历史轨迹数据的学习型索引实现

汇报：B21150201王文珍

研究背景和意义

研究背景和意义

随着车载传感网、智能设备和移动计算技术的不断发展，轨迹数据的规模呈现爆炸式增长。这些数据包含了移动对象丰富的时空信息，成为智能交通、智慧城市等应用的重要信息来源。为了支持高效的轨迹查询和分析，构建合理的时空数据索引结构至关重要。

研究背景和意义

轨迹数据查询

历史查询： 可以挖掘出移动对象的行为模式、热点区域的分布特点以及长期趋势

提供上下文支持

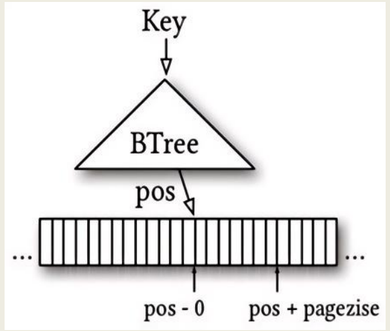
奠定基础

当前查询

未来查询

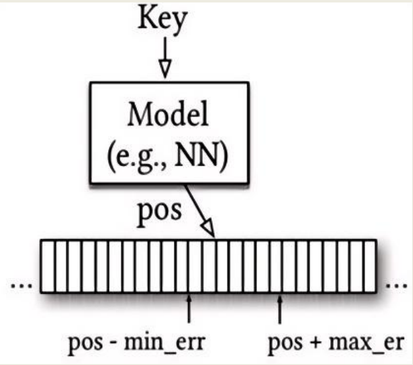
构建索引的面临的挑战：针对轨迹数据的复杂性和分布不均衡问题，在保证查询效率的同时压缩索引体积是一个核心问题

传统索引



随着数据规模爆发性增长，传统索引处理海量数据时面临随着数据量增大结构变大的问题，这使得查询时待搜索的节点变多而需要较多查询时间。

学习型索引



将索引问题转化为函数拟合问题，通过机器学习模型对数据分布进行建模，以预测记录的位置或区间。不仅能大幅减少索引大小，还能在数据分布良好建模的场景下实现近似常数时间查询，极大地提升了轨迹数据管理效率。

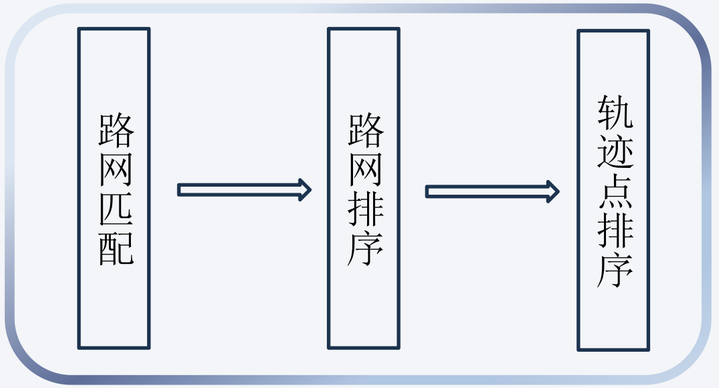
研究方法

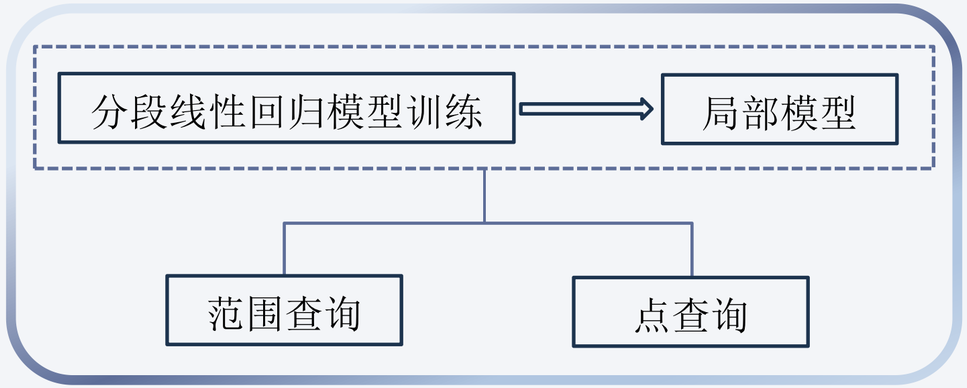
研究方法

索引构建总框架

01 轨迹数据排序

对初始轨迹数据进行路网匹配，采用深度优先搜索对路段进行拓扑有序编号，并结合路段内相对距离计算轨迹点在路网中的位置，实现三维轨迹向二维映射。第二步结合Hilbert曲线与勒贝格测度设计映射函数，将二维轨迹点映射为一维有序序列。





02 模型建立

通过分段线性回归模型预测轨迹点在排序数组中的位置，并用分片预测函数确定其所属分片；随后对分片进行磁盘分配，并建立局部模型存储每个分片的页面分割值和页面编号，以精确预测轨迹点的存储页面。

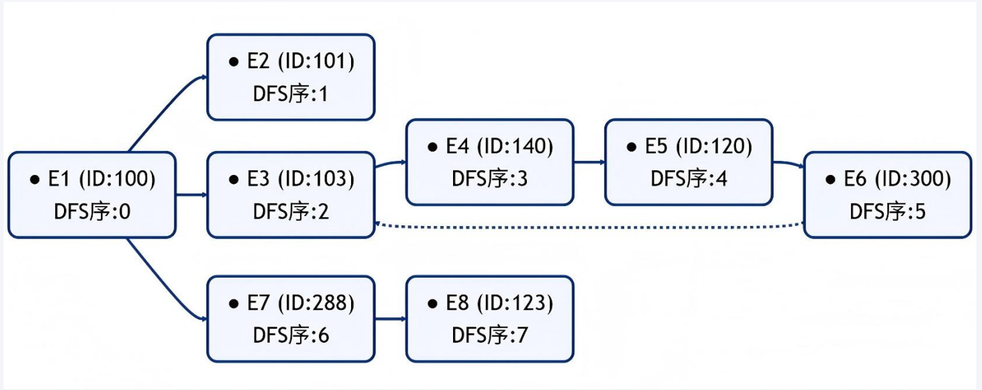
研究方法

轨迹数据排序

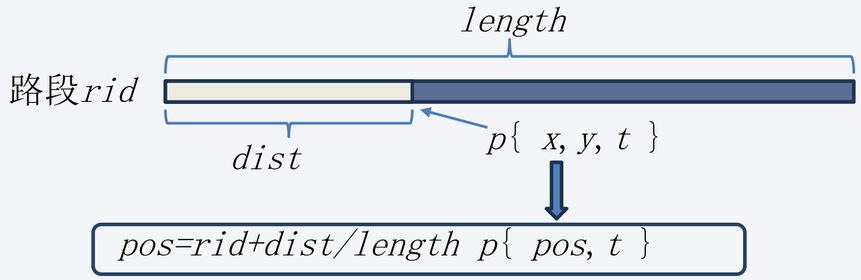
路网匹配

轨迹点p表示为$p=\{ x, y, t \}$，x、y分别表示经度和纬度，t表示时间戳。本文采用隐马尔可夫模型将轨迹数据匹配到路网，路网匹配过程中会计算路段的长度，以及计算轨迹点到所在路段起点的距离。

路网排序



采用DFS的路网遍历策略，由一个起点出发，深度遍历路网图，按照遍历顺序对路段进行排序，并根据排序顺序为每个路段分配新的编号rid。

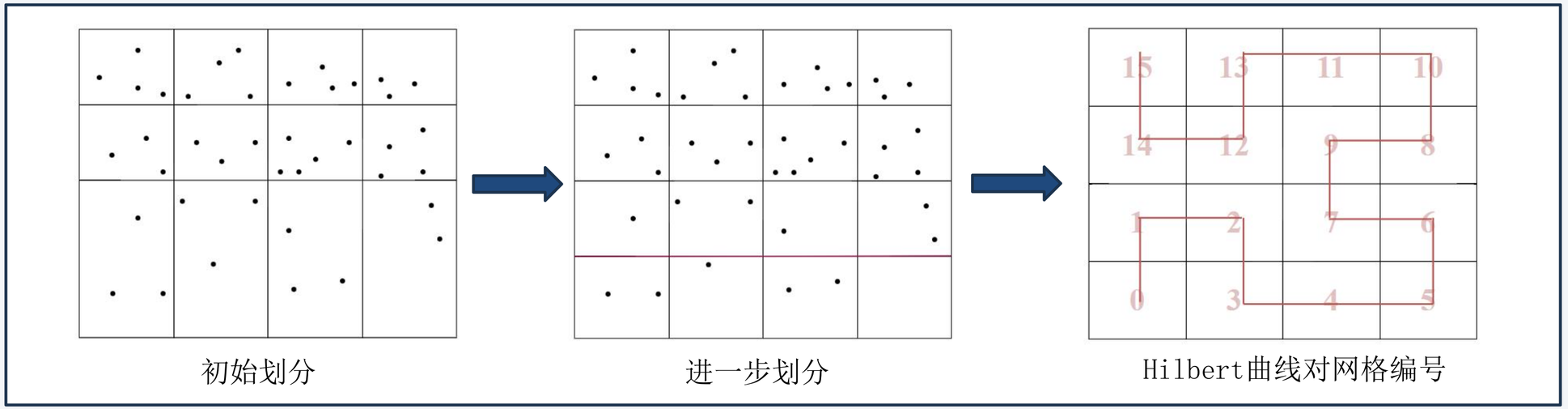


研究方法

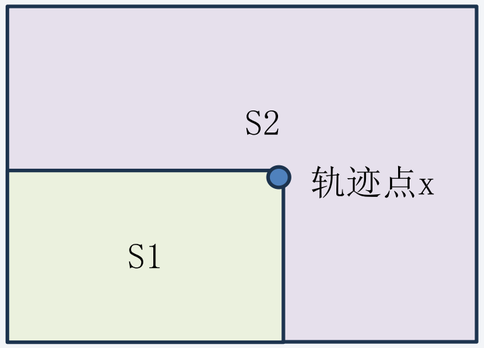
轨迹数据排序

映射函数设计

网格编号



映射函数



引入勒贝格测度的思想构建映射函数
Ci为轨迹点x所在网格的编号

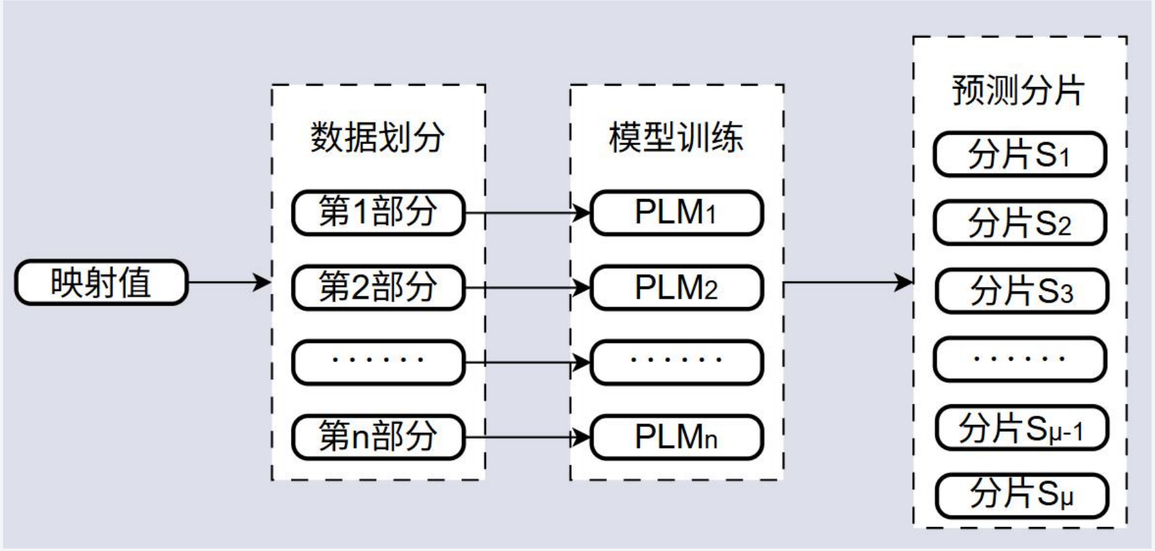
映射函数M(x)=Ci+S1/S2

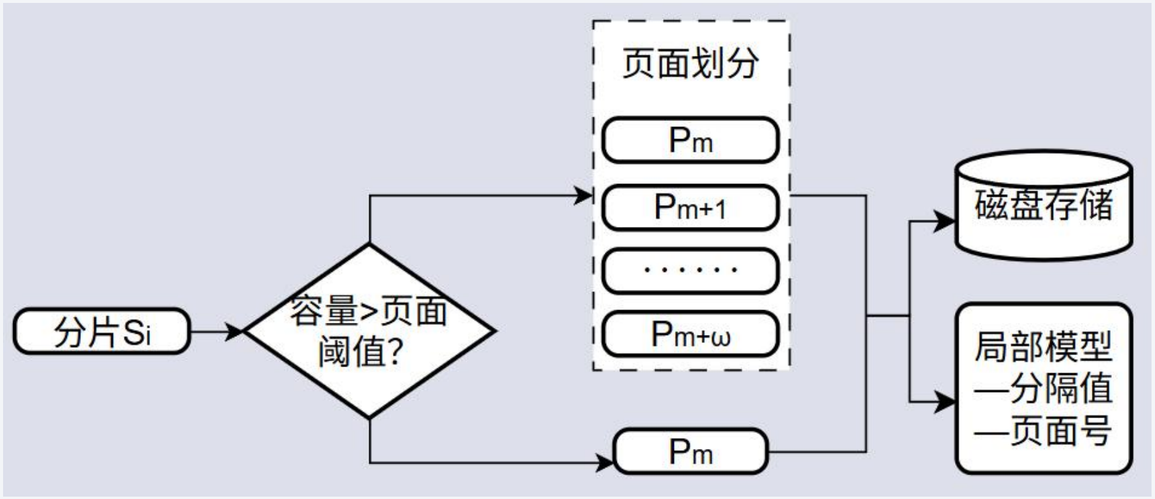
研究方法

01 分片预测

02 局部模型

模型建立

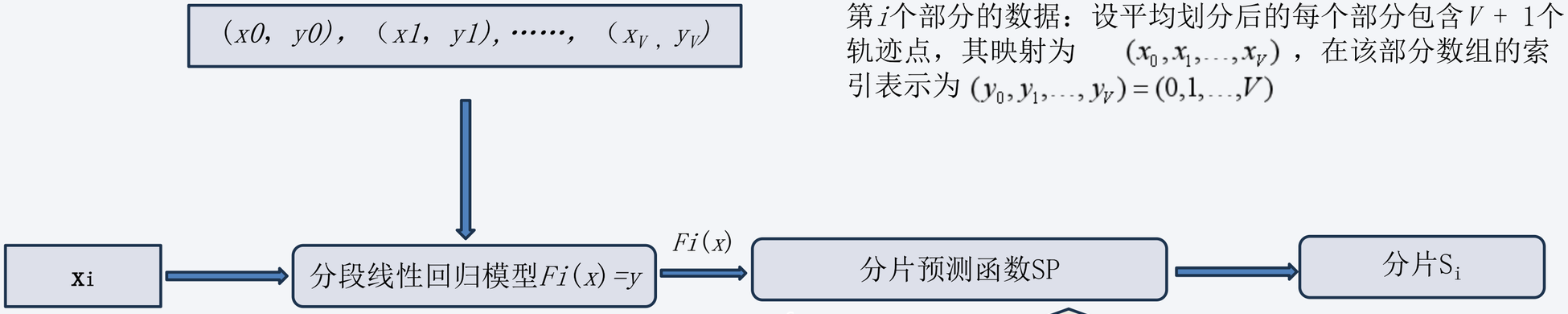




研究方法

模型建立

分片预测



第i个部分的数据：设平均划分后的每个部分包含$V+1$个轨迹点，其映射为 $(x_0,x_1,\ldots,x_V)$，在该部分数组的索引表示为 $(y_0,y_1,\ldots,y_V)=(0,1,\ldots,V)$

$$SP(x)=F_i(x)/\psi+i\times D,i=0,1,2\cdots n-1$$

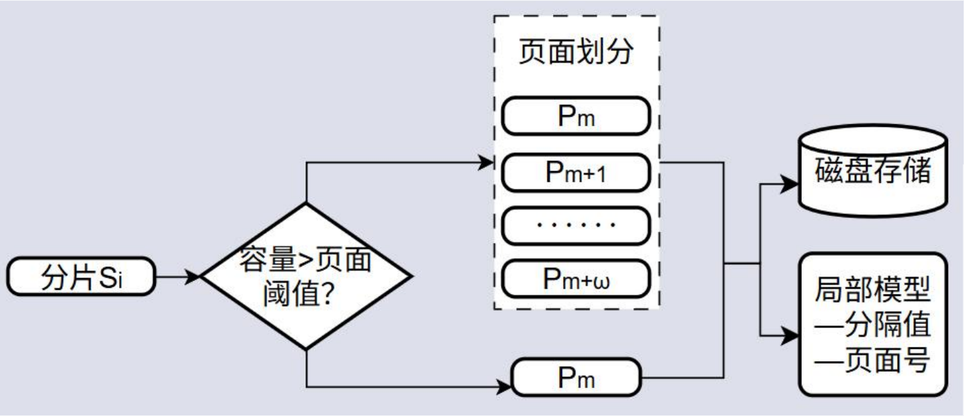
ψ是分片中估计键的平均数量，该值设定和磁盘页面可以存储的键值数量一致，D为每个分段线性回归模型可以生成的分片数

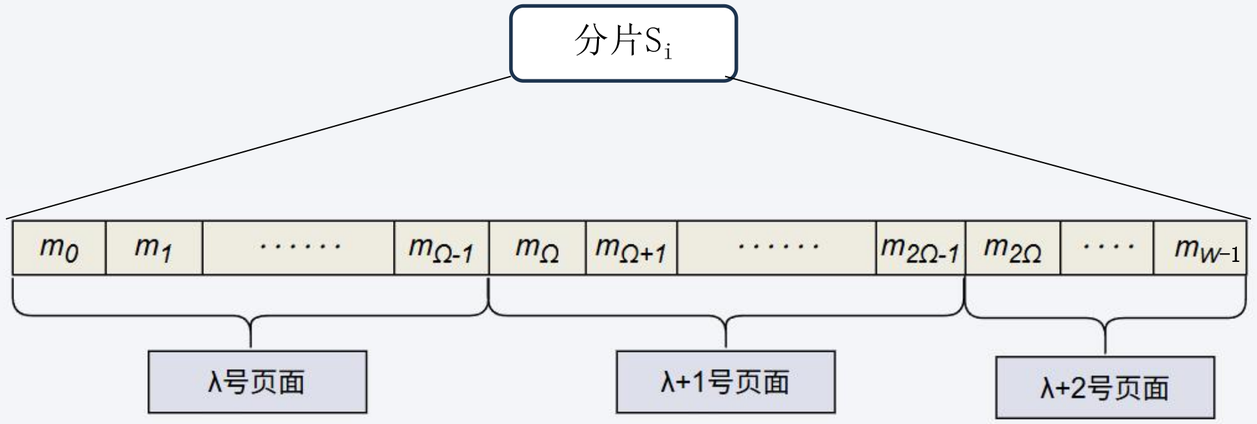
$$D=\left\lceil\frac{V+1}{\psi}\right\rceil$$

研究方法

模型建立

磁盘页面分配和局部模型构建





局部模型并不是一个机器学习模型，只是一个简单的轻量级的辅助数据结构，该数据结构用于存储每个页面数据所在的全局页面号和分割值。局部模型LM 数据结构为$LM=[PA,PM]$ 。PA 用于存储磁盘页面号，PM 用于存储页面分割值。假设每个磁盘页面容量 Ω 个轨迹点。

$$W>\Omega \quad \lceil W/\Omega \rceil = 3$$
$$PA=[\lambda,\lambda+1,\lambda+2]$$
$$PM=[m_\Omega, m_{2\Omega}]$$

$$W \le \Omega$$
$$PA=[\lambda], PM=[\ ]$$

研究内容

查询设计

01 点查询

02 范围查询

利用希尔伯特曲线的连续性，遍历所有边界网格单元，计算其编码，并根据相邻编码是否落在查询范围内，确定各个查询区间的起止位置。

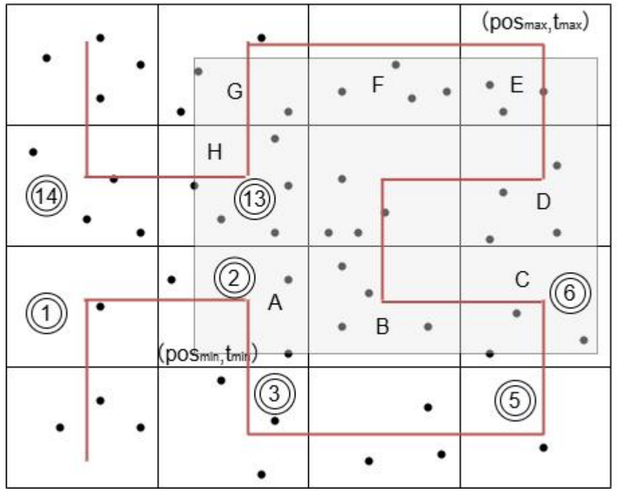
遍历边界的8个网格{*A, B, C, D, E, F, G, H* }

*A*前后相邻编码的网格都不在查询范围内，所以起点集合*begin*=[2]，终点集*end*=[2]。

网格*C*的前相邻网格不在查询范围内，则将其编码加入起点集合，*begin*=[2,6], *end*=[2]；

网格*H*的后相邻网格不在查询范围内，将其编码加入终点集合，*begin*=[2,6]， *end*=[2,13]

最后得到的查询希尔伯特区间为range[2,2]，rang2=[6,13]

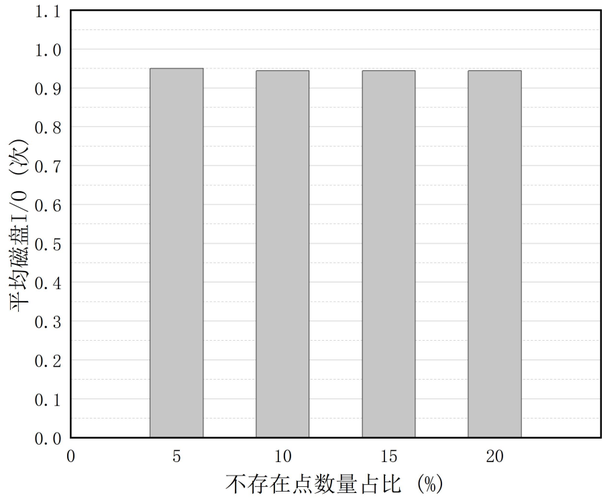


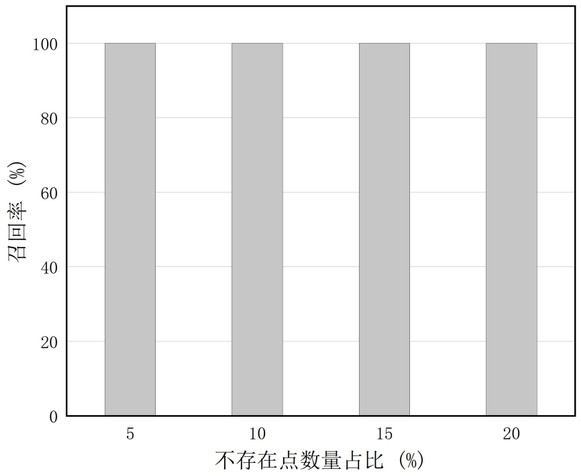
实验结果

实验结果

实验使用了GEOLIFE轨迹数据集和来自电子地图的北京四环内路网数据。经过路网匹配和路网排序，共获得3511335个包含路网位置与时间戳的轨迹点。在实验中，网格分割值、模型参数与局部模型在查询前均加载至内存。所构建的索引占用内存为2.69MB，得益于采用参数较少、计算简单的分段线性回归模型，显著降低了存储开销。

点查询：设计四类1000个点查询，每类点查询中分别有5%，10%，15%，20%在轨迹数据中不能查找到对应的轨迹点样本。



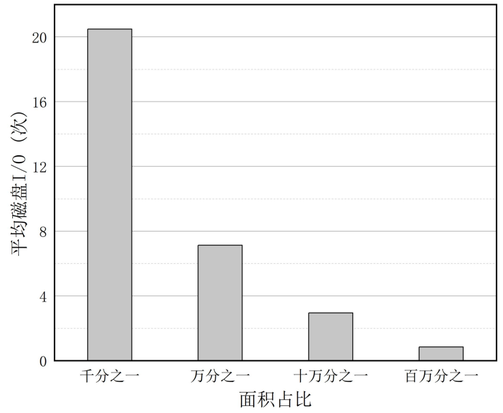


由结果可知，平均点查询所需的磁盘IO不超过一次，原因是本文的索引模型是先预测分片之后再放入磁盘中，并结合局部模型精确定位页面地址，使得每次查询最多只会产生一次磁盘IO。

范围查询：范围查询根据覆盖面积占二维空间总面积的比例分成四类：占总面积的千分之一、万分之一、十万分之一和百万分之一，两个方向的长度不超过所在方向长度的1/4。每类生成100个矩形框查询，共400个查询。

平均磁盘I/O	召回率
14.595	100%

召回率100%的原因是进行范围查询时，对于边界没有完全覆盖的网格是当作完全覆盖的网格进行索引，最后再过滤掉不满足该范围的点，避免了产生漏查的情况。



为观察模型范围查询时不同空间尺度（千分之一、万分之一、十万分之一和百万分之一）下的响应趋势，进一步将查询按面积比例分组并计算平均磁盘I/O。每类分别有400个矩形框查询。

总结与展望

总结与展望

总结

首先采用图结构与DFS遍历方法对道路进行排序，然后通过Hilbert曲线和勒贝格测度设计了映射函数将轨迹点映射到一维空间，最后实现了学习型索引。

展望

（1）当前的双重降维策略虽提高了构建效率，但可能在一定程度上损失了原始数据的空间分布信息，未来可引入更强的空间保持映射方法以减少信息损失。

（2）可探索对现有分段线性回归模型的优化，增强其对数据分布的学习与拟合能力，同时控制存储开销。

（3）针对轨迹数据的时序性与动态性，未来可结合增量学习或局部重训练策略，提升索引结构的在线更新与自适应能力。

THANK YOU FOR WATCHING

谢谢