($\mathcal{Q}, \mathcal{R}$)

1: $l \leftarrow -1$;
2: $HQ \leftarrow$ Haar wavelet coefficients of $\mathcal{Q}$; # 待查询轨迹获取哈尔小波系数
3: $SSQ = \sum_{i=0}^{n-1} ||HQ_i||^2$; # 所有系数的平方和
4: sendToRemoteSites($\mathcal{R}$, SSQ);

generateInfo()

1: $l \leftarrow l + 1$;
2: return $\widehat{\mathcal{Q}}_l$; # 返回第 l 层哈尔小波系数

框架中插入欧式距离成为可能。本节将详细介绍将欧式距离跟 FTB 相结合的查询算法 ED-FTB。ED-FTB 维持了 FTB 框架的主要结构，只对本章第一节所介绍的接口进行了具体实现。

算法5介绍了协调者节点的上的函数接口的实现方法。在 coordinatorInit 函数中，我们对待查询轨迹进行哈尔小波变换，并计算出其所有系数的平方和。接着将该平方和发送给所有远程结点。远程结点在将来将会利用该值进行上、下界计算。由于 FTB 框架是迭代式由粗到细的通信计算框架，在每轮的迭代中会将某一层的概要数据（哈尔小波系数）发送给远程结点。因此，在初始化过程中我们用 l 来记录当前已经发送到哪一层的数据，并将 l 初始化为 -1 以便从第 0 层开始。此外，我们在 generateInfo 中准备将要发送的下一层概要数据，以便协调者结点发送。

接着，算法6介绍运行在远程结点的函数。对于 RemoteInit 函数，若远程结点是第一次接受查询，则会为每条候选轨迹进行哈尔小波变换并计算系数的平方和。若不是，则为该结点所包含的每条轨迹初始化界特征信息（下界初始化为 0，上界初始化为正无穷），并接受查询轨迹的系数平方和。在查询执行过程中 UpdateBounds 函数为候选更新上、下界。直接实现方式为根据上、下界的计算公式，计算出这两个值。但由于查询过程中的主要计算开销就是对所有候选更新界特征。所以，降低该过程的计算开销很有必要。

为降低更新界特征的计算开销，本文引入了在更新界过程中进行剪枝的思想。在介绍此方法前，我们再次回顾下我们的下界。我们的下界计算主要包含 3 个算

算法 6 *ED-FTB*在远程结点

/* *ED-FTB* 远程结点函数接口实现 */

remoteInit(TS_r, S_r**)**

1: **if** 第一次接受查询 **then**
2: **for all** $\mathcal{C}$ in TS_r **do**
3: $HC \leftarrow$ Haar wavelet coefficients of $\mathcal{C}$; # 候选轨迹获取哈尔小波系数
4: $SSC = \sum_{i=0}^{n-1} ||HQ_i||^2$; # 所有系数的平方和
5: **for all** $\mathcal{C}$ in TS_r **do**
6: $C\_BF \leftarrow \langle 0, \infty \rangle$; # 为 $\mathcal{C}$ 初始化界特征
7: $S_r = S_r \cup C\_BF$;
8: $SSQ \leftarrow$getFromCoordinator();

updateBounds(S_r, M**)** //边更新边过滤。

1: **for all** $\mathcal{C}$ in TS_r **do**
2: $a = \sum_{i=l+1}^{L-1} \sum_{j=0}^{2^i-1} ||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2$;$b = \sum_{i=l+1}^{L-1} \sum_{j=0}^{2^i-1} ||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2$;
3: $tmp = a + b + S_0(\mathcal{Q}, \mathcal{C})$;
4: $lb = tmp - 2\sqrt{ab}$; $ub = tmp + 2\sqrt{ab}$;
5: **if** $lb < gkub$ **&&** $(lb = lb + \sum_0^l SED_i(\mathcal{Q}, \mathcal{C})) < gkub$ **then**
6: 将轨迹 $\mathcal{C}$ 的界特征的下界更新为 lb;
7: **if** $ub < gkub$ **&&** $(ub = ub + \sum_0^l SED_i(\mathcal{Q}, \mathcal{C})) < gkub$) **then**
8: $ub = ub + \sum_0^l SED_i(\mathcal{Q}, \mathcal{C})$;
9: 将轨迹 $\mathcal{C}$ 的界特征的上界更新为 ub;
10: **else**
11: 将该轨迹从 S_r 中移除;

子：首先是 $\sum_{i=0}^{l} SED_i(\mathcal{Q}, \mathcal{C})$，其计算复杂度为 O($2^l$)。其次是 $\sum_{i=l+1}^{L-1} \sum_{j=0}^{2^i-1} ||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2$，由于该部分可以通过 $SSQ - \sum_{i=0}^{l} \sum_{j=0}^{2^i-1} ||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2$ 计算，而且 $\sum_{i=0}^{l-1} \sum_{j=0}^{2^i-1} ||\boldsymbol{d}_{i,j}^{\mathcal{Q}}||^2$ 的值已知。故该部分的计算复杂度也为 $O(2^l)$。同理，第三部分 $\sum_{i=l+1}^{L-1} \sum_{j=0}^{2^i-1} ||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2$ 计算复杂度也为 $O(2^l)$。除这三个算子的计算法外，剩余部分的计算复杂度为 $O(1)$。

基于以上分析。假设我们已知若轨迹下界值超过 α, 那该条轨迹即不可能称为候选。此时，我们可以在更新下界的同时进行剪枝（算法6：UpdateBounds 函数）。具体的做法是我们首先计算出第二和第三两个算子，并根据这两个算子的值计算出下界中除 $\sum_{i=0}^{l} SED_i(\mathcal{Q}, \mathcal{C})$ 以外部分的值。若此时计算出来的值已超过 $gkub$，则无需继续求解下界和上界，直接将该候选删除。若仍小于 $gkub$，则计算出完整的下界，并判断此时下界值是否小于 $gkub$，若小于则停止计算上界并将该候选删除。当更新完下界后，我们先计算出上界中除 $\sum_{i=0}^{l} SED_i(\mathcal{Q}, \mathcal{C})$ 以外部分的值。若该

部分值超过 $gkub$，则该轨迹的上界对选取新一轮的全局最小的 k 个上界没有帮助，无需计算该轨迹的具体上界值。否则，计算出具体的上界并更新界特征。此时，需要注意的是，在 FTB 框架算法的第 7-9 行中，我们仅需传递上界小于 $gkub$ 的 k 个最小上界。若个数不足 k 个，则只传递满足 $gkub$ 的上界。

4.3.2 ED-FTB 算法性能分析

在分析前，我们先介绍对比算法 LEEWAVE-CL。LEEWAVE-CL 是在 LEEWAVE 算法的基础上使用了本文所提下界（比原始的下界更紧）。LEEWAVE-CL 算法也是迭代式算法。在它的每次迭代中，远程结点根据获取的概要数据，为每个候选计算如下两个算子：$\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}_{i,j}^{\mathcal{C}}||^2}$ 和 $S_0(\mathcal{Q},\mathcal{C})+\sum_{i=0}^{l}SED_i(\mathcal{Q},\mathcal{C})$。然后协调者节点获取者些参数并为每个候选计算上、下界，并利用界特征进行过滤。然后，将候选列表发给对应远程结点。LEEWAVE-CL 方法与本文方法的最大不同就是，它在协这结点进行界特征的计算和过滤，而 ED-FTB 是在所有远程结点进行。接下来，我们将从时间和通信两个方面对 ED-FTB 和 LEEWAVE-CL 进行对比分析。在此之前，我们使用如下标记：(i) 使用 $|C_i|$ 和 $|CS_i|$ 分别表示第 i（$i \geq 0$）次迭代前候选轨迹的数量和包含候选轨迹的远程结点数量。由于采用相同的上、下界，这两个算法的迭代次数相同，我们假定迭代次数为 λ（$\lambda \leq \log_2 n$）。

时间复杂度: 不考虑系统初始化时对所有候选进行哈尔小波变换的时间开销，ED-FTB 和 LEEWAVE-CL 两者的时间开销来自迭代式计算时更新上下界，所以他们的时间复杂度一样。计算复杂度最坏的情况就是所有候选都集中在一个结点上，则更新上、下界的时间复杂度为 $O(\sum_{i=0}^{\lambda-1} d\cdot|C_i|\cdot 2^i)$。此外，由于 k 值一般较小，算法运行过程中设计到的找 top$k$ 上界的时间较低，可以忽略。总的来说，ED-FTB 和 LEEWAVE-CL 的时间复杂度均为 $O(\sum_{i=0}^{\lambda-1} d\cdot|C_i|\cdot 2^i)$。但 ED-FTB 由于在各个结点进行界特征的计算，且采用了边计算边过滤的策略，故其实际计算开销低于 LEEWAVE-CL。

通信复杂度: ED-FTB 和 LEEWAVE-CL 的通信开销主要集中在迭代式计算过

程中。ED-FTB 在第 i 轮迭代时，需要花费 $O(d \cdot 2^i \cdot |CS_i| + |C_i|)$ 代价以将第 i 层概要数据发送给候选结点，并至多接收 $|C_i|$ 个上界（用于计算全局的剪枝阈值）。在 λ 次迭代后，总的通信开销为 $O(\sum_{i=0}^{\lambda-1}(d \cdot 2^i \cdot |CS_i| + |C_i|))$。LEEWAVE-CL 在第 i 轮迭代需要花费 $O(|CS_i| \cdot (d \cdot 2^i + |C_{i+1}|))$ 来讲概要数据和所有候选的列表发送给所有候选结点。此外，还需要花费 $O(d \cdot |C_{i+1}|)$ 代价以接受每个候选轨迹的两个算子值以便在协调者结点计算出上、下界。所以，当 λ 次迭代后，其总的通信开销为 $O(|CS_i| \cdot (d \cdot 2^i + |C_{i+1}|) + d \cdot |C_{i+1}|)$。对比两个算法的开销值，我们可以发现 ED-FTB 的能节省更多通信开销。

4.4 实验分析

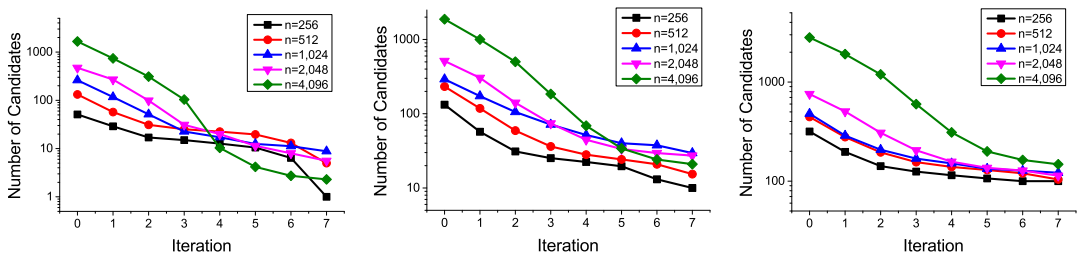
本节将在真实世界的轨迹数据上评估本章节所提出的算法。本小姐首先介绍使用的真实世界数据集合实验设置。接着从多个角度验证算法的性能（[1]）。

4.4.1 实验设置

本章工作在实验中使用北京出租车数据集，该数据集已被广泛应用于轨迹数据分析。该数据集采集了北京市三万多辆出租车在 2013 年 10 月份到 12 月份三个月内的行驶 GPS 轨迹。每个 GPS 轨迹点包含的数据维度较多，我们仅选取了位置(经度和纬度)、时间、速度和角度这五个维度的值，并进行了归一化预处理。我们从该数据集中截取了两个子数据集：*T-Small* 和 *T-Big* 以分别验证算法的有效性和可扩展性。*T-Small* 截取了 10 月 1 至 7 号间上午 8 点到 10 点的轨迹数据，并从中选出最长的 1 万条轨迹进行分析。*T-Big* 截取了 11 月 1 号至 12 月 31 号间每天上午 8 至 10 点和下午 5 至 7 点两个时间段的 1 百万条轨迹数据。为满足实验需求，这两个数据集的每条轨迹长度均超过 4,096。

在本章的实验中,我们将比较ED-FTB和LEEWAVE-CL算法的性能。LEEWAVE-CL 算法在已有 LEEWAVE 算法的基础上使用了本文所提供的下界。两个算法均用

[1]扩展



(a) $k = 1$ (b) $k = 10$ (c) $k = 100$

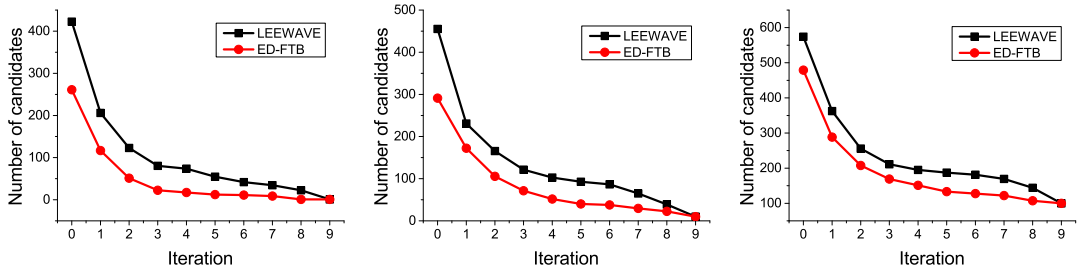
图 4.2: ED-FTB 剪枝效果图

JAVA 实现，并运行在一个包含 12 个结点的 Spark 集群上。每个结点包含 8 核因特尔 E5335 2.0 GHz 中央处理器和 16GB 的内存。我们在 *T-Small* 数据集上验证了算法的有效性，在 *T-Big* 数据集上验证了算法的可扩展性。

4.4.2 算法有效性

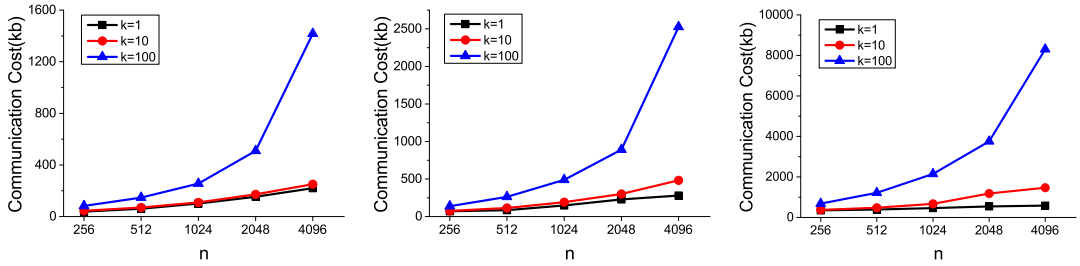
首先，我们通过观察迭代过程中每轮结束后的剩余候选数来研究 ED-FTB 算法的剪枝效果。在该组实验中，我们将 M 设置为 10,000，并研究了不同长度轨迹剪枝的效果。图 4.2 介绍了前 8 轮迭代中每轮迭代后的候选数。我们可以看出候选数在前 5 轮迭代中下降的较快且已经过滤掉绝大多数候选。这说明了我们的上下级收敛的很快，具有较好的剪枝效果。此外，我们可以发现短轨迹在前几轮中候选数下降的比长轨迹快。这是由于相同数据量的概要数据下短轨迹比长轨迹包含更多的原始轨迹的信息。最后，对比不同的 k 值，我们发现 k 值越小，每轮所剩候选数越少。即 k 越小，剪枝效果越好。这是由于 k 值越小，我们所获得的全局第 k 小上界就越小。而无论 k 值如何选取，每个候选的下界值不变。因此，越小的全局上界能通过下界剪枝掉更多的候选。同时，由于实际应用中 k 值通常都是一个很小的数。因而我们的算法能往往取得较好的效果。

然后，我们对比研究了 ED-FTB 的剪枝效果和 ED-FTB 使用 LEEWAVE 中的下界后的剪枝效果，以分析下界对剪枝效果的影响。在该组实验中，我们将 M 值设置为 10,000，轨迹长度设为 $1,024$。图4.3对比展示了不同 k 值下，不同下界对剪枝的影响。其中 LEEWAVE 代表 ED-FTB 算法使用 LEEWAVE 算法中的下界后的



(a) $k = 1$ (b) $k = 10$ (c) $k = 100$

图 4.3: 下界对剪枝结果的影响

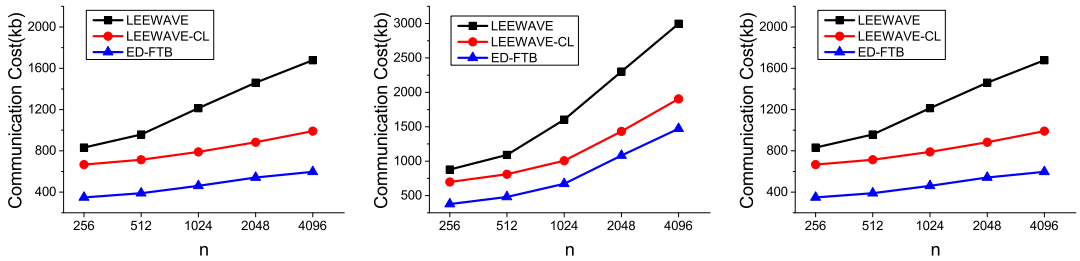


(a) $M = 100$ (b) $M = 1,000$ (c) $M = 10,000$

图 4.4: n,k 和 M 对 ED-FTB 算法通信开销的影响

结果。从图中，我们可以看出由于本文所提出的下界比已有算法的下界更紧，导致每轮迭代结束后 ED-FTB 所剩的候选数更少。这说明了本文所提下界的优越性。此外我们可以发现 k 值越小，两条曲线间的间隔越大。更能说明本文下界所取得的优越性越明显。

其次，我们研究了 n,k 和 M 对 ED-FTB 算法通信开销的影响。在该组实验中，我们将 n 的值从 256 变化到 $4,096$，将 k 的值从 1 变化到 100。图4.4展示了不同 M 值下（M 从 100 编化到 $10,000$），ED-FTB 算法通信开销随参数变化的结果。首先，我们看出随着 k 值的增加，通信开销逐步增加。这是由于 k 值越大，每轮所剩候选数越多，候选所在的远程结点也就越多。从而每轮需要将概要数据发送到更多的候选结点中。其次，随着 n 值的增加通信开销也在增加。这是由于长轨迹需要发送更多的数据才能达到与短轨迹形同的剪枝效果。最后，对比三幅图可以发现，随着远程结点数据的增加，通信开销也在增加。这是由于在每轮迭代中，远程结点增加后，候选轨迹会分布到更多的结点中。因而，算法需要将概要数据发送到更多



(a) $k = 1$ (b) $k = 10$ (c) $k = 100$

图 4.5: Communication cost comparison on T-Small

的结点中。因而总的通信开销也会增加。但是，对于极端情况 $M = 10,000$ 时，此时每个远程结点仅包含一条轨迹时我们的通信开销仍不到 10Mb，远小于直接方法的开销（约 640Mb）。因此，ED-FTB 算法具有较高的通信性能。

接下来，我们将 ED-FTB 与 LEEWAVE 和 LEEWAVE-CL 进行通信性能比较。在该组实验中我们将轨迹长度 n 从 256 变化到 4,096。图4.5展示了当 k 取值为 1，10 和 100 下的结果。从该组图中我们可以看出，本文所提 ED-FTB 算法性能最好，LEEWAVE-CL 其次，最差的是 LEEWAVE。LEEWAVE-CL 比 LEEWAVE 好的原因是它使用了本文所提下界进行剪枝。本文所提下界由于比 LEEWAVE 中的更加紧凑，因而剪枝效果更好。最终导致通信开销更低。而 ED-FTB 算法比 LEEWAVE-CL 好的原因是，本文采用的是在远程结点剪枝的策略，只需发送概要数据。而 LEEWAVE 和 LEEWAVE-CL 均是采用在协调者结点剪枝的策略，除了发概要数据还要发送和接受其他额外数据。因而开销更高。最后，我们可以发现，随着轨迹长度 n 和 k 值的增加，EDD-FTB 算法所节省的通信开销逐步增大。这进一步说明了本文算法的优越性。

4.4.3 算法可扩展性

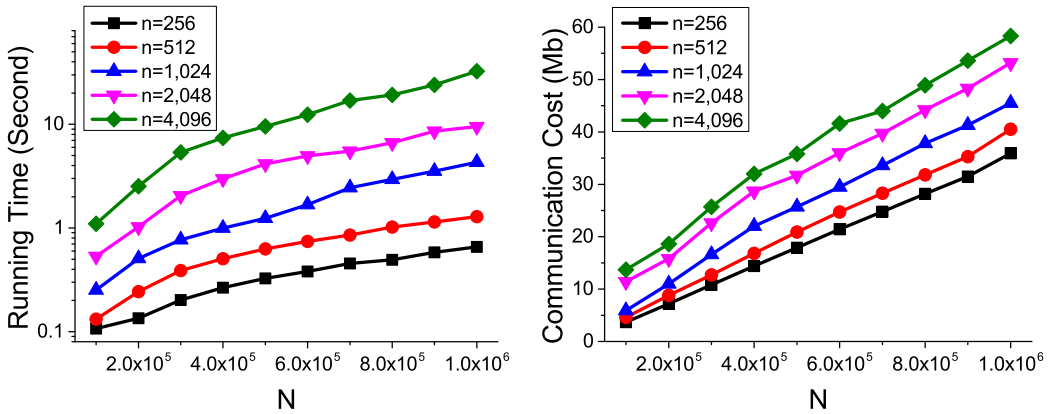
[2] 本小节我们将在 *T-Big* 数据集上从时间和通信两个角度研究了 ED-FTB 算法的可扩展性。首先，考虑了轨迹数量 N（即数据量大小）和轨迹长度 n 对可扩展性的影响。在该组实验中，我们将 k 设置为 1，M 设置为 10,000。我们将轨迹长度

[2]重做实验

从 256 到 4096 进行指数级变化，同时将轨迹数量从 10 万到 1 百万进行线性变换。图4.6分别对这两个角度的结果进行了介绍。其中图4.6(a)介绍了时间性能受 n 和 N 的影响。我们可以看到对于任意长度的轨迹数据集，ED-FTB 算法的运行时间都随着轨迹数据量的增加而线性增加。此外，随着轨迹长度的指数增加，算法的运行时间也指数增加，即运行时间与轨迹长度呈一定的线性关系。这一结果反映了由于 ED-FTB 的剪枝效果较好，导致迭代结束后，所剩候选数较少。即我们对 ED-FTB 算法时间复杂度分析部分的 N' 较少。结合以上两点，我们可以看出 ED-FTB 算法运行时间随着数据集的大小而线性变换，因此运行效率具有较好的可扩展性。

4.5 本章小结

本章节介绍了利用 FTB 框架在嵌入欧式距离下的具体实现算法 ED-FTB。本章节，首先针对欧式距离，提出了基于哈尔小波系数的概要数据。接着，提出了基于 Haar 小波系数的欧式距离上、下界。该上、界能够随着小波粒度的增加而逐渐变紧，这使得我们利用将该距离嵌入到 FTB 框架称为可能。然后，我们提出了 ED-FTB 算法以实现基于欧式距离的查询。在我们的查询算法中，我们引入了边计算下界边剪枝的查询优化措施，以提高执行效率。通过在真实数据集上进行的实验，表明所提方法剪枝效果由于现有方案，且具有较好的可扩展性。



(a) Running time in respect of n,N (b) Communication cost in respect of n,N

图 4.6: Scalability of ED-FTB on T-Big

4.6 附件

引理 4.2.1. 给定两条轨迹 $\mathcal{Q}$ 和 $\mathcal{C}$，$H\mathcal{Q}$ 和 $H\mathcal{C}$ 分别表示 $\mathcal{Q}$ 和 $\mathcal{C}$ 经过哈尔小波变换后的系数序列。我们有如下结论：$SED(\mathcal{Q},\mathcal{C}) = SED(H\mathcal{Q}, H\mathcal{C})$。

证明. 首先，根据 $S_i(\mathcal{Q},\mathcal{C})$ 和 $SED_i(\mathcal{Q},\mathcal{C})$ 定义（定义4.5）我们有

$$\begin{aligned}
& S_{i+1}(\mathcal{Q},\mathcal{C}) = \sum_{j=0}^{2^{i+1}-1} SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,j}, \boldsymbol{a}^{\mathcal{C}}_{i+1,j}) \\
= \;& SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,0}, \boldsymbol{a}^{\mathcal{C}}_{i+1,0}) + SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,1}, \boldsymbol{a}^{\mathcal{C}}_{i+1,1}) + \cdots + \\
& SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,2^{i+1}-2}, \boldsymbol{a}^{\mathcal{C}}_{i+1,2^{i+1}-2}) + SED(\boldsymbol{a}^{\mathcal{Q}}_{i+1,2^{i+1}-1}, \boldsymbol{a}^{\mathcal{C}}_{i+1,2^{i+1}-1}) \\
= \;& SED(\boldsymbol{a}^{\mathcal{Q}}_{i,0}, \boldsymbol{a}^{\mathcal{C}}_{i,0}) + SED(\boldsymbol{d}^{\mathcal{Q}}_{i,0}, \boldsymbol{d}^{\mathcal{C}}_{i,0}) + \cdots + \\
& SED(\boldsymbol{a}^{\mathcal{Q}}_{i,2^i-1}, \boldsymbol{a}^{\mathcal{C}}_{i,2^i-1}) + SED(\boldsymbol{d}^{\mathcal{Q}}_{i,2^i-1}, \boldsymbol{d}^{\mathcal{C}}_{i,2^i-1}) \\
= \;& \sum_{j=0}^{2^i-1} SED(\boldsymbol{a}^{\mathcal{Q}}_{i,j}, \boldsymbol{a}^{\mathcal{C}}_{i,j}) + \sum_{j=0}^{2^i-1} SED(\boldsymbol{d}^{\mathcal{Q}}_{i,j}, \boldsymbol{d}^{\mathcal{C}}_{i,j}) \\
= \;& S_i(\mathcal{Q},\mathcal{C}) + SED_i(\mathcal{Q},\mathcal{C})
\end{aligned} \tag{4.14}$$

根据如上公式，对于叶子层结点，我们有：

$$\begin{aligned}
S_L(\mathcal{Q},\mathcal{C}) &= S_0(\mathcal{Q},\mathcal{C}) + \sum_{i=0}^{L-1} SED_i(\mathcal{Q},\mathcal{C}) \\
&= SED(H\mathcal{Q}, H\mathcal{C})
\end{aligned} \tag{4.15}$$

此外，第 L 层为叶子节点层，包含了轨迹的原始信息。所以我们又有 $SED(\mathcal{Q},\mathcal{C}) = S_L(\mathcal{Q},\mathcal{C})$。此时结合公式4.15 我们得到 $SED(\mathcal{Q},\mathcal{C}) = SED(H\mathcal{Q}, H\mathcal{C})$。原问题得证。 □

Property 4.2.2. HLB 会随着粒度的概要数据粒度的增加而逐渐上升，即 $HLB_l \leq HLB_{l+1}$。至此，我们根据哈尔小波变换得到原始轨迹不同粒度概要数据，并根据概要数据提出了欧式距离的上、下界。

证明. 我们的策略是证明 $HLB_{l+1} - HLB_l \geq 0$。我们首先将其左半部分展开:

$$HLB_{l+1} - HLB_l = SED_{l+1}(\mathcal{Q},\mathcal{C}) - \sum_{j=0}^{2^{l+1}-1}(||\boldsymbol{d}^{\mathcal{Q}}_{l+1,j}||^2 + ||\boldsymbol{d}^{\mathcal{C}}_{l+1,j}||^2) + 2(\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{Q}}_{i,j}||^2}\cdot\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{C}}_{i,j}||^2} - \sqrt{\sum_{i=l+2}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{Q}}_{i,j}||^2}\cdot\sqrt{\sum_{i=l+2}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{C}}_{i,j}||^2})$$

接着,我们将 $SED_{l+1}(\mathcal{Q},\mathcal{C})$ 展开得到 $SED_{l+1}(\mathcal{Q},\mathcal{C}) = \sum_{j=0}^{2^{l+1}-1}(||\mathbf{d}^{\mathcal{Q}}_{l+1,j}||^2 + ||\mathbf{d}^{\mathcal{C}}_{l+1,j}||^2 - 2\mathbf{d}^{\mathcal{Q}}_{l+1,j}\cdot\mathbf{d}^{\mathcal{C}}_{l+1,j})$。我们的问题变为证明如下不等式成立。

$$\begin{aligned}\sum_{j=0}^{2^{l+1}-1}\boldsymbol{d}^{\mathcal{Q}}_{l+1,j}\cdot\boldsymbol{d}^{\mathcal{C}}_{l+1,j} \quad \leq \quad & \sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{Q}}_{i,j}||^2}\cdot\sqrt{\sum_{i=l+1}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{C}}_{i,j}||^2} \\ & -\sqrt{\sum_{i=l+2}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{Q}}_{i,j}||^2}\cdot\sqrt{\sum_{i=l+2}^{L-1}\sum_{j=0}^{2^i-1}||\boldsymbol{d}^{\mathcal{C}}_{i,j}||^2} \qquad (4.16)\end{aligned}$$

对于不等式 4.16的左半部分我们有 $\sum_{j=0}^{2^{l+1}-1}\boldsymbol{d}^{\mathcal{Q}}_{l+1,j}\cdot\boldsymbol{d}^{\mathcal{C}}_{l+1,j} \leq \sqrt{\sum_{j=0}^{2^{l+1}-1}||\boldsymbol{d}^{\mathcal{Q}}_{l+1,j}||^2}\cdot\sqrt{\sum_{j=0}^{2^{l+1}-1}||\boldsymbol{d}^{\mathcal{C}}_{l+1,j}||^2}$。所以我们的目标变为证明 $\sqrt{\sum_{j=0}^{2^{l+1}-1}||\mathbf{d}^{\mathcal{Q}}_{l+1,j}||^2}\cdot\sqrt{\sum_{j=0}^{2^{l+1}-1}||\mathbf{d}^{\mathcal{C}}_{l+1,j}||^2}$ 小于不等式4.16的右半部分。为方便表示，我们令 $x = \sum_{j=0}^{2^{l+1}-1}||\boldsymbol{d}^{\mathcal{Q}}_{l+1,j}||^2$，$y = \sum_{j=0}^{2^{l+1}-1}||\boldsymbol{d}^{\mathcal{C}}_{l+1,j}||^2$，$\alpha = \sum_{i=l+2}^{L-1}\sum_{j=0}^{2^l-1}||\boldsymbol{d}^{\mathcal{Q}}_{i,j}||^2$，$\beta = \sum_{i=l+2}^{L-1}\sum_{j=0}^{2^l-1}||\boldsymbol{d}^{\mathcal{C}}_{i,j}||^2$。则不等于4.16等价于:

$$\sqrt{x\cdot y} + \sqrt{\alpha\cdot\beta} \leq \sqrt{(\alpha+x)\cdot(\beta+y)} \qquad (4.17)$$

我们将如上不等式两边平方得到如下不等式:

$$2\sqrt{x\cdot y\cdot\alpha\cdot\beta} \leq \alpha\cdot y + \beta\cdot x \qquad (4.18)$$

根据算数-几何平均不等式，我们得不等式4.18 成立。原问题得证。 □

Property 4.2.3. *HLB 会随着粒度的概要数据粒度的增加而逐渐上升,即 $HLB_l \leq HLB_{l+1}$。*

证明. 性质4.2.2的证明过程与性质4.2.2类似，故不再赘述。 □

第五章　FLB 框架的应用

本章主要介绍如何使用 FLB 框架处理基于动态时间卷曲距离的查询。首先，章节5.1介绍了利用包围信封为 DTW 距离提供概要数据。然后，章节5.2介绍了基于包围信封的 DTW 距离下界。其次，章节5.3介绍了算法 DTW-FLB，以处理当使用动态时间卷曲距离作为距离度量准则时的查询。接着，章节5.3展示了 DTW-FLB 算法的有效性和可扩展性。再其次，章节 5.5小结本章的研究内容。最后，本章涉及到的证明内容放在附件部分。

5.1　基于动态时间卷曲距离的概要数据

5.1.1　基于动态时间卷曲距离的轨迹相似度度量

给定查询轨迹 $\mathcal{Q}$ 表示为 $\mathcal{Q} = \{\boldsymbol{q}_0, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_{n-1}\}$，以及一条候选轨迹 $\mathcal{C}$ 表示为 $\mathcal{C} = \{\boldsymbol{c}_0, \boldsymbol{c}_1, \cdots, \boldsymbol{c}_{n-1}\}$，其中 n 为轨迹长度，每个轨迹点来自 d 维空间，即 $\boldsymbol{q}_i, \boldsymbol{c}_i \in R^d$。为使用动态时间卷曲距离匹配 $\mathcal{Q}$ 和 $\mathcal{C}$，我们构建了一个 $n \times n$ 的代价矩阵 $cost$。其中第 i 行第 j 列的元素记录了使用欧式距离匹配 $\boldsymbol{q}_i$ 和 $\boldsymbol{c}_j$ 两点之间的代价。动态时间卷曲距离的目的就是发现一条从如图5.1所示左下角出发到右上角结束的代价最小的路线/卷曲路径。一条卷曲路径需满足如下 3 个特别的约束：

- **边界条件**: 路径从代价矩阵的 $[0,0]$ 号格子出发并在 $[n-1,n-1]$ 号格子结束。
- **连续性**: 如果路径中相邻的两次权重值分别取自代价矩阵的 $[i,j]$ 和 $[i',j']$ 两个单元，那么这两个单元的位置关系必满足如下条件：$i'-i \leq 1$ 和 $j'-j \leq 1$。这个约束使得路径在扩展过程中只能从一个格子扩展到其半径为 1 周边的格子。
- **单调性**: 如果路径中相邻的两次权重值分别取自代价矩阵的 $[i,j]$ 和 $[i',j']$ 两个单元，那么这两个单元的位置关系还须满足如下条件：$i'-i \geq 0$ 和 $j'-j \geq 0$。

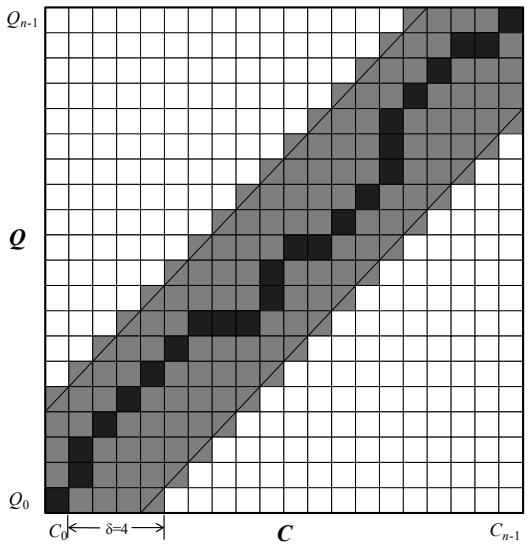


图 5.1: 带约束的动态时间卷曲

这个约束使得路径搜索过程中只能前进不能后退。

根据以上描述，我们形式化给出动态时间卷曲距离的定义:

$$DTW(\mathcal{Q},\mathcal{C}) = \min \sum_{k=1}^{K} \boldsymbol{w}_k \qquad (n \leq K < 2n) \tag{5.1}$$

其中，$\boldsymbol{w}_k$ 是卷曲路径 $\mathcal{W}$ 的第 k 个元素，其值来源于代价矩阵的第 $[i,j]$ 个格子。$\mathcal{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_K\}$, 代表了一条匹配 $\mathcal{Q}$ 和 $\mathcal{C}$ 卷曲路径。

动态时间卷曲距离的计算过程可通过动态规划算法实现，因为其计算过程可以看作如下最优子结构的递归:

$$DTW = \gamma(n-1, n-1) \tag{5.2}$$

$$\begin{aligned} \gamma(i,j) = {} & SED(\boldsymbol{q}_i, \boldsymbol{c}_j) + \min(\gamma(i-1,j-1), \gamma(i-1,j), \\ & \gamma(i,j-1)) \quad s.t. \quad |i-j| \leq \delta \end{aligned} \tag{5.3}$$

其中 δ 是一个全局约束，用于限制查询路径的可以偏离对角线的范围。图5.1中显示了当 $\delta = 4$ 时，灰色部分即为卷曲路径的查询范围。使用全局约束的原因有两个:（i）该约束可以提高查询的效率，δ 使得动态时间卷曲距离的计算复杂度由原

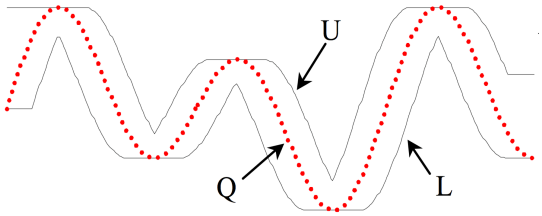


图 5.2: 包围信封

来的 n^2 降低为 $\frac{n^2}{\delta}$。（ii）该约束还能有效抑制无效的卷曲路径的产生。比如，一轨迹的一小段匹配到另一轨迹的一大段上情况。

5.1.2 基于包围信封的概要数据

包围信封（Bounding Envelope，BE）概念来源于时间序列数据分析，其目的是使用一上一下两条边界曲线来包住给定的时间序列。如图5.2所示，红色虚线为一条时间序列，U 和 L 两条线分别代表了这条时间序列的上界和下界。给定时间序列，构造其包围信封的方法有许多。由于本文的目的是利用包围信封构造概要数据，故所构造的包围信封包含的数据量要低。此外，所构造的包围信封还应具有多粒度特性，以适应 FLB 框架的需求。

为此，引入基于时间序列划分（Time Series Segmentation）的包围信封。其做法是给定一条时间序列 $\mathcal{S}$，将其均匀划分成多个等长度的片段。我们使用 $s_l^p = [lb, ub, len]$ 来表示其中一条片段 $\{\boldsymbol{q}_i, \boldsymbol{q}_{i+1}, \cdots, \boldsymbol{q}_j\}$ 的最小包围盒（Minimum Bounding Rectangle, MBR）。其中 l 代表划分的粒度（粒度越细，每个片段的长度越短），p 代表该片段在当前划分下所对应的位置。lb 和 ub 分别代表该片段的最小值和最大值，len 代表该片段的长度。由 $\{s_l^1, \cdots s_l^{n'}\}$ 所构成的列表代表了该次划分所对应的包围信封，其用 $\mathcal{S}_l$ 表示。在第 0 层，整个时间序列被看做一个划分，接着将其均匀切成 R 个不相交的相连子片段。这个过程可以一直持续下去，直到每个片段的长度为 1 截止。

表格5.1介绍了给定时间序列 $\mathcal{S} = \{5, 2, 4, 3\}$，计算其不同粒度包围盒的过程。其中，$R$ 的值设置为 2。即从粗粒度到细粒度切分的过程中，每次将一个片段均匀

$\mathcal{S}_0$	$s_0^0 = [2,5,4]$			
$\mathcal{S}_1$	$s_1^0 = [2,5,2]$		$s_1^1 = [3,4,2]$	
$\mathcal{S}_2$	$s_2^0 = [5,5,1]$	$s_2^1 = [2,2,1]$	$s_2^2 = [4,4,1]$	$s_2^3 = [3,3,1]$
Q	5	2	4	3

表 5.1: 时间序列 $\mathcal{S}$ 的多粒度包围盒

划分成 2 个子片段。不失一般性，接下来的示例和实验室中，我们都将 R 的值设置为 2。值得注意的是，本文采用的划分为均匀划分，而相关文章中所提非均匀划分的方法也可以被使用。此外，由于是均匀划分，我们可以推算出每个片段的长度，故其值无需保存。

5.2 动态时间卷曲距离的下界

本节将介绍如何根据包围信封计算动态时间卷曲距离的下界。首先将介绍点的上、下界，然后再介绍如何利用点的范围计算轨迹的下界。

5.2.1 满足 DTW 约束的包围信封及下界

令 $\boldsymbol{q}$ 和 $\boldsymbol{c}$ 表示两个来自 d 维实数空间的点，其中 $\boldsymbol{c}$ 的值已知，$\boldsymbol{q}$ 的值不知道，但知道其在一个 d 维矩形 $\{\boldsymbol{u}, \boldsymbol{l}\}$ 里，即 $\forall j \in [0, d)$, $\boldsymbol{l}^j \leq \boldsymbol{q}^j \leq \boldsymbol{u}^j$。对 $\boldsymbol{q}$ 和 $\boldsymbol{c}$ 之间的实际欧式距离我们无法计算出来，但可以通过如下引理计算出距离的下界。

引理 5.2.1. *给定 $\boldsymbol{q}$、$\boldsymbol{c}$ 以及 $\boldsymbol{q}$ 的包围矩形 $\{\boldsymbol{u}, \boldsymbol{l}\}$，我们有 $SED\_LB(\{\boldsymbol{u}, \boldsymbol{l}\}, \boldsymbol{c}) \leq SED(\boldsymbol{q}, \boldsymbol{c})$, 其中 $SED\_LB(\{\boldsymbol{u}, \boldsymbol{l}\}, \boldsymbol{c})$ 的计算过程如下:*

$$SED\_LB(\{\boldsymbol{u}, \boldsymbol{l}\}, \boldsymbol{c}) = \sum_{j=0}^{d-1} \begin{cases} (\boldsymbol{c}^j - \boldsymbol{u}^j)^2 & \text{if } \boldsymbol{u}^j \leq \boldsymbol{c}^j, \\ 0 & \text{if } \boldsymbol{l}^j \leq \boldsymbol{c}^j \leq \boldsymbol{u}^j, \\ (\boldsymbol{l}^j - \boldsymbol{c}^j)^2 & \text{if } \boldsymbol{c}^j \leq \boldsymbol{l}^j. \end{cases} \tag{5.4}$$

接着介绍如何构建满足动态时间卷曲距离的包围信封。根据包围信封的介绍，我们知道给定查询轨迹 $\mathcal{Q}$，构建包围信封的最重要就是构造轨迹的上、下边界。同

时，考虑到动态时间卷曲距离的全局约束特性。我们使用如下方法构建轨迹的上、下边界: We next introduce how to construct a bounding envelope $\{\mathcal{U}, \mathcal{L}\}$ for trajectory $\mathcal{Q}$, as defined below.

$$\begin{aligned} \mathcal{U} &= \{\boldsymbol{u}_0, \boldsymbol{u}_1, \cdots, \boldsymbol{u}_{n-1}\} \quad , \quad \forall j\, \boldsymbol{u}_i^j = \max(\boldsymbol{q}_{i-\delta}^j : \boldsymbol{q}_{i+\delta}^j) \\ \mathcal{L} &= \{\boldsymbol{l}_0, \boldsymbol{l}_1, \cdots, \boldsymbol{l}_{n-1}\} \quad , \quad \forall j\, \boldsymbol{l}_i^j = \min(\boldsymbol{q}_{i-\delta}^j : \boldsymbol{q}_{i+\delta}^j) \end{aligned} \tag{5.5}$$

公式5.5中，$\mathcal{U}$ 和 $\mathcal{L}$ 分别代表轨迹的上边界和下边界，δ 为动态时间卷曲距离中的全局约束。根据如上方式计算出来的包围盒，我们称之为满足动态时间卷曲约束的包围信封（Dynamic Time Warping constraint Bounding Envelope，DTWBE）。此时，根据 DTWBE$\{\mathcal{U}, \mathcal{L}\}$ 和 $\mathcal{C}$，我们可以给出如下距离定义:

$$LB\_Keogh(\{\mathcal{U}, \mathcal{L}\}, \mathcal{C}) = \sum_{i=0}^{n-1} SED\_LB(\{\boldsymbol{u}_i, \boldsymbol{l}_i\}, \boldsymbol{c}_i) \tag{5.6}$$

该定义可以看做是原始一维的 LB_Keogh 下界在多维情况下的扩展，并得到如下下界定理:

定理 5.2.2. *给定轨迹 C 和待查询轨迹 Q 满足动态时间卷曲约束的包围信封 $\{\mathcal{U}, \mathcal{L}\}$，我们有如下结论: $LB\_Keogh(\{\mathcal{U}, \mathcal{L}\}, \mathcal{C}) \leq DTW(Q, C)$。*

5.2.2 基于多粒度包围信封的下界

为计算上一小节介绍的 LB_Keogh 下界，需要传递 DTWBE，其数据量为 $2*n$。因此，不满足我们降低通信的目标。为此，本文首先设计了基于 DTWBE 的多粒度包围信封。多粒度信封的第 l 层边界计算方式如下:

$$\begin{aligned} \hat{\mathcal{U}}_l &= \{\hat{\boldsymbol{u}}_{l,0}, \hat{\boldsymbol{u}}_{l,1}, \cdots, \hat{\boldsymbol{u}}_{l,2^l-1}\} \quad , \quad \forall j\, \hat{\boldsymbol{u}}_{l,i}^j = \max(\boldsymbol{u}_{i\cdot 2^{L-l}}^j : \boldsymbol{u}_{(i+1)\cdot 2^{L-l}-1}^j) \\ \hat{\mathcal{L}}_l &= \{\hat{\boldsymbol{l}}_{l,0}, \hat{\boldsymbol{l}}_{l,1}, \cdots, \hat{\boldsymbol{l}}_{l,2^l-1}\} \quad , \quad \forall j\, \hat{\boldsymbol{l}}_{l,i}^j = \min(\boldsymbol{l}_{i\cdot 2^{L-l}}^j : \boldsymbol{l}_{(i+1)\cdot 2^{L-l}-1}^j) \end{aligned} \tag{5.7}$$

第 l 层的包围信封计算过程可看作将原来的 DTWBE 均匀切分成 2^l 块，然后对每个信封块计算仅使用两个 d 维点来构成新的包围信封。根据对公式 5.7的观察，我们得到两种结论：（i）第 l 层的包围信封 $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$ 数据量是第 $l-1$ 层的两倍；（ii）第 $l-1$ 层的数据被第 l 层包含。因此，在由粗到细发送包围信封这一概要数据时，除第 0 层外，其他层只需要发送一半的信息量（这一半数据的每个值需要用 0 和 1 标注其位置是在左边还是右边，但位置信息消耗的开销较小可忽略）。那么假定远程结点已获取到查询轨迹 $\mathcal{Q}$ 的第 l 层包围信封 $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$，我们可通过如下公式来计算给定的候选与 $\mathcal{Q}$ 的下界。

定理 5.2.3. *给定轨迹 $\mathcal{Q}$ 的第 l 层包围信封 $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$ 和候选轨迹 $\mathcal{C}$, 我们得到如下结论：$LB\_HPAA(\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}, \overline{\mathcal{C}}_l) \leq DTW(\mathcal{Q}, \mathcal{C})$。其中 LB_HPAA 的计算公式如下：*

$$LB\_HPAA(\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}, \overline{\mathcal{C}}_l) = 2^{L-l} \cdot \sum_{i=0}^{2^l-1} SED\_LB(\{\hat{\boldsymbol{u}}_{l,i}, \hat{\boldsymbol{l}}_{l,i}\}, \bar{\boldsymbol{c}}_{l,i})$$

以上定理说明了给定查询轨迹的第 l 层包围信封和候选误轨迹差树的第 l 层均值，就能为原始轨迹和候选轨迹计算出动态时间卷曲距离的下界。同时该下界的计算复杂度是线性的，远小于距离本身二次的复杂度。因此，通过计算下界可为剪枝候选提供帮助。

此外，为配合 FLB 框架，我们需要证明，LB_HPAA 能随着粒度的增加而逐渐变紧。为此我们给出如下定理：

定理 5.2.4. *LB_HPAA 下界能随着查询轨迹包围信封层次的增加而逐渐变紧。也就是说：$LB\_HPAA(\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}, \overline{\mathcal{C}}_l) \leq LB\_HPAA(\{\hat{\mathcal{U}}_{l+1}, \hat{\mathcal{L}}_{l+1}\}, \overline{\mathcal{C}}_{l+1})$。*

5.3 基于动态时间距离的查询算法:DTW-FLB

5.3.1 DTW-FLB 算法实现

在上一节，我们已经利用多粒度概要数据得到越来越紧的动态时间卷曲距离下界。这使得在 FLB 框架中插入欧式距离成为可能。本节将详细介绍动态时间卷

算法 7 *DTW-FLB 在协调者节点*

/* *DTW-FLB 协调者结点函数接口实现* */

coordinatorInit($\mathcal{Q}, \mathcal{R}$)

1: 根据公式5.5构建 DTWBE $\{\mathcal{U}, \mathcal{L}\}$;
2: **for** $i = 1$,$i < \log_2 n$;$i++$ **do**
3: 根据公式5.7构建第 i 层包围信封 $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$;
4: $l \leftarrow -1$;

generateInfo()

1: $l \leftarrow l + 1$;
2: **return** $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$;

曲距离跟 FLB 相结合的查询算法 DTW-FLB。DTW-FLB 维持了 FLB 框架的主要结构，只对本章第一节所介绍的接口进行了具体实现。

我们首先介绍如何实现那些运行在协调者节点的上的函数。在 coordinatorInit 函数中，我们首先对待查询轨迹计算出对应的 DTWBE，接着基于 DTWBE，我们构造多粒度的包围信封（即概要数据）。由于 FLB 框架也是迭代式由粗到细的通信计算框架，在每轮的迭代中会将某一层的概要数据（包围信封）发送给远程结点。因此，在初始化过程中我们用 l 来记录当前已经发送到哪一层的数据，并将 l 初始化为 -1，以便从第 0 层开始发送。此外，我们在 generateInfo 中准备将要发送的下一层概要数据，以便协调者结点发送。在发送数据时需要注意的是，由于 $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$ 的一半数据量已经包含在 $\{\hat{\mathcal{U}}_{l-1}, \hat{\mathcal{L}}_{l-1}\}$ 中，我们只需传输剩下的一半数据。

接着，介绍运行在远程结点的函数。对于 RemoteInit 函数，若远程结点是第一次接受查询，则会为每条候选轨迹进行哈尔小波变换。若不是，则为该结点所包含的每条轨迹初始化界特征信息（下界初始化为 0，不记录上界）。在查询执行过程中 UpdateBounds 函数为候选更新下界。直接实现方式为根据下界的计算公式直接算出该值。但由于查询过程中的主要计算开销就是对所有候选更新界特征。因此，降低该过程的计算开销很有必要。为降低更新界特征的计算开销，与上一章一样采用在更新界过程中进行剪枝的思想。根据 LB_HPAA 的定义，我们可知该下界是一系列点距离下界的累加和。由于每个点距离下界都是大于 0 的数，因此我们可以在累加的过程中，通过判断当前已经累加的值是否已经超过剪枝的阈值。若

算法 8 *DTW-FLB 在远程结点*

/* *DTW-FLB 远程结点函数接口实现* */

remoteInit(TS_r, S_r**)**

1: **if** 第一次接受查询 **then**
2: 构建 R-Tree 索引。
3: **for all** $\mathcal{C}$ in TS_r **do**
4: $\overline{\mathcal{C}} \leftarrow \mathcal{C}$ 的均值序列; # 候选轨迹获取所有层的均值构成的序列
5: **for all** $\mathcal{C}$ in TS_r **do**
6: $C\_BF \leftarrow 0$; # 为 $\mathcal{C}$ 初始化下界
7: $S_r = S_r \cup C\_BF$;

updateBounds(S_r, m**)** $/* m = \{\hat{\boldsymbol{u}}_{l,i}, \hat{\boldsymbol{l}}_{l,i}\} */$

1: **if** $l == 0$ **then**
2: 使用索引树进行剪枝。
3: **for all** $\mathcal{C}$ in TS_r **do**
4: $factor = 2^{L-l}$;
5: $c.lb = 0$;
6: **for** $i = 0, i < 2^l; i++$ **do**
7: $p\_lb = SED\_LB(\{\hat{\boldsymbol{u}}_{l,i}, \hat{\boldsymbol{l}}_{l,i}\}, \bar{\boldsymbol{c}}_{l,i})$;
8: $c.lb += p\_lb$;
9: **if** $c.lb > \theta$ **then**
10: 将该轨迹从 S_r 中移除;
11: 将 $\mathcal{C}$ 的下界更新为 $c.lb$;

超过了则可提前将该值过滤掉。

除了以上步骤外，我们引入了 R-tree 来对轨迹构建索引以提高查询效率。具体地：首先，每个远程结点对自己局部轨迹数据构建 R-tree 索引。接着，当该结点第 0 层包围信封时，使用 R-tree 来剪枝。剪枝具体过程入下：(i) 遍历 R-tree，找到跟查询轨迹包围信封相交的叶子节点。(ii) 遍历这些叶子节点所包含的轨迹，并将这些轨迹列为候选。

5.3.2 DTW-FLB 算法性能分析

本章节将从时间复杂度和空间复杂度量方面对 DTW-FLB 进行性能分析。在此之前，我们仍使用 $|C_i|$ 和 $|CS_i|$ 分别表示第 i（$i \geq 0$）次迭代前候选轨迹的数量和包含候选轨迹的远程结点数量，并假定迭代次数为 λ（$\lambda \leq \log_2 n$）。由于仅使用下界进行剪枝，因此存在着当概要数据发送完毕后，仍然存在着超过 k 个候选的

情况。此时，我们使用 N' 来表示迭代计算结束后剩余候选的个数。

时间复杂度: 不考虑系统初始化时对所有候选进行哈尔小波变换的时间开销，DTW-FLB 算法的时间主要来自两个方面：（i）算法 3 第 2 阶段的迭代更新的时间；（ii）对迭代结束后的候选计算真实动态时间卷曲距离的时间。第一方面的时间复杂度为 $\sum_{i=0}^{\lambda} d{\cdot}|C_i|{\cdot}2^i$，该值与算法 ED-FTB 的迭代计算复杂度相同。第二方面的计算复杂度为 $O(d{\cdot}\delta{\cdot}N'{\cdot}n^2)$。因此，总的时间复杂度为 $O(d{\cdot}(\sum_{i=0}^{\lambda}|C_i|{\cdot}2^i{+}\delta{\cdot}N'{\cdot}n^2))$。

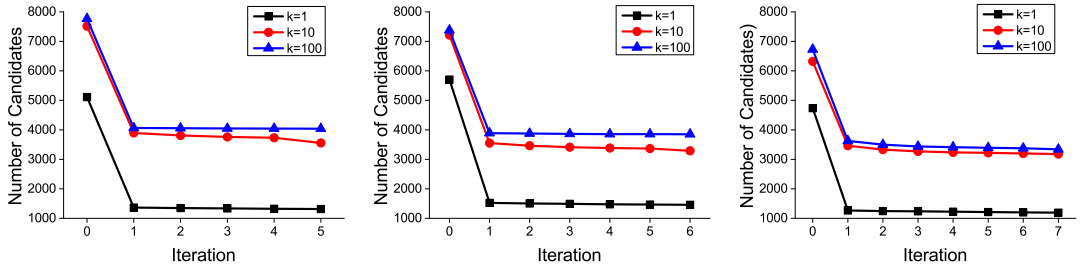
通信复杂度: DTW-FLB 算法的通信开销跟时间开销一样也是来自迭代发送包围信封和迭代结束后发送轨迹计算真实距离两个阶段。在迭代式发送包围阶段，第 i 次迭代中，协调者需要将第 i 层包围信封发送到候选结点中。由于，第 i 层包围信封已经有一半数据存在于第 $i-1$ 层包围信封中且已被候选结点接受。因此，发送的数据量为 $O(d \cdot 2^i \cdot |CS_i|)$。远程结点在接受到包围信封后，便更新下界并将局部最小的 k 个下界发送给协调者结点。这一过程发送的数据量最多为 $O(|C_i|)$。总的来说，迭代过程中产生的通信开销为。假设迭代结束后，所剩 N'（$N' > k$）个结点分布在 m' 个远程结点中。此时最多需要将原始轨迹发送到 m' 个结点中。其所对应的开销为 $O(d \cdot n \cdot m')$。结合这两个阶段，DTW-FLB 的总通信开销为 $O(\sum_{i=0}^{\log_2 n}(d \cdot 2^i \cdot |CS_i| + |C_i|) + d \cdot n \cdot m'))$。

通过以上对 DTW-FLB 的分析，我们可以发现该算法对迭代剪枝的效果十分敏感。即剪枝结束后所剩候选的个数即包含这些候选的节点的个数，会严重影响到计算时间和通信开销。由于，DTW-FLB 会在每一轮迭代中都会通过对部分候选轨迹计算真实的 DTW 距离用以更新全局过滤的阈值。该方法能保证该过滤阈值会不断逼近真实的第 k 小距离，因此最终会取得较好的剪枝效果。

5.4 实验分析

5.4.1 实验设置

本章工作在实验中使用北京出租车数据集，该数据集已被广泛应用于轨迹数据分析。该数据集采集了北京市三万多辆出租车在 2013 年 10 月份到 12 月份三个



(a) $n = 1,024$ (b) $n = 2,048$ (c) $n = 4,096$

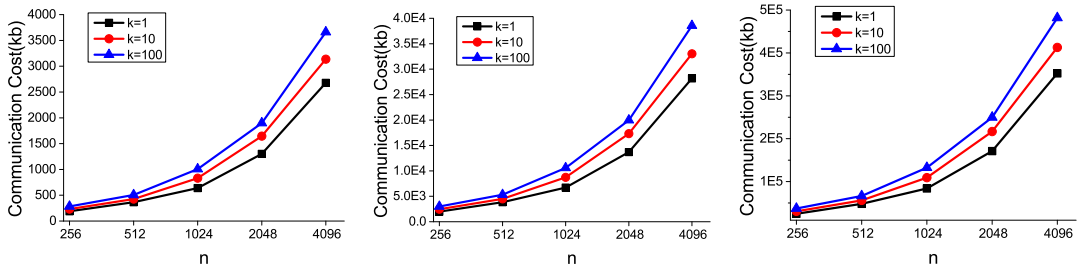
图 5.3: DTW-FLB 剪枝效果

月内的行驶 GPS 轨迹。每个 GPS 轨迹点包含的数据维度较多，我们仅选取了位置(经度和纬度)、时间、速度和角度这五个维度的值，并进行了归一化预处理。我们从该数据集中截取了两个子数据集：*T-Small* 和 *T-Big* 以分别验证算法的有效性和可扩展性。*T-Small* 截取了 10 月 1 至 7 号间上午 8 点到 10 点的轨迹数据，并从中选出最长的 1 万条轨迹进行分析。*T-Big* 截取了 11 月 1 号至 12 月 31 号间每天上午 8 至 10 点和下午 5 至 7 点两个时间段的 1 百万条轨迹数据。为满足实验需求，这两个数据集的每条轨迹长度均超过 4,096。

在本章的实验中，DTW-FLB 算法使用 JAVA 实现。需要注意的是，在我们介绍中 δ 的取值为整数来表示约束的范围。但为适应不同长度轨迹的需求，我们在实验中使用轨迹长度的百分比来表示约束范围。实验运行在一个包含 12 个结点的 Spark 集群上。每个结点包含 8 核因特尔 E5335 2.0 GHz 中央处理器和 16GB 的内存。

5.4.2 算法有效性

首先，我们通过汇报迭代过程中每轮结束后所剩的候选的个数以展示 DTW-FLB 的剪枝效果。在该组实验中，我们设置轨迹长度 n 从 256 变化到 4,096，k 的值从 1 变化到 100。图5.3展示了随着 k 和 n 变化后的剪枝的效果。从该图中我们可以看出在最开始的两轮迭代中，候选的个数急剧下降。此外，两轮迭代后所剩候选数较少。这说明，即使使用粗粒度的概要数据索引和剪枝策略的混合使用能起到较好的剪枝效果。在接下来的迭代中，我们可以发现候选个数降低比较平缓。由



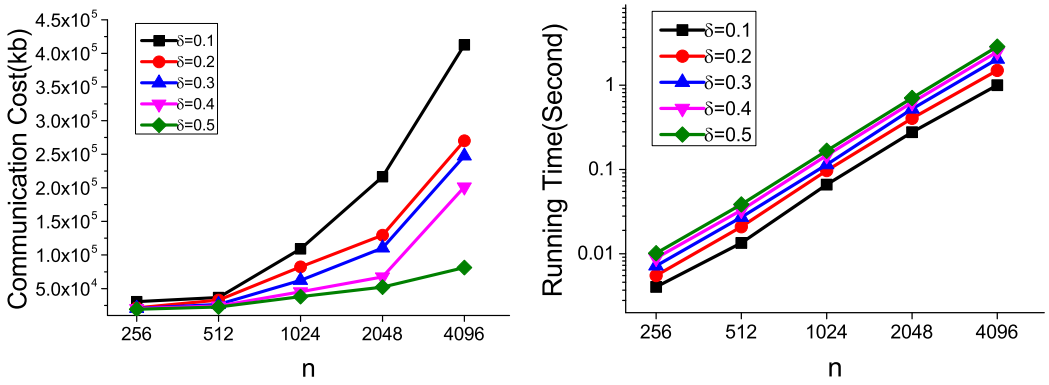
(a) $M = 100$ (b) $M = 1,000$ (c) $M = 10,000$

图 5.4: DTW-FLB 通信开销与 n, M 和 k 之间的关系

此可见，本文提出的由粗到细使用多粒度概要数据进行剪枝的思想具有较高的优越性，即它能够使用较少的数据剪枝掉大多数候选，从而达到了降低通信开销的目的。此外，可以发现随着 k 值的降低，我们能取得更好的剪枝效果。究其原因，k 值较小时，所选取的全局过滤阈值随之降低即更接近真实的第 k 小真实距离值。

接着，我们研究了 n，k 和 M 对 DTW-FLB 算法通信开销的影响。在该组实验中，我们将 M 值从 100 变化到 10,000，将 k 从 1 变化到 100，并将 n 从 256 变化到 4,096。实验结果如图5.4 所示，我们可以看到随着 n 和 M 的指数增加，通信开销随之指数增长。这实际反映了通信开销与 n 和 M 之间存在着线性关系。究其原因，随着 M 的增加，候选轨迹会存在于更多的远程结点中。所以协调者结点需要将概要数据发送到更多的节点中，这导致了通信开销的增加。此外，当 n 增加时，长轨迹需要传递更多的数据才能跟短轨迹达到相同的剪枝效果。最后，随着 k 值的增加，通信开销也会随之增长。这一结论跟图5.3所得到的结果一致，反映了随着 k 的增加，剪枝阈值越松，进而每轮迭代中所剩候选越多。

然后，我们从时间和通信开销两个角度研究了 DTW 距离计算中 δ 的值对 DTW-FLB 算法的影响。在该组实验中，我们将 k 和 m 的值分别固定为 10 和 10,000。实验结果如图5.5所示。我们可以发现随着 δ 的增加通信开销逐步减小。其原因是 δ 值的增加会导致我们的包围信封更加紧凑，从而更利于剪枝。同时，随着 δ 的增加时间开销逐步增加。这是因为 δ 值越大导致最后一步计算 DTW 真实距离时的搜索空间越大。此外，结合前面对 DTW-FLB 的时间复杂度分析部分，我们可以发现



(a) 通信开销与 δ 关系　　(b) 时间开销与 δ 关系

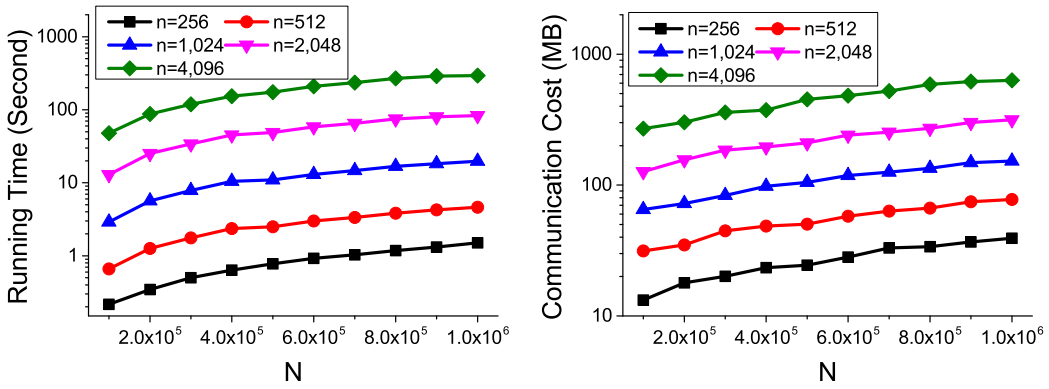
图 5.5: DTW-FLB 性能与 δ 间的关系

这一实验结果跟我们的理论分析是一致的。

5.4.3 算法可扩展性

本小节我们将从时间和通信两个角度研究了 DTW-FLB 算法的可扩展性。首先，考虑了轨迹数量 N（即数据量大小）和轨迹长度 n 对可扩展性的影响。在该组实验中，我们将 k 设置为 1，m 设置为 $10,000$。我们将轨迹长度从 256 到 4096 进行指数级变化，同时将轨迹数量从 10 万到 1 百万进行线性变换。图5.6分别对这两个角度的结果进行了介绍。其中图5.6(a)介绍了时间性能受 n 和 N 的影响。我们可以，看到对于任意长度的轨迹数据集，DTW-FLB 算法的运行时间都随着轨迹数据量的增加而线性增加。此外，随着轨迹长度的指数增加，算法的运行时间也指数增加，即运行时间与轨迹长度呈一定的线性关系。这一结果反映了由于 DTW-FLB 的剪枝效果较好，导致迭代结束后，所剩候选数较少。即我们对 DTW-FLB 算法时间复杂度分析部分的 N' 较少。结合以上两点，我们可以看出 DTW-FLB 算法运行时间随着数据集的大小而线性变换，因此运行效率具有较好的可扩展性。

接着，我们在图5.6(b)中介绍了通信性能受 n 和 N 的影响。从图中可以看出，通信量随着轨迹的数据的线性增加而呈指数增加。这是由于 DTW-FLB 具有较好的剪枝效果，导致迭代结束后，不仅所剩候选较少，而且所剩下的候选结点也较少。因此，迭代结束后的通信较少。而根据我们前面对 DTW-FLB 的通信开销分析



(a) 时间可扩展与 n 和 N 之间的关系　　(b) 通信可扩展性与 n 和 N 之间的关系

图 5.6: The scalability of DTW-FLB

可知通信开销主要来自迭代式计算部分和迭代后将轨迹发送给所有的候选结点部分。由于后一部分开销较少，故主要开销集中在第一部分，即迭代式通信部分。根据图5.3分析可知，前两次迭代已经过滤掉很过结点，后面的候选变话较小。此时，第一部分的开销主要跟包围盒大小相关。而包围盒大小是呈指数变换（当前大小是上一次的 2 倍）。所以，我们的实验结果跟我们的理论分析一致。此外，从图中可以看出通信开销随着轨迹长度的指数变化而指数增加，即通信开销与轨迹长度呈线性关系。这反应了对于长轨迹需要更多的概要数据才能达到与短轨迹相同的剪枝效果。

5.5 本章小结

本章节介绍了利用 FLB 框架在嵌入动态时间卷曲距离下的具体实现算法 DTW-FLB。本章节，首先提出了针对动态时间卷曲距离的包围信封以作为概要数据。接着，提出了基于包围信封的动态时间距离的下界。该下界能够随着包围信封粒度的增加而逐渐变紧，这使得我们利用将该距离嵌入到 FLB 框架称为可能。为此，我们提出了 DTW-FLB 算法以实现基于动态时间卷曲距离的查询。在我们的查询算法中，我们引入了索引、边计算下界边剪枝等查询优化措施，以提高效率。通过在真实数据集上进行的实验，表明 DTW-FLB 算法剪枝效果较好、算法效率高且具有较好的可扩展性。

5.6 附件

引理 5.2.1. 给定 $\boldsymbol{q}$、$\boldsymbol{c}$ 以及 $\boldsymbol{q}$ 的包围矩形 $\{\boldsymbol{u},\boldsymbol{l}\}$，我们有 $SED\_LB(\{\boldsymbol{u},\boldsymbol{l}\},\boldsymbol{c}) \leq SED(\boldsymbol{q},\boldsymbol{c})$。

证明. 对于点的任意维度，如第 j 维，我们有如下结论

$$\begin{cases} (\boldsymbol{u}^j - \boldsymbol{c}^j)^2 \leq (\boldsymbol{q}^j - \boldsymbol{c}^j)^2 & \text{if } \boldsymbol{u}^j \leq \boldsymbol{c}^j, \\ 0 \leq (\boldsymbol{q}^j - \boldsymbol{c}^j)^2 & \text{if } \boldsymbol{l}^j \leq \boldsymbol{c}^j \leq \boldsymbol{u}^j, \\ (\boldsymbol{c}^j - \boldsymbol{l}^j)^2 \leq (\boldsymbol{q}^j - \boldsymbol{c}^j)^2 & \text{if } \boldsymbol{c}^j \leq \boldsymbol{l}^j. \end{cases} \tag{5.8}$$

由于 $SED\_LB(\{\boldsymbol{u},\boldsymbol{l}\},\boldsymbol{c})$ 可看做上式左边部分在各个维度的累加，$SED(\boldsymbol{q},\boldsymbol{c})$ 是上式右半部分的累加。故累加所有维度的值后，我们可以得到 $SED\_LB(\{\boldsymbol{u},\boldsymbol{l}\},\boldsymbol{c}) \leq SED(\boldsymbol{q},\boldsymbol{c})$。 □

定理 5.2.2. 给定轨迹 C 和待查询轨迹 Q 满足动态时间卷曲约束的包围信封 $\{\mathcal{U},\mathcal{L}\}$，我们有如下结论: $LB\_Keogh(\{\mathcal{U},\mathcal{L}\},\mathcal{C}) \leq DTW(Q,C)$。

证明. 根据 $LB\_Keogh(\{\mathcal{U},\mathcal{L}\},\mathcal{C})$ 和 $DTW(Q,C)$ 的定义，我们的目标即是证明如下不等式成立。

$$\sum_{i=0}^{n-1} SED\_LB(\{\boldsymbol{u}_i,\boldsymbol{l}_i\},\boldsymbol{c}_i) \leq \sum_{k=1}^{K} \boldsymbol{w}_k \tag{5.9}$$

此外我们还有 $n \leq K$，我们接下来的策略是证明上述不等式左边的每个值都能在右边找到一个比它大或相等的元素:

$$\begin{aligned} & SED\_LB(\{\boldsymbol{u}_i,\boldsymbol{l}_i\},\boldsymbol{c}_i) \leq \boldsymbol{w}_k \\ \Leftrightarrow\ & SED\_LB(\{\boldsymbol{u}_i,\boldsymbol{l}_i\},\boldsymbol{c}_i) \leq ||\boldsymbol{c}_i^j - \boldsymbol{q}_x^j||^2 \\ \Leftrightarrow\ & \sum_{j=0}^{d-1} SED\_LB(\{\boldsymbol{u}_i^j,\boldsymbol{c}_i^j\},\boldsymbol{c}_i^j) \leq \sum_{j=0}^{d-1} (\boldsymbol{c}_i^j - \boldsymbol{q}_x^j)^2 \end{aligned} \tag{5.10}$$

其中 x 与 i 满足如下不等式约束 $i-\delta \leq x \leq i+\delta$。同时，根据 $\boldsymbol{u}_i$ 和 $\boldsymbol{l}_i$ 的定义，我

们有 $\forall j\ \boldsymbol{l}_i^j \le \boldsymbol{q}_x^j \le \boldsymbol{u}_i^j$. 因此，我们进一步的得到如下不等式：

$$\begin{aligned}
&\begin{cases}
(\boldsymbol{u}_i^j - \boldsymbol{c}_i^j)^2 \le (\boldsymbol{c}_i^j - \boldsymbol{q}_x^j)^2 & \text{if } \boldsymbol{u}_i^j \le \boldsymbol{c}_i^j, \\
0 \le (\boldsymbol{c}_i^j - \boldsymbol{q}_x^j)^2 & \text{if } \boldsymbol{l}_i^j \le \boldsymbol{c}_i^j \le \boldsymbol{u}_i^j, \\
(\boldsymbol{c}_i^j - \boldsymbol{l}_i^j)^2 \le (\boldsymbol{c}_i^j - \boldsymbol{q}_x^j)^2 & \text{if } \boldsymbol{c}_i^j \le \boldsymbol{l}_i^j.
\end{cases} \\
\Leftrightarrow\ & SED_LB(\{\boldsymbol{u}_i^j, \boldsymbol{c}_i^j\}, \boldsymbol{c}_i^j) \le (\boldsymbol{c}_i^j - \boldsymbol{q}_x^j)^2 \\
\Leftrightarrow\ & \sum_{j=0}^{d-1} SED_LB(\{\boldsymbol{u}_i^j, \boldsymbol{c}_i^j\}, \boldsymbol{c}_i^j) \le \sum_{j=0}^{d-1} (\boldsymbol{c}_i^j - \boldsymbol{q}_x^j)^2
\end{aligned}$$

因此, 不等式 5.10 成立。原问题得证。 □

定理 5.2.3. *给定轨迹 $\mathcal{Q}$ 的第 l 层包围信封 $\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}$ 和候选轨迹 $\mathcal{C}$, 我们得到如下结论：* $LB\_HPAA(\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}, \overline{\mathcal{C}}_l) \le DTW(\mathcal{Q}, \mathcal{C})$。

证明. 从定理5.2.2可知 LB_keogh 是原始 DTW 距离的下界。因此，若我们能得到 $LB\_HPAA(\{\hat{\mathcal{U}}_l, \hat{\mathcal{L}}_l\}, \overline{\mathcal{C}}_l) \le LB\_Keogh(\{\mathcal{U}, \mathcal{L}\}, \mathcal{C})$, 则原问题得证。根据 LB_HPAA 的定义，我们可知第 l 层包围信封的每个元素对应着 LB_Keogh 的 s ($s = 2^{L-l}$) 个元素。如果 LB_HPAA 的每个元素满足如下不等式，则原问题得证。

$$s \cdot SED\_LB(\{\hat{\boldsymbol{u}}_{l,i}, \hat{\boldsymbol{l}}_{l,i}\}, \overline{\boldsymbol{c}}_{l,i}) \le \sum_{p=i\cdot s}^{(i+1)\cdot s-1} SED\_LB(\{\boldsymbol{u}_p, \boldsymbol{l}_p\}, \boldsymbol{c}_p)$$

根据不等式5.11，由于 $\forall j\ \hat{\boldsymbol{l}}_{l,i}^j \le \boldsymbol{l}_p^j \le \boldsymbol{u}_p^j \le \hat{\boldsymbol{u}}_{l,i}^j$，我们有 $SED\_LB(\{\hat{\boldsymbol{u}}_{l,i}, \hat{\boldsymbol{l}}_{l,i}\}, \boldsymbol{c}_p) \le SED\_LB(\{\boldsymbol{u}_p, \boldsymbol{l}_p\}, \boldsymbol{c}_p)$。

$$\begin{cases}
(\hat{\boldsymbol{u}}_{l,i}^j - \boldsymbol{c}_p^j)^2 \le (\boldsymbol{u}_p^j - \boldsymbol{c}_p^j)^2 & \text{if } \boldsymbol{u}_p^j \le \hat{\boldsymbol{u}}_{l,i}^j \le \boldsymbol{c}_p^j, \\
0 \le (\boldsymbol{u}_p^j - \boldsymbol{c}_p^j)^2 & \text{if } \boldsymbol{u}_p^j \le \boldsymbol{c}_p^j \le \hat{\boldsymbol{u}}_{l,i}^j, \\
0 \le 0 & \text{if } \boldsymbol{l}_p^j \le \boldsymbol{c}_p^j \le \boldsymbol{u}_p^j, \\
0 \le (\boldsymbol{l}_p^j - \boldsymbol{c}_p^j)^2 & \text{if } \hat{\boldsymbol{l}}_{l,i}^j \le \boldsymbol{c}_p^j \le \boldsymbol{l}_p^j, \\
(\hat{\boldsymbol{l}}_{l,i}^j - \boldsymbol{c}_p^j)^2 \le (\boldsymbol{l}_p^j - \boldsymbol{c}_p^j)^2 & \text{if } \boldsymbol{c}_p^j \le \hat{\boldsymbol{l}}_{l,i}^j \le \boldsymbol{l}_p^j.
\end{cases} \tag{5.11}$$

所以,我们的问题变为求证 $s{\cdot}SED\_LB(\{\hat{\boldsymbol{u}}_{l,i},\hat{\boldsymbol{l}}_{l,i}\},\bar{\boldsymbol{c}}_{l,i}) \le \sum_{p=i\cdot s}^{(i+1)\cdot s-1} SED\_LB(\{\hat{\boldsymbol{u}}_{l,i},\hat{\boldsymbol{l}}_{l,i}\},\boldsymbol{c}_p)$ 成立。为简化证明，我们只证明第一个轨迹片段（即 $i=0$ 时）成立。此外我们使用 $\hat{\boldsymbol{u}}$,$\hat{\boldsymbol{l}}$ 和 $\bar{\boldsymbol{c}}$ 来分别表示 $\hat{\boldsymbol{u}}_{l,0}$,$\hat{\boldsymbol{l}}_{l,0}$ 和 $\bar{\boldsymbol{c}}_{l,0}$。对于其他片段，证明方法相同。此时，我们的目标就是证明如下不等式:

$$s \cdot SED\_LB(\{\hat{\boldsymbol{u}},\hat{\boldsymbol{l}}\},\bar{\boldsymbol{c}}) \le \sum_{i=0}^{s-1} SED\_LB(\{\hat{\boldsymbol{u}},\hat{\boldsymbol{l}}\},\boldsymbol{c}_i) \tag{5.12}$$

根据 SED_LB 定义，我们展开不等式5.12。此时，若对于任一维度 j 有如下不等式成立则不等式5.12也成立。

$$s \cdot SED\_LB(\{\hat{\boldsymbol{u}}^j,\hat{\boldsymbol{l}}^j\},\bar{\boldsymbol{c}}^j) \le \sum_{i=0}^{s-1} SED\_LB(\{\hat{\boldsymbol{u}}^j,\hat{\boldsymbol{l}}^j\},\boldsymbol{c}_i^j) \tag{5.13}$$

对于以上不等式的右半部分，$\boldsymbol{c}_i^j$ 和 $\{\hat{\boldsymbol{u}}^j,\hat{\boldsymbol{l}}^j\}$ 之间存在 3 种大小关系。不失一般性，我们假设这 3 种关系均出现在这 s 对元组中。为此，我们首先将所有 $\boldsymbol{c}_i$ 进行重新排序，使得排序后的结果满足:（i）从 $\boldsymbol{c}_0$ 到 $\boldsymbol{c}_{p-1}$ 中的元素,满足 $\boldsymbol{c}_i^j \le \hat{\boldsymbol{u}}^j$;（ii）从 $\boldsymbol{c}_p$ 到 $\boldsymbol{c}_{q-1}$ 中的元素，满足 $\hat{\boldsymbol{l}}^j \le \boldsymbol{c}_i^j \le \hat{\boldsymbol{u}}^j$;（iii）从 $\boldsymbol{c}_q$ 到 $\boldsymbol{c}_{s-1}$ 中的元素满足 $\boldsymbol{c}_i^j \le \hat{\boldsymbol{l}}^j$，其中 $0 \le p \le q \le s$。那么根据 $\bar{\boldsymbol{c}}$ 的定义,我们得到下述结论:

$$\sum_{i=0}^{p-1}(\boldsymbol{c}_i^j - \bar{\boldsymbol{c}}^j) = \sum_{i=p}^{s-1}(\bar{\boldsymbol{c}}^j - \boldsymbol{c}_i^j) = \sum_{i=p}^{q-1}(\bar{\boldsymbol{c}}^j - \boldsymbol{c}_i^j) + \sum_{i=q}^{s-1}(\bar{\boldsymbol{c}}^j - \boldsymbol{c}_i^j) \tag{5.14}$$

然后，我们考虑不等式5.13的左半部分。其根据 $\hat{\boldsymbol{u}}^j$ 和 $\bar{\boldsymbol{c}}^j$ 之间的大小关系可分为 3 种情况。（i）$\hat{\boldsymbol{u}}^j \le \bar{\boldsymbol{c}}^j$ 的情况，此时左半部分的值为 $s(\bar{\boldsymbol{c}}^j - \hat{\boldsymbol{u}}^j)^2$。此时，对于右半部分我们有

$$\begin{aligned}
&\sum_{i=0}^{s-1}(\{\hat{\boldsymbol{u}}^j - \hat{\boldsymbol{l}}^j\},\boldsymbol{c}_i^j)^2 = \sum_{i=0}^{p-1}(\boldsymbol{c}_i^j - \hat{\boldsymbol{u}}^j)^2 + \sum_{i=q}^{s-1}(\hat{\boldsymbol{l}}^j - \boldsymbol{c}_i^j)^2 \\
\ge\ & \frac{1}{s-q+p}(\sum_{i=0}^{p-1}(\boldsymbol{c}_i^j - \hat{\boldsymbol{u}}^j) + \sum_{i=q}^{s-1}(\hat{\boldsymbol{l}}^j - \boldsymbol{c}_i^j))^2 \quad (AM-GM\square\square\square) \\
\ge\ & \frac{1}{s-q+p}(\sum_{i=p}^{q-1}(\bar{\boldsymbol{c}}^j - \boldsymbol{c}_i^j) + p(\bar{\boldsymbol{c}}^j - \hat{\boldsymbol{u}}^j) + \sum_{i=q}^{s-1}(\hat{\boldsymbol{l}}^j - 2\boldsymbol{c}_i^j + \bar{\boldsymbol{c}}^j))^2
\end{aligned}$$

$$\begin{aligned} &\geq \frac{1}{s-q+p}(\sum_{i=p}^{q-1}(\bar{\boldsymbol{c}}^j-\hat{\boldsymbol{u}}^j)+p(\bar{\boldsymbol{c}}^j-\hat{\boldsymbol{u}}^j)+\sum_{i=q}^{s-1}(\bar{\boldsymbol{c}}^j-\hat{\boldsymbol{l}}^j))^2 \\ &= \frac{s^2}{s-q+p}(\bar{\boldsymbol{c}}^j-\hat{\boldsymbol{u}}^j)^2 \geq s(\bar{\boldsymbol{c}}^j-\hat{\boldsymbol{u}}^j)^2 \end{aligned} \tag{5.15}$$

因此，第一种情况下原问题得证。(ii) $\bar{\boldsymbol{c}}^j \leq \hat{\boldsymbol{l}}^j$ 的情况，此时证明过程与第一种情况类似。(iii) $\hat{\boldsymbol{l}}^j \leq \bar{\boldsymbol{c}}^j \leq \hat{\boldsymbol{u}}^j$ 的情况，此时不等式5.13成立，因其左半部分值为 0，而有半部分是在大于 0。结合以上 3 种情况，原问题得证。 □

定理 5.2.4. *LB_HPAA 下界能随着查询轨迹包围信封层次的增加而逐渐变紧。也就是说：$LB\_HPAA(\{\hat{\mathcal{U}}_l,\hat{\mathcal{L}}_l\},\overline{\mathcal{C}}_l) \leq LB\_HPAA(\{\hat{\mathcal{U}}_{l+1},\hat{\mathcal{L}}_{l+1}\},\overline{\mathcal{C}}_{l+1})$。*

证明. 根据多粒度包围信封的计算法方式我们可知，当包围信封从第 l 层转到第 $l+1$ 层时，其每个元素分为两个元素用于分别表示左右两边的最值。所以，我们只要证明如下不等式即可:

$$\begin{aligned} 2\cdot SED\_LB(\{\hat{\boldsymbol{u}}_{l,i},\hat{\boldsymbol{l}}_{l,i}\},\bar{\boldsymbol{c}}_{l,i}) \leq\ & SED\_LB(\{\hat{\boldsymbol{u}}_{l+1,2i},\hat{\boldsymbol{l}}_{l+1,2i}\},\bar{\boldsymbol{c}}_{l+1,2i}) \\ & +SED\_LB(\{\hat{\boldsymbol{u}}_{l+1,2i+1},\hat{\boldsymbol{l}}_{l+1,2i+1}\},\bar{\boldsymbol{c}}_{l+1,2i+1}) \end{aligned}$$

为简化问题，我们分别使用 $\hat{\boldsymbol{u}}$，$\hat{\boldsymbol{u}}_L$ 和 $\hat{\boldsymbol{u}}_R$ 来分别表示 $\hat{\boldsymbol{u}_{l,i}}$,$\hat{\boldsymbol{u}}_{l+1,2i}$ 和 $\hat{\boldsymbol{u}}_{l+1,2i+1}$。并且使用 $\bar{\boldsymbol{c}}$，$\bar{\boldsymbol{c}}_L$ 和 $\bar{\boldsymbol{c}}_R$ 来分别代表 $\bar{\boldsymbol{c}}_{l,i}$，$\bar{\boldsymbol{c}}_{l+1,2i}$ 和 $\bar{\boldsymbol{c}}_{l+1,2i+1}$。那么不等式可重新表示为如下形式:

$$\begin{aligned} 2\cdot SED\_LB(\{\hat{\boldsymbol{u}},\hat{\boldsymbol{l}}\},\bar{\boldsymbol{c}}) \ &\leq\ SED\_LB(\{\hat{\boldsymbol{u}}_L,\hat{\boldsymbol{l}}_L\},\bar{\boldsymbol{c}}_L) \\ &+\ SED\_LB(\{\hat{\boldsymbol{u}}_R,\hat{\boldsymbol{l}}_R\},\bar{\boldsymbol{c}}_R) \end{aligned} \tag{5.16}$$

此时如果我们能证明对任一维度 j，都能满足如下不等式，则不等式5.16成立。

$$\begin{aligned} 2\cdot SED\_LB(\{\hat{\boldsymbol{u}}^j,\hat{\boldsymbol{l}}^j\},\bar{\boldsymbol{c}}^j) \ &\leq\ SED\_LB(\{\hat{\boldsymbol{u}}^j_L,\hat{\boldsymbol{l}}^j_L\},\bar{\boldsymbol{c}}^j_L) \\ &+\ SED\_LB(\{\hat{\boldsymbol{u}}^j_R,\hat{\boldsymbol{l}}^j_R\},\bar{\boldsymbol{c}}^j_R) \end{aligned} \tag{5.17}$$

对于不等式5.17左半部分，$\bar{\boldsymbol{c}}^j$ 和 $\{\hat{\boldsymbol{u}}^j, \hat{\boldsymbol{l}}^j\}$ 之间存在以下三种关系：

（i）$\bar{\boldsymbol{c}}^j \geq \hat{\boldsymbol{u}}^j$，此时左半部分等价于 $2 \cdot (\bar{\boldsymbol{c}}^j - \hat{\boldsymbol{u}}^j)^2$。而右半部分，我们首先假设 $\bar{\boldsymbol{c}}_L^j \leq \bar{\boldsymbol{c}}^j \leq \bar{\boldsymbol{c}}_R^j$。此时我们得到如下结论 $SED\_LB(\{\hat{\boldsymbol{u}}_L^j, \hat{\boldsymbol{l}}_L^j\}, \bar{\boldsymbol{c}}_L^j) \geq (\bar{\boldsymbol{c}}_L^j - \hat{\boldsymbol{u}}^j)^2$。因此，不等式5.17 左半部分等于 $2 \cdot SED(\bar{\boldsymbol{c}}, \hat{\boldsymbol{u}})$。对于其右半部分，我们有

$$
\begin{aligned}
& SED\_LB(\{\hat{\boldsymbol{u}}_L^j, \hat{\boldsymbol{l}}_L^j\}, \bar{\boldsymbol{c}}_L^j) + SED\_LB(\{\hat{\boldsymbol{u}}_R^j, \hat{\boldsymbol{l}}_R^j\}, \bar{\boldsymbol{c}}_R^j) \\
= \;& SED\_LB(\{\hat{\boldsymbol{u}}_L^j, \hat{\boldsymbol{l}}_L^j\}, \bar{\boldsymbol{c}}_L^j) + (\bar{\boldsymbol{c}}_R^j - \hat{\boldsymbol{u}}_R^j)^2 \quad (\hat{\boldsymbol{u}}_R^j \leq \hat{\boldsymbol{u}}^j \leq \bar{\boldsymbol{c}}^j \leq \bar{\boldsymbol{c}}_R^j) \\
\geq \;& (\bar{\boldsymbol{c}}_L^j - \hat{\boldsymbol{u}}^j)^2 + (\bar{\boldsymbol{c}}_R^j - \hat{\boldsymbol{u}}_R^j)^2 \\
\geq \;& (\bar{\boldsymbol{c}}_L^j - \hat{\boldsymbol{u}}^j)^2 + (\bar{\boldsymbol{c}}_R^j - \hat{\boldsymbol{u}}^j)^2 \qquad (\bar{\boldsymbol{u}}_R^j \leq \hat{\boldsymbol{u}}^j \leq \bar{\boldsymbol{c}}_R^j) \\
\geq \;& \frac{1}{2} \cdot (\bar{\boldsymbol{c}}_L^j - \hat{\boldsymbol{u}}^j + \bar{\boldsymbol{c}}_R^j - \hat{\boldsymbol{u}}^j)^2 \qquad (AM - GM\ Inequality) \\
= \;& \frac{1}{2} \cdot (2\bar{\boldsymbol{c}}^j - 2\hat{\boldsymbol{u}}^j)^2 \qquad (\bar{\boldsymbol{c}}_L^j + \bar{\boldsymbol{c}}_R^j = 2\bar{\boldsymbol{c}}^j) \\
= \;& 2 \cdot (\bar{\boldsymbol{c}}^j - \hat{\boldsymbol{u}}^j)^2
\end{aligned}
$$

对于 $\bar{\boldsymbol{c}}_R^j \leq \bar{\boldsymbol{c}}^j \leq \bar{\boldsymbol{c}}_L^j$ 的情况，我们通过相同的方法得到同样的结论。因此，此种情况下不等式 5.16成立。(ii) $\bar{\boldsymbol{c}}^j \leq \hat{\boldsymbol{l}}^j$, 可通过跟第一种相同的方法证明不等式 5.17 成立。(iii) $\hat{\boldsymbol{l}}^j \leq \bar{\boldsymbol{c}}^j \leq \hat{\boldsymbol{u}}^j$，此时不等式5.17 的左半部分为 0，其右半部分始终大于 0。因此，结论也成立。综合以上几种情况, 不等式5.16 始终成立。原问题得证。 □

第六章　总结与展望

本文重点研究了分布式轨迹数据上的 k 近邻查询。本章对全文研究内容和技术模型进行总结归纳，并展望未来的重点研究工作。

6.1　研究总结

随着移动设备的广泛应用，各行各业产生了海量的同时分布在各存储结点的轨迹数据，面临着如何利用该数据的问题。轨迹 k 近邻查询是轨迹数据挖掘研究的重要内容，也是基于位置服务的基本功能。针对海量轨迹数据，分布式场景下的 k 近邻查询问题应运而生。本文重点研究了基于协调者结点-远程结点分布式架构下的 k 近邻查询，旨在从以下三个方面解决相关问题: （1）目前存在着许多轨迹距离度量方式，缺乏适用于所有度量方式的查询处理框架。设计统一的处理框架可以满足不同应用场景的需求，使得处理的问题不再受限。（2）如何降低查询的通信开销；相对于传统集中式环境下的查询，分布式环境下通信开销成为算法的首要瓶颈。（3）如何降低查询的时间开销；提高查询的处理效率，对查询处理有着重要意义。。具体地，本文的研究问题和技术贡献总结如下。

首先，针对缺乏适用于所有度量方式的查询处理框架，本文基于多粒度概要数据模型和基于概要数据的上、下界提出了两种查询处理框架: FTB 和 FLB。传统的近邻查询针对具体的距离度量方式，设计具体的查询实现算法。而本文提出的框架能满足不同距离度量方式的需求。其中 FTB 适用于那些根据概要数据能同时计算出上、下界的场景。而 FLB 适用于那些根据概要数据仅能计算出距离下界的场景。针对具体的距离度量方式，我们只需研究出适合的概要数据并提供界信息，便能应用到对应的框架中。

其次，针对分布式查询的通信开销问题，本文强调了通信开销对查询方法性能的影响。现有基于集中式环境和分布式环境的查询往往关注方法的时效性，鲜有考

虑通信开销对查询算法的影响。而通信开销是分布式算法的性能的首要指标。基于此本文设计的使用概要数据进行剪枝的思想能大大减少所要传输的数据量。进一步地，我们分别针对欧式距离设计了基于小波变换的多粒度概要数据，针对动态时间卷曲距离设计了基于包围信封的多粒度概要数据。本文设计的迭代式算法将概要数据由粗到细粒度发送到远程结点，并在每个结点进行剪枝。理论分析和基于公开数据集的实验表明本文所提方法能显著降低查询的通信开销。

最后，针对**分布式 k 近邻轨迹查询的时效问题**，本文强调了时间效率队查询方法性能的影响。现有的查询方法大都使用索引的方式来提高查询效率。但本文处理的分布式场景中，不要求各个节点构建轨迹索引。为此，本文提出通过构建距离上、下界进行剪枝的策略能有效地降低查询方法的复杂度。特别地，针对欧式距离应用的场景，设计在更新上、下界的过程中进行剪枝的策略以进一步降低计算开销。针对动态时间卷曲距离应用的场景，引入了使用多个下界进行级联剪枝和在更新下界过程中剪枝的两种剪枝策略，以提高查询效率。理论分析和基于公开数据集的实验表明本文所提方法能显著提高查询的时间效率。

6.2 研究展望

本文重点研究了分布式 k 近邻轨迹查询问题。作者认为还可以从下面三个方面进行扩展。

首先，本文针对语义轨迹上的查询可以进行拓展研究。本文所研究的轨迹数据来自欧式空间，如何结合含非欧空间的语义信息进行近邻查询会是一个有趣的问题。一些前人工作 [57, 58, 75] 已经考虑了对语义轨迹上的查询。因此，基于轨迹的查询也是未来研究的重点。

其次，本文研究了所提两个查询框架分别在欧式和动态时间卷曲这两个常用距离的具体实现。附件部分还指出了在最长公共子串距离下的应用。但轨迹距离度量方式还有许多，在以后的工作中需要针对具体的度量方式设计适合的概要数据，并为概要数据提供距离范围计算方法。

最后，本文所提算法是针对协调者-远程结点这一分布式场景设计的查询算法。随着区块链[1]、以太坊[2]等基于 P2P 系统和概念的兴起，基于 P2P 这一分布式架构的轨迹等时间序列上的近邻查询会成为未来的热点。

[1]https://blockchain.info/

[2]https://www.ethereum.org/

参考文献

[1] KEOGH E J, CHAKRABARTI K, PAZZANI M J, et al. Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases[J]. Knowledge and Information System (KIS), 2001, 3(3) : 263 – 286.

[2] ZHENG Y. Computing with spatial trajectories[M]. New York,USA : Springer, 2011 : 16.

[3] ZHENG Y. Trajectory Data Mining: An Overview[J]. ACM TIST, 2015, 6(3) : 29:1 – 29:41.

[4] WANG H, ZHENG K, XU J, et al. SharkDB: An In-Memory Column-Oriented Trajectory Storage[C] // Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM. 2014 : 1409 – 1418.

[5] TAN H, LUO W, NI L M. CloST: a hadoop-based storage system for big spatio-temporal data analytics[C] // Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management(CIKM). 2012 : 2139 – 2143.

[6] ZHANG Z, JIN C, MAO J, et al. TrajSpark: A Scalable and Efficient In-Memory Management System for Big Trajectory Data[C] // Proceedings of the First International APWeb-WAIM Joint Conference on Web and Big Data, Part I. 2017 : 11 – 26.

[7] XIE D, LI F, PHILLIPS J M. Distributed Trajectory Similarity Search[J]. Proceedings of the 43rd International Conference on Very Large Data Bases (PVLDB), 2017, 10(11) : 1478 – 1489.

[8] DENG Z, HU Y, ZHU M, et al. A scalable and fast OPTICS for clustering trajectory big data[J]. Cluster Computing, 2015, 18(2) : 549 – 562.

[9] COSTA G, MANCO G, MASCIARI E. Dealing with trajectory streams by clustering and mathematical transforms[J]. Journal of Intelligent Information Systems,, 2014, 42(1) : 155 – 177.

[10] YU Y, WANG Q, WANG X, et al. Online clustering for trajectory data stream of moving objects[J]. Computer Science and Information Systems, 2013, 10(3) : 1293 – 1317.

[11] MAO J, SONG Q, JIN C, et al. TSCluWin: Trajectory Stream Clustering over Sliding Window[C] // Proceedings of the 21st International Conference onDatabase Systems for Advanced Applications (DASFAA), Part II. 2016 : 133 – 148.

[12] NEHME R V, RUNDENSTEINER E A. SCUBA: Scalable Cluster-Based Algorithm for Evaluating Continuous Spatio-temporal Queries on Moving Objects[C] // Proceedings of the 10th International Conference on Extending Database Technology (EDBT). 2006 : 1001 – 1019.

[13] SACHARIDIS D, PATROUMPAS K, TERROVITIS M, et al. On-line discovery of hot motion paths[C] // Proceedings of the 11th International Conference on Extending Database Technology (EDBT). 2008 : 392 – 403.

[14] ZHENG K, ZHENG Y, YUAN N J, et al. On discovery of gathering patterns from trajectories[C] // Proceedings of the 29th IEEE International Conference on Data Engineering (ICDE). 2013 : 242 – 253.

[15] TANG L A, ZHENG Y, YUAN J, et al. On Discovery of Traveling Companions from Streaming Trajectories[C] // Proceedings of the 28th IEEE International Conference on Data Engineering (ICDE). 2012 : 186 – 197.

[16] LI X, CEIKUTE V, JENSEN C S, et al. Effective Online Group Discovery in Trajectory Databases[J]. IEEE Transaction on Knowledge and Data Engineering (TKDE), 2013, 25(12) : 2752 – 2766.

[17] DUAN X, JIN C, WANG X, et al. Real-Time Personalized Taxi-Sharing[C] // Proceedings of the 21st International Conference onDatabase Systems for Advanced Applications (DASFAA), Part II. 2016 : 451 – 465.

[18] ZHANG Z, WANG Y, MAO J, et al. DT-KST: Distributed Top-k Similarity Query on Big Trajectory Streams[C] // Proceedings of the 22nd International Conference onDatabase Systems for Advanced Applications (DASFAA), PartI. 2017 : 199 – 214.

[19] HSU C-C, KUNG P-H, YEH M-Y, et al. Bandwidth-efficient distributed k-nearest-neighbor search with dynamic time warping[C] // Proceedings of the 2015 International Conference on Big Data(ICBD). 2015 : 551 – 560.

[20] ZEINALIPOUR-YAZTI D, LAOUDIAS C, COSTA C, et al. Crowdsourced trace similarity with smartphones[J]. IEEE Transaction on Knowledge and Data Engineering (TKDE), 2013, 25(6) : 1240 – 1253.

[21] COSTA C, LAOUDIAS C, ZEINALIPOUR-YAZTI D, et al. SmartTrace: Finding similar trajectories in smartphone networks without disclosing the traces[C] // Proceedings of the 27th IEEE International Conference on Data Engineering (ICDE). 2011 : 1288 – 1291.

[22] CHEN L, OZSU M T, ORIA V. Robust and fast similarity search for moving object trajectories[C] // Proceedings of the 2005 ACM International Conference on Management of Data (SIGMOD). 2005 : 491 – 502.

[23] CHEN L, NG R T. On The Marriage of Lp-norms and Edit Distance[C] // Proceedings of the 30th International Conference on Very Large(VLDB). 2004 : 792 – 803.

[24] ZHU H, LUO J, YIN H, et al. Mining Trajectory Corridors Using Frechet Distance and Meshing Grids[C] // Proceedings of the 14th Pacific-Asia Advances in Knowledge Discovery and Data Mining (PAKDD). 2010 : 228 – 237.

[25] GUO N, MA M, XIONG W, et al. An Efficient Query Algorithm for Trajectory Similarity Based on Frechet Distance Threshold[J]. International Journal of Geo-Information, 2017, 6(11) : 326.

[26] LIN B, SU J. One Way Distance: For Shape Based Similarity Search of Moving Object Trajectories[J]. GeoInformatica, 2008, 12(2) : 117 – 142.

[27] LIU H, SCHNEIDER M. Similarity measurement of moving object trajectories[C] // Proceedings of the 2012 ACM International Workshop on GeoStreaming (SIGSPATIAL). 2012 : 19 – 22.

[28] ZHAO X, XU W. A New Measurement Method to Calculate Similarity of Moving Object Spatio-Temporal Trajectories by Compact Representation[J]. International Journal of Computational Intelligence Systems, 2011, 4(6) : 1140 – 1147.

[29] ZHENG B, YUAN N J, ZHENG K, et al. Approximate keyword search in semantic trajectory database[C] // Proceedings of the 31st IEEE International Conference on Data Engineering (ICDE). 2015 : 975 – 986.

[30] NEHAL M, A. S M, MOSTAFA T, et al. Review on trajectory similarity measures[C] // Proceedings of the 7th IEEE International Conference on Intelligent Computing and Information Systems. 2016 : 613 – 619.

[31] TOOHEY K, DUCKHAM M. Trajectory similarity measures[J]. Proceedings of the 2015 ACM International Workshop on GeoStreaming (SIGSPATIAL), 2015, 7(1) : 43 – 50.

[32] VERNICA R, CAREY M J, LI C. Efficient parallel set-similarity joins using MapReduce[C] // Proceedings of the 2010 ACM International Conference on Management of Data (SIGMOD). 2010 : 495 – 506.

[33] KIM Y, SHIM K. Parallel top-k similarity join algorithms using MapReduce[C] // Proceedings of the IEEE 28th International Conference on Data Engineering (ICDE). 2012 : 510 – 521.

[34] 江俊文, 王晓玲. 轨迹数据压缩综述 [J]. 华东师范大学学报 (自然科学版), 2015(5) : 61 – 76.

[35] HERSHBERGER J, SNOEYINK J. Speeding Up the Douglas-Peucker Line-Simplification Algorithm[J]. Proceedings of International Symposium on Spatial Data Handling, 1992 : 134 – 143.

[36] MERATNIA N, de BY R A. Spatiotemporal Compression Techniques for Moving Point Objects[C] // Proceedings of the 9th International Conference on Extending Database Technology (EDBT). 2004 : 765 – 782.

[37] LIU J, ZHAO K, SOMMER P, et al. Bounded Quadrant System: Error-bounded trajectory compression on the go[C] // Proceedings of the 31st IEEE International Conference on Data Engineering (ICDE). 2015 : 987 – 998.

[38] AGRAWAL R, FALOUTSOS C, SWAMI A N. Efficient Similarity Search In Sequence Databases[C] // Proceedings of International Conference on Foundations of Data Organization and Algorithms. 1993 : 69 – 84.

[39] FALOUTSOS C, RANGANATHAN M, MANOLOPOULOS Y. Fast subsequence matching in time-series databases[J]. Proceedings of the 1994 ACM International Conference on Management of Data (SIGMOD), 1994, 23(2) : 419 – 429.

[40] RAFIEI D, MENDELZON A O. Similarity-based queries for time series data[J]. Proceedings of the 1997 ACM International Conference on Management of Data (SIGMOD), 1997, 26(2) : 13 – 25.

[41] RAFIEI D. On similarity-based queries for time series data[C] // Proceedings of the IEEE 15th International Conference on Data Engineering (ICDE). 1999 : 410 – 417.

[42] SHATKAY H. The Fourier Transform - A Primer[M]. Providence, Rhode Island 02912, USA : Brown University, 1995 : 186 – 195.

[43] CHAN F-P, FU A-C, YU C. Haar wavelets for efficient similarity search of time-series: with and without time warping[J]. IEEE Transaction on Knowledge and Data Engineering (TKDE), 2003, 15(3) : 686 – 705.

[44] VITTER J S. Random Sampling with a Reservoir[J]. ACM Transactions on Mathematical Software (TOMS), 1985, 11(1) : 37 – 57.

[45] KEOGH E J, CHU S, HART D M, et al. An Online Algorithm for Segmenting Time Series[C] // Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM). 2001 : 289 – 296.

[46] MERATNIA N, de BY R A. Spatiotemporal Compression Techniques for Moving Point Objects[C] // Proceedings of the 2004 International Conference on Extending Database Technology (EDBT). 2004 : 765 – 782.

[47] POTAMIAS M, PATROUMPAS K, SELLIS T K. Sampling Trajectory Streams with Spatiotemporal Criteria[C] // Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM). 2006 : 275 – 284.

[48] YIN H, WOLFSON O. A Weight-based Map Matching Method in Moving Objects Databases[C] // Proceedings of the 16th International Conference on Scientific and Statistical Database Management (SSDBM). 2004 : 437 – 438.

[49] LERIN P M, YAMAMOTO D, TAKAHASHI N. Encoding Travel Traces by Using Road Networks and Routing Algorithms[C] // Intelligent Interactive Multimedia: Systems and Services. 2012 : 233 – 243.

[50] KELLARIS G, PELEKIS N, THEODORIDIS Y. Trajectory Compression under Network Constraints[C] // Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases(SSTD). 2009 : 392 – 398.

[51] KELLARIS G, PELEKIS N, THEODORIDIS Y. Map-matched trajectory compression[J]. Journal of Systems and Software, 2013, 86(6) : 1566 – 1579.

[52] SONG R, SUN W, ZHENG B, et al. PRESS: A Novel Framework of Trajectory Compression in Road Networks[J]. Procedings of the 40th International Conference on Very Large Data Bases(PVLDB), 2014, 7(9) : 661 – 672.

[53] SCHMID F, RICHTER K, LAUBE P. Semantic Trajectory Compression[C] // Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases(SSTD). 2009 : 411 – 416.

[54] RICHTER K, SCHMID F, LAUBE P. Semantic trajectory compression: Representing urban movement in anutshell[J]. Journal of Spatial Information Science, 2012, 4(1) : 3 – 30.

[55] SU H, ZHENG K, ZENG K, et al. STMaker - A System to Make Sense of Trajectory Data[J]. Procedings of the 40th International Conference on Very Large Data Bases(PVLDB), 2014, 7(13) : 1701 – 1704.

[56] SU H, ZHENG K, ZENG K, et al. Making sense of trajectory data: A partition-and-summarization approach[C] // Proceedings of the 31st IEEE International Conference on Data Engineering (ICDE). 2015 : 963 – 974.

[57] XIANGYE X, YU Z, QIONG L, et al. Finding Similar Users Using Category-based Location History[C] // Proceedings of the 18th International Conference on Advances in Geographic Information Systems(SIGSPATIAL). 2010 : 442 – 445.

[58] ZHENG K, SHANG S, YUAN N J, et al. Towards efficient search for activity trajectories[C] // Proceedings of the IEEE 29th International Conference on Data Engineering (ICDE). 2013 : 230 – 241.

[59] SHANG S, DING R, YUAN B, et al. User oriented trajectory search for trip recommendation[C] // Proceedings of the 15th International Conference on Extending Database Technology, (EDBT). 2012 : 156 – 167.

[60] BOTEA V, MALLETT D, NASCIMENTO M A, et al. PIST: An Efficient and Practical Indexing Technique for Historical Spatio-Temporal Point Data[J]. GeoInformatica, 2008, 12(2) : 143 – 168.

[61] CHAKKA V P, EVERSPAUGH A, PATEL J M. Indexing large trajectory data sets with SETI[C] // Proceedings of the First Biennial Conference on Innovative Data Systems CIDR : Vol 1001. 2003 : 12.

[62] CUDRÉ-MAUROUX P, WU E, MADDEN S. TrajStore: An adaptive storage system for very large trajectory data sets[C] // Proceedings of the 26th IEEE International Conference on Data Engineering (ICDE). 2010 : 109 – 120.

[63] WANG H, ZHENG K, ZHOU X, et al. SharkDB: An In-Memory Storage System for Massive Trajectory Data[C] // Proceedings of the 2015 ACM International Conference on Management of Data (SIGMOD). 2015 : 1099 – 1104.

[64] WANG H, ZHENG K, XU J, et al. SharkDB: An In-Memory Column-Oriented Trajectory Storage[C] // Proceedings of the 23rd ACM International Conference on

Conference on Information and Knowledge Management (CIKM). 2014 : 1409 – 1418.

[65] ZHENG B, WANG H, ZHENG K, et al. SharkDB: an in-memory column-oriented storage for trajectory analysis[J]. World Wide Web, 2018, 21(2) : 455 – 485.

[66] AGRAWAL R, FALOUTSOS C, SWAMI A N. Efficient Similarity Search In Sequence Databases[C] // Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms (FODO). 1993 : 69 – 84.

[67] RAFIEI D, MENDELZON A O. Efficient retrieval of similar shapes[J]. VLDB Journal, 2002, 11(1) : 17 – 27.

[68] NIKOS P, IOANNIS K, GERASIMOS M, et al. Similarity Search in Trajectory Databases[C] // Proceedings of the 14th International Symposium on Temporal Representation and Reasoning. 2007 : 129 – 140.

[69] FRENTZOS E, GRATSIAS K, PELEKIS N, et al. Algorithms for Nearest Neighbor Search on Moving Object Trajectories[J]. GeoInformatica, 2007, 11(2) : 159 – 193.

[70] GÜTING R H, BEHR T, XU J. Efficient k-nearest neighbor search on moving object trajectories[J]. VLDB Journal, 2010, 19(5) : 687 – 714.

[71] KOLLIOS G, GUNOPULOS D, TSOTRAS V J. Nearest Neighbor Queries in a Mobile Environment[C] // Proceedings of the International Workshop on Spatio-Temporal Database Management (STDBM). 1999 : 119 – 134.

[72] CHEN Z, SHEN H T, ZHOU X, et al. Searching trajectories by locations: an efficiency study[C] // Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD). 2010 : 255 – 266.

[73] GEORGIOS S, DIMITRIOS S, ALEXANDRA V. Efficient Identification and Approximation of K-nearest Moving Neighbors[C] // Proceedings of the 21st ACM International Conference on Advances in Geographic Information Systems(SIGSPATIAL). 2013 : 264 – 273.

[74] ZHAO Z, SAALFELD A. Linear-time Sleeve-fitting Polyline Simplification Algorithms[J]. Proceedings of Autocarto, 1997.

[75] WANG S, BAO Z, CULPEPPER J S, et al. Answering Top-k Exemplar Trajectory Queries[C] // 33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017. 2017 : 597 – 608.

[76] SACHARIDIS D, SKOUTAS D, SKOUMAS G. Continuous monitoring of nearest trajectories[C] // Proceedings of the 22nd ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL). 2014 : 361 – 370.

[77] BAKALOV P, TSOTRAS V J. Continuous Spatiotemporal Trajectory Joins[C] // Proceedings of the Second International Conference on GeoSensor Networks (GSN). 2006 : 109 – 128.

[78] LIAN X, CHEN L, YU J X. Pattern Matching over Cloaked Time Series[C] // Proceedings of the 24th International Conference on Data Engineering (ICDE). 2008 : 1462 – 1464.

[79] YEH M, WU K, YU P S, et al. PROUD: a probabilistic approach to processing similarity queries over uncertain data streams[C] // Proceedings of the 12th International Conference on Extending Database Technology (EDBT). 2009 : 684 – 695.

[80] TRAJCEVSKI G, TAMASSIA R, DING H, et al. Continuous probabilistic nearest-neighbor queries for uncertain trajectories[C] // Proceedings of the 12th International Conference on Extending Database Technology (EDBT). 2009 : 874 – 885.

[81] CHUNYANG M, HUA L, LIDAN S, et al. KSQ: Top-k Similarity Query on Uncertain Trajectories[J]. IEEE Transaction on Knowledge and Data Engineering (TKDE), 2013, 25(9) : 2049 – 2062.

[82] GOCE T, ROBERTO T, F C I, et al. Ranking continuous nearest neighbors for uncertain trajectories[J]. VLDB Journal, 2011, 20(5) : 767 – 791.

[83] LI G, LI Y, SHU L, et al. CkNN Query Processing over Moving Objects with Uncertain Speeds in Road Networks[C] // Proceedings of the 13th Asia-Pacific Web Conference on Web Technologies and Applications(APWeb). 2011 : 65 – 76.

[84] LU J, GÜTING R H. Parallel Secondo: Boosting Database Engines with Hadoop[C] // Proceedings of the 18th IEEE International Conference on Parallel and Distributed Systems(ICPADS). 2012 : 738 – 743.

[85] MA Q, YANG B, QIAN W, et al. Query processing of massive trajectory data based on mapreduce[C] // Proceedings of the First International CIKM Workshop on Cloud DataManagement (CloudDb). 2009 : 9 – 16.

[86] YANG B, MA Q, QIAN W, et al. TRUSTER: TRajectory Data Processing on ClUSTERs[C] // Proceedings of the 14th International Conference Database Systems for Advanced Applications(DASFAA). 2009 : 768 – 771.

[87] ELDAWY A, MOKBEL M F. SpatialHadoop: A MapReduce framework for spatial data[C] // Proceedings of the 31st IEEE International Conference on Data Engineering (ICDE). 2015 : 1352 – 1363.

[88] ALY A M, MAHMOOD A R, HASSAN M S, et al. AQWA: Adaptive Query-Workload-Aware Partitioning of Big Spatial Data[J]. Procedings of the 41st International Conference on Very Large Data Bases(PVLDB), 2015, 8(13) : 2062 – 2073.

[89] DEAN J, GHEMAWAT S. MapReduce: Simplified Data Processing on Large Clusters[C] // Proceedings of the 6th Symposium on Operating System Design and Implementation (OSDI). 2004 : 137 – 150.

[90] DEAN J, GHEMAWAT S. MapReduce: simplified data processing on large clusters[J]. ACM Communication, 2008, 51(1) : 107 – 113.

[91] AJI A, WANG F, VO H, et al. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce[J]. Procedings of the 39th International Conference on Very Large Data Bases(PVLDB), 2013, 6(11) : 1009 – 1020.

[92] NISHIMURA S, DAS S, AGRAWAL D, et al. MD-HBase: design and implementation of an elastic data infrastructure for cloud-scale location services[J]. Distributed and Parallel Databases(DPD), 2013, 31(2) : 289 – 319.

[93] HUANG S, WANG B, ZHU J, et al. R-HBase: A Multi-dimensional Indexing Framework for Cloud Computing Environment[C] // Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM). 2014 : 569 – 574.

[94] HUGHES J N, ANNEX A, EICHELBERGER C N, et al. Geomesa: a distributed architecture for spatio-temporal fusion[C] // SPIE Defense+ Security. 2015 : 94730F – 94730F.

[95] XIE X, MEI B, CHEN J, et al. Elite: an elastic infrastructure for big spatiotemporal trajectories[J]. VLDB Journal, 2016, 25(4) : 473 – 493.

[96] YOU S, ZHANG J, GRUENWALD L. Large-scale spatial join query processing in Cloud[C] // Proceedings of the 31st IEEE International Conference on Data Engineering(ICDE) Workshops. 2015 : 34 – 41.

[97] YU J, WU J, SARWAT M. GeoSpark: a cluster computing framework for processing large-scale spatial data[C] // Proceedings of the 23rd International Conference on Advances in Geographic Information Systems (SIGSPATIAL). 2015 : 70:1 – 70:4.

[98] TANG M, YU Y, MALLUHI Q M, et al. Locationspark: a distributed in-memory data management system for big spatial data[J]. Procedings of the 42nd International Conference on Very Large Data Bases(PVLDB), 2016, 9(13) : 1565 – 1568.

[99] XIE D, LI F, YAO B, et al. Simba: Efficient In-Memory Spatial Analytics[C] // Proceedings of the 2016 ACM International Conference on Management of Data (SIGMOD). 2016 : 1071 – 1085.

[100] YUAN M, DENG K, ZENG J, et al. OceanST: a distributed analytic system for large-scale spatiotemporal mobile broadband data[J]. Procedings of the 40th International Conference on Very Large Data Bases(PVLDB), 2014, 7(1561-1564).

[101] CHRISTENSEN R, WANG L, LI F, et al. STORM: Spatio-Temporal Online Reasoning and Management of Large Spatio-Temporal Data[C] // Proceedings of the 2015 ACM International Conference on Management of Data (SIGMOD). 2015 : 1111 – 1116.

[102] GOWANLOCK M G, CASANOVA H. Distance threshold similarity searches on spatiotemporal trajectories using GPGPU[C] // Proceedings of the 21st International Conference on High Performance Computing(HiPC). 2014 : 1 – 10.

[103] ZHANG J, YOU S, LE G. U2STRA:high-performance data management of ubiquitous urban sensing trajectories on GPGPUs[C] // Proceedings of the 2012 ACM Workshop on City Data Management Workshop(CDMW). 2012 : 5 – 12.

[104] LEAL E, GRUENWALD L, ZHANG J, et al. TKSimGPU: A parallel top-K trajectory similarity query processing algorithm for GPGPUs[C] // Proceedings of the 2015 IEEE International Conference on Big Data (ICBD). 2015 : 461 – 469.

[105] LEAL E, GRUENWALD L, ZHANG J, et al. Towards an Efficient Top-K Trajectory Similarity Query Processing Algorithm for Big Trajectory Data on GPGPUs[C] // Proceedings of the 2016 IEEE International Conference on Big Data (ICBD). 2016 : 206 – 213.

[106] ZEINALIPOUR-YAZTI D, LIN S, GUNOPULOS D. Distributed spatio-temporal similarity search[C] // Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM). 2006 : 14 – 23.

[107] PAPADOPOULOS A, MANOLOPOULOS Y. Distributed Processing of Similarity Queries[J]. Distributed and Parallel Databases, 2001, 9(1) : 67 – 92.

[108] YEH M Y, WU K L, YU P S, et al. LeeWave: level-wise distribution of wavelet coefficients for processing kNN queries over distributed streams[J]. Procedings of the 34th International Conference on Very Large Data Bases(PVLDB), 2008, 1(1) : 586 – 597.

[109] KASHYAP S, KARRAS P. Scalable kNN search on vertically stored time series[C] // Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining SIGKDD. 2011 : 1334 – 1342.

附录　主要缩写符号对照表

ED	欧式距离（Euclidean Distance）
DTW	动态时间卷曲（Dynamic Time Warping）
LCSS	最长公共子串（Longest Common Subsequence）
EDR	实值序列上的编辑距离（Edit Distance on Real sequence）
HD	霍斯托夫距离（Hausdorff Distance）
FD	弗雷歇距离（Fréchet Distance）
OWD	一路距离 (One Way Distancen)
P2P	点对点 (Peer-To-Peer)
DP	道格拉斯-普克 (Douglas-Peuker)
SVD	奇异值分解 (Singular Value Decomposition)
DFT	离散傅里叶变换 (Discrete Fourier Transform)
HWT	哈尔小波变换 (Haaar Wavelet Transform)
PAA	分段聚合近似（Piecewise Aggregate Approximation）
APCA	自适应分段常量近似（Adaptive Piecewise Constant Approximation）
SED	欧式距离的平方 (Squared Euclidean Distance)

致　　谢

回首十一年的本、硕、博三阶段求学时光，经历了失落与荣耀。求学路上，目睹了一届届同学离开校园走上工作岗位的背影。此刻的我，即将博士毕业，感慨良多。在此期间，有幸得到老师、同学和亲友们的指导与帮助，在此谨对他们表示衷心的感谢！

首先，我要郑重感谢我人生中的三位引路人。他们分别是吉根林、周傲英和金澈清三位教授。吉根林教授是我在南京师范大学本、硕阶段的指导老师。他对待学生视若己出，为我们创造了良好的学习环境与和热烈活跃的团队氛围。并在我即将硕士毕业的十字路口，为我介绍了人生中第二位引路人周傲英教授。周老师是国内首屈一指的数据库领域专家，他在数据库领域的远瞩高瞻、对新兴学科和领域也能高屋建瓴。为此，我毅然来到他在华东师范大学的团队，拜师学艺。进入团队，我很快遇到了我的第三位引路人金澈清教授。金老师对待学术事无巨细且严谨有方，对待学生谆谆善诱。难以忘怀，在 2017 年国庆等节假日期间，金老师牺牲自己跟家庭团聚的时间帮我们修改论文。在周、金两位老师的指导下，我不仅学术水平突飞猛进发表若干学术论文，而且项目管理和实现能力也得到较大提高，成功带领实验室同学完成两项校企合作项目。除此之外，周、金两位老师在我母亲生病期间给我很大安慰和温暖，帮我度过了人生中最艰难的时刻。

其次，我十分感谢毛嘉莉老师，毛老师在我读博四年时光里，始终像知心大姐一样给我关心和鼓励。此外，在我几次投稿期间花费许多精力和时间与我讨论研究问题和研究方法，修改学术论文，让我学习了大量的学术论文写作技巧。然后，我还要感谢南京师范大学赵斌老师。感谢赵老师始兄长般的热情和帮助。也特别感谢华东师范大学钱卫宁教授、王晓玲教授、何晓丰教授、张蓉教授、高明副教授和张召副教授在日常学习中与研究生课程中给予的多方面帮助和指导。

另外，感谢所有读书期间陪伴我的同学和朋友，你们是我的美好记忆。感谢已经毕业的孔超、房俊华和王科强等师兄，以及马建松、康强强、段小艺、孔扬鑫、宋秋革、包婷、杨小林和廖春和等师弟师妹，感谢你们在我研究生前期给予的学习上的帮助和生活上的快乐，难忘一起写论文和一起做项目的点点滴滴。感谢王艺霖、杨小林、戚晓冬，你们为我的论文提出了宝贵的修改意见。特别感谢陪我度过

研究生时光的方祝和和肖冰，谢谢你们几年来对我的关心和包容。感谢一起毕业的毛嘉莉博士、郭敬伟博士、刘辉平博士和朱涛博士，与你们一起毕业是我的荣幸。也祝尚在奋斗的乔宝强、戚晓冬博士、申弋斌博士、方祝和等博士顺利完成学业，早日毕业。感谢在 LBS 课题组一起学习的陈鹤、王婧、李敏茜、施晋、华嘉逊、濮敏等师弟师妹们。此外，我还想感谢燕存、李文俊、丁敬恩、徐寅等朋友，谢谢你们这些年一直对我的关心和照顾。

最后，着重感谢我的家人，对我攻读博士学位的大力支持。

章志刚

二零一八年四月

攻读博士学位期间发表的学术论文、科研情况以及奖项

■ 已公开发表论文

[1] **Zhigang Zhang**, Xiaodong Qi, Yilin Wang, Cheqing Jin, Jiali Mao, Aoyin Zhou. Distributed Top-k Similarity Query on Big Trajectory Streams. *Frontiers of Computer Science* (FCS), to be published, 2018.

[2] **Zhigang Zhang**, YilinWang, Jiali Mao, Cheqing Jin, Aoying Zhou.DT-KST: Distributed Top-k Similarity Query on Big Trajectory Streams. In Proceedings of *the 22th Database Systems for Advanced Applications* (DASFAA), pages 199-214, 2017.

[3] **Zhigang Zhang**, Cheqing Jin, Jiali Mao, Xiaolin Yang, Aoying Zhou.TrajSpark: A Scalable and Efficient in-memory Management System for Big Trajectory Data. In Proceedings of *the Web and Big Data - First International Joint Conference,Part I* (APWeb-WAIM), pages 11-26, 2017.

[4] **章志刚**, 金澈清, 王晓玲, 周傲英. 面向海量低质手机轨迹数据的重要位置发现. *软件学报*, 27 卷, 7 期, 1700-1714, 2016.

[5] Xiaolin Yang, **Zhigang Zhang**, Yilin Wang, Cheqing Jin. Distributed Continuous KNN Query over Moving Objects. *Information Scineces* , to be published, 2017.

[6] 毛嘉莉, 金澈清, **章志刚**, 周傲英. 轨迹大数据异常检测: 研究进展及系统框架. *软件学报*, 28 卷, 1 期, 17-34, 2017.

[7] 王艺霖, **章志刚**, 金澈清. 智能交通卡记录中的公交站点恢复方法. *华东师范大学学报(自然科学版)*, 5 卷, 201-212, 2017.

[8] Cheqing Jin, Ashwin Lall, Jun(Jim) Xu,**Zhigang Zhang**, Aoying Zhou. Distributed Error Estimation of Functional Dependency. *Information Scineces* , volume 345, pages 11-26, 2017.

[9] Jiali Mao, Qiuge Song, Cheqing Jin, **Zhigang Zhang**, Aoying Zhou.TSCluWin: Trajectory Stream Clustering over Sliding Window. In Proceedings of *the 21th Database Systems for Advanced Applications* (DASFAA), pages 133-148, 2016.

[10] 包婷, **章志刚**, 金澈清. 基于手机大数据的城市人口流动分析系统. *华东师范大学学报(自然科学版)*, 5 卷, 162-171, 2015.

■ **参加的科研项目**

[1] 面向智慧城市的大规模数据计算理论和关键技术（国家自然科学基金面上项目，编号：U1501252）

[2] 数据质量管理中的完整性约束关键技术研究（国家自然科学基金面上项目，编号：61370101）

[3] 基于超云平台的社会化移动网络大数据管理与分析关键技术研究（国家自然科学基金面上项目，编号：U1401256）

[4] 基于空间包含关系的面向对象聚类方法研究（国家自然科学基金面上项目，编号：40871176）

[5] 移动环境下以商户为中心的客户定向机制（国家自然科学基金面上项目，编号：61402180）