轨迹数据处理应用

安祺

2019.5.9

内容概要

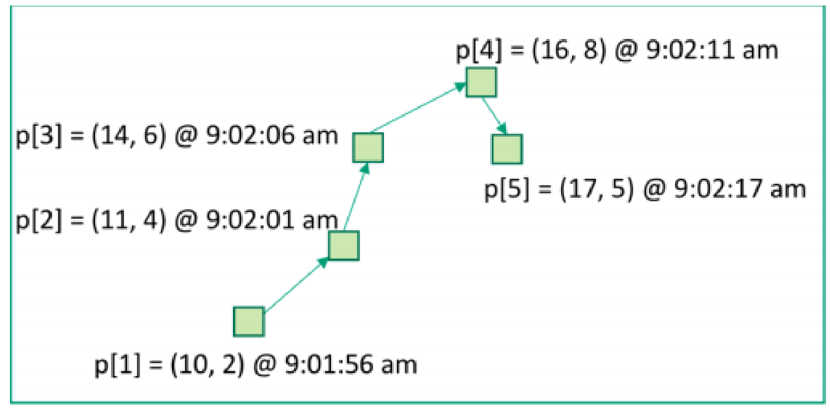
- 什么是轨迹数据
- 对轨迹数据的应用
  - 利用轨迹数据进行违章停车检测

    Detecting Illegal Vehicle Parking Events using Sharing Bikes' Trajectories (KDD 2018)
  - 利用轨迹数据进行自行车道路规划

    Planning Bike Lanes based on Sharing-Bikes’ Trajectories (KDD 2017)

什么是轨迹

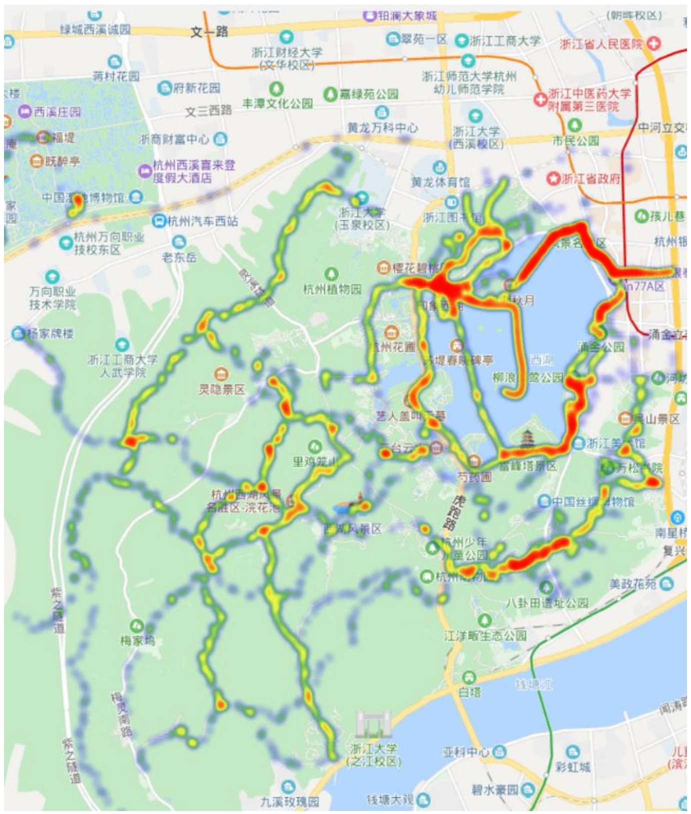
一个坐标



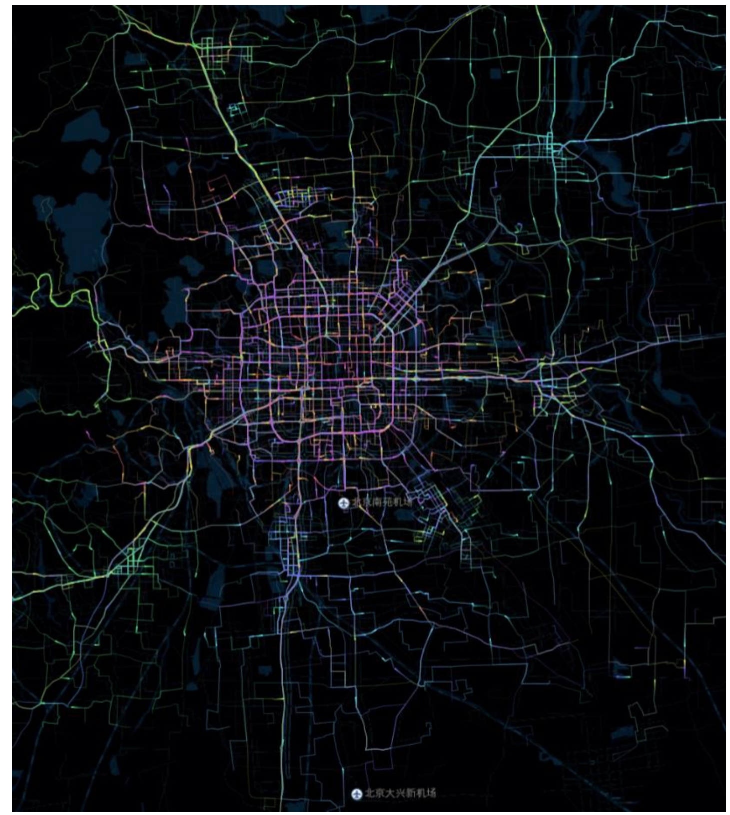
一条轨迹

轨迹是一个依时间顺序排序的坐标序列

轨迹中包含的信息



杭州旅游热路线热力图



北京市公交线路图

来源:www.echartsjs.com

轨迹数据

```
460040084802501,116.362144,39.838600,2016-01-04 23:58:16,2,57
460040084802501,116.362144,39.838600,2016-01-04 23:58:27,2,57
460040084802501,116.362144,39.838600,2016-01-04 23:58:37,2,57
460040084802501,116.362144,39.838600,2016-01-04 23:58:47,2,57
460040084802501,116.362094,39.838600,2016-01-04 23:58:58,2,57
460040084802501,116.362094,39.838600,2016-01-04 23:59:08,2,57
460040084802501,116.362094,39.838600,2016-01-04 23:59:19,2,57
460040089004206,116.391536,39.906288,2030-08-21 08:01:57,2,50
460040089004206,116.391536,39.906288,2030-08-21 08:02:08,2,50
460040089004206,116.391536,39.906338,2030-08-21 08:02:18,2,50
460040089004206,116.391536,39.906388,2030-08-21 08:02:29,2,50
460040089004206,116.391536,39.906388,2030-08-21 08:02:39,2,50
460040089004206,116.391536,39.906438,2030-08-21 08:02:50,2,50
460040089004206,116.391536,39.906488,2030-08-21 08:03:00,2,50
460040160309675,116.596792,39.839828,2016-01-04 23:58:18,2,57
460040160309675,116.596792,39.839828,2016-01-04 23:58:29,2,57
460040160309675,116.596792,39.839828,2016-01-04 23:58:39,2,57
460040160309675,116.596792,39.839828,2016-01-04 23:58:50,2,57
460040160309675,116.596792,39.839828,2016-01-04 23:59:00,2,57
460040160309675,116.596792,39.839828,2016-01-04 23:59:11,2,57
460040160309675,116.596792,39.839828,2016-01-04 23:59:21,2,57
460040160005570,116.065592,39.644192,2016-01-04 23:58:19,1,23
460040160005570,116.065592,39.644192,2016-01-04 23:58:30,1,23
460040160005570,116.065592,39.644192,2016-01-04 23:58:41,1,23
```

轨迹ID, 经度, 纬度, 时间, 方向

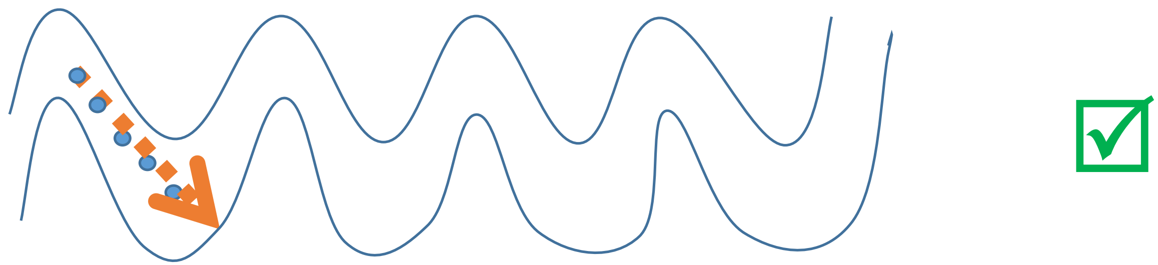

```
[
    [1164383,401471,-11,-13,1,-49,-26,-14,99,-170,4,-36,
    87,-2,16,-141,-2,-15,-47,-6,-168,-9,-2,22,-74,-4,
    -138,10,12,-152,9,-55,-17,-111,13,-176,-20,-38,1,
    -57,31,-54,28,-85,-5,-126,-13,-62,1,-34,-84,1,-3,
    -218,15,6,78,2,4,-52,70,1,7,-142,99,2,21,-5,229,
    -150,16,-23,0,-180,168,1,-4,-257,82,0,33,-22,78,20],

    [1164311,399594,8,1,33,0,56,1,29,3,30,27,8,-5,21,
    -14,3,-2,1,-1,-4,-3,-13,-13,-21,-19,-9,-9,-16,-9,
    -93,-6,8,50,-100,-1,-1,0,-171,-3,-1,0,0,-1,0,-8,0,
    -15,1,-8,0,-28,0,-4,1,-11,0,-19,0,-12,2,-22,1,-26,1,
    -25,0,-25,1,-24,0,-4,2,-25,0,-1,0,-7,1,-28,2,-25,17,
    -30,0,-1,0,-2,2,-55,-44,-1,0,-32,1,0,48,3,11,0,12,
    -34,0,-7,-24,-9,0,-12,1,-12,0,-1,7,0,8,0,8,0,41,1,1,
    -10,0,-4,1,-5,0,-6,0,-3,2,-21,1,-12,0,-8,0,-2,2,0,
    31,-6,6,0,51,2,0,1,-70,0,7,13,7,5,30,22,1,1],

    [1163521,398776,16,1,55,2,47,1,0,1,0,0,0,4,0,8,0,1,
    -1,7,0,11,-1,16,0,4,0,15,-1,45,0,1,-1,16,-1,54,0,29,
    0,1,0,2,0,1,0,2,0,4,0,37,-1,7,0,7,0,18,107,0,2,0,30,
    0,129,2,56,2,36,1,22,1,53,1,47,2,7,0,8,0,8,0,41,1,2,
    1,17,0,4,0,10,0,33,1,7,0,18,0,1,-32,0,-2,37,0,1,0],

    [1163823,399822,54,2,1,-26,2,-29,1,-1,47,0,2,0,12,
    -5,1,-25,1,-10,0,-8,0,-1,1,-26,1,-11,1,-20,1,-31,1,
    0,72,2,57,-13,0,-18,1,-5,0,-6,0,-1,0,-8,0,-15,1,-8,
    0,-28,0,-4,1,-11,0,-19,0,-12,2,-22,1,-26,1,-25,0,
    -25,1,-24,0,-4,2,-25,0,-1,1,0,13,0,56,1,14,1,65,1,3,
    0,19,0,52,0,23,-1,3,-3,60,4,86,1,8,0,12,0,41,0,46,0,
```


起点坐标, {位移序列}

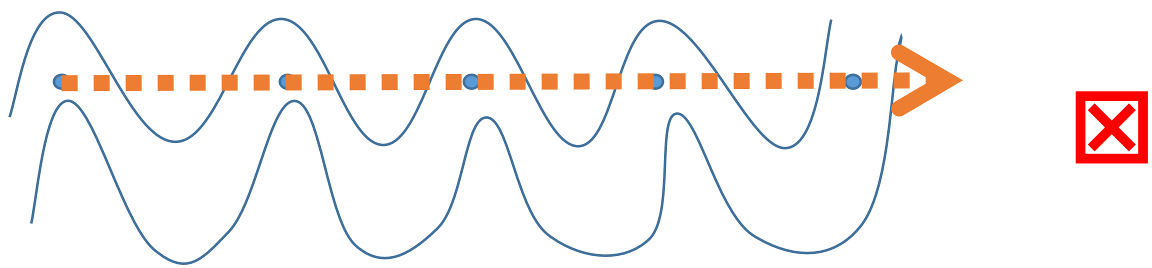
轨迹数据处理可能面临的问题

- 数据误差（设备，网络等带来的坐标精度、时间上的误差）
- 采样率问题
  - 如：当前车速为50km/h，得到了5个坐标

每隔2s记录一次位置

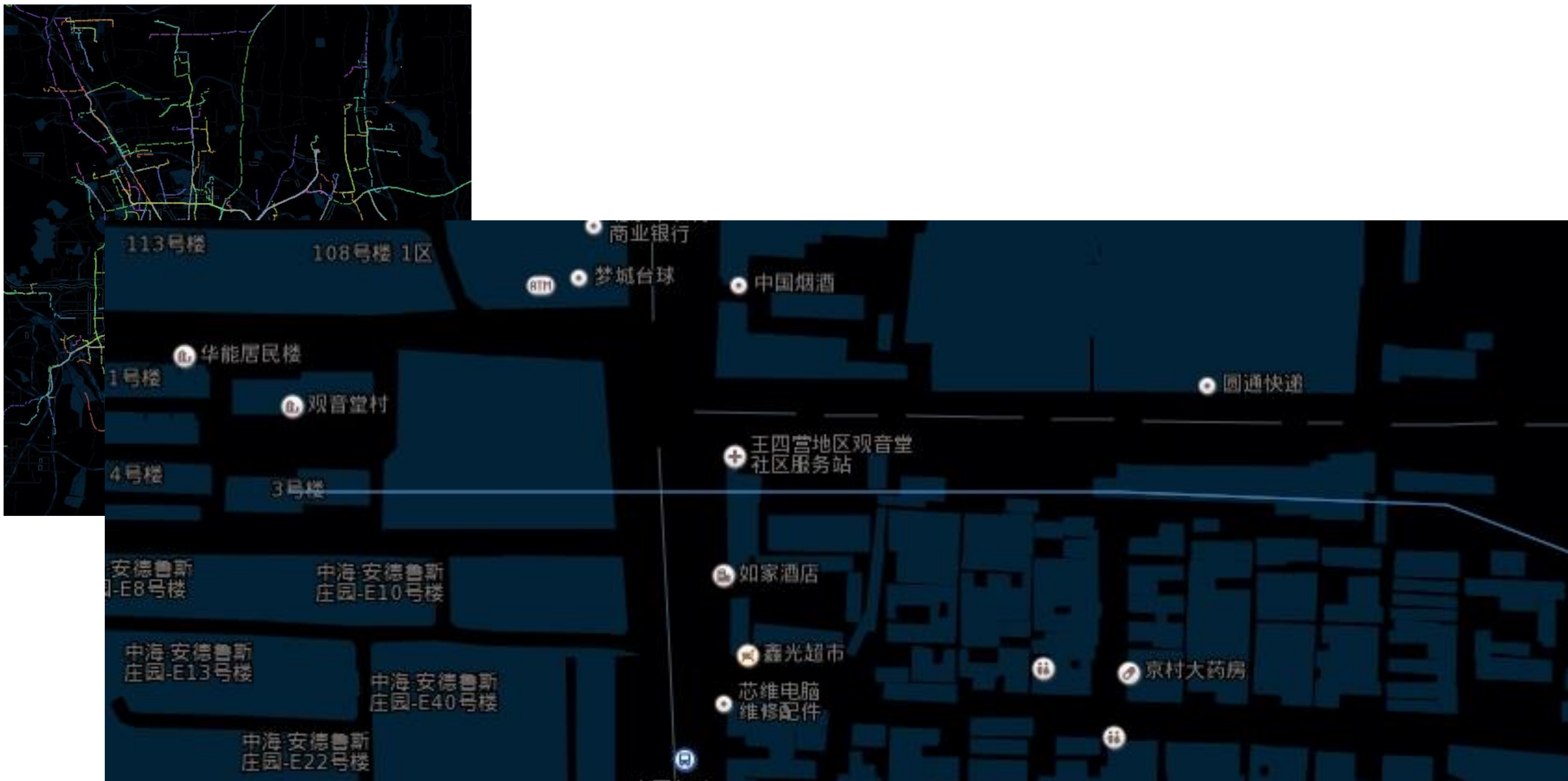


每隔2min记录一次位置



轨迹数据处理可能面临的问题

- 地图匹配



Detecting Illegal Vehicle Parking Events using Sharing Bikes' Trajectories

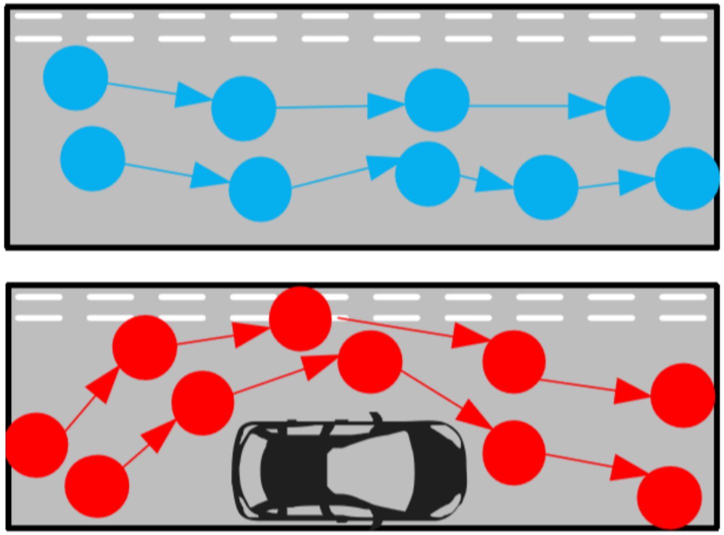
KDD 2018

问题背景

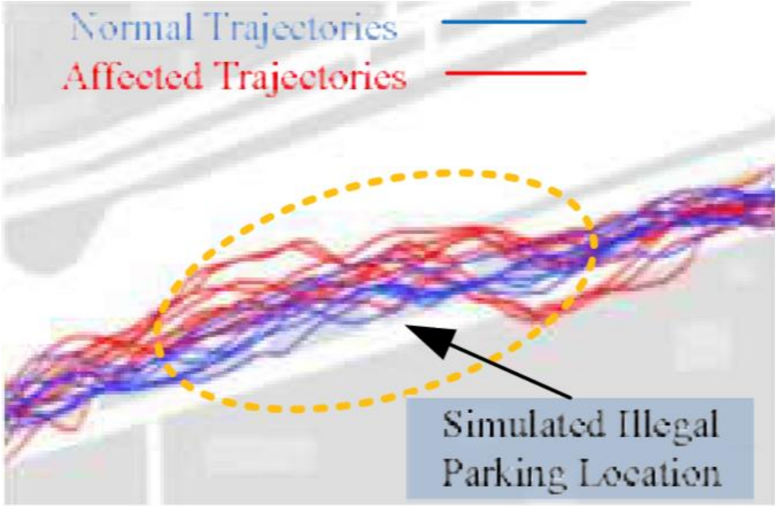
- **违章停车**
  - 引起交通拥堵
  - 引发交通事故
  - 尾气排放更多，污染环境

- **传统方法**
  - 无法有效覆盖整个城市
  - 依靠人力监查，人力有限
  - 摄像头监控，成本高

想法产生



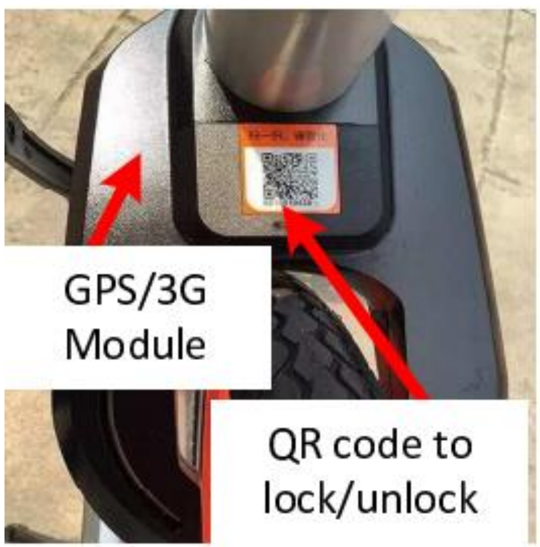

Simulated Illegal
Parking Location


Normal Trajectories
Affected Trajectories
Simulated Illegal
Parking Location

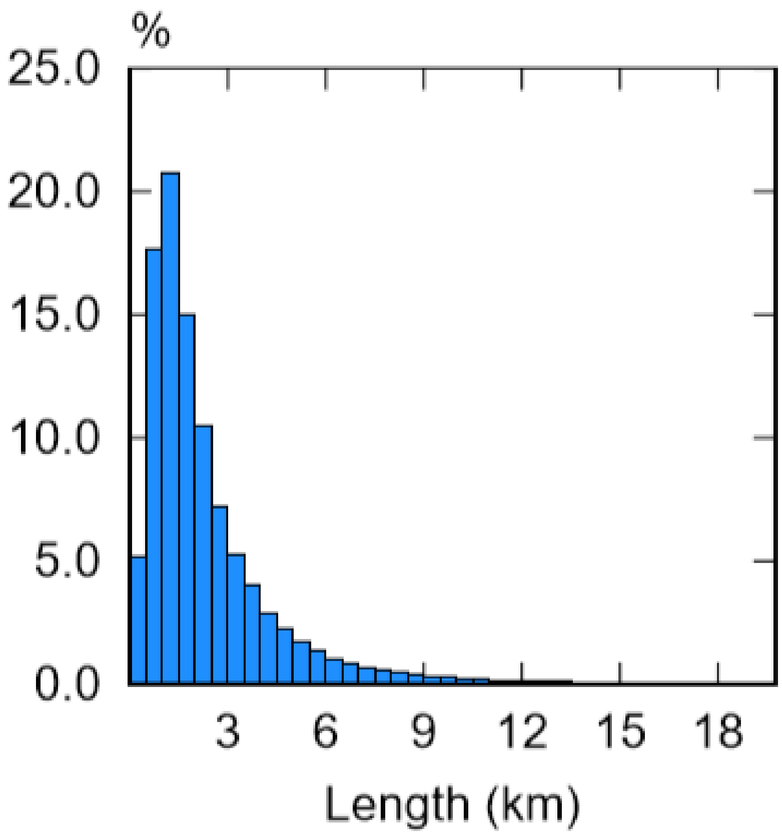

Beijing

数据依托

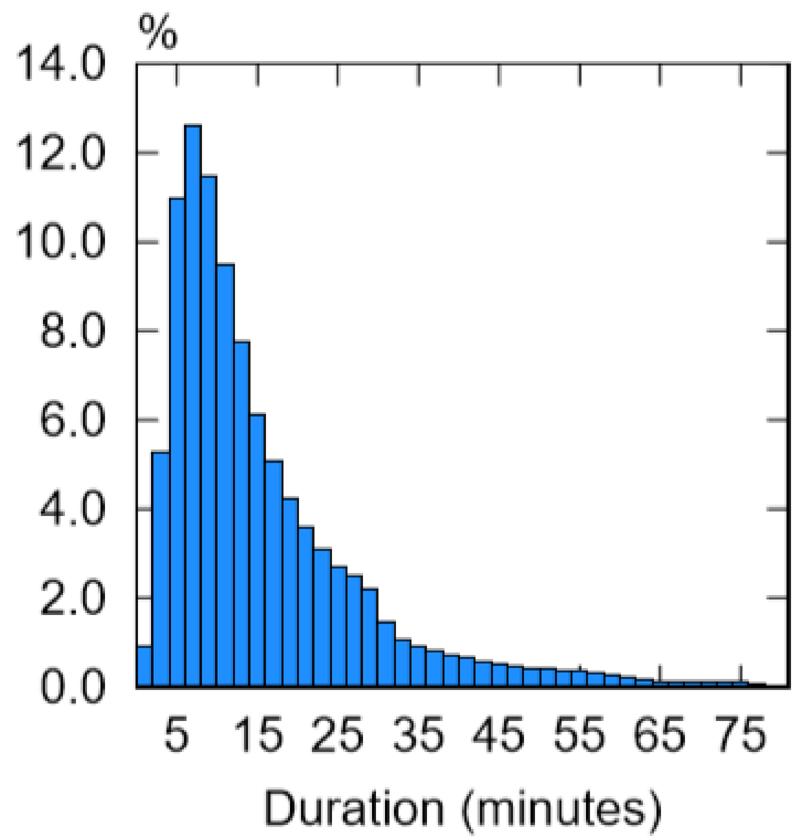
- Mobike的轨迹数据
  - 无站式共享单车，反映出自行用户的真实轨迹
  - 找车方便，轨迹记录准确
  - 覆盖全城



数据总览

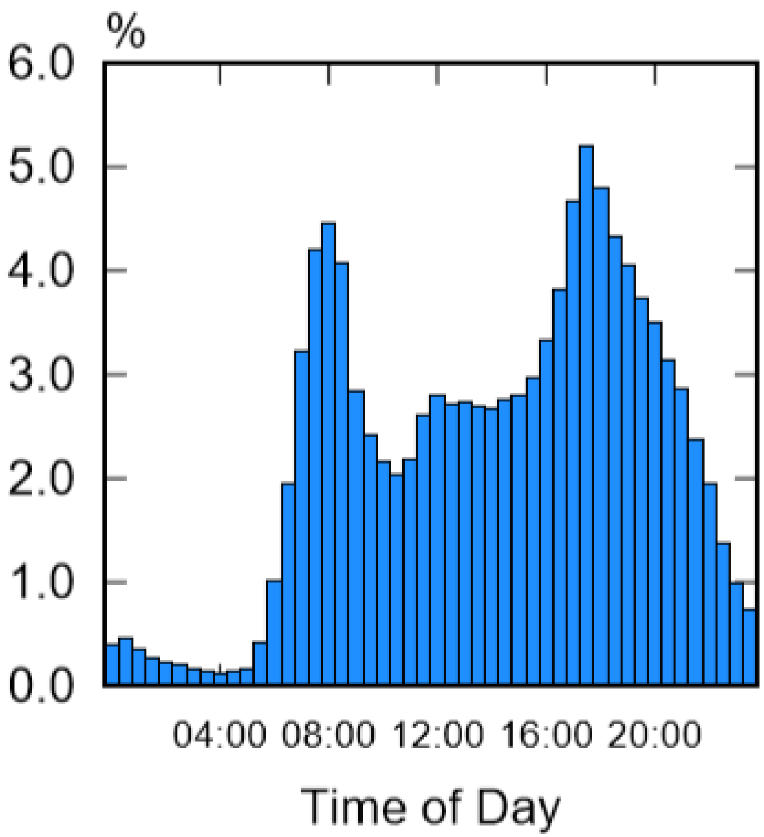


骑行轨迹长度

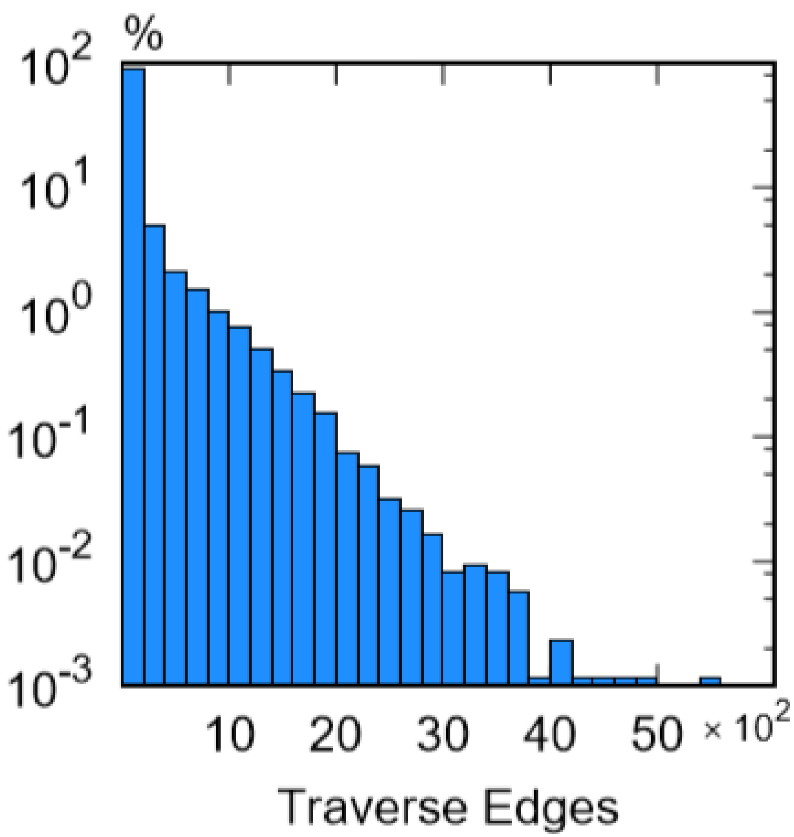


骑行轨迹时间

数据总览



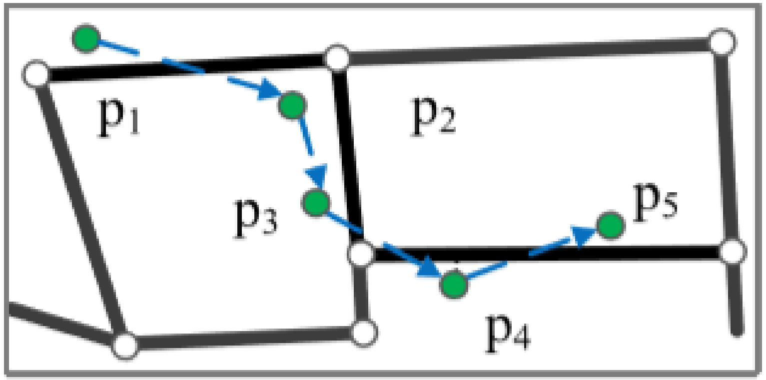
骑行发生时段



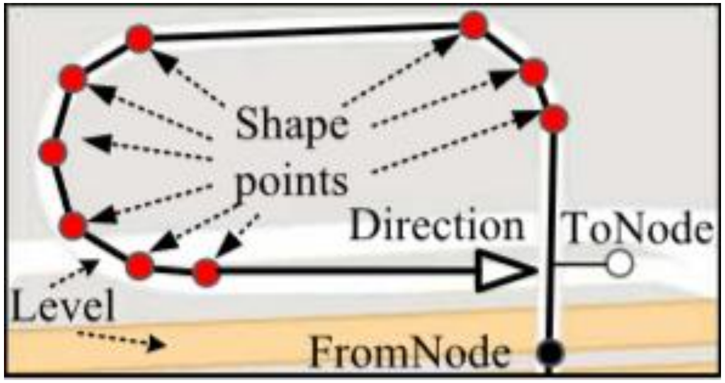
骑行经历路段数量

问题建模

- 将城市的道路网以图的形式表示 G=(V,E)
  - V：道路交汇路口集合，E={e} 道路路段集合
  - ei属性: (1) 等级; (2) 形状；(3) 方向



道路网



路段等级和形状

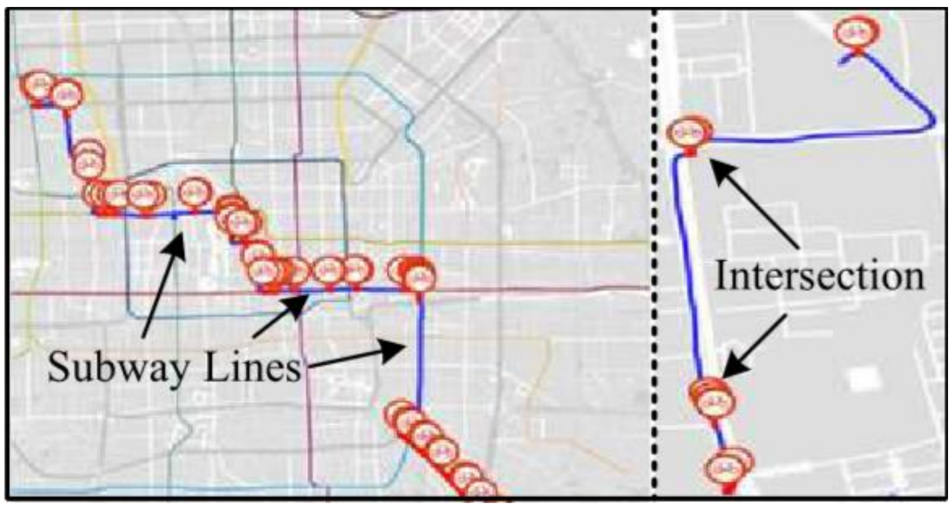
违停检测是指在时段ti到ti+1内，ei上妨碍正常轨迹的障碍的检测

系统结构

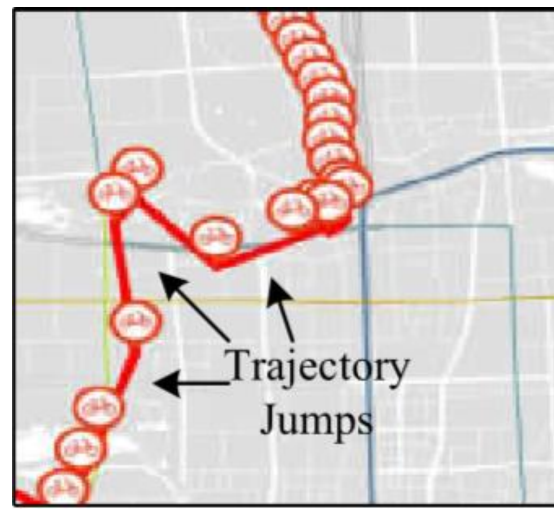
- 预处理阶段
  - 数据清洗，清除掉异常的GPS点
  - 地图匹配，轨迹中的点进行划分，映射到路段中
  - 建立索引，依照路段的id为轨迹建立索引，加速轨迹检索过程
- 违停检测阶段
  - 为每个道路建立基准路线
  - 轨迹特征提取
  - 基于分布统计的检测方法

数据清洗

- 速度异常(速度过快或过慢)
- 低采样率(智能手机中GPS模型的异常导致地采样)



行驶速度异常



低采样率

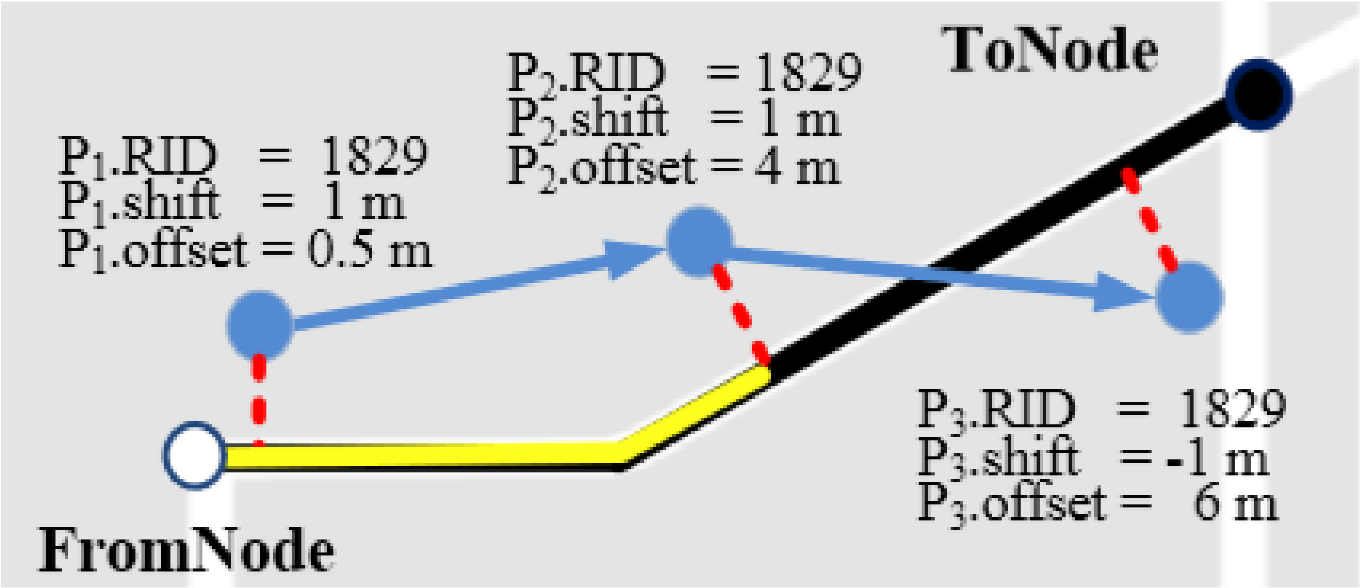
清洗方法：设置能够反映速度和采样率的阈值，低于阈值的连续点序被删除

地图匹配

(1)行驶速度更低 (2)行驶方向灵活 (3)不受道路限制 (4)行驶路程更短

方法1：调整匹配方法

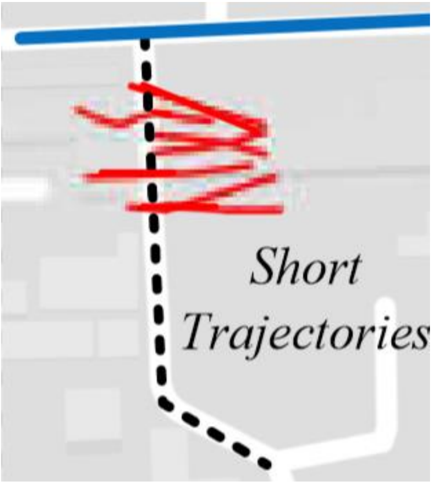
- 删除高等级路段，所有道路都视为双向道路，不使用速度限制
- 每个GPS点的新属性：RID，shift，offset

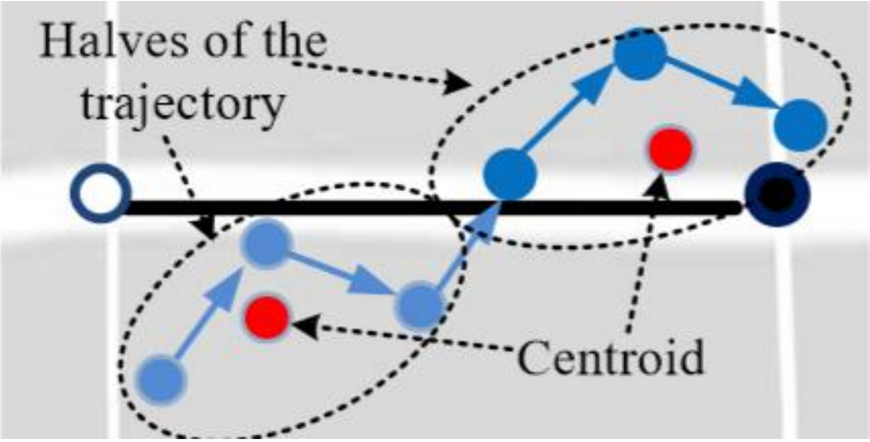


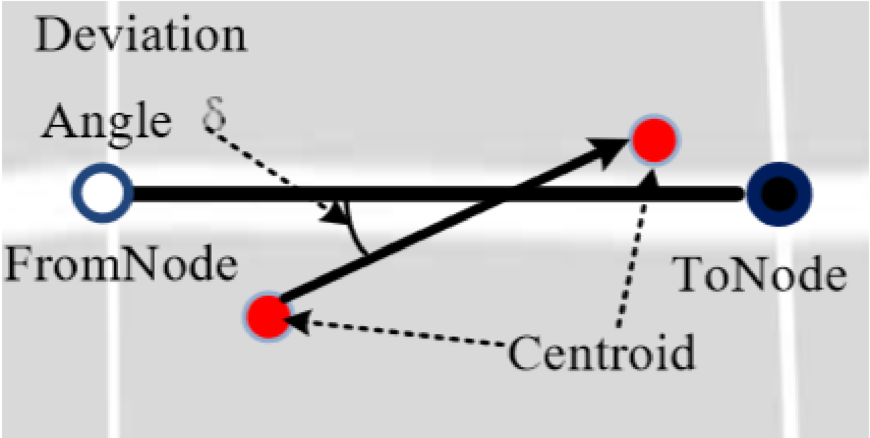
地图匹配

方法2:基于图形的精细化

- 删除距离错误的轨迹
- 删除方向错误的轨迹





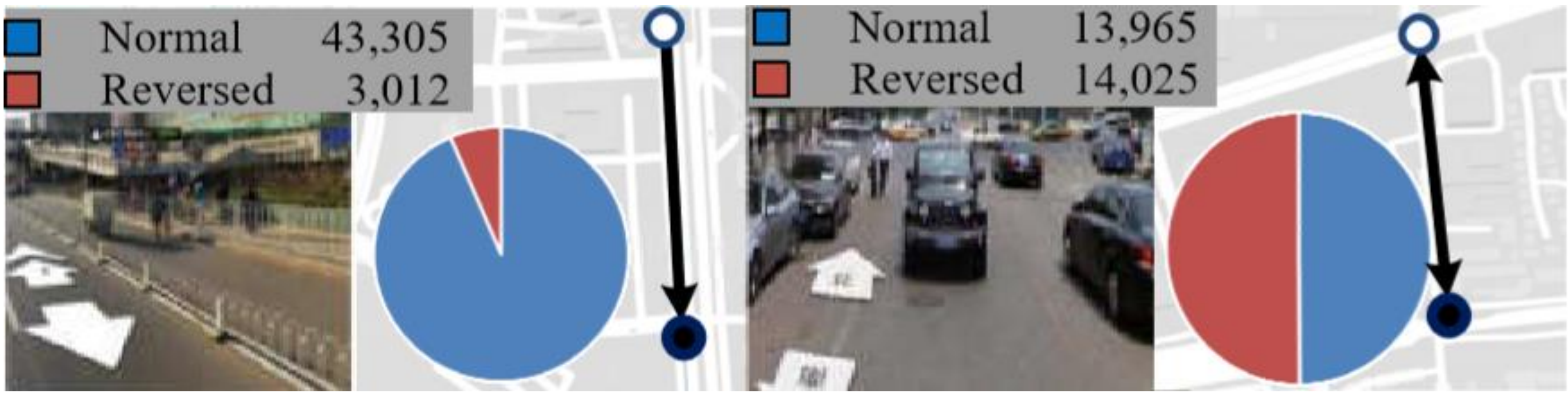


删除平均shift高于阈值的轨迹点

删除与道路夹角大于阈值角度的点

地图匹配

方法3:删除逆行轨迹

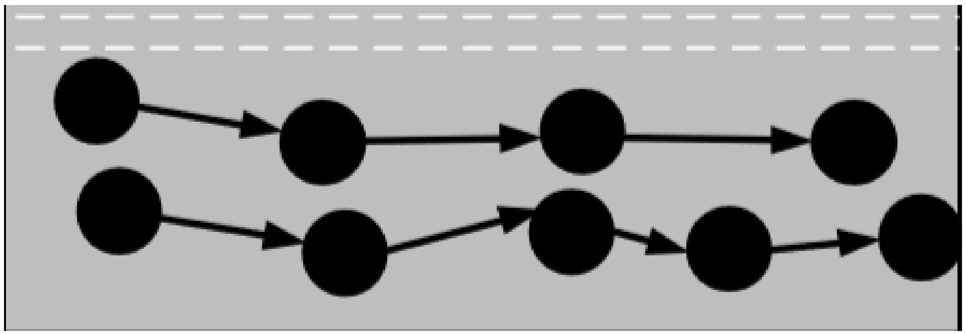


违停检测

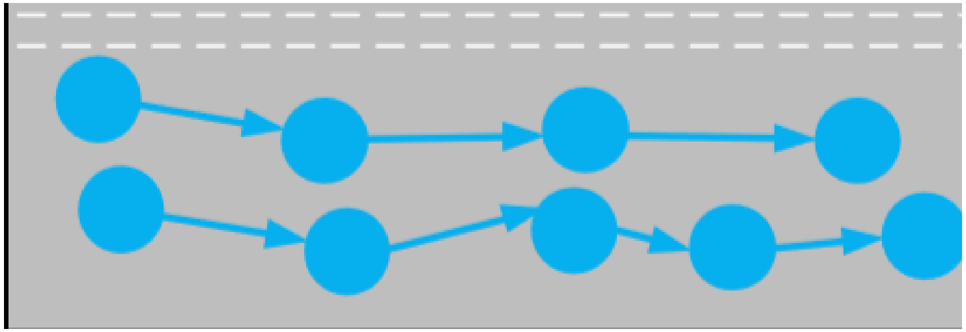
- 数据皆无标记 → - 标记正常情况
- 违停情况复杂 → - 正常轨迹的模式稳定而且简单
- 骑行行为多样 → - 限定时段，从时段中的所有轨迹中提取特征
- GPS经度有限 → - 通过建立基准轨迹来描述每个路段上的正常轨迹的特征

聚合轨迹的与道路边界的偏移分布作为特征来检测违停现象

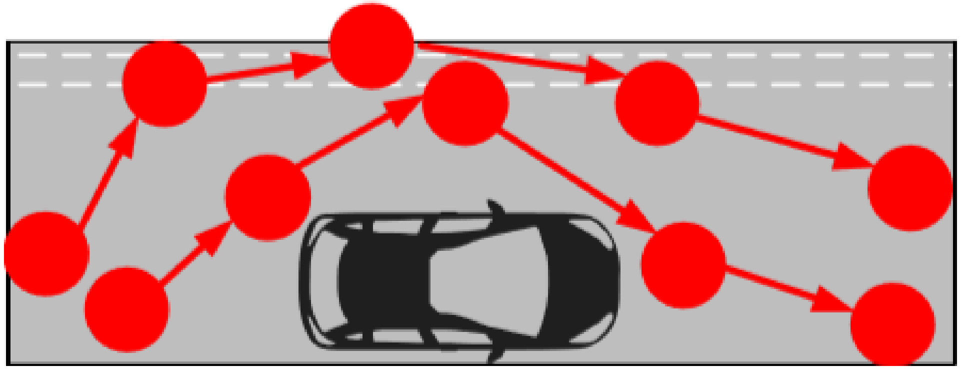
主要思想



基准轨迹



正常轨迹



存在违停

基准轨迹建模

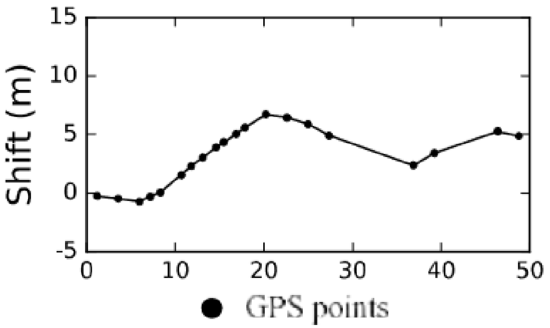
- 简单建模
  - 直接使用道路形状作为基准轨迹
- 利用夜间轨迹建模
  - 利用11:00 pm – 7:00am 时段的轨迹作为基准轨迹
  - 利用这个时段中，6个月的历史轨迹数据来进行基准轨迹建模

基准特征提取

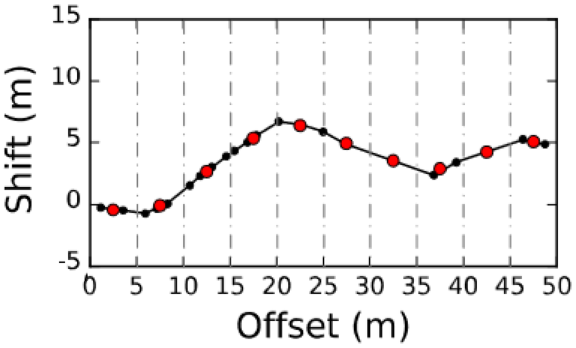
- 保证用于处理的轨迹采样公平
  - 每条轨迹进行进一步划分（50m一段）
  - 每条轨迹的GPS点重新采样（每5m采样一个点）

- 提取平均偏移
- Offset 5m范围内所有点的平均偏移

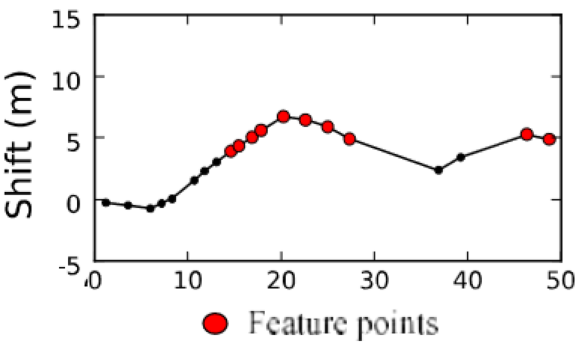
- 提取最大偏移
  - 每条轨迹的最大偏移，每50m道路中的top-10偏移点作为最终特征



原始轨迹



平均偏移

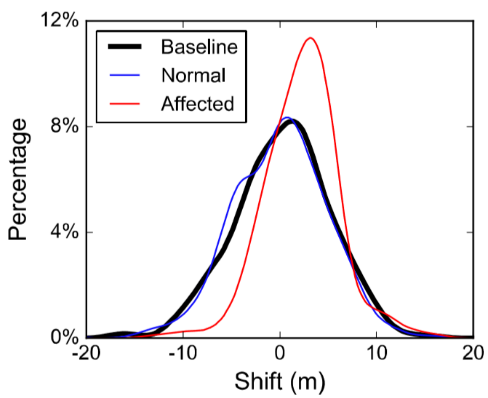


最大偏移

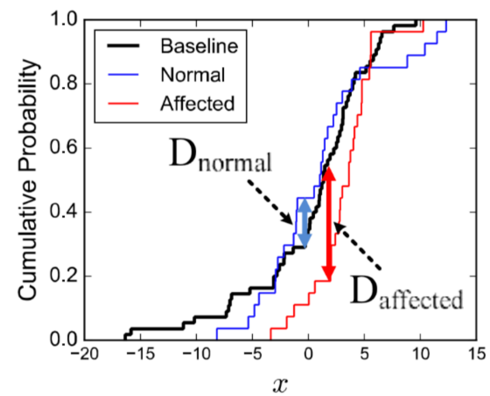
违停检测

如果两条轨迹的特征足够相似，那么它们服从同一分布

- KS测试统计计算 $$D_{n,m} = \sup_x |F_{1,n}(x) - F_{2,m}(x)|$$



偏移分布



KS检测

- 阈值选择 $$D_{n,m} > c(\alpha)\sqrt{\frac{n+m}{nm}} \ \& \ c(\alpha) = \sqrt{-\frac{1}{2}ln(\frac{\alpha}{2})}$$

实验数据

- 北京道路网络
  - 来自Bing Map，377,559个路口，501,462路段
- 摩拜轨迹
  - {bike ID, UID, temporal Range, { start, end },{ GPS Points}}
  - 08/01/2017 - 02/08/2018
- 用于验证的真实标记
  - 海淀区和朝阳区的违停记录
  - {road ID, Time Stamp, Photo, Label}
  - 32条道路，454真实标记，其中159有违停现象
  - 海淀区 12/26/2017-12/30/2017
  - 朝阳区 01/12/2018-02/09/2018

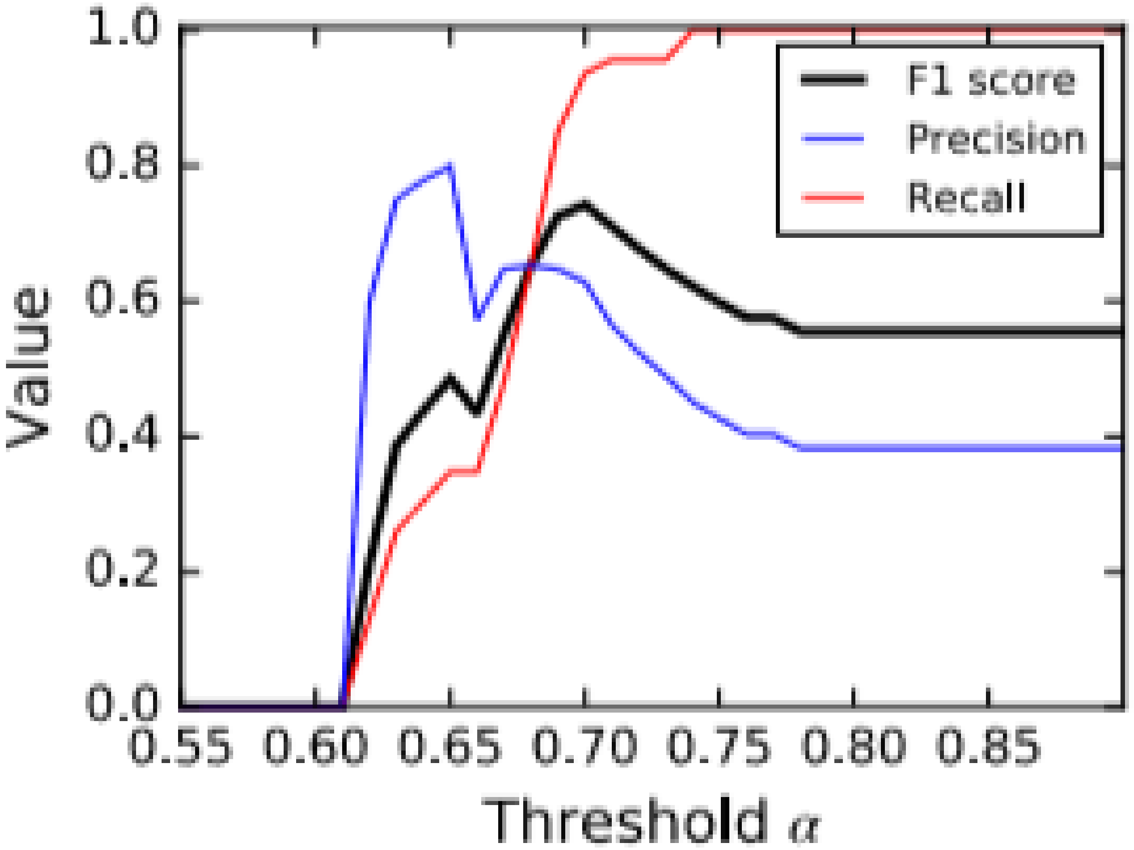
实验结果

- 阈值选择

$P = N_{TP} / (N_{TP} + N_{FP})$

$R = N_{TP} / (N_{TP} + N_{FN})$

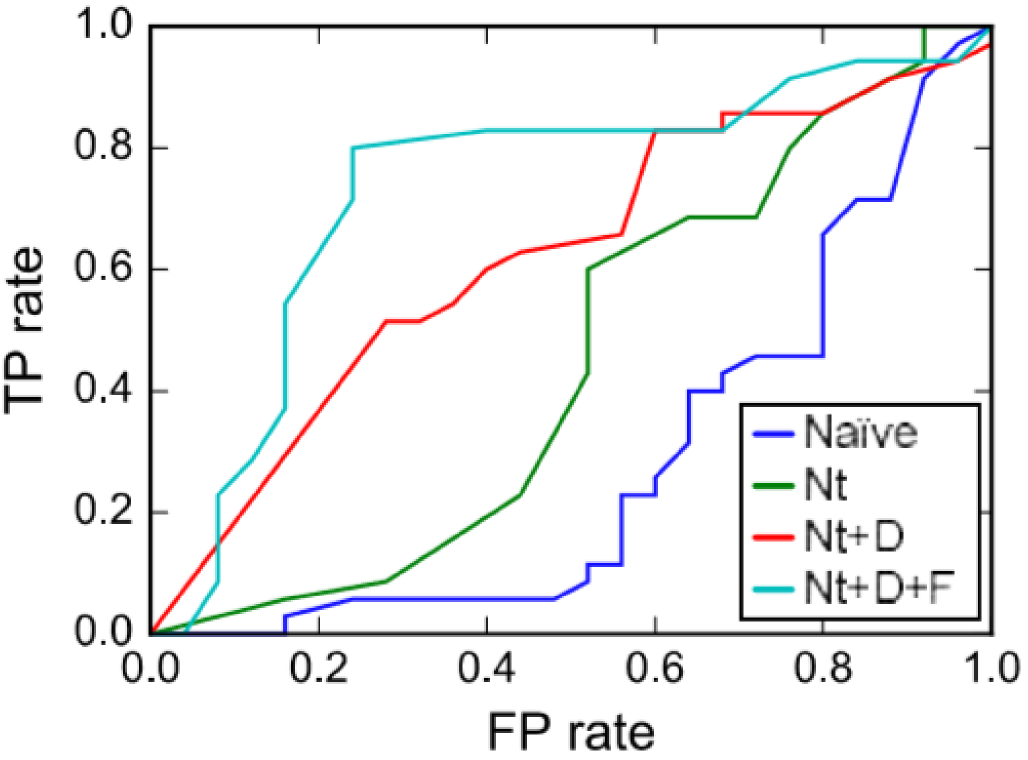
$F_1 = 2PR / (P + R)$



实验结果

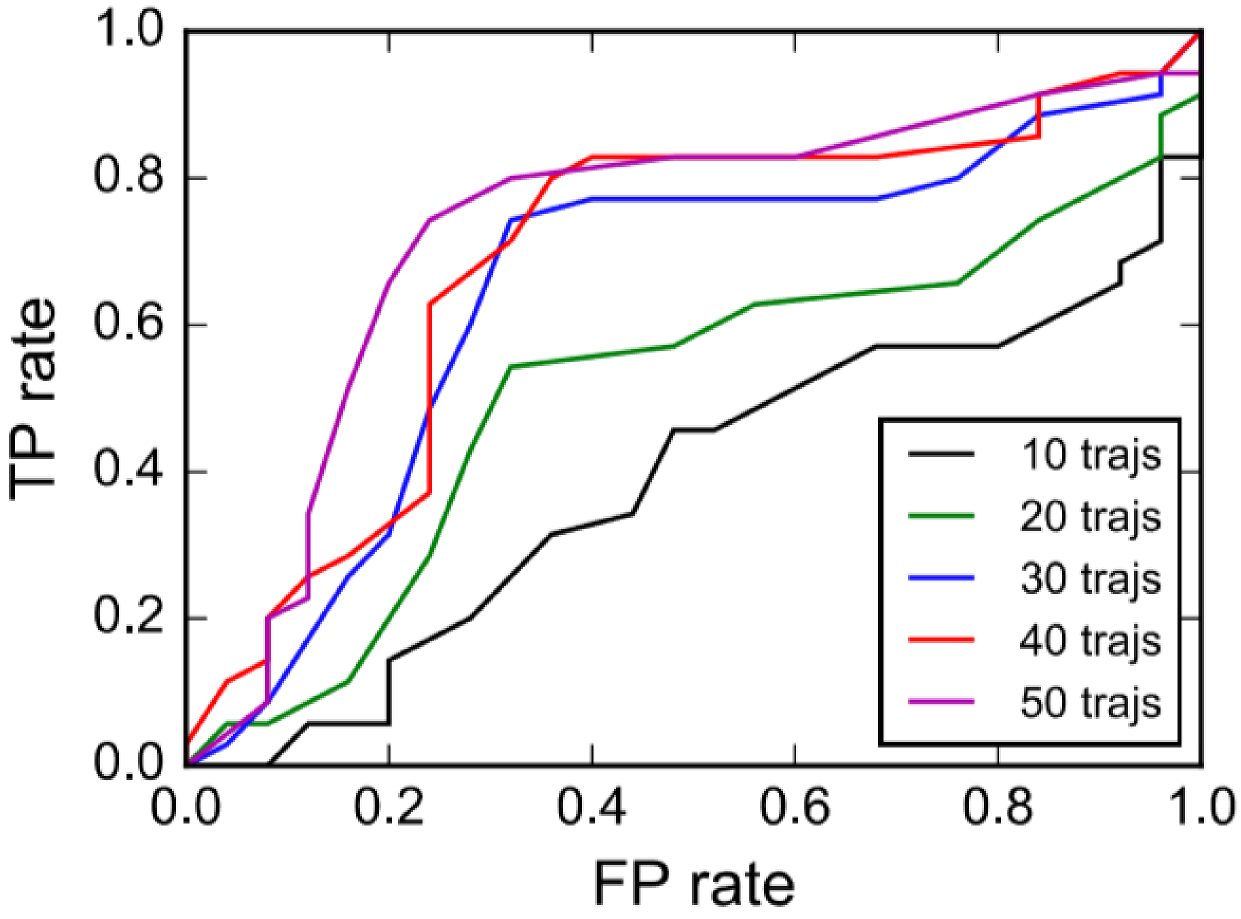
- 基准轨迹构建方式选择

- *Naïve:* 直接使用道路形状方式
- *Nt:* 夜间轨迹聚类方式
- *Nt + Dir:* 在Nt的基础上，提出了逆行轨迹，并且用平均偏移作为轨迹特征
- *Nt + Dir + T:* 与上一方式基准轨迹相同，用最大偏移作为特征



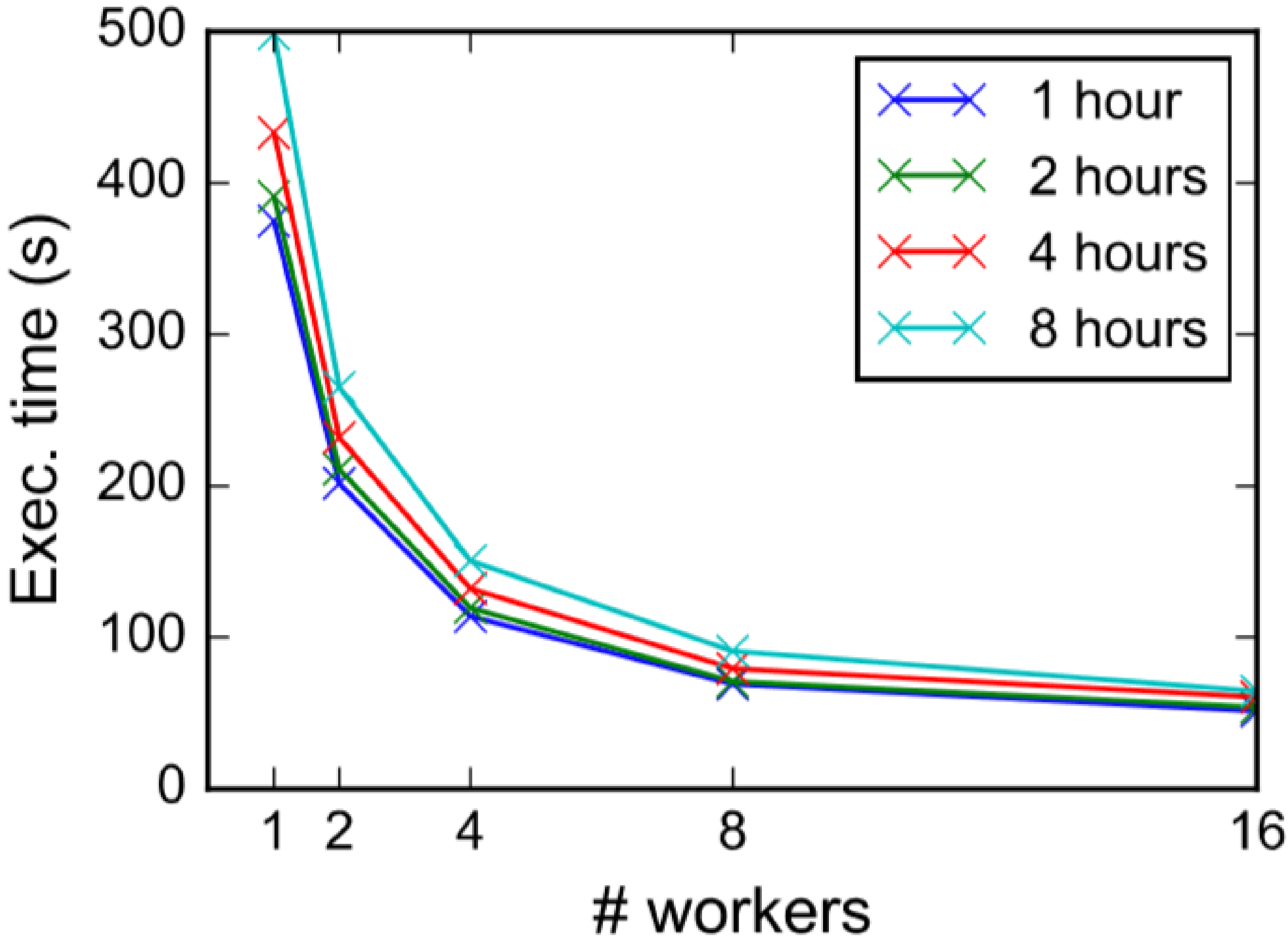
实验结果

- 轨迹数量的影响



实验结果

- 云平台性能



500
400
300
200
100
0
Exec. time (s)
1
2
4
8
16
workers
1 hour
2 hours
4 hours
8 hours

Planning Bike Lanes based on Sharing-Bikes' Trajectories

KDD 2017

问题背景

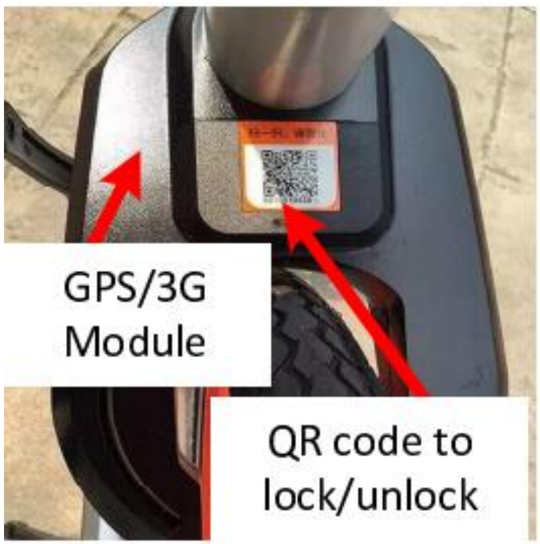
- **自行车出行**
  - 是一种方便环保的出行方式
  - 能有效减少交通拥堵
  - 对政府来说合理规划自行车道，一方面能够推行绿色生活方式，另一方面治理交通拥堵
- **现实问题**
  - 预算限制
  - 施工方便
  - 实用性

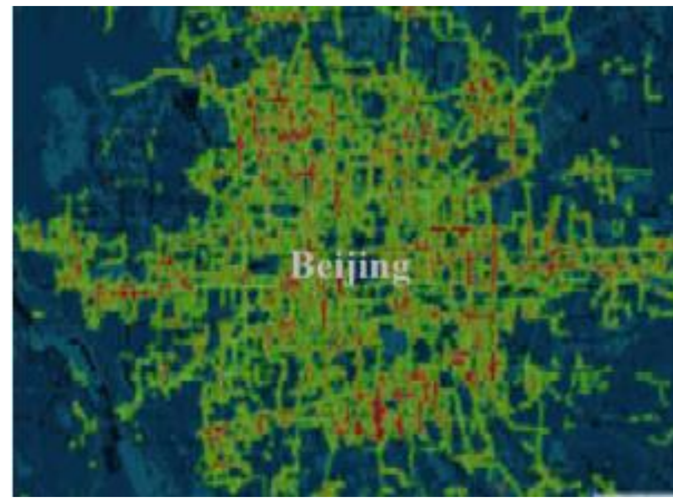
相关研究

- 数据驱动的城市规划
  - Urban computing with taxicabs. In Proceedings of the 13th international conference on Ubiquitous computing. ACM 2011.
- 轨迹数据挖掘
  - Summarizing trajectories into k-primary corridors: a summary of results. In SIGSPATIAL GIS.ACM 2012.
  - Fast and exact network trajectory similarity computation: a case-study on bicycle corridor planning. In UrbComp. ACM 2013.
- 传统自行车道规划
- 有站式共享单车轨迹的应用
  - Traffic prediction in a bike-sharing system. In SIGSPATIAL GIS. ACM 2015.
  - Rebalancing Bike Sharing Systems: A Multi-source Data Smart Optimization. In SIGKDD. ACM 2016.

数据依托

- 将Mobike的轨迹数据集用于自行车道规划的优势
  - 无站式共享单车，覆盖全城，反映出自行车道的真实需求
  - 找车方便，轨迹记录准确





问题建模

- 将城市的道路网以图的形式表示 G = (V,E)
  - V：道路交汇路口集合，E={e} 道路路段集合

挑选出最合适的道路修建自行车道

从图中选择出满足约束条件的边

问题定义

预算约束：资金预算体现为挑选的边子集的总长度

$$\sum_{e_i \in E'} e_i.c \leq B$$

施工约束：施工队的数量体现为最后的子图中最多k个连通子图

$$C(E') \leq k$$

问题定义

实用约束：使用率体现为尽可能方便更多的用户，尽可能多的覆盖轨迹中的连续路段

$$tr_i.g = \sum_{s_j \in S_i} \alpha^{\frac{s_j.\ell}{min(e.\ell)}} \times \frac{s_j.\ell}{min(e.\ell)}, \alpha \geq 1 \quad E'.g = \sum_{tr_i \in Tr \& tr_i \cap E' \neq \emptyset} tr_i.g$$

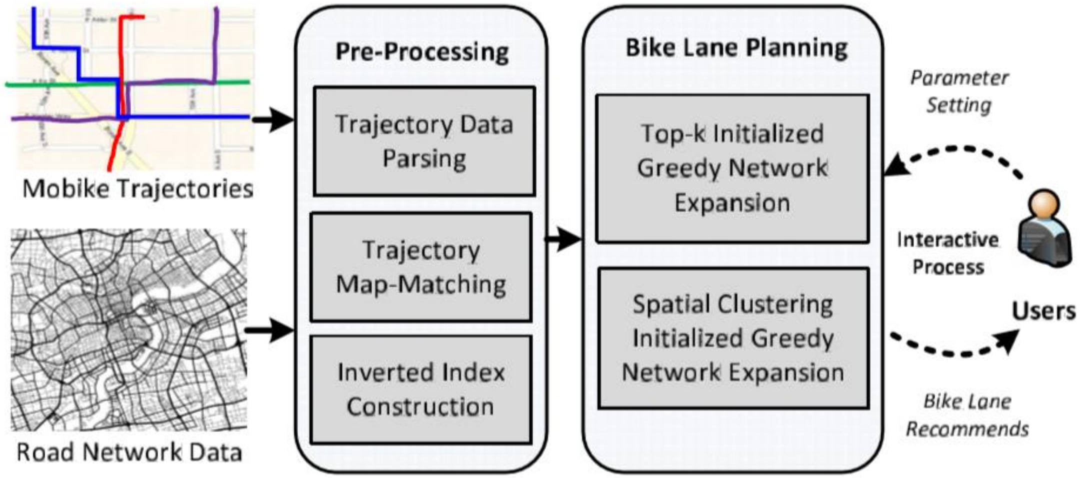
条件：G, Tr, α, k, B

目标：找到一个得分g最高的子集E’

NP难的整数规划问题

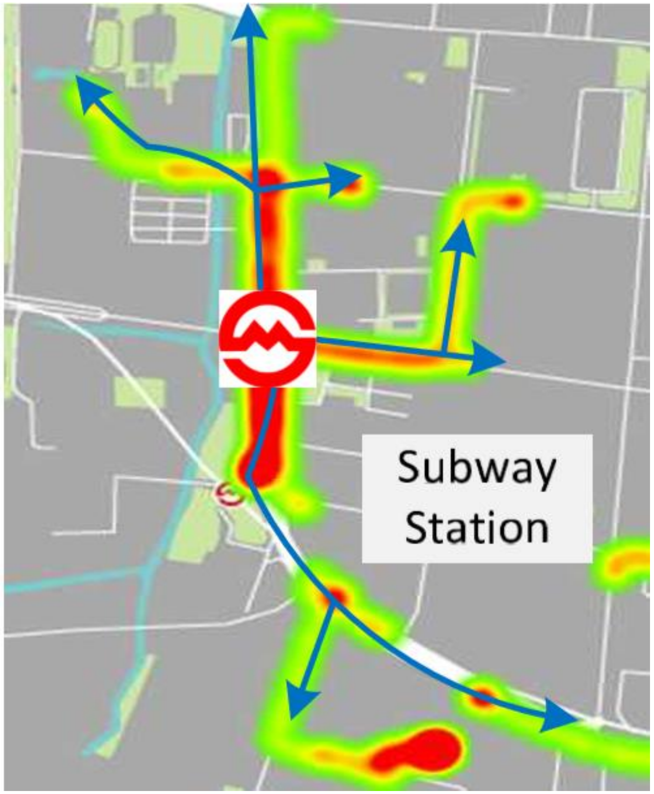
系统结构

- 预处理阶段
  - 轨迹预处理，主要是数据清洗
  - 轨迹的地图匹配，运用了一种投票式的地图匹配算法
  - 为路段建立倒排索引，给每条轨迹设置道路路段的ID，方便查找
- 道路规划阶段
  - 初始化（两种方法）
  - 网络扩增
  - 结束约束



算法设计

空间热点



星型分布模式

基于贪心的网络扩增算法

Input: Road Network $G = (V, E)$, Inverted index I, Trajectory Dataset Tr, Total budget B, tuning parameter α, and a value k.

选择热度最高的k个路段, 作为E’的初始结果, 将该k个路段的分支作为下一轮次的候选路段

Output: Result road segment set E'.

//Stage 1: Initialization

Road Segment Set $E' \leftarrow k$ starting road segments

约束2:k个连通子图

Candidate set $C \leftarrow$ adjacent road segments of E'

Remaining Budget $B \leftarrow B - \sum_{e_i \in E'} e_i.c$

//Stage 2: Network Expansion

while Budget $B > 0$ **do**

约束1:预算

  $MaxGain \leftarrow 0$; $e_{next} \leftarrow \emptyset$

  for $e_i \in$ Candidate set C **do**

    if $e_i.c < B$ **then**

      Retrieve trajectories Tr' from I based on $E' \cup e_i$

      Calculate beneficial score difference per cost $\Delta g = \frac{g'-g}{e_i.c}$

      if $MaxGain < \Delta g$ **then**

约束3:最优子结构

        $MaxGain = \Delta g$; $e_{next} \leftarrow e_i$

  $E' \leftarrow E' \cup e_{next}$; $B \leftarrow B - e_{next}.c$

  Candidate Set $C \leftarrow C \cup$ none-selected adjacent edges of e_{next}

更新条件,进行下一轮次迭代

//Stage 3: Termination

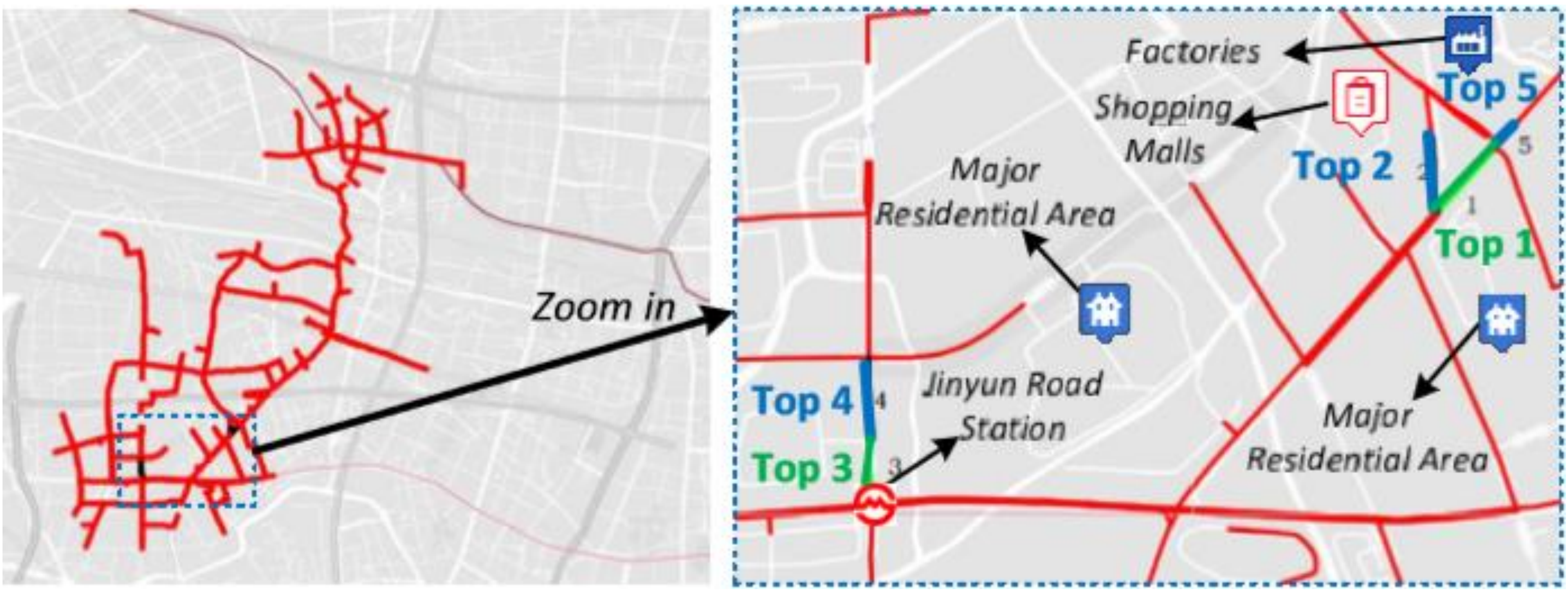
return E'

算法细节

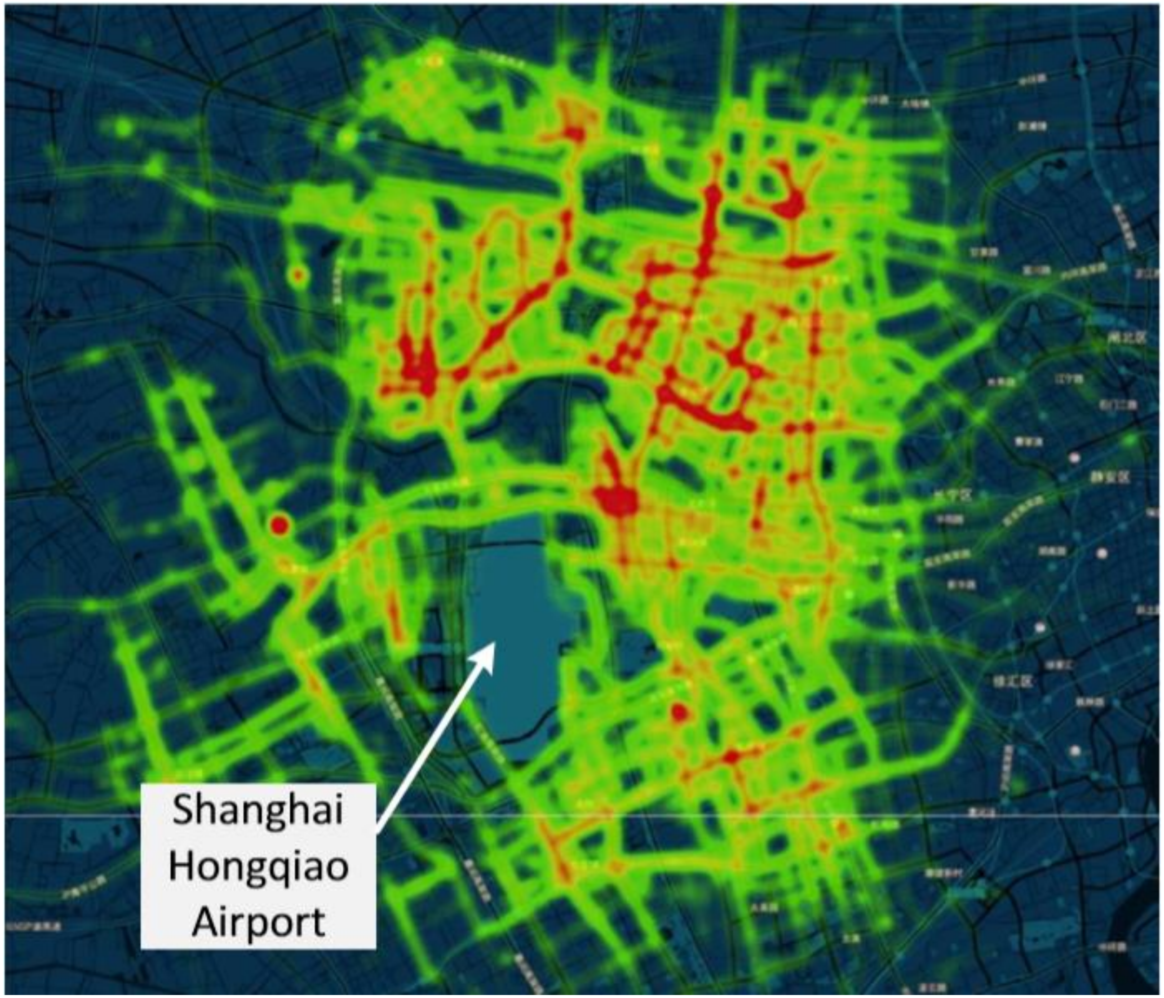
Edge ID	e_1	e_2	e_4	e_5	e_6	e_7	e_8	e_{10}	e_{11}
Δ Gain	1	4	5	7	8	4	5	9	5
Cost	2	2	2	2	5	2	1	5	2

算法优化

- 优化初始化算法
  - Top-k方法

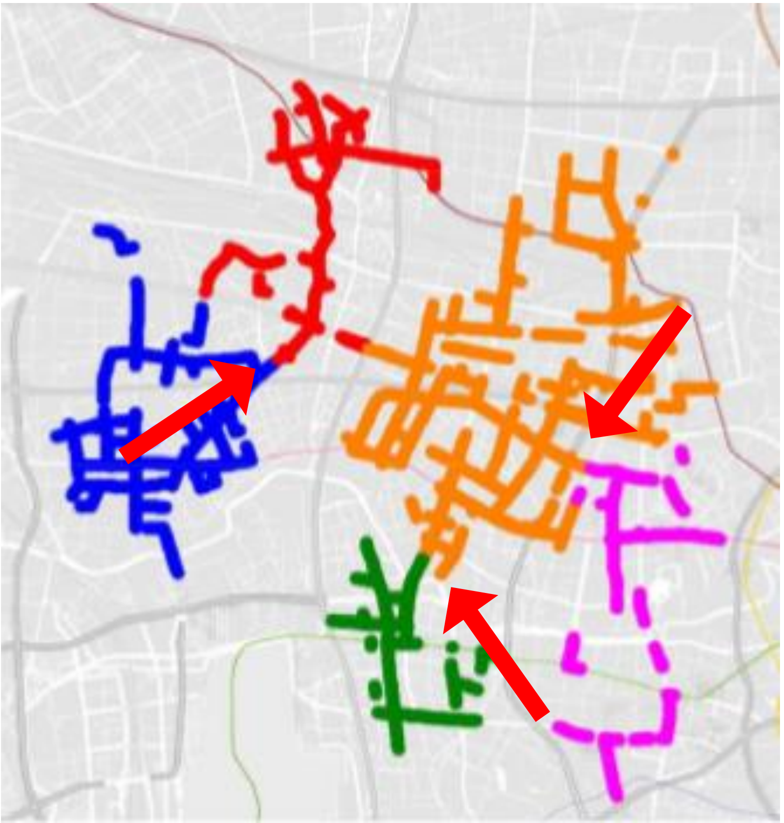


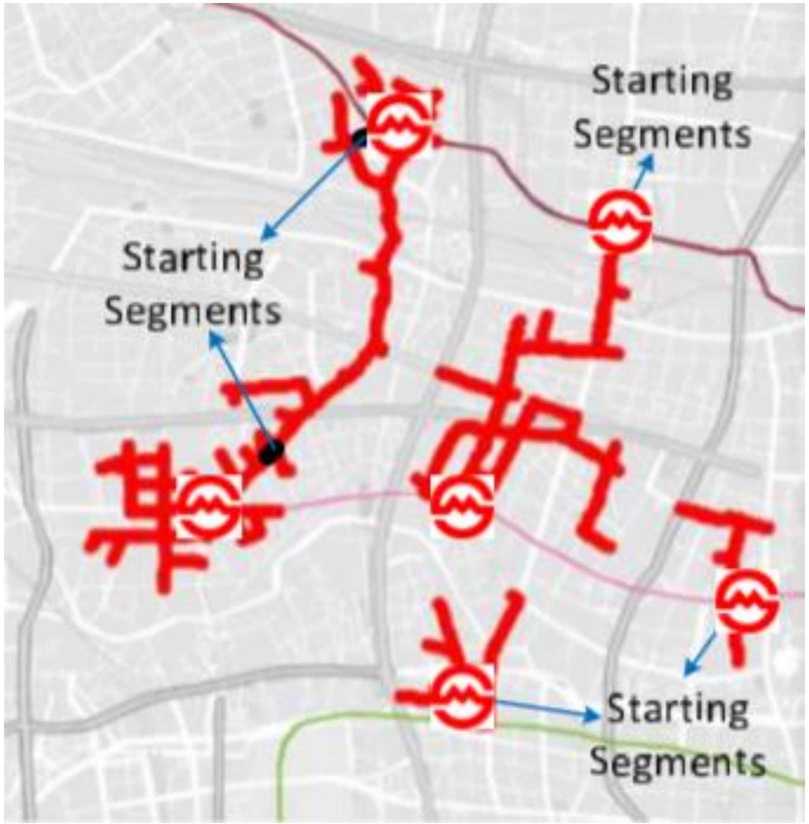
算法优化



- 起始路段候选集
  - 所有路段中，轨迹覆盖数量最高的前1%的路段作为起始路段的候选集合
- 空间聚类
  - 运用空间聚类的方法把候选集合进行聚类，然后从每个聚类中选择热度最高的路段作为算法的输入

优化结果

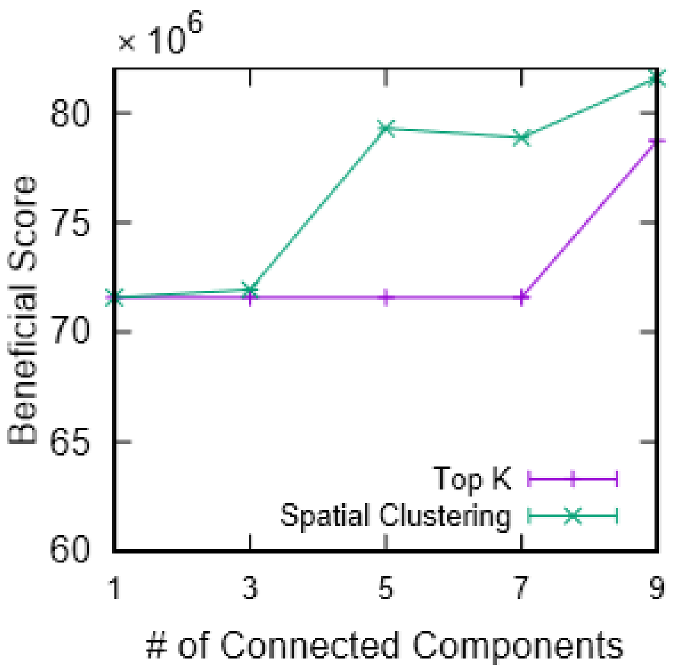




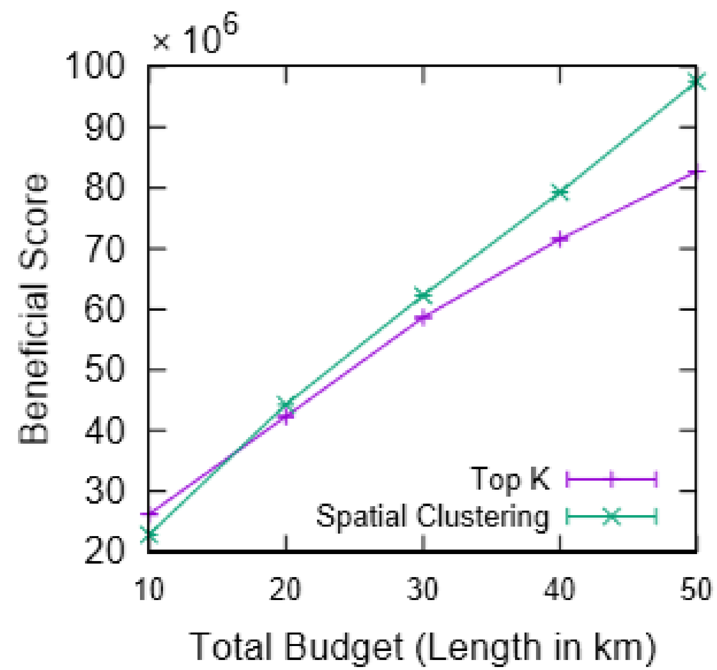
实验数据

- 上海道路网络
  - 来自Bing Map，333,766个路口，440,922路段
- 摩拜轨迹
  - {bike ID, UID, temporal Range, { start, end },{ GPS Points}}
  - 09/01/2016 - 09/30/2016
  - 13,063个用户, 3,971辆单车, 230,303条轨迹，共19,039,283个GPS点

实验结果



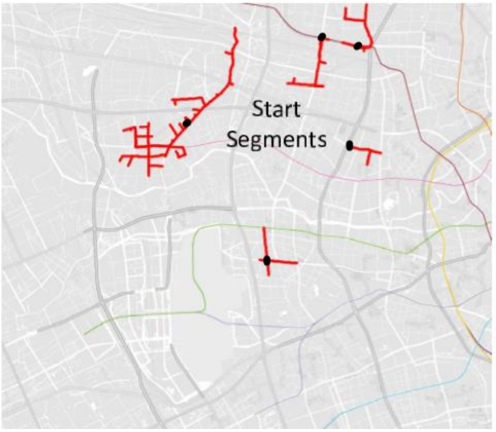
不同k值



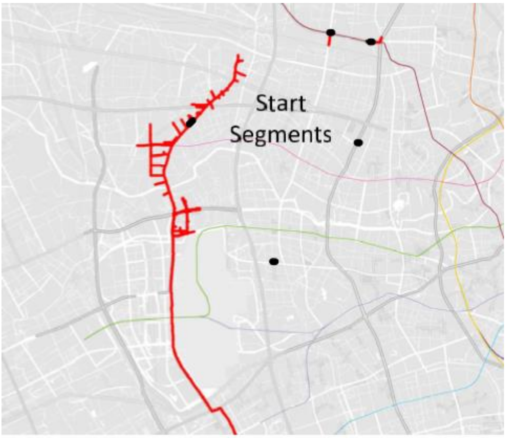
不同预算

实验结果

$\alpha = 1$



$\alpha = 1.03$



总结

- 结合现实问题
- 数据预处理
  - 数据清洗
  - 数据映射
  - 数据索引