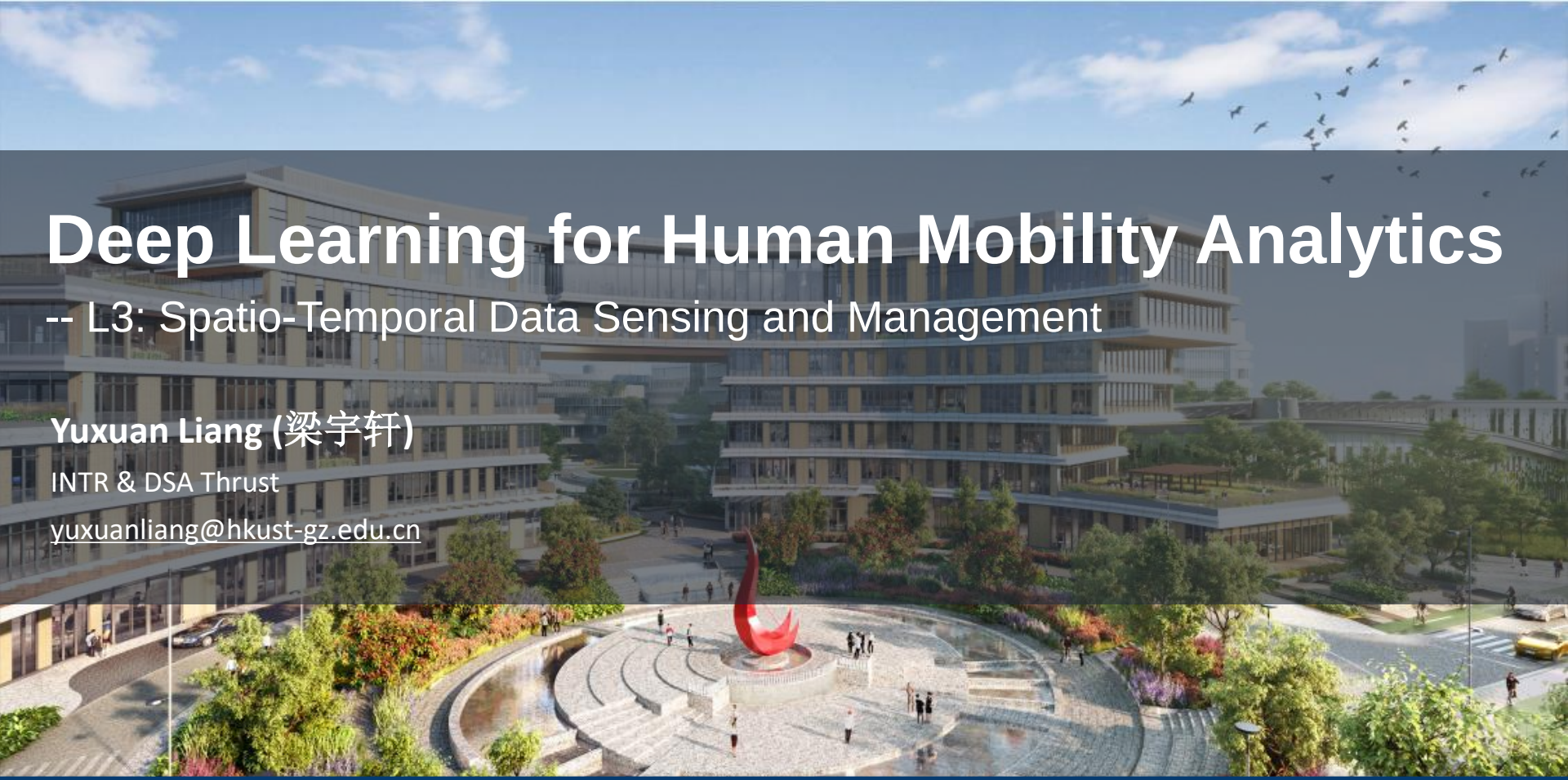


Deep Learning for Human Mobility Analytics

-- L3: Spatio-Temporal Data Sensing and Management

Yuxuan Liang (梁宇轩)

INTR & DSA Thrust

yuxuanliang@hkust-gz.edu.cn

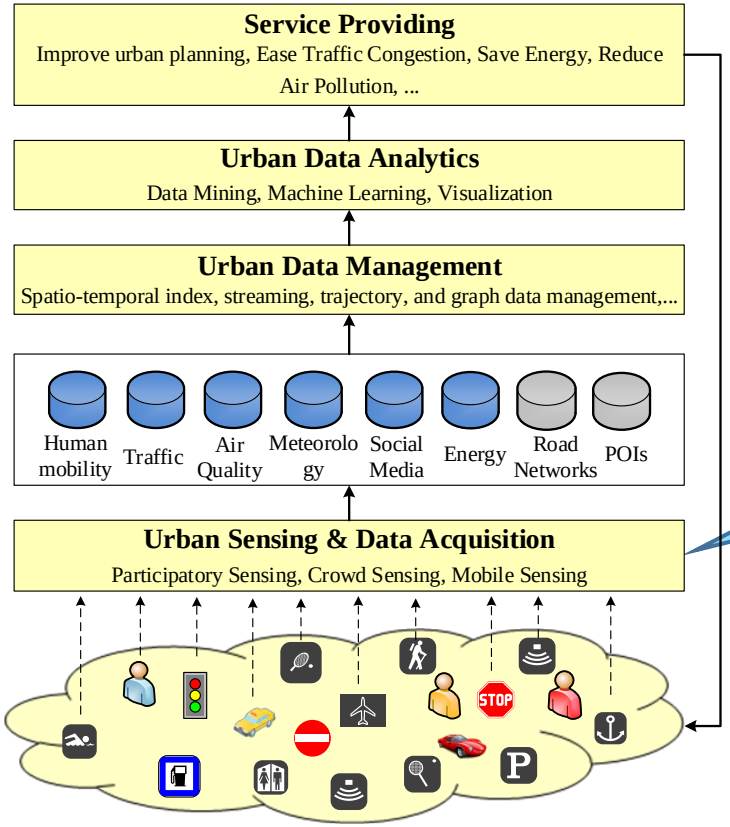
香港科技大学(广州)
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

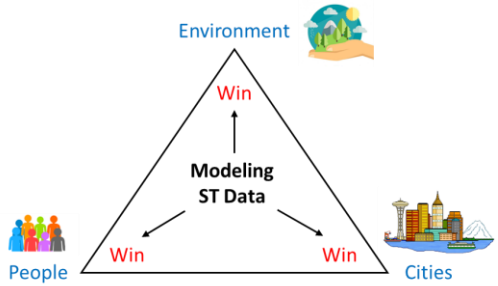
Notes

13	16 Sep	Transfer to **an online course for project discussion** (each team 20 mins) in **early November**. Details will be informed in late October		Mid-Autumn Festival
3	23 Sep	L3: Spatio-Temporal Data Sensing and Management	✓	To form team (2 people)
4	30 Sep	L4: Cross-Domain Data Fusion and Multimodal Learning	✓	

Revisit our Last Course

1st Stage: Urban Sensing & Data Acquision



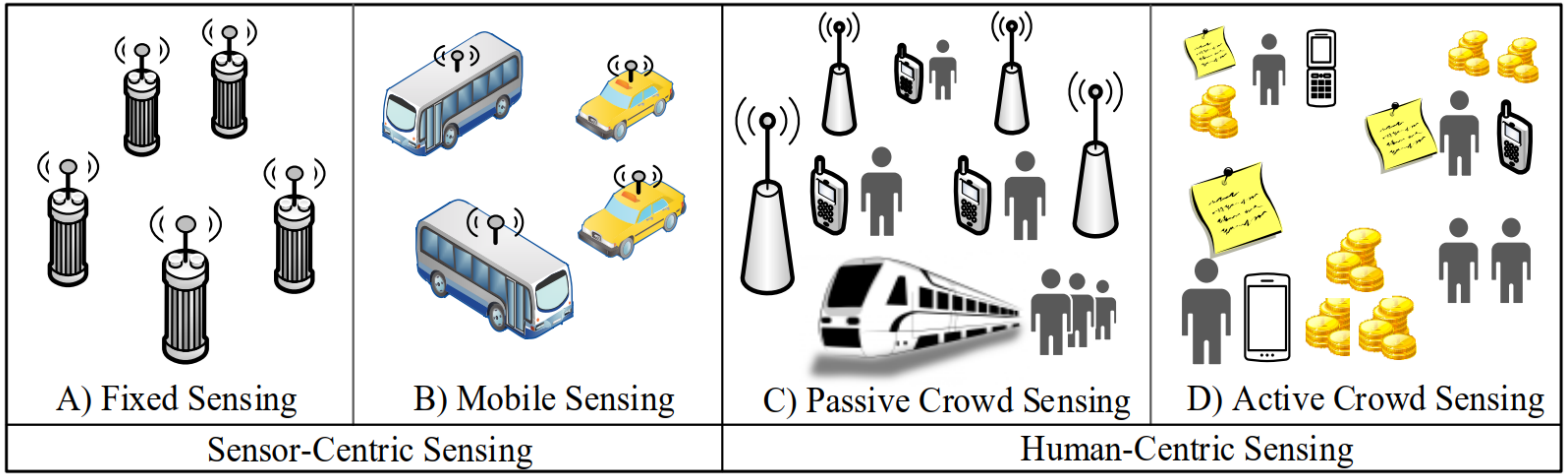


Collecting urban data through
- Sensor-centric sensing
- Human-centric sensing

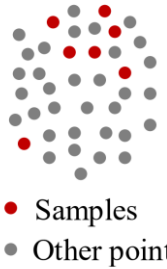
Urban Sensing & Data Acquisition
Crowd Sensing: Passive Sensing, Participatory Sensing, Task Assignment...
Fixed sensing: Sensor Deployment, Location Selection for Sensors...

Spatio-Temporal Sensing Modes

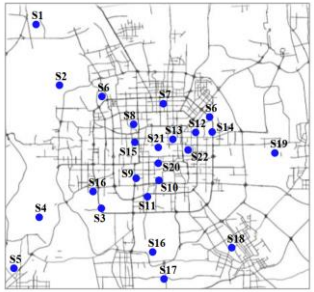
- Sensor-centric sensing
  - Fixed (location) sensing
  - Mobile (objects) sensing
- Human-centric sensing
  - Passive crowd sensing
  - Active crowd sensing



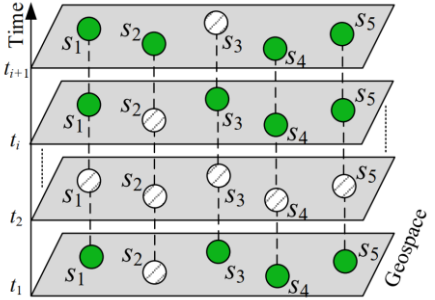
Challenges of Urban Sensing



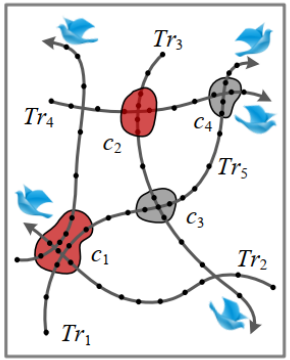
Biased distribution

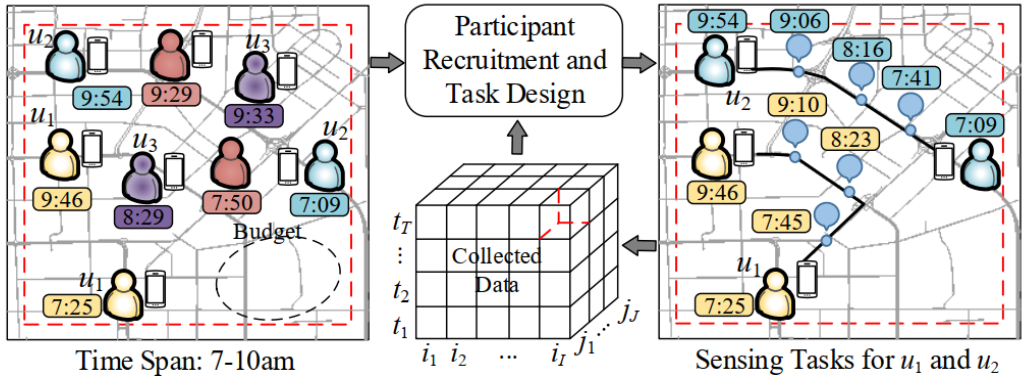


Data sparsity



Data missing





Resource deployment

Biased Distribution

- Inferring Gas Consumption and Pollution Emission of Vehicles throughout a City

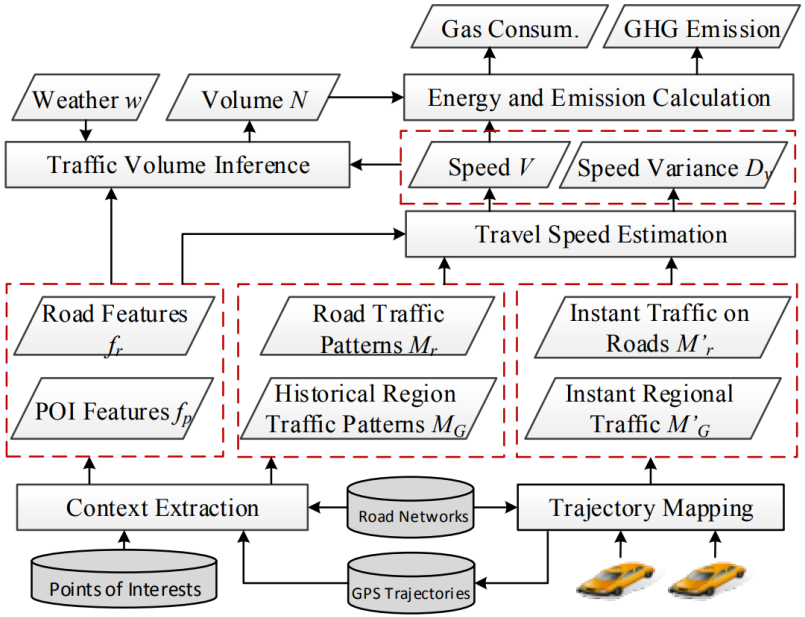
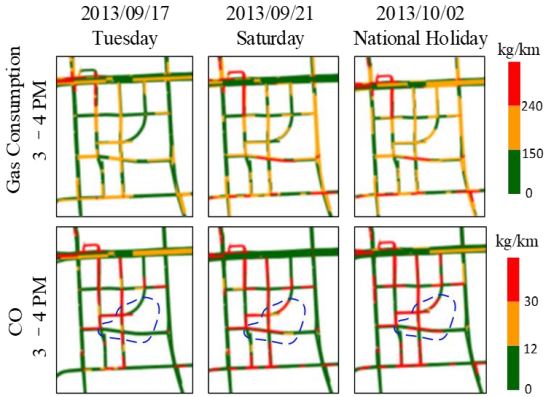
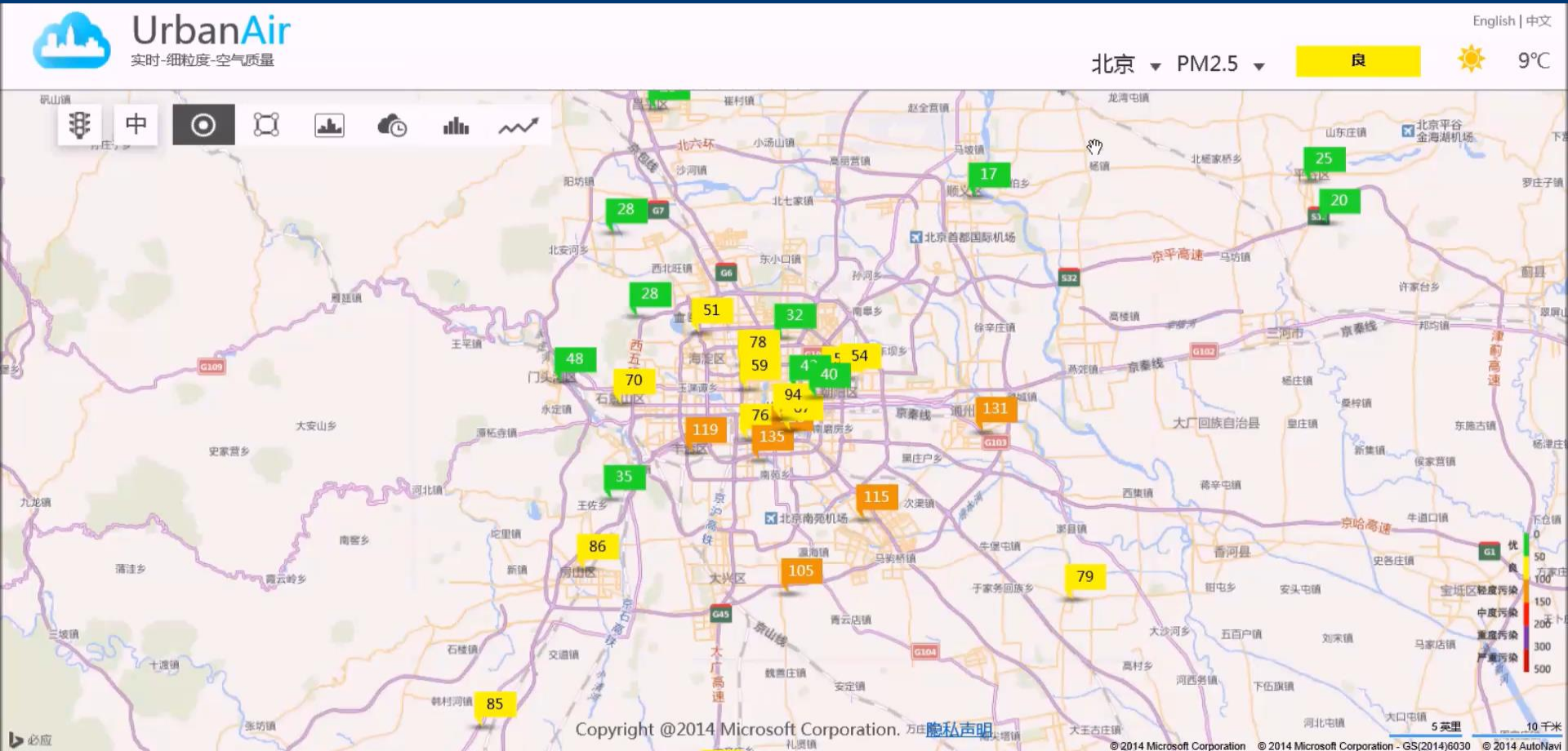


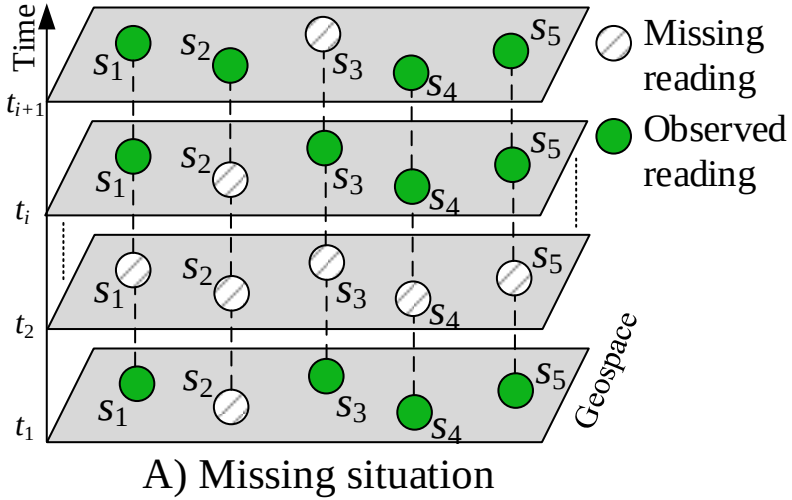
Figure 2. Framework of our method



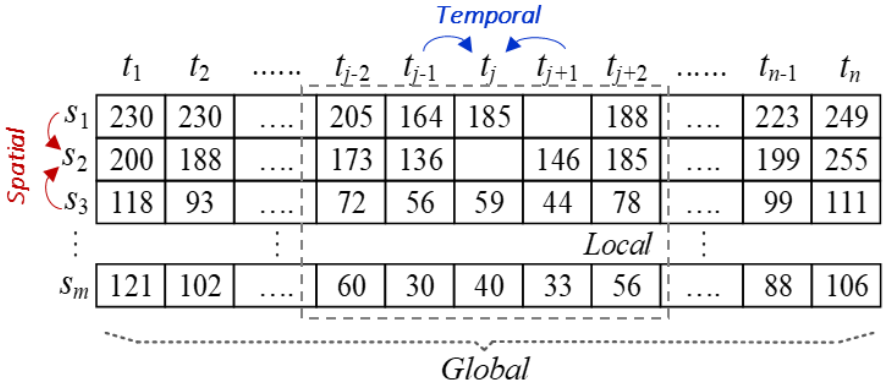
Data Sparsity

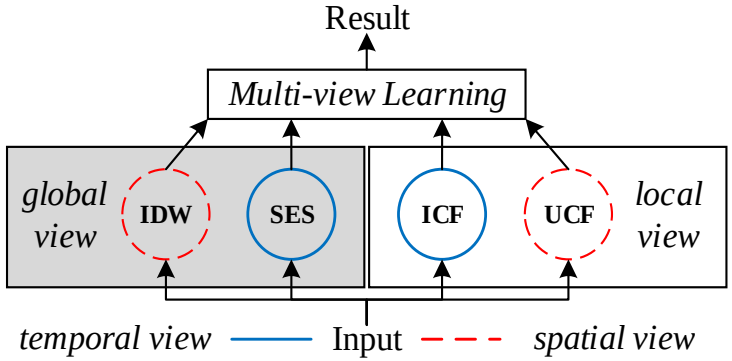


Data Missing



A) Missing situation



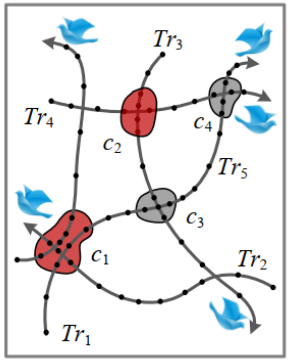


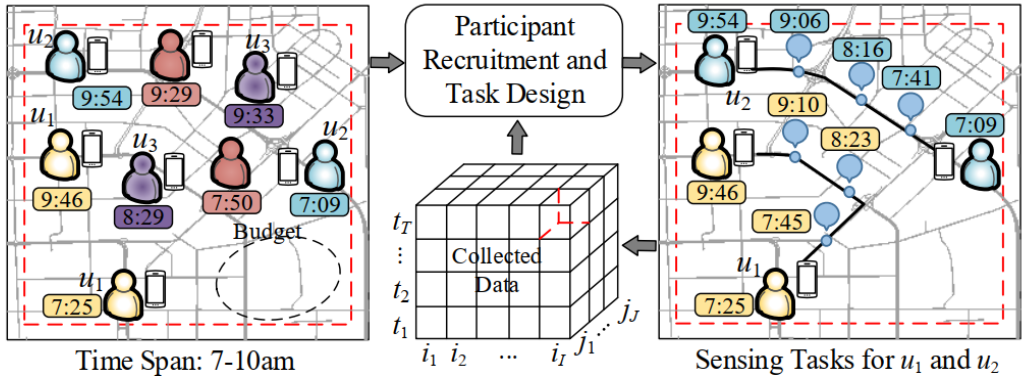
Challenges of Urban Sensing

Biased distribution

Data sparsity

Data missing

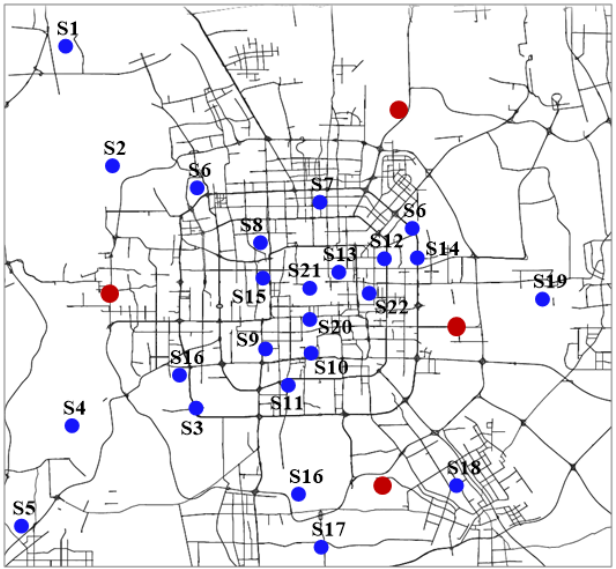




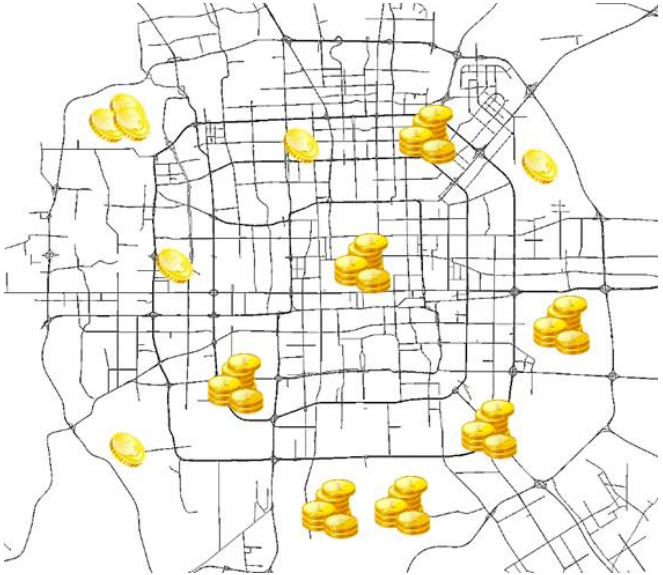
Resource deployment

A limited resource (budget, labors, land…)

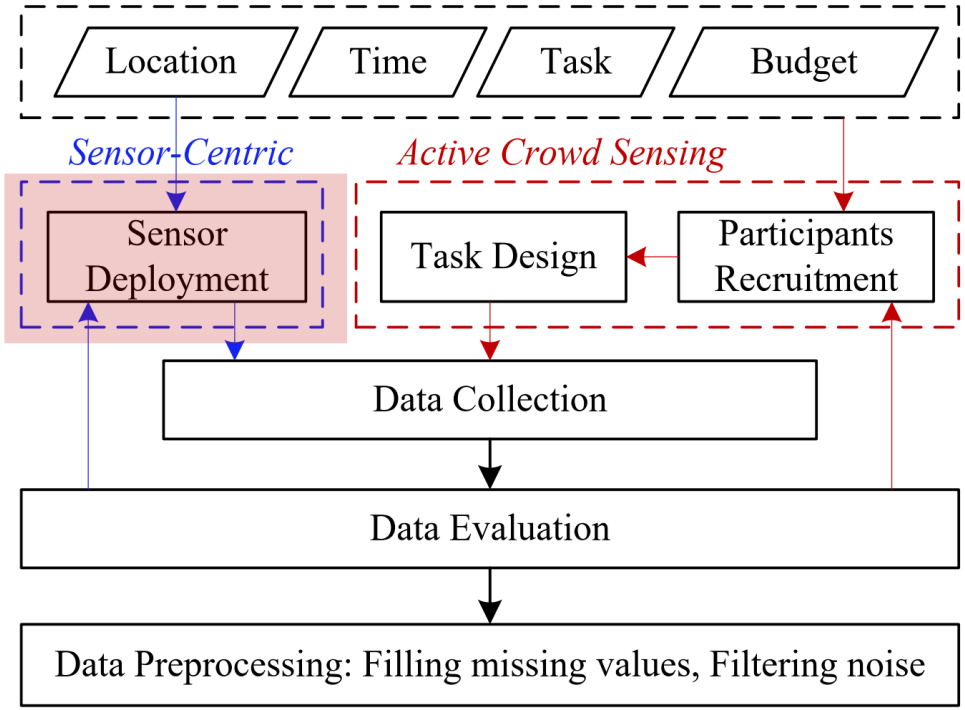
- Static sensing: Where to deploy sensor to maximize the gain?



- Crowdsensing: How to arrange the incentives dynamically?



General Framework of Urban Sensing



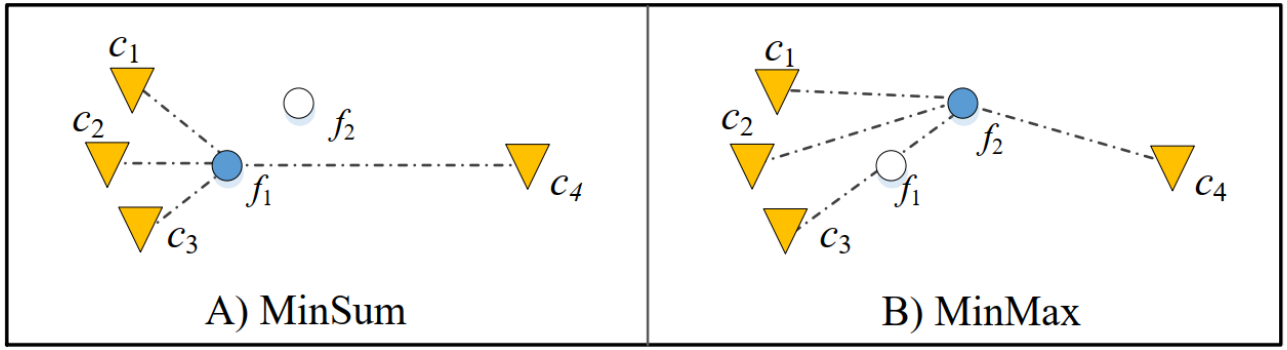
Resource Deployment and Location Selection

- Optimal meeting points problems
- Maximum coverage problem -- submodular maximization
- Learning-to-rank problems
- Uncertainty minimization problems (entropy)

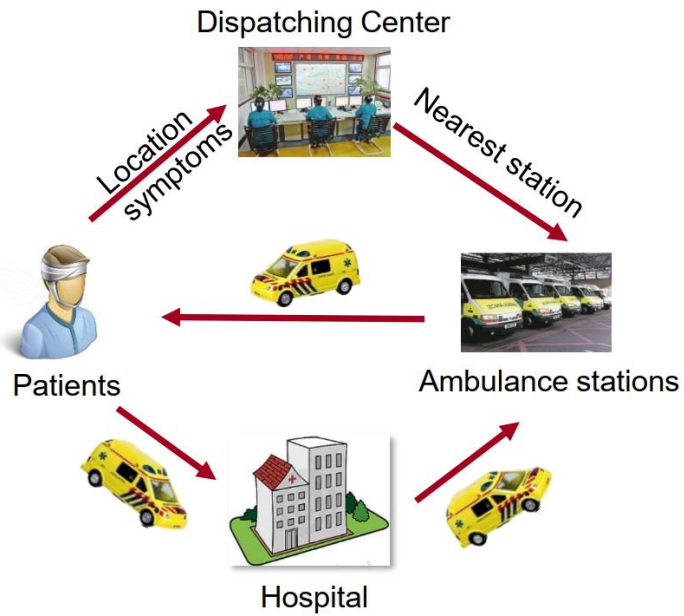
Optimal Meeting Points

- locate k facilities such that
  - MinSum: Minimizing the average cost to reach all clients can be minimized
  - MinMax: Minimizing the maximum cost to reach those clients

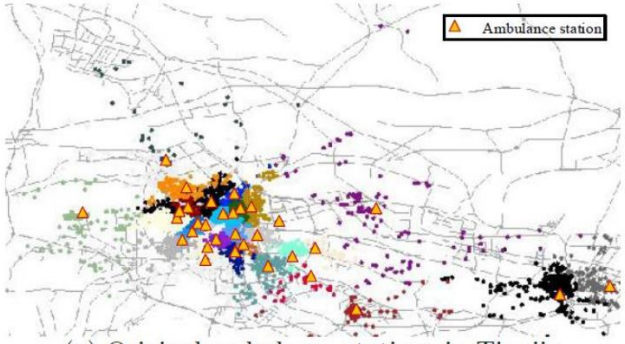
NP-hard!



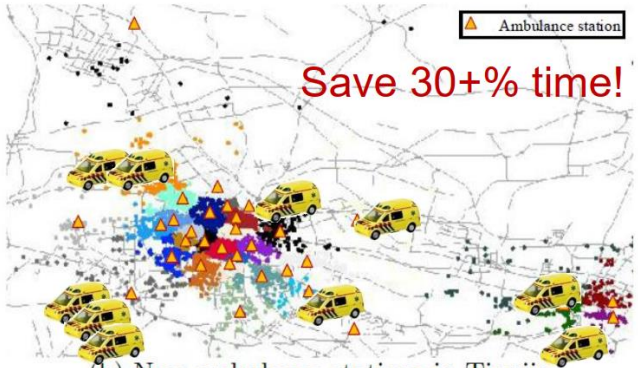
Improving Medical Emergency Services using Big Data



- Select locations for Ambulance Stations
- Dynamic ambulance allocation



(a) Original ambulance stations in Tianjin

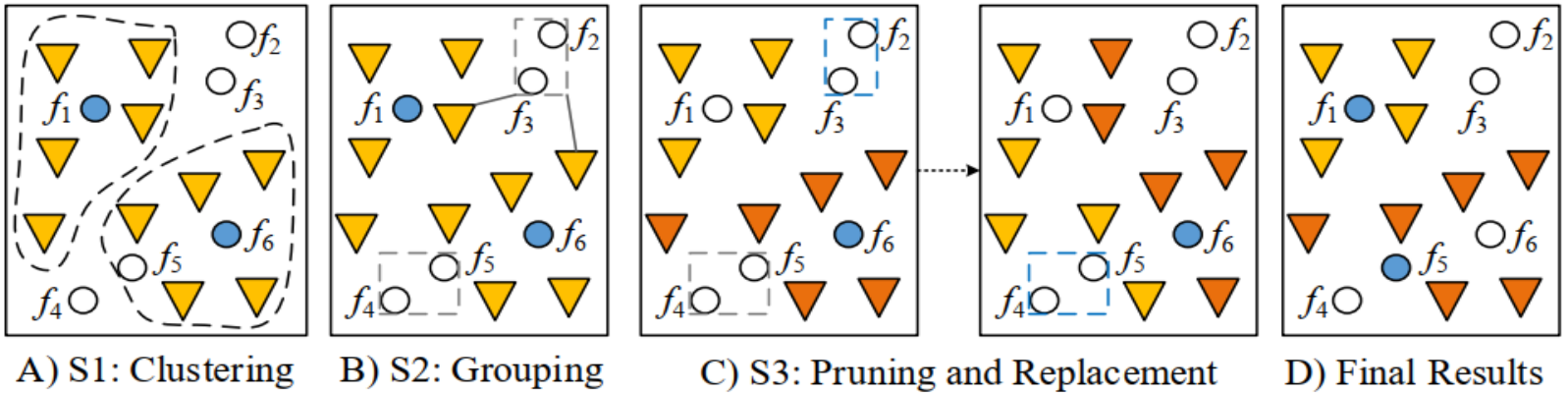


(b) New ambulance stations in Tianjin

Yuhong Li et al., Location Selection for Ambulance Stations: A Data-Driven Approach, ACM SIGSPATIAL 2015

Deploying Ambulance Stations Based on Optimal Meeting Point Model

- Methodology
  - k-medoids clustering
  - Candidate grouping
  - Iterations: pruning and replacement

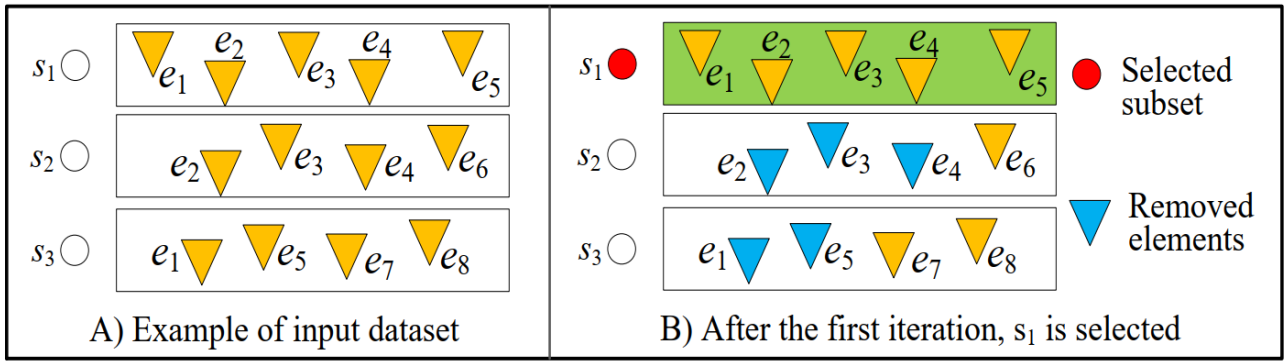


Yuhong Li et al., Location Selection for Ambulance Stations: A Data-Driven Approach, ACM SIGSPATIAL 2015

Maximum Coverage Problem

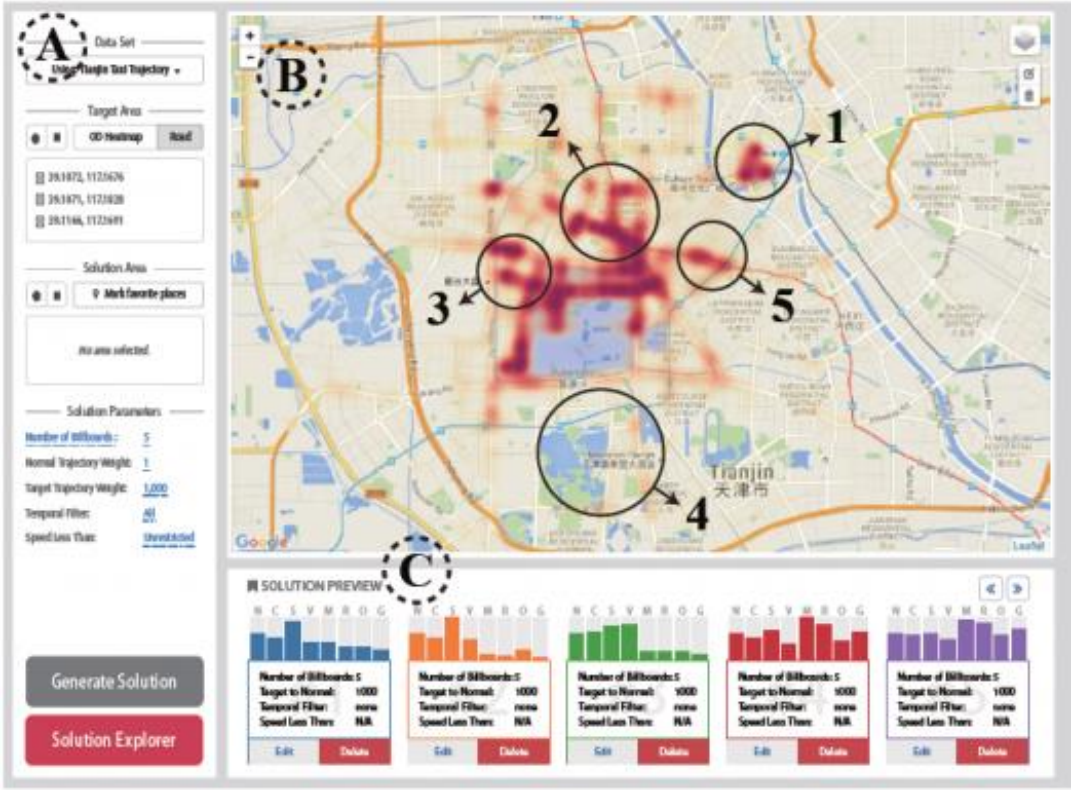
- Select a subset of (k) candidates to maximize
  - Nodes from a graph
  - Trajectories from a road network
  - Topic coverage from a document corpus

NP-hard!



$(1 - 1/e)$ approximation guarantee

Finding Top-k Most Influential Location Set



A submodular maximization problem, NP-hard!

Interactive Visual Data Analytics

Dongyu Liu et al., SmartAdP: Visual Analytics of Large-scale Taxi Trajectories for Selecting Billboard Locations, TVCG

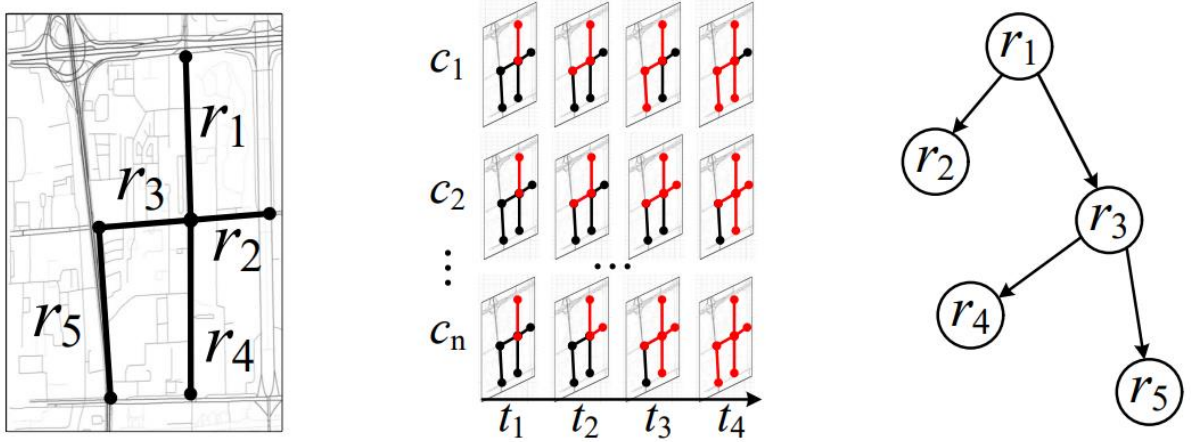
Inferring Traffic Cascading Patterns

A submodular maximization problem, NP-hard!

Yuxuan Liang et al., Inferring Traffic Cascading Patterns, SIGSPATIAL 2017

Inferring Traffic Cascading Patterns

- Target: to uncover how traffic congestion propagates thru road networks



(a) Road Network (b) Observed Cascades (c) Cascading Pattern

Yuxuan Liang et al., Inferring Traffic Cascading Patterns, SIGSPATIAL 2017

Inferring Traffic Cascading Patterns

- Also a maximum coverage problem

Cascading Pattern Construction

Model formulation

- Given a propagation tree T, the likelihood of a cascade c:

$$f(c|T)$$

- Given a network G considering all possible trees T, the likelihood of a cascade c:

$$f(c|G) = \sum_{T \in \mathrm{T}_c(G)} f(c|T) f(T|G)$$

For a specific cascade, this part is equal.

simplify

Conditional independence

Joint likelihood of C

$$f(C|G) = \prod_{c \in C} f(c|G) \qquad f(c|G) \propto \sum_{T \in \mathrm{T}_c(G)} f(c|T)$$

Optimization target: $\hat{G} = \arg\max_{|G| \le k} f(C|G)$ difficult to optimize

$\hat{G}$ best explains the observed cascades,
and the maximization is over all possible graphs G at most k edges

Alternative Optimization (CasInf)

$$\hat{G} = \arg\max_{|G| \le k} f(C|G)$$

1. matrix tree theorem
2. Combine environmental intensity
3. Log-likelihood

$$\hat{G} = \arg\max_{|G| \le k} F(C|G)$$

- Alternative target
- It satisfies submodularity, a natural diminishing returns property.

Optimization: greedy algorithm

Algorithm 1 Approximate algorithm for *CasInf*

Input:
k : the number of edges in cascading patterns we infer.
C : the set of cascades obtained in a time span.
$\mathcal{D}$: a constant denoting the spatial constraint.
Output: G: the inferred cascading pattern.

```
G ← K̂;
P ← all pairs (j, i): ∃c ∈ C with t_j < t_i and d_{j,i} < D
while |G| ≤ k do
    for all (j, i) ∈ P\G do
        δ_{j,i} = 0;
        for all c : t_j < t_i do
            ε_g ← environmental intensity inference
            w_c(m, n) ← weight of (m, n) in G ∪ (j, i);
            for all m : t_m < t_i and m ≠ j do
                δ_{c,j,i} = δ_{c,j,i} + w_c(m, i);
            end for
            δ_{j,i} = δ_{j,i} + log((δ_{c,j,i} + w_c(j, i)) / δ_{c,j,i});
        end for
    end for
    (j*, i*) ← argmax_{(j,i)∉G} δ_{j,i};
    G ← G ∪ (j*, i*);
end while
```

In each iteration, we select

$$e_i = \arg\max_{e \in G_i \setminus G_{i-1}} F(C|G_{i-1} \cup e) - F(C|G_{i-1}).$$

Yuxuan Liang et al., Inferring Traffic Cascading Patterns, SIGSPATIAL 2017

Inferring Traffic Cascading Patterns

- Also a maximum coverage problem

Cascading Pattern Construction

Model formulation

- Given a propagation tree T, the likelihood of a cascade c:

$$f(c|T)$$

- Given a network G considering all possible trees T, the likelihood of a cascade c:

$$f(c|G) = \sum_{T \in \mathrm{T}_c(G)} f(c|T) f(T|G)$$

For a specific cascade, this part is equal.

simplify

Conditional independence

Joint likelihood of C

$$f(C|G) = \prod_{c \in C} f(c|G) \qquad f(c|G) \propto \sum_{T \in \mathrm{T}_c(G)} f(c|T)$$

Optimization target: $\hat{G} = \arg\max_{|G| \le k} f(C|G)$ difficult to optimize

$\hat{G}$ best explains the observed cascades,
and the maximization is over all possible graphs G at most k edges

Alternative Optimization (CasInf)

$$\hat{G} = \arg\max_{|G| \le k} f(C|G)$$

1. matrix tree theorem
2. Combine environmental intensity
3. Log-likelihood

$$\hat{G} = \arg\max_{|G| \le k} F(C|G)$$

- Alternative target
- It satisfies submodularity, a natural diminishing returns property.

Optimization: greedy algorithm

Algorithm 1 Approximate algorithm for *CasInf*

Input:
k : the number of edges in cascading patterns we infer.
C : the set of cascades obtained in a time span.
$\mathcal{D}$: a constant denoting the spatial constraint.
Output: G: the inferred cascading pattern.

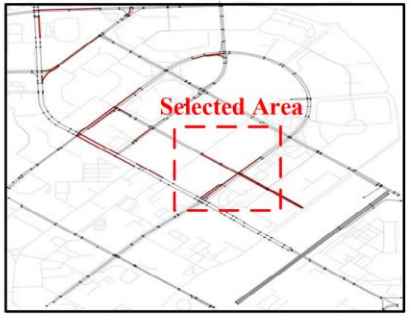
```
G ← K̂;
P ← all pairs (j, i): ∃c ∈ C with t_j < t_i and d_{j,i} < D
while |G| ≤ k do
    for all (j, i) ∈ P\G do
        δ_{j,i} = 0;
        for all c : t_j < t_i do
            ε_g ← environmental intensity inference
            w_c(m, n) ← weight of (m, n) in G ∪ (j, i);
            for all m : t_m < t_i and m ≠ j do
                δ_{c,j,i} = δ_{c,j,i} + w_c(m, i);
            end for
            δ_{j,i} = δ_{j,i} + log((δ_{c,j,i} + w_c(j, i)) / δ_{c,j,i});
        end for
    end for
    (j*, i*) ← argmax_{(j,i)∉G} δ_{j,i};
    G ← G ∪ (j*, i*);
end while
```

In each iteration, we select

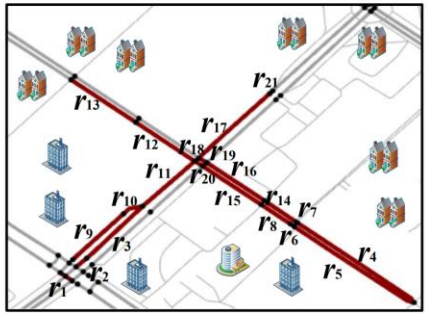
$$e_i = \arg\max_{e \in G_i \setminus G_{i-1}} F(C|G_{i-1} \cup e) - F(C|G_{i-1}).$$

Yuxuan Liang et al., Inferring Traffic Cascading Patterns, SIGSPATIAL 2017

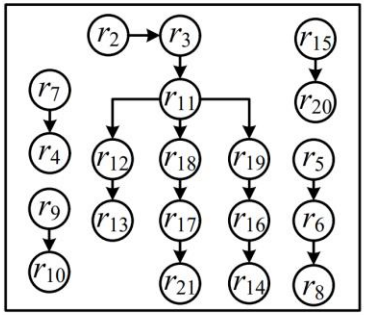
Inferring Traffic Cascading Patterns



(a) Wangjing district

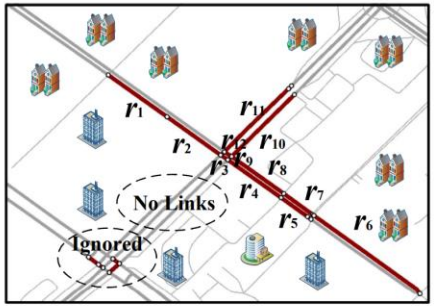


(b) Traffic cascading pattern between 8 and 10 am

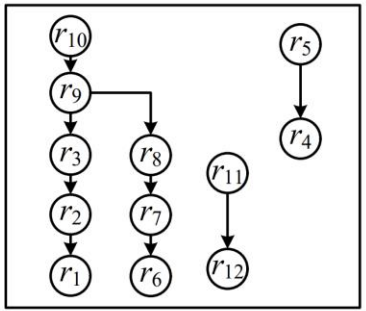


(c) Cascading pattern in (b)

(d) Selected area



(e) Traffic cascading pattern between 7 and 9 pm

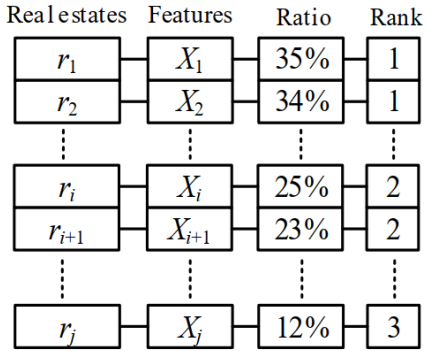


(f) Cascading pattern in (e)

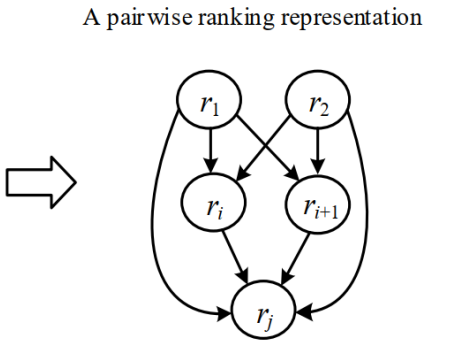
Yuxuan Liang et al., Inferring Traffic Cascading Patterns, SIGSPATIAL 2017

Learning to Rank

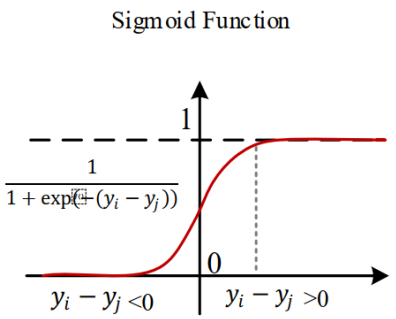
- Rank a set of locations so that the top-k best candidates can be selected to deploy resources or facilities
- Methods
  - Pointwise
  - Pairwise
  - Listwise



A) Ranking of Real Estates



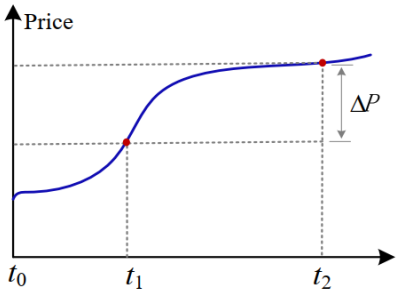
B) Graphical Representation of Orders



C) Keeping Pairwise Orders

Ranking and Clustering Real Estates using Big Data

- Values (learned from big data)
  - Increase more in a rising market
  - Decrease less in a falling market



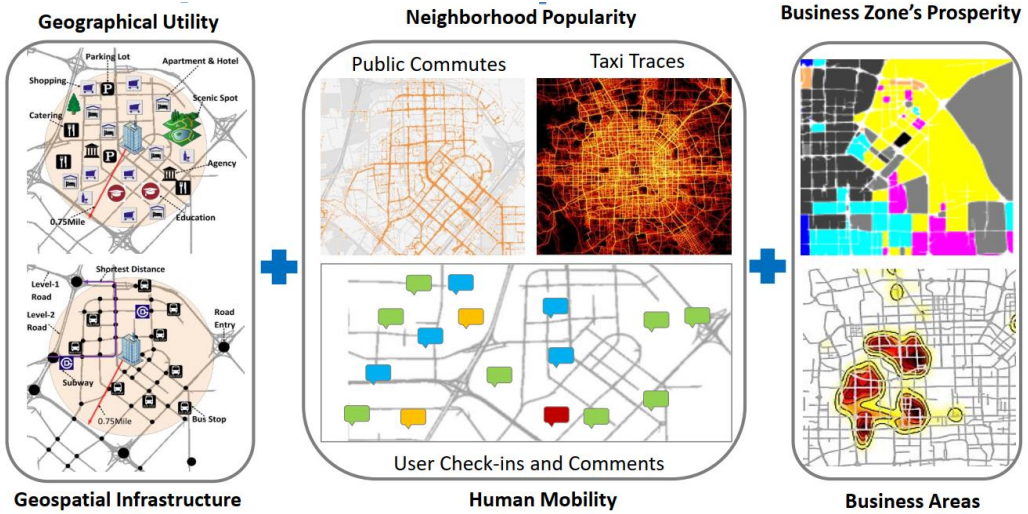
A) The price of a real estate

House	Increase	Rank↓
H1	35%	R1
H5	29%	R1
H4	13%	R2
….	….	….
H2	9%	R3
H3	2%	R3
H6	-1.5%	R4
H7	-6.1%	R5

B) Rank of estates by ΔP

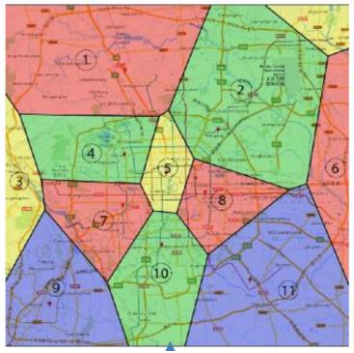
Yanjie Fu et al., Sparse Real Estate Ranking with Online User Reviews and Offline Moving Behaviors, ICDM 2014

Learning-to-Rank Problems



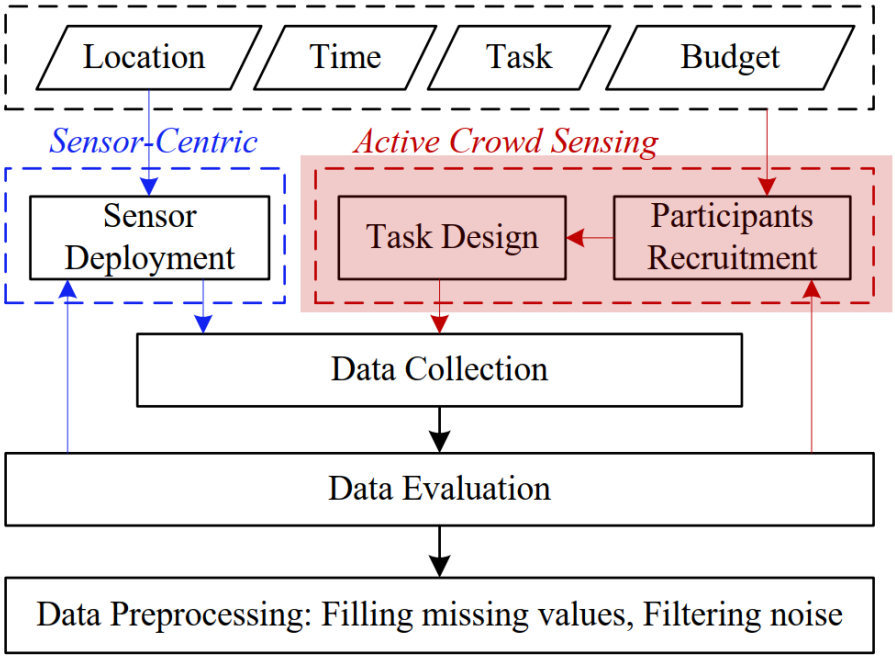
Ranking of Real Estates

Clusters of Real Estates



Yanjie Fu et al., Sparse Real Estate Ranking with Online User Reviews and Offline Moving Behaviors, ICDM 2014

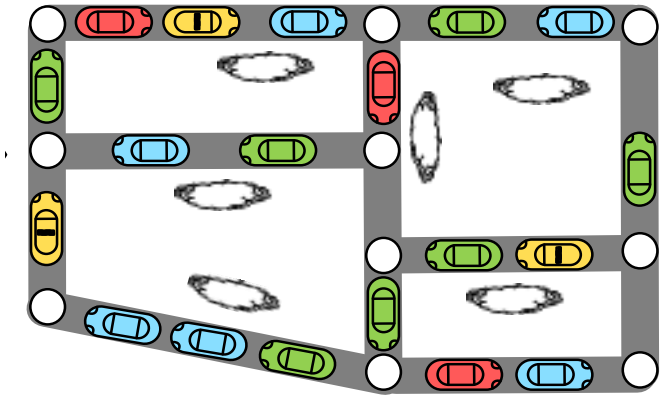
General Framework of Urban Sensing



Urban Sensing Based on Human Mobility

Ubicomp 2016

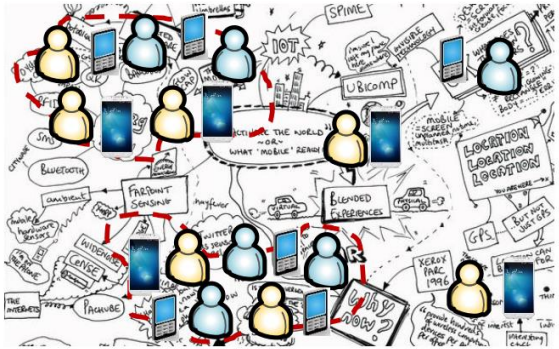
Shenggong Ji, et al. Urban Sensing Based on Human Mobility, UBICOMP 2014



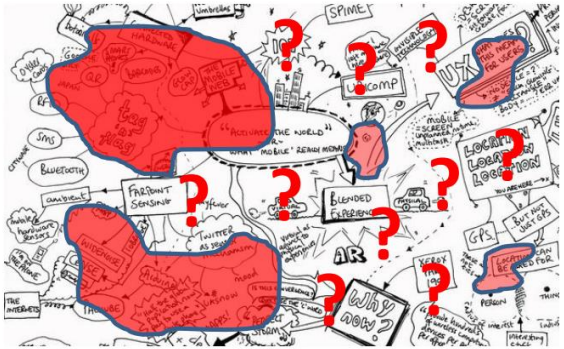
Urban Sensing

- Collecting urban data
  - Noise, temperature, air quality
  - Human as a sensor
- Brings challenges to
  - City-scale real-time monitoring
  - Further data analytics

Skewed human mobility

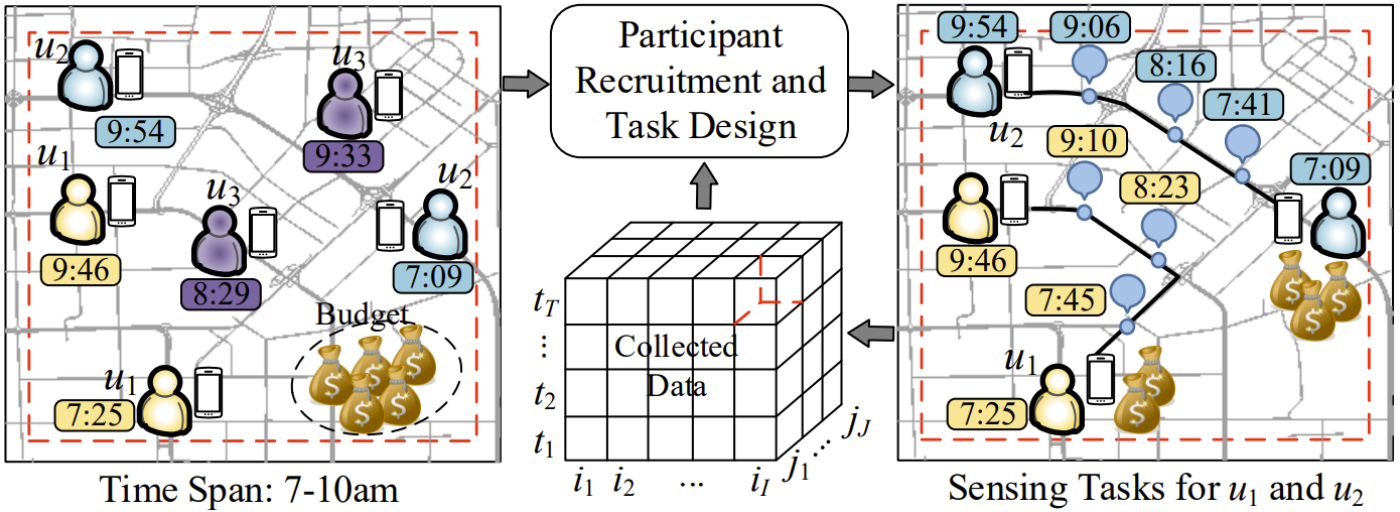


Imbalanced data coverage

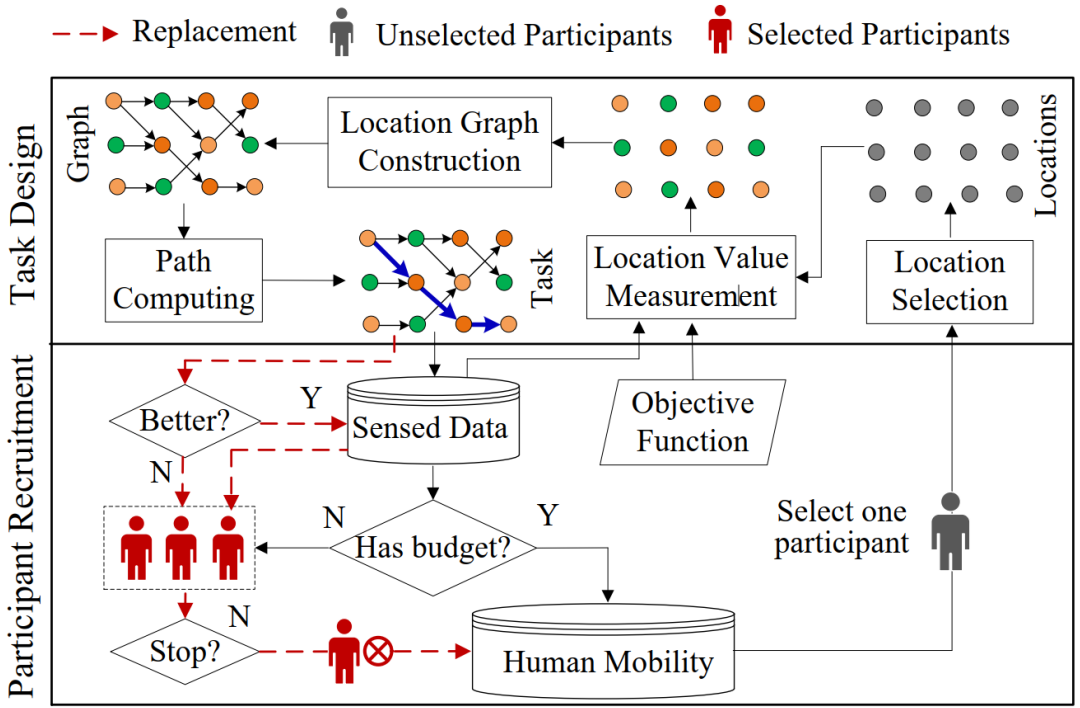


An Urban Sensing Framework

- Consider real-world human mobility
- Maximize the amount and balance of collected data
- Given a limited budget



Human Mobility-Based Urban Sensing Framework

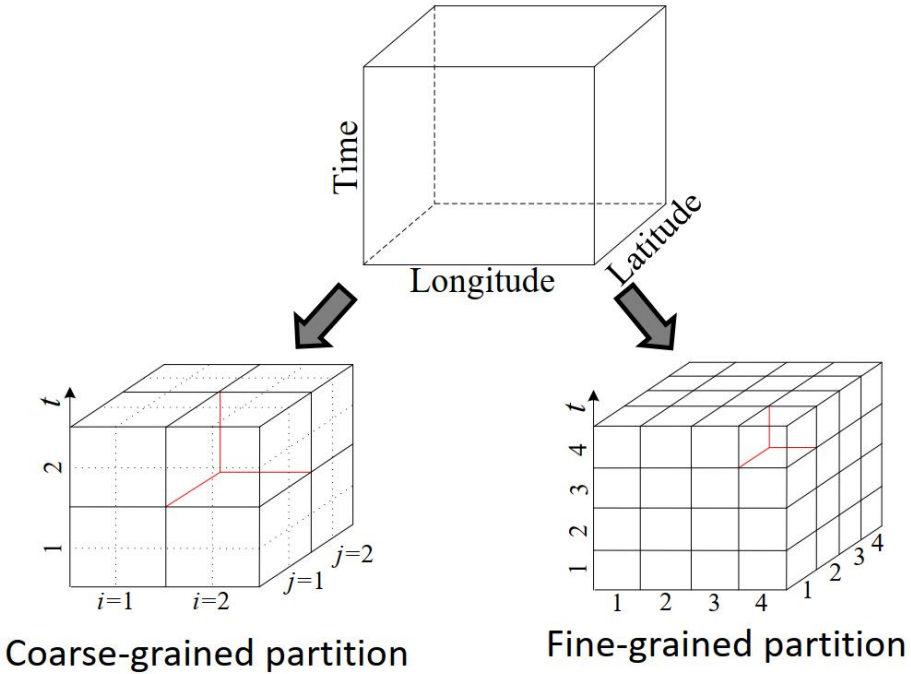


- A participant recruitment mechanism
  - random recruitment
  - replacement-based refinement

$$\max \phi = \alpha \times E + (1 - \alpha) \times \log_2 Q$$

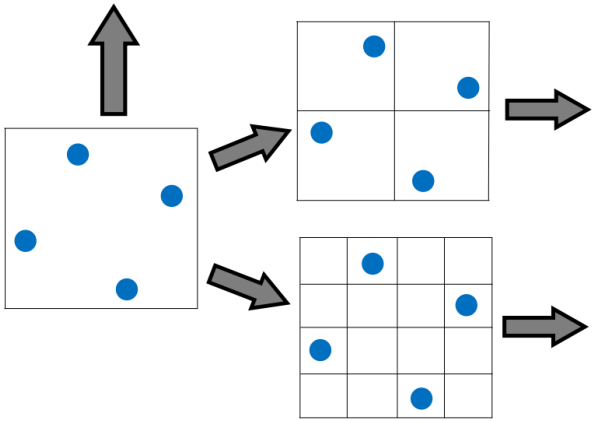
α: the relative preference of data balance to data amount

- application specific



Hierarchical Entropy-based Objective Function

Data amount: $Q = 4$



$$\left(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}\right) \rightarrow E_1 = 4 \times \left(-\frac{1}{4}\log_2\frac{1}{4}\right) = 2$$

$$\left(0,\frac{1}{4},0,0,0,0,0,\frac{1}{4},\frac{1}{4},0,0,0,0,0,\frac{1}{4},0\right)$$

$$E_2 = 4 \times \left(-\frac{1}{4}\log_2\frac{1}{4}\right) + 12 \times (0\log_2 0) = 2$$

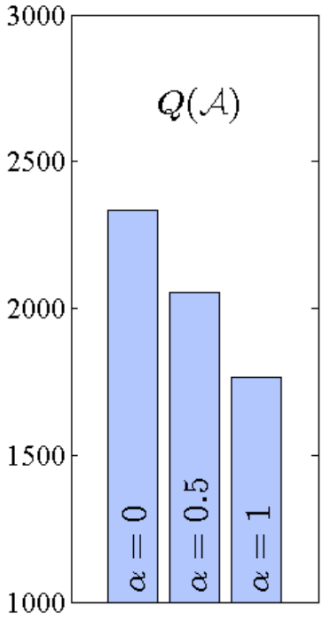
Coarse-grained

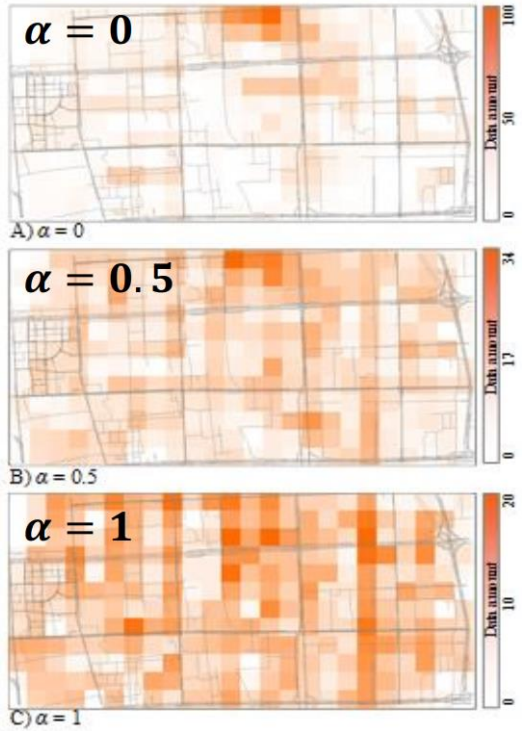
Fine-grained

Data balance: $E = \frac{(2\times E_1 + 1\times E_2)}{2} = 3$

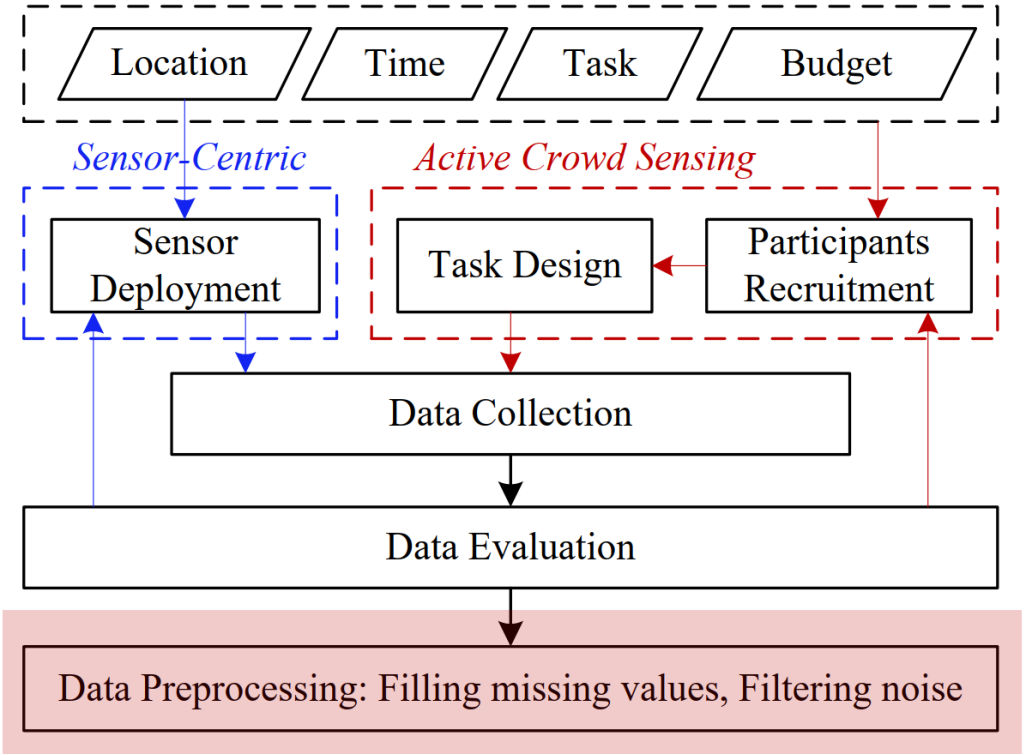
Real-World Evaluation

- Datasets
  - Human mobility dataset from a real-world noise sensing experiment
    - Sensing region: 6.6km × 3.3km
    - Sensing time interval: 6:00 am ~ 22:00 pm
  - 244 participant candidates with mobility information





General Framework of Urban Sensing

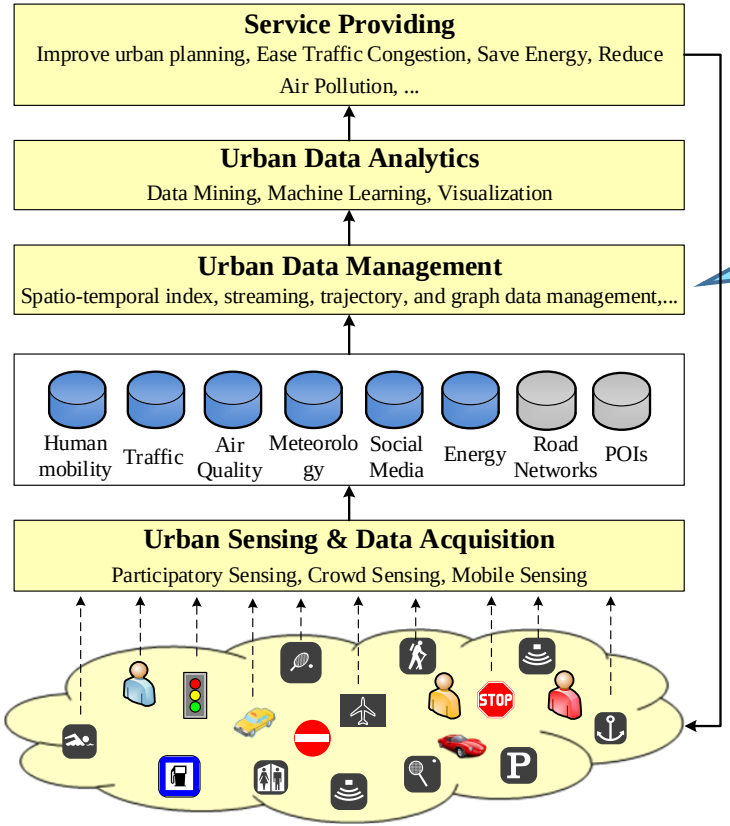


Objectives of this Course

To introduce

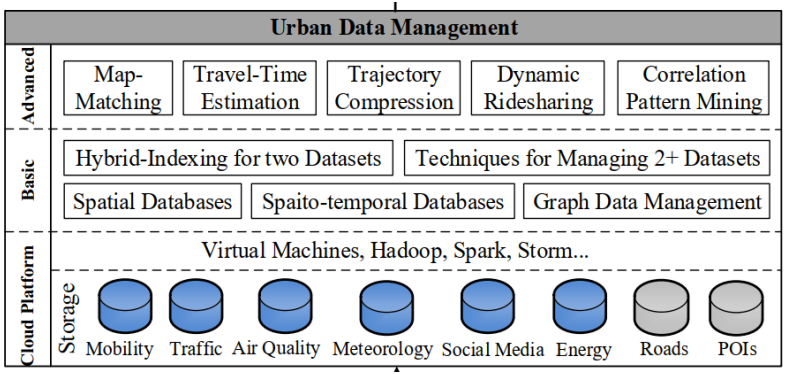
- Basic data management techniques
  - Spatial databases
  - Spatio-temporal databases
- Advanced applications
  - Map matching
  - Trajectory compression
  - Large-scale dynamic ridesharing
- Managing spatio-temporal data on the cloud

2nd Stage: Urban Data Management



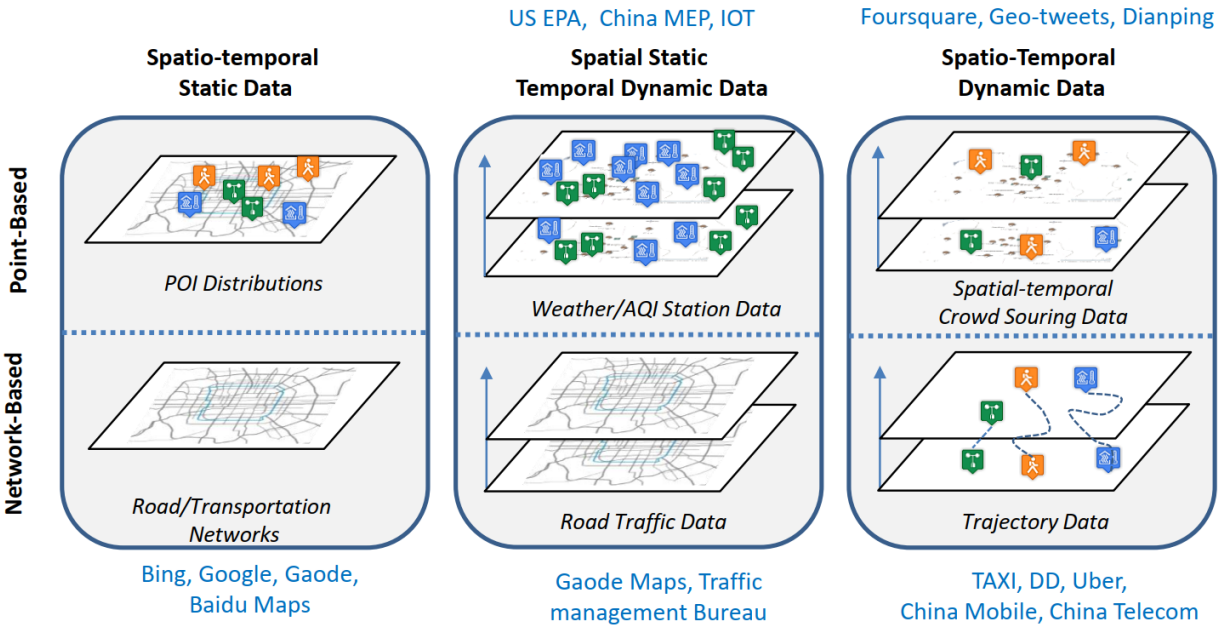
Manage cross-domain spatio-temporal data using

- Data: ST properties
- Indexing and retrieval algorithms
- Cloud computing platforms



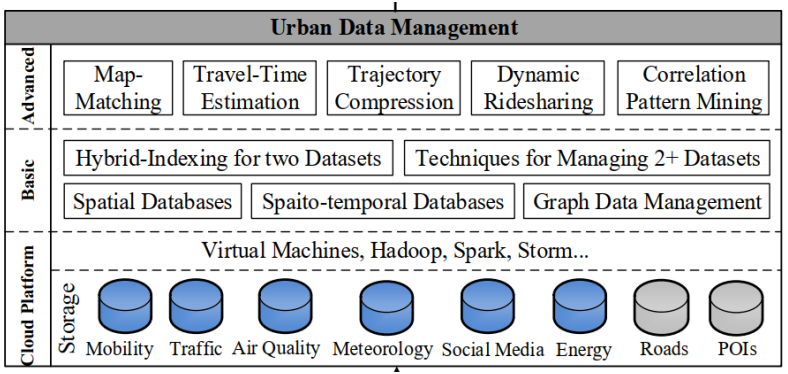
Spatio-Temporal Big Data

- Data Structures
- Spatio-Temporal (ST) Properties



Spatio-Temporal Data Management

- Basic data management techniques
  - Spatial databases
  - Spatio-temporal databases
- Advanced applications
  - Map matching
  - Trajectory compression
  - Large-scale dynamic ridesharing
  - Road network recovery
- Managing spatio-temporal data on the cloud

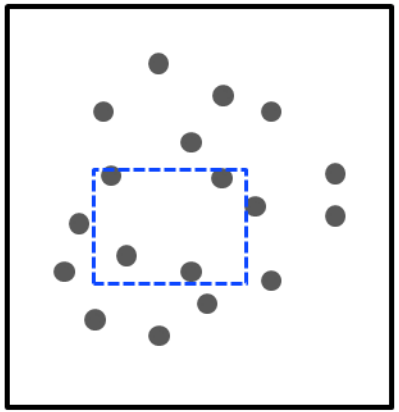


ST Data Management

- Basic techniques
  - Part 1: Spatial databases
  - Part 2: Spatio-temporal databases

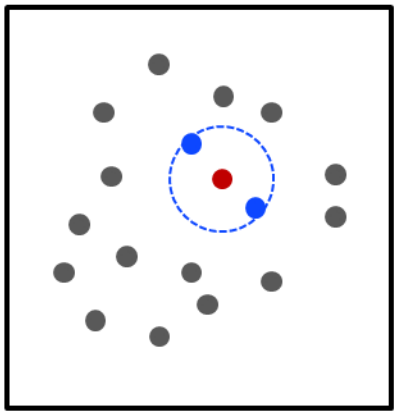
Spatial Queries

Region (Range) Query



Ask for objects that lie partially or fully inside a specified region.

Nearest Neighbour Queries



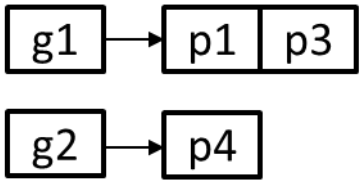
Given a point or an object, find the nearest object that satisfies given conditions

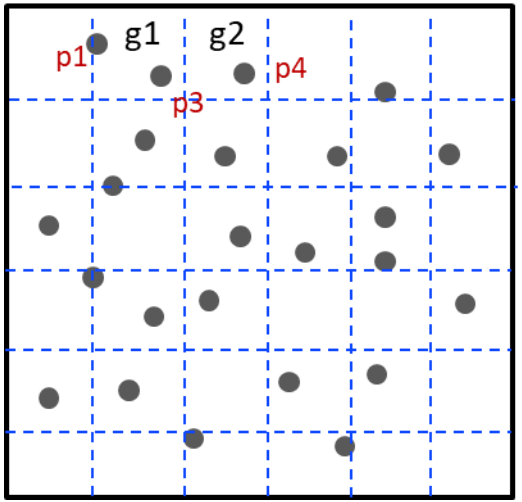
Spatial Indexing Structures

- Space Partition-Based Indexing Structures
  - Grid-based
  - Quad-tree
  - K-D tree
- Data-Driven Indexing Structures
  - R-Tree

Grid-based Spatial Indexing

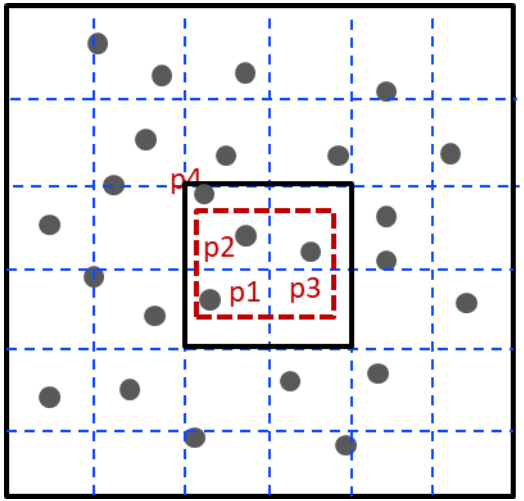
- Indexing
  - Partition the space into disjoint and uniform grids
  - Build inverted index between each grid and the points in the grid

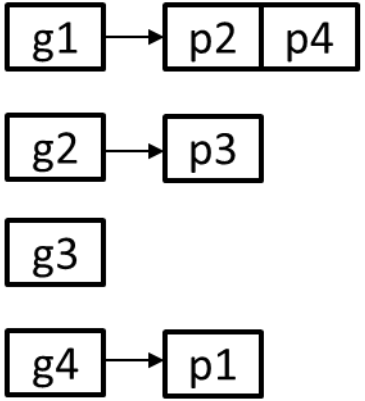




Grid-based Spatial Indexing

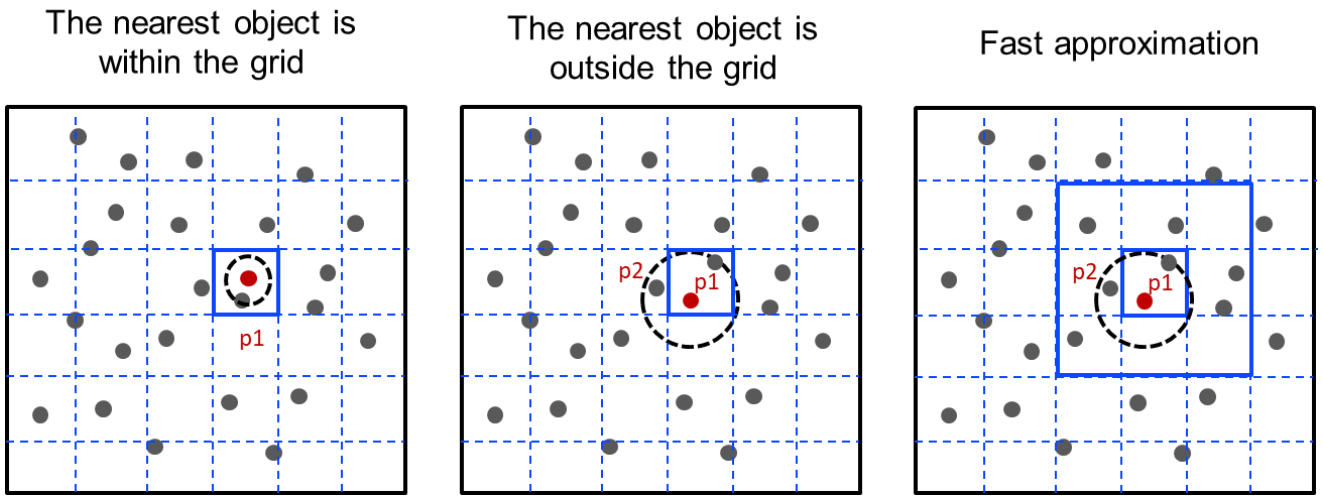
- Range Query
  - Find the grids intersecting the range query
  - Retrieve the points from the grids and identify the points in the range





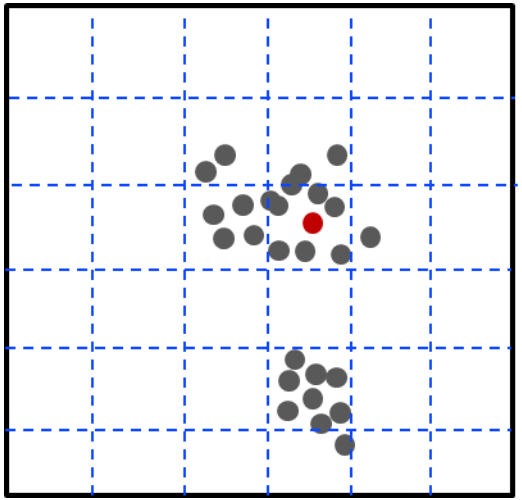
Grid-based Spatial Indexing

- Nearest neighbor query
  - Euclidian distance
  - Road network distance is quite different



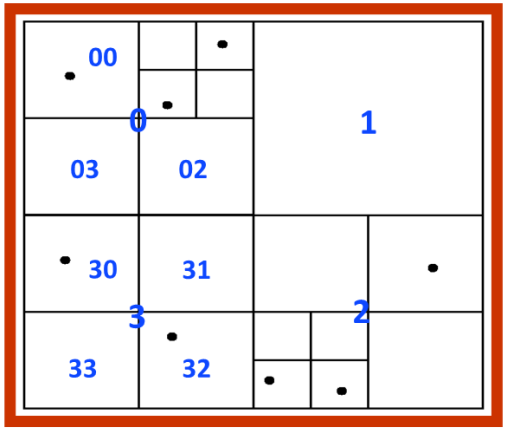
Grid-based Spatial Indexing

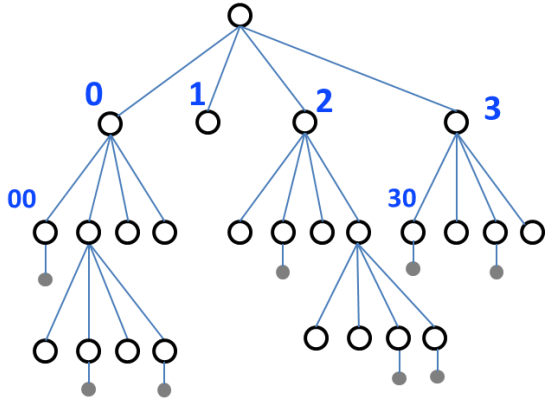
- Advantages
  - Easy to implement and understand
  - Very efficient for processing range and nearest queries
- Disadvantages
  - Index size could be big
  - Difficult to deal with unbalanced data



Quad-Tree

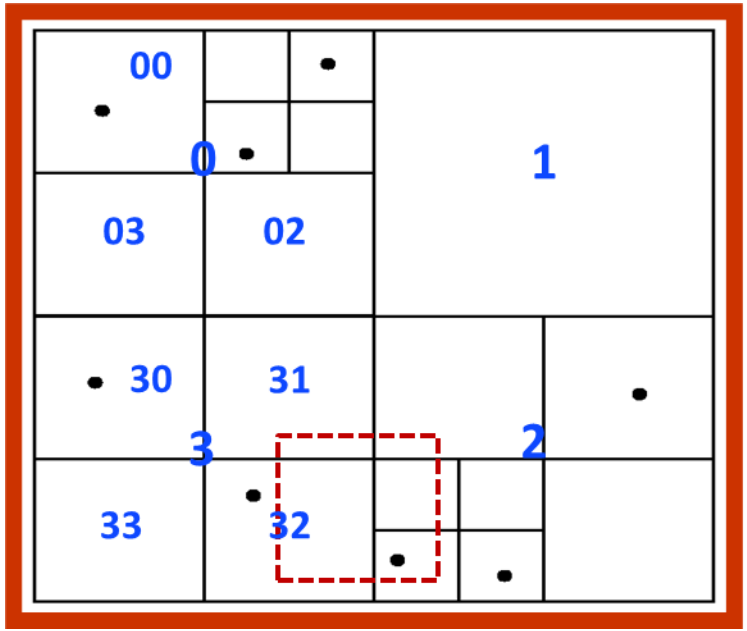
- Indexing
  - Each node of a quad-tree is associated with a rectangular region of space; the top node is associated with the entire target space
  - Each non-leaf node divides its region into four equal sized quadrants
  - Leaf nodes have between zero and some fixed maximum number of points (set to 1 in example)

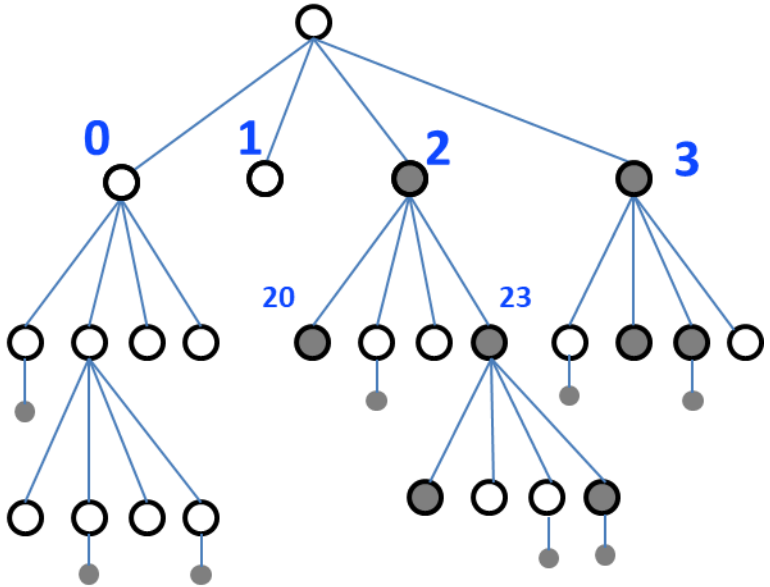




Quad-Tree

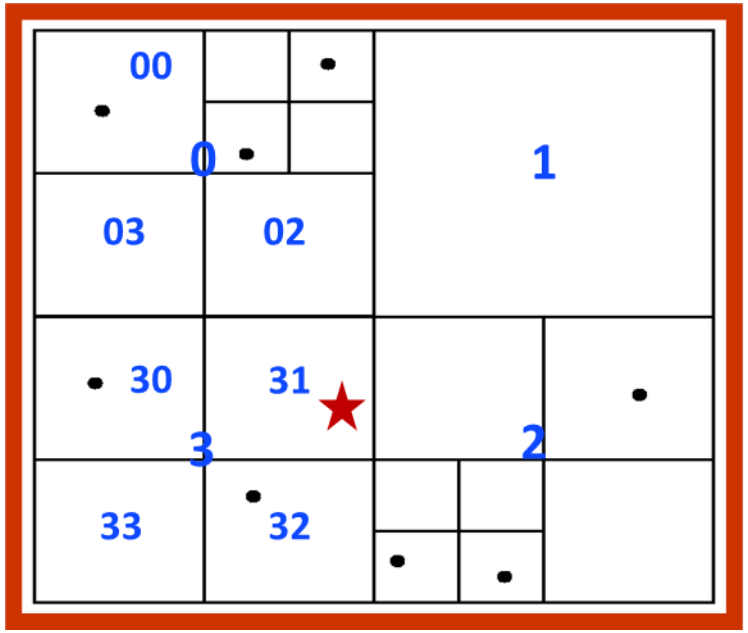
- Range query

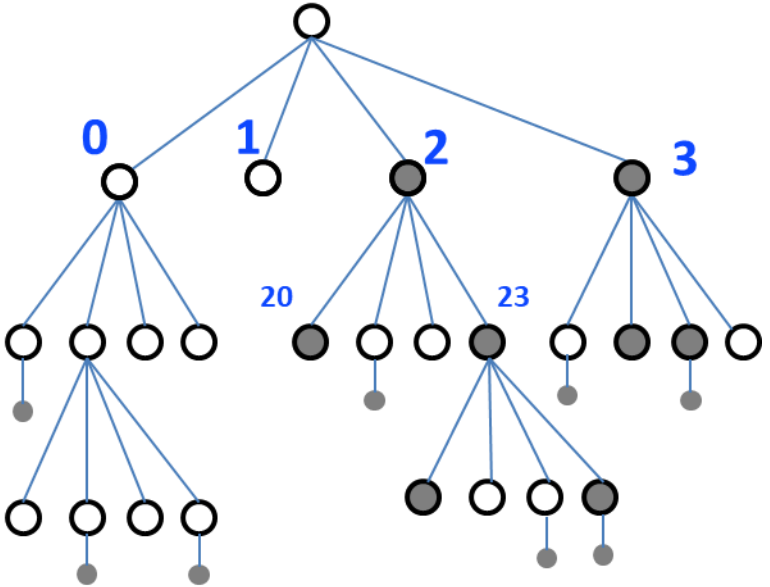




Quad-Tree

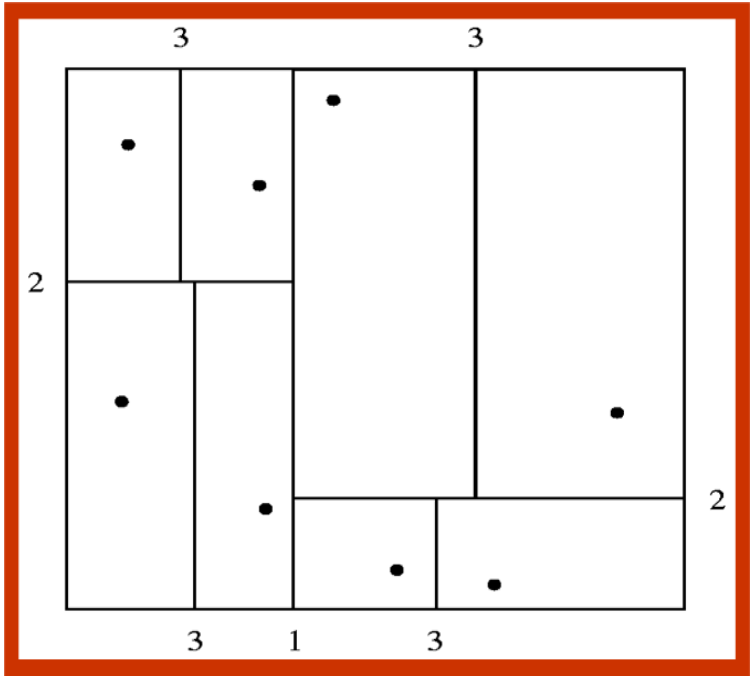
- Nearest Neighbor Query (hard)



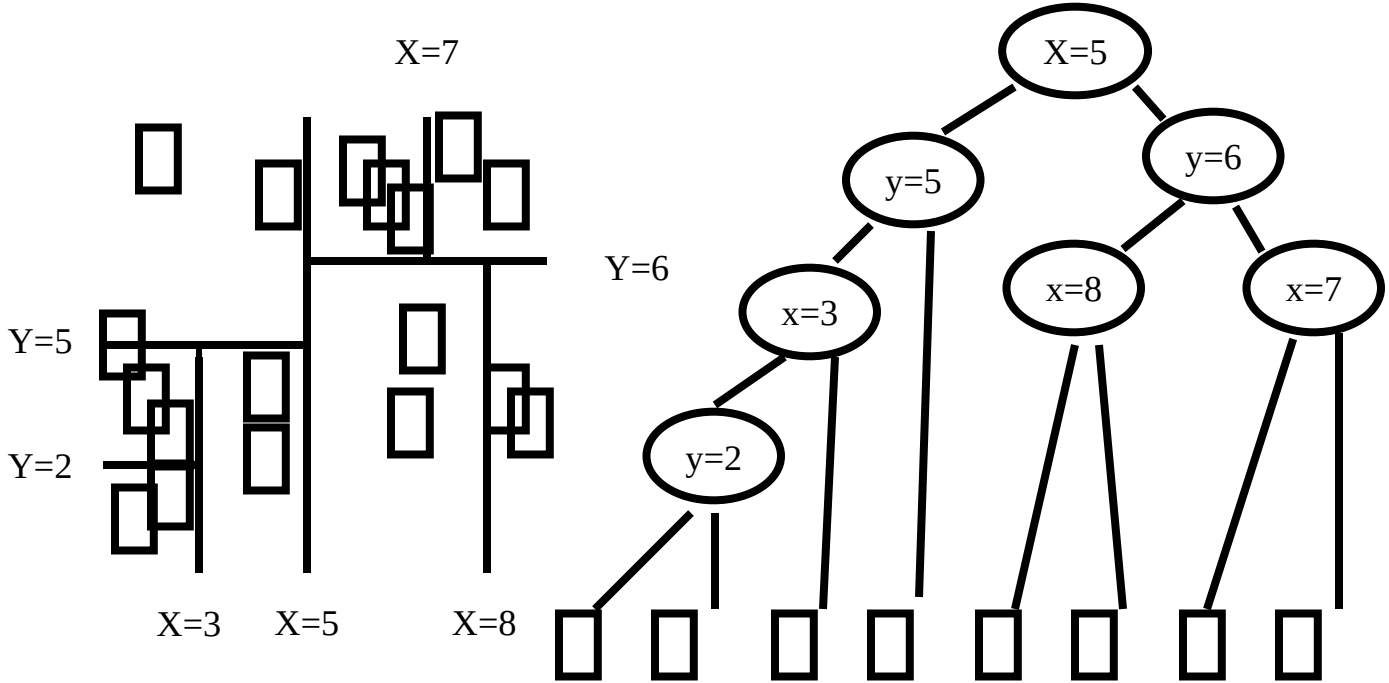


K-D-Tree

- Each line in the figure (other than the outside box) corresponds to a node in the k-d tree
  - the maximum number of points in a leaf node has been set to 1
- The numbering of the lines in the figure indicates the level of the tree at which the corresponding node appears

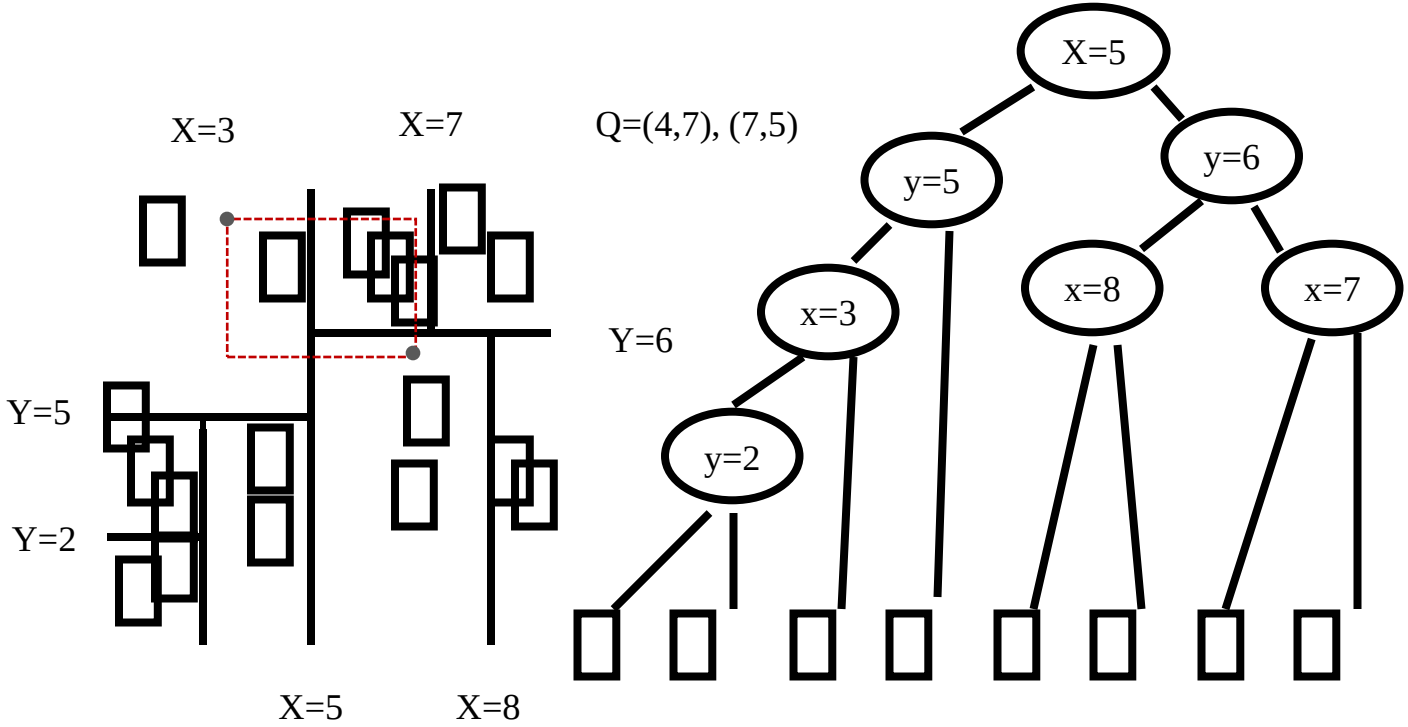


K-D-Tree Example



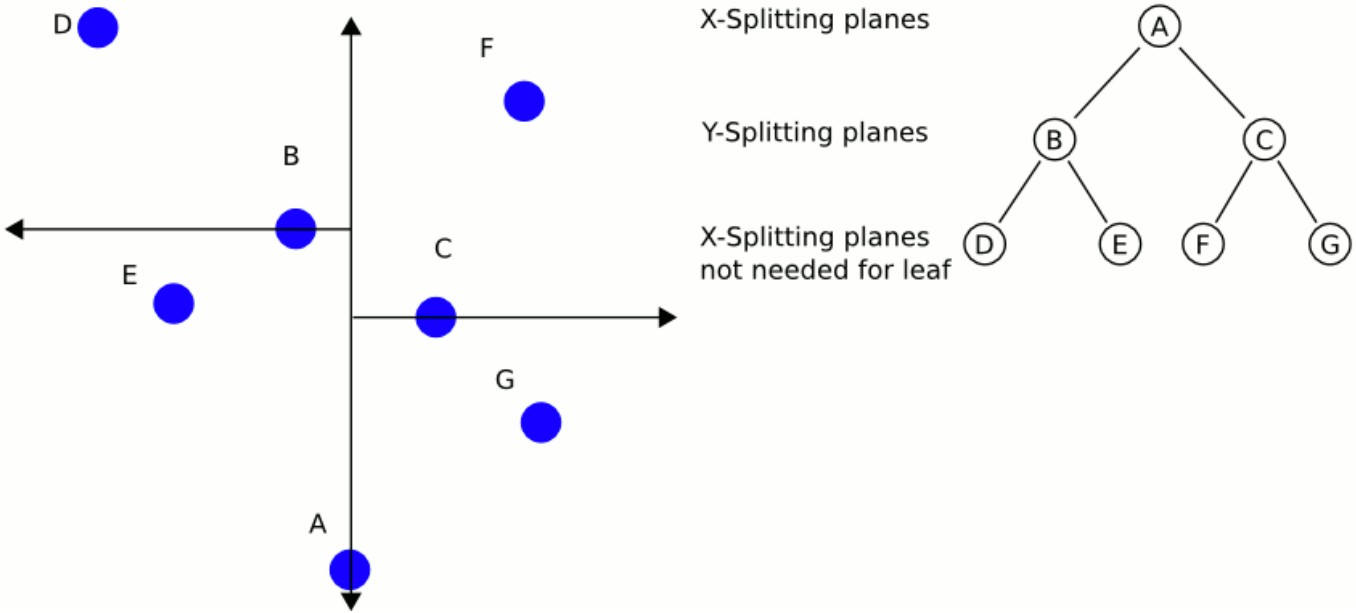
K-D-Tree Example

- Range query



K-D-Tree

- Nearest neighbor query

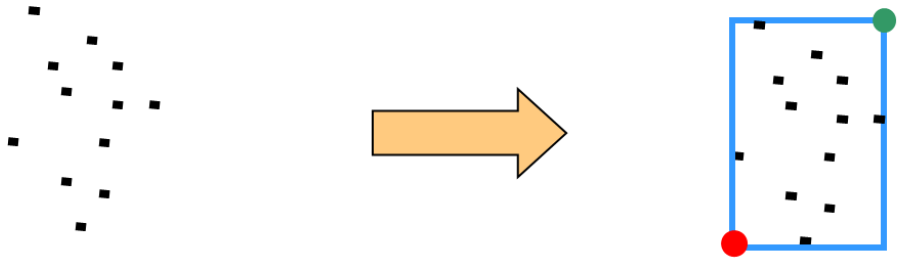


Spatial Indexing Structures

- Space Partition-Based Indexing Structures
  - Grid-based
  - Quad-tree
  - k-D tree
- Data-Driven Indexing Structures
  - R-Tree

R-Trees

- Build a Minimum Bounding Rectangle (MBR)

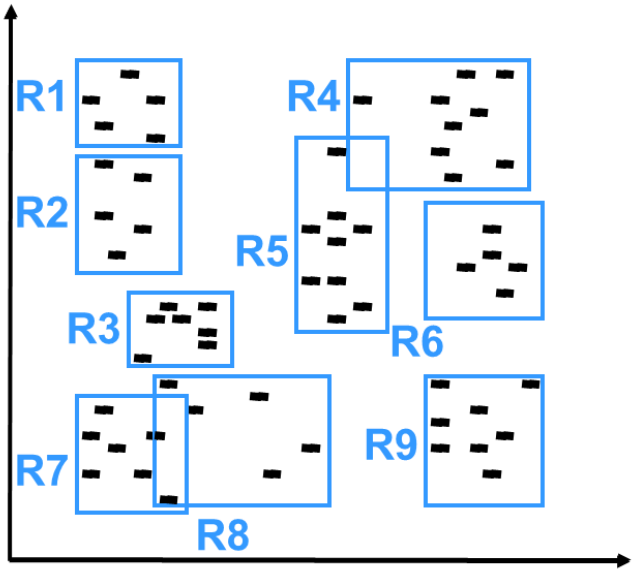


$$\mathbf{MBR} = \{(\mathbf{L.x,L.y})(\mathbf{U.x,U.y})\}$$

Note that we only need two points to describe an MBR, we typically use lower left, and upper right.

R-Trees

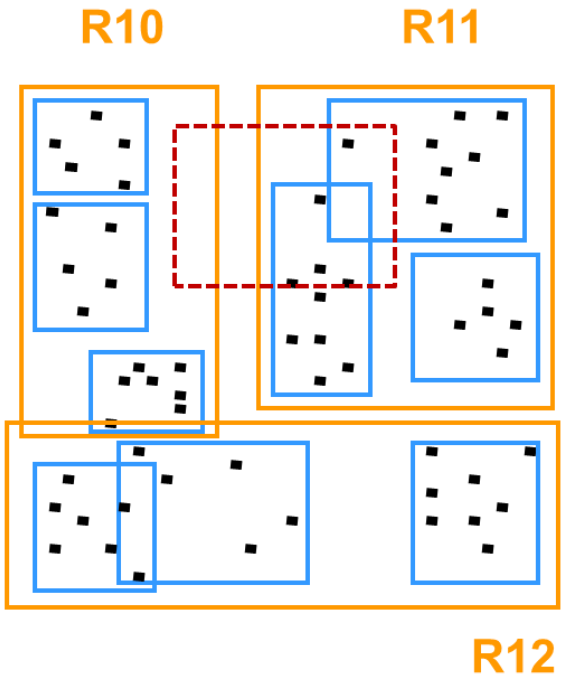
- We can group clusters of data points into MBRs
  - Can also handle line-segments, rectangles, polygons, in addition to points

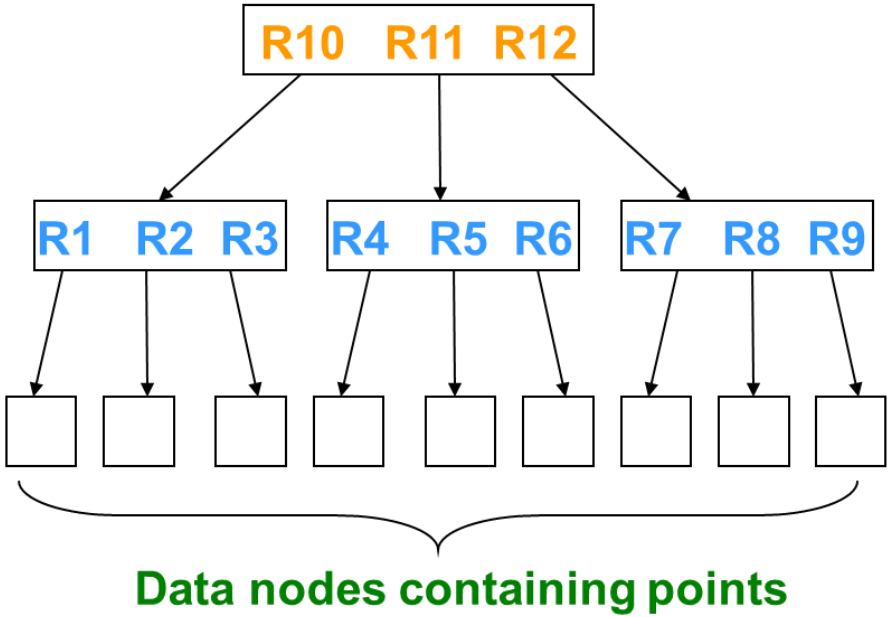


We can further recursively group MBRs into larger MBRs….

R-Tree Structure

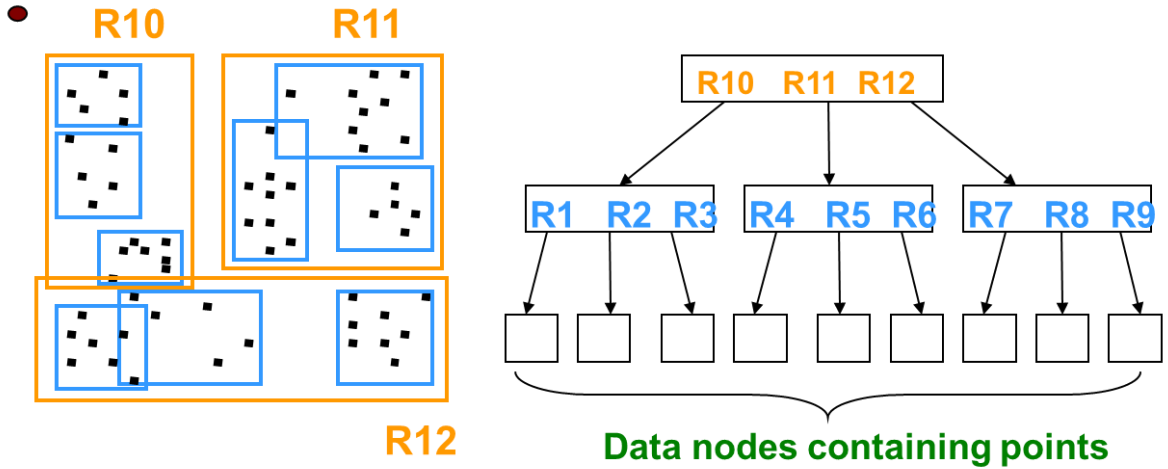
- Nested MBRs are organized as a tree





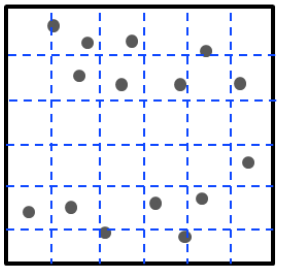
Nearest Neighbour Search

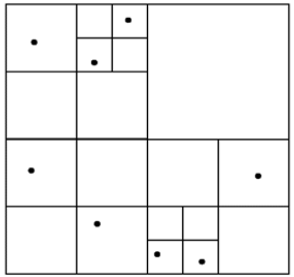
- Given an MBR, we can compute lower bounds on nearest object
- Once we know there is an item within some distance d, we can prune away all items/MBRs at distance $> d$
  - Even if we haven't actually found the nearest item yet
  - Similar technique possible for k-d trees and quad-trees as well

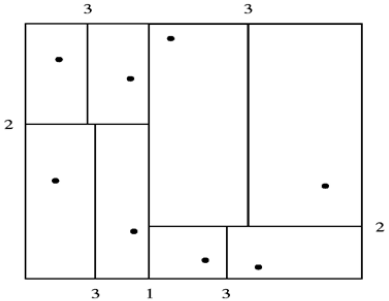


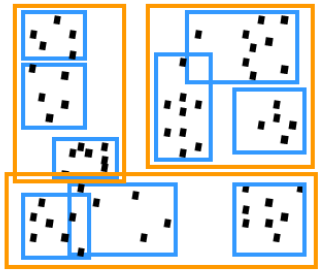
Comparison among Spatial Indices

	Unbalanced data	Range query	Nearest neighbor	Construction	Balanced structure	Storage
Grid-based	Poor	Good	Normal	Easy	Yes	Big
Quad-Tree	Good	Best	Poor	Easy	No	Median
KD-Tree	Good	Normal	Good	Easy	Almost	Median
R-Tree	Good	Normal	Best	Difficult	Yes	Small









ST Data Management

- Basic techniques
  - Part 1: Spatial databases
  - Part 2: Spatio-temporal databases

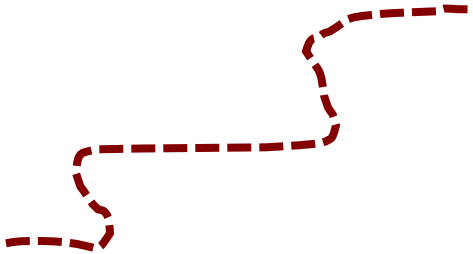
Part 2: Spatio-Temporal Databases

- Many moving objects
- Current status: (x, y, t)
- Past: a trajectory (x_1, y_1, t_1), (x_2, y_2, t_2),…,

Spatio-Temporal Databases

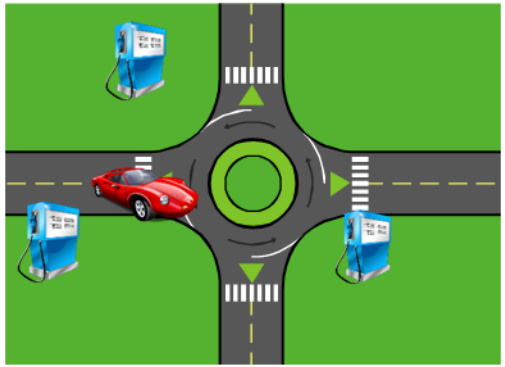
- Querying current status of a moving object
  - Scenarios:
    - Query point is moving but objects fixed
    - Query point is stationary but query objects are moving
    - Both query points and objects are moving
  - Type of queries: KNN and Range queries

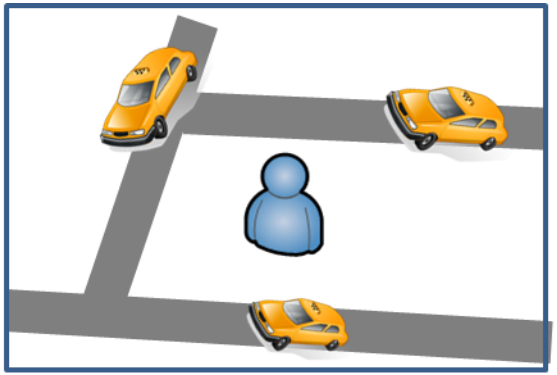
- Query the past of a of a moving object
  - Trajectory management
  - KNN and Range Queries

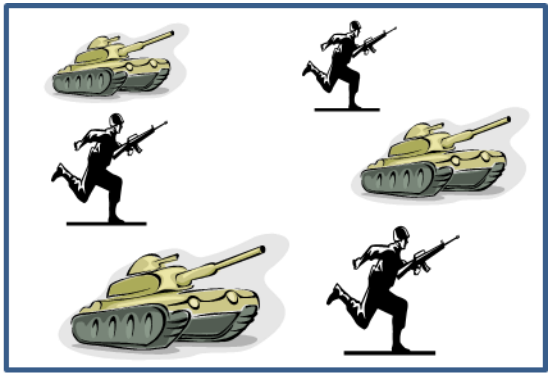


Querying current status of a moving object

- Examples:
  - Show me the closest gas station when I am driving
  - Tell me the top-2 closest vacant taxis that would pass me
  - How many moving soldiers are around a moving tank?





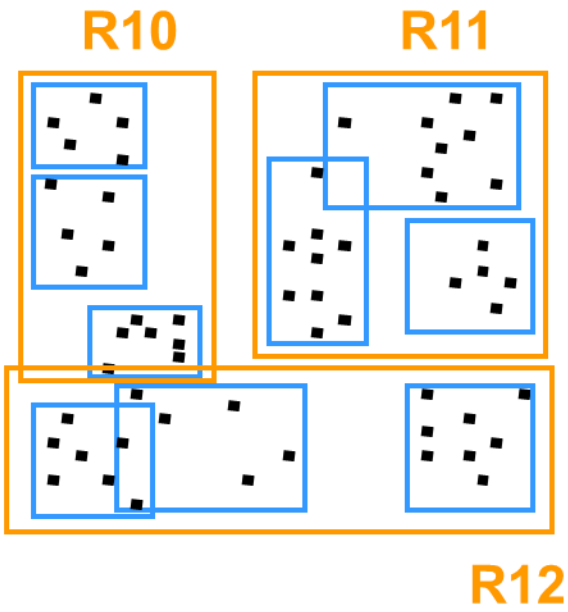


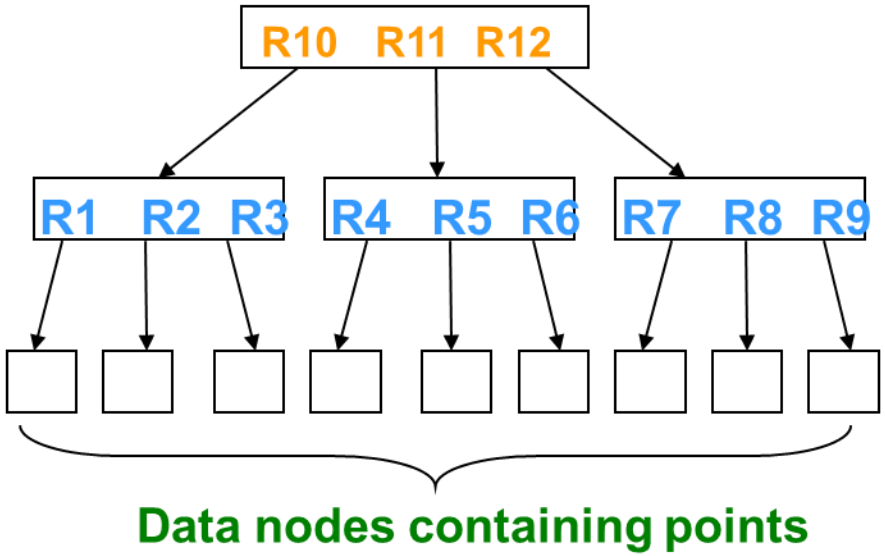
Querying Current Status of a Moving Object

- KNN and Range queries
- Instantaneous and continuous
- Indexing structures
  - View temporal as an additional dimension
    - 3D R-Tree
    - ST R-Tree
    - TB-Tree
  - Multiple version-based
    - HR-tree
    - MR-tree
    - HR+-tree
    - MV3R-tree

Index for the Query of Moving Objects

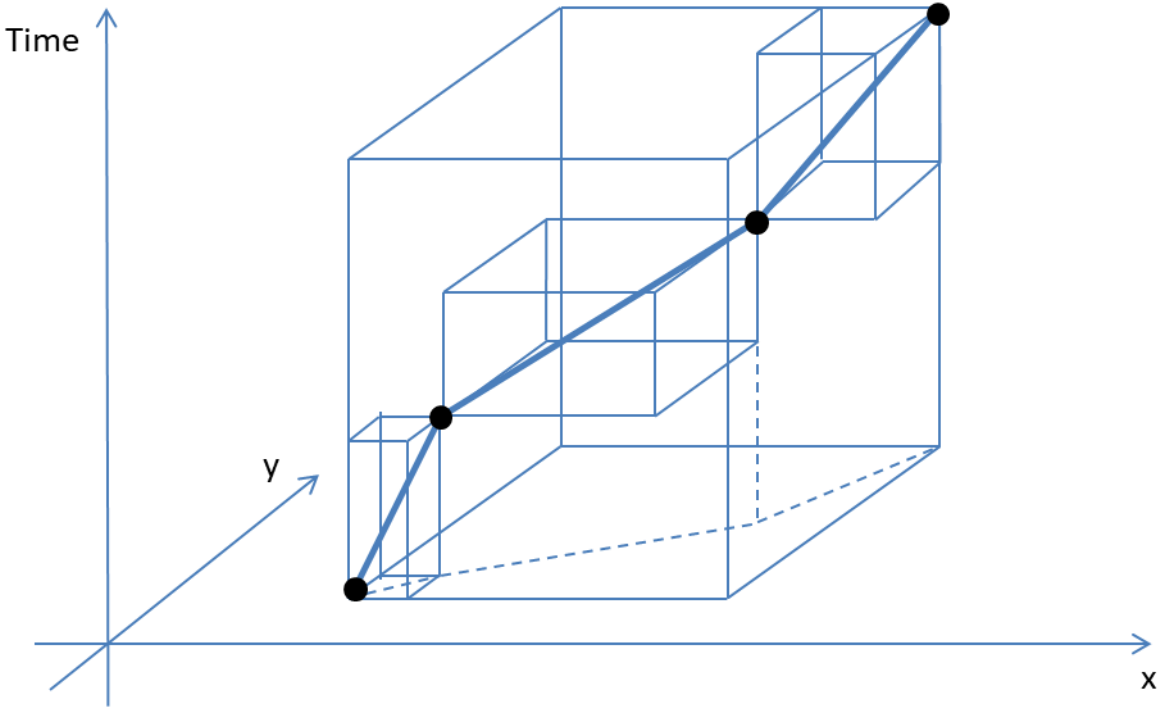
- R-Tree





Index for the Query of Moving Objects

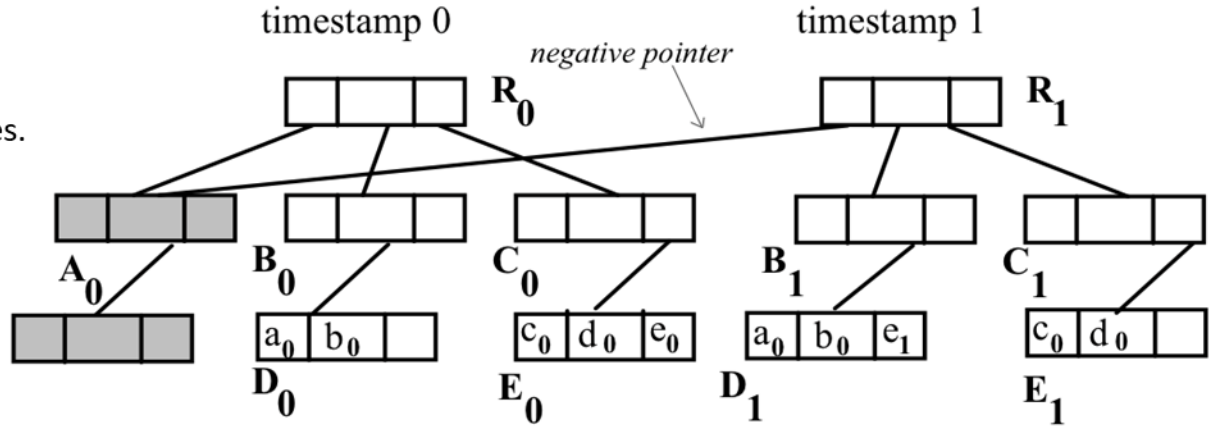
- 3D R-tree



Index for the Query of Moving Objects

- Multi-version R-tree (HR-tree [Tao2001a], HR+-tree[Tao2001b], MR-tree[Xu2005])

For each timestamp, an R-tree is created. So, there are many R-trees. These R-trees are indexed.

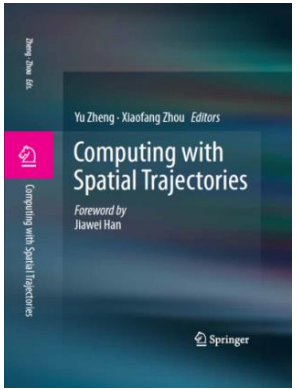


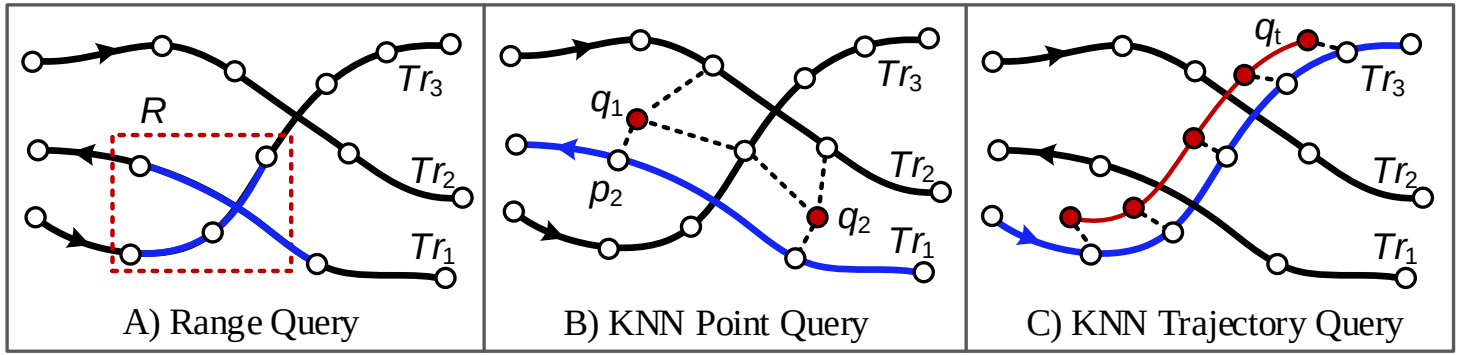
HR-tree [Tao2001]

Query for trajectories in a given region and in a given time interval:
1. The R-tree at the timestamp is found first
2. The trajectories in the specified region are retrieved from the R-tree.

Trajectory Data Management

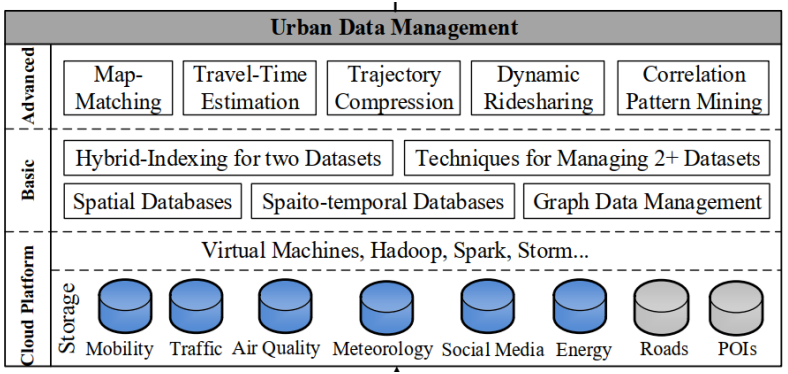
- (ST) Range queries
- KNN queries
  - Point KNN queries
  - Trajectory KDD queries





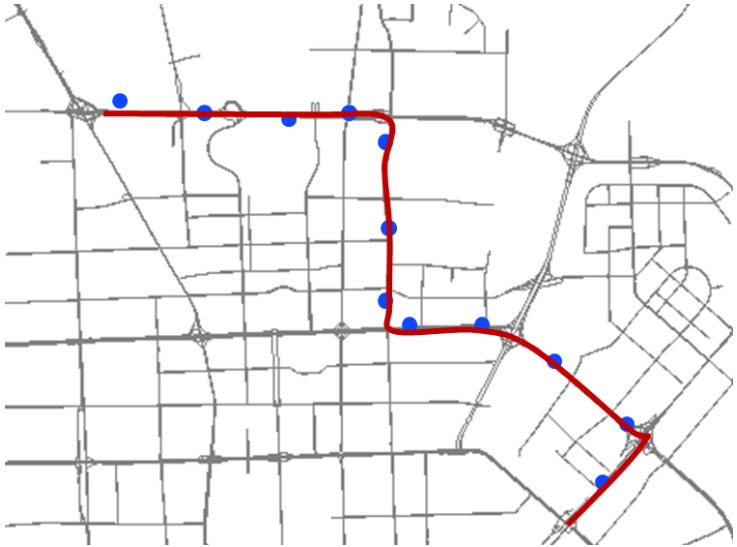
Urban Data Management

- Basic Data management Techniques
  - Spatial Databases
  - Spatio-Temporal Databases
- Advanced applications
  - Map Matching
  - Trajectory compression
  - Large-Scale Dynamic Ridesharing
- Managing Spatio-Temporal Data on the Cloud



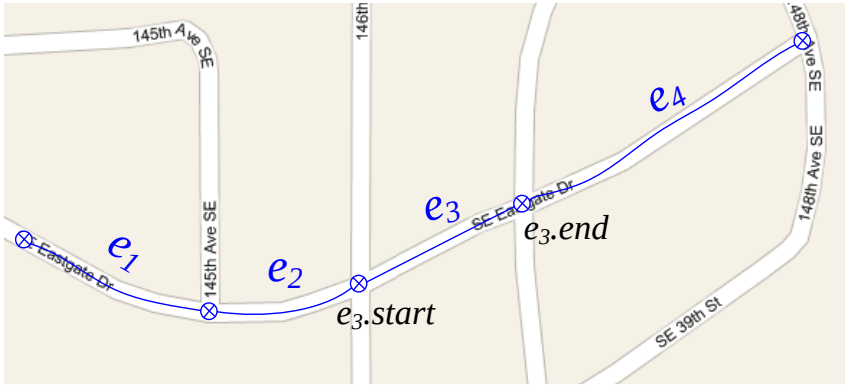
Map-matching

- Problem
  - Map a GPS trajectory onto a road network
  - a sequence of GPS points → a sequence of road segments



Spatial Data

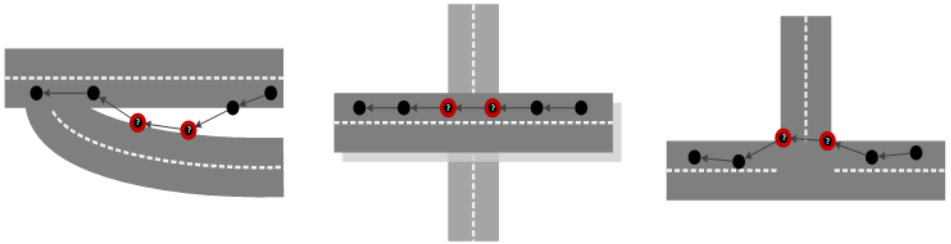
- Road network: G=(V, E)
  - V is a set of nodes
  - E is a set of road segments
  - $e \in E$, consists of two terminal nodes and a sequence of intermediate points describing the segment with a polyline
  - Properties: $e.len,\ e.dir,\ e.lanes$



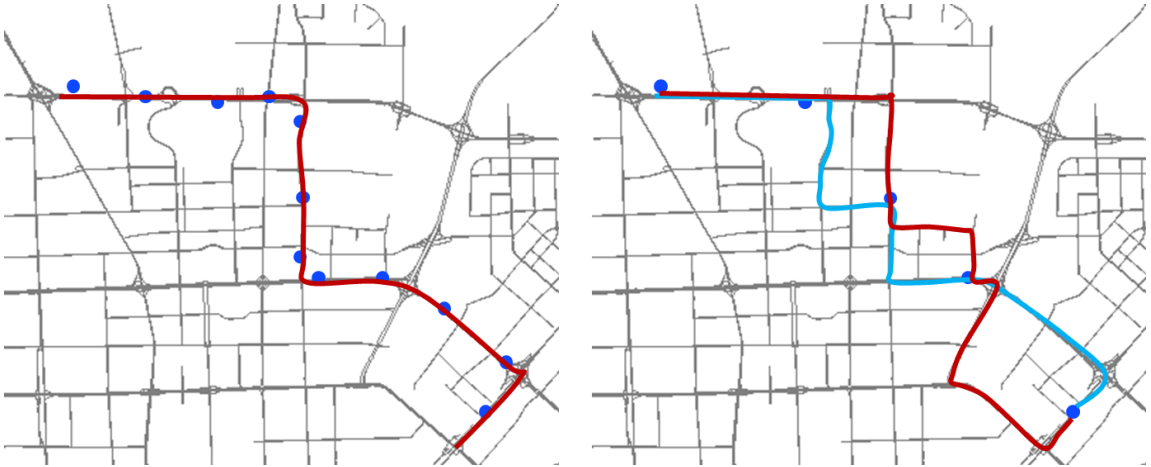
Map-Matching

- Why it is important
  - A fundamental step in many transportation applications
    - Navigation and driving
    - Traffic analysis
    - Taxi dispatching and recommendations
  - Examples:
    - Find the vehicles passing No.1 Duxue Road
    - Calculate the average travel time from HKUST(GZ) to Dongchong
    - When will the NO. 33 bus arrive at HKUST(GZ) stop?
    - ….

Why difficult

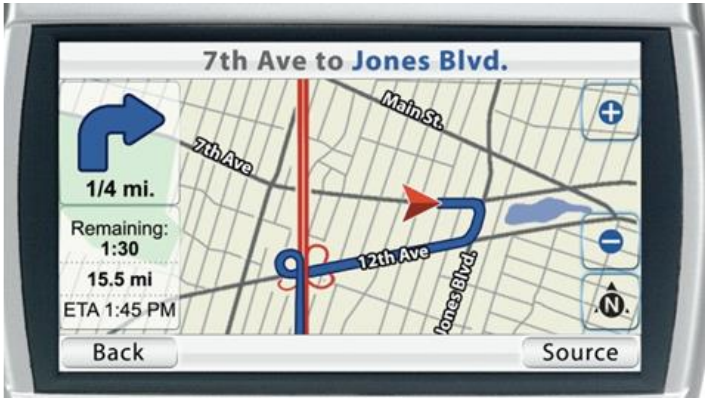


(a) Parallel roads b) Overpass c) Spur



Map-Matching

- Simple solution for high-sampling-rate data
  - Weighted distance



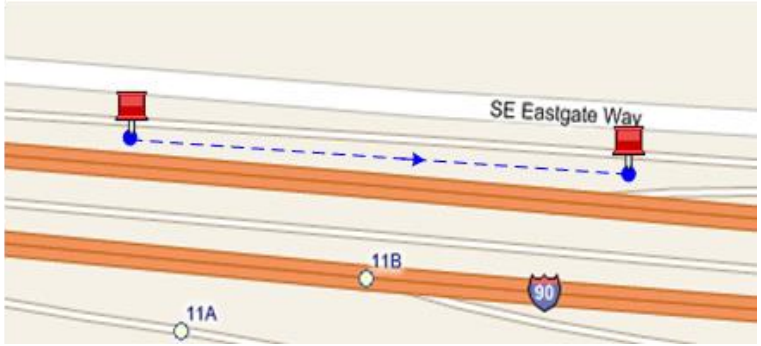
Yin Lou, Chengyang Zhang, Yu Zheng, et al. Map-Matching for Low-Sampling-Rate GPS Trajectories. In ACM SIGSPATIAL GIS 2009

Map-Matching

- According to the additional information used
  - Geometric
  - Topological
  - Probabilistic
  - Advanced techniques
- According to the range of sampling points
  - Local/incremental
  - Global
  - Advanced

Map-matching

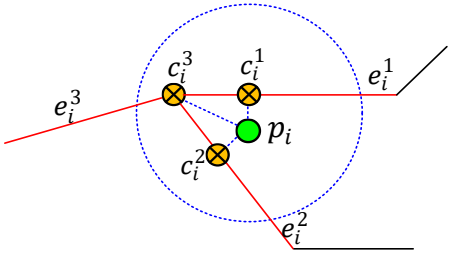
- Insights
  - Consider both local and global information
  - Incorporating both spatial and temporal features



Yin Lou, Chengyang Zhang, Yu Zheng, et al. Map-Matching for Low-Sampling-Rate GPS Trajectories. In ACM SIGSPATIAL GIS 2009

Map-matching

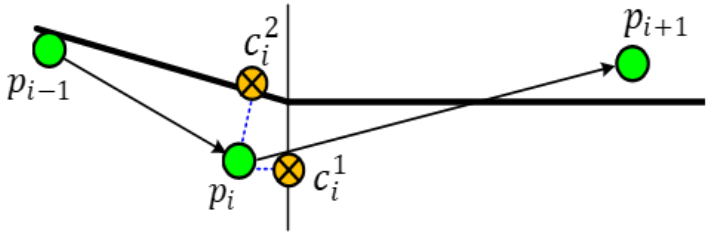
- Solution (incorporating spatial information)
  - Model local possibility



$$N\left(c_i^j\right) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i^j-\mu)^2}{2\sigma^2}}$$

where $x_i^j = dist(c_i^j, p_i)$ is the distance between p_i and c_i^j.

  - Considering context (global)



$$V\left(c_{i-1}^t \rightarrow c_i^s\right) = \frac{d_{i-1\rightarrow i}}{w_{(i-1,t)\rightarrow(i,s)}}$$

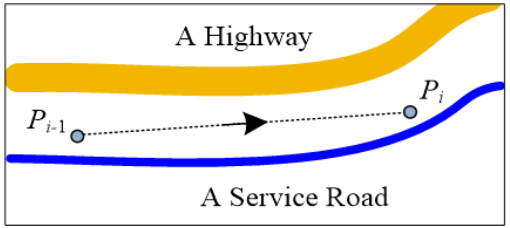
where $d_{i-1\rightarrow i} = dist(p_i, p_{i-1})$ is the Euclidean distance between p_i and p_{i-1}, and $w_{(i-1,t)\rightarrow(i,s)}$ is the length of shortest path from c_{i-1}^t to c_i^s.

$$F_s\left(c_{i-1}^t \rightarrow c_i^s\right) = N(c_i^s) * V\left(c_{i-1}^t \rightarrow c_i^s\right), \qquad 2 \le i \le n$$

Map-matching

- Solution
  - Considering temporal information

$$F_t\left(c_{i-1}^t \rightarrow c_i^s\right) = \frac{\sum_{u=1}^{k}\left(e'_u.v \times \bar{v}_{(i-1,t)\rightarrow(i,s)}\right)}{\sqrt{\sum_{u=1}^{k}(e'_u.v)^2} \times \sqrt{\sum_{u=1}^{k}\bar{v}^2_{(i-1,t)\rightarrow(i,s)}}}$$



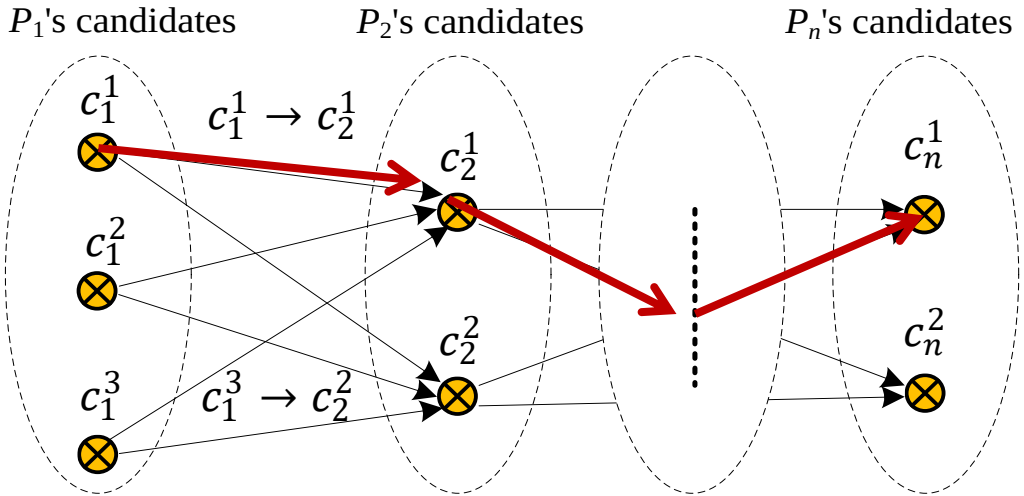
$$\bar{v}_{(i-1,t)\rightarrow(i,s)} = \frac{\sum_{u=1}^{k} l_u}{\Delta t_{i-1\rightarrow i}}$$

Map-matching

- Aggregating
  - Spatial and temporal information
  - Local and global information

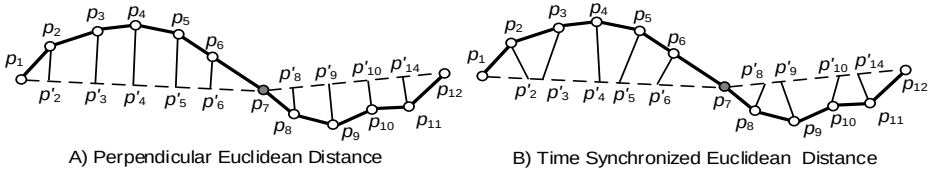
$$F\left(c_{i-1}^{t} \rightarrow c_{i}^{s}\right) = F_s\left(c_{i-1}^{t} \rightarrow c_{i}^{s}\right) * V\left(c_{i-1}^{t} \rightarrow c_{i}^{s}\right)$$

- Dynamic programing



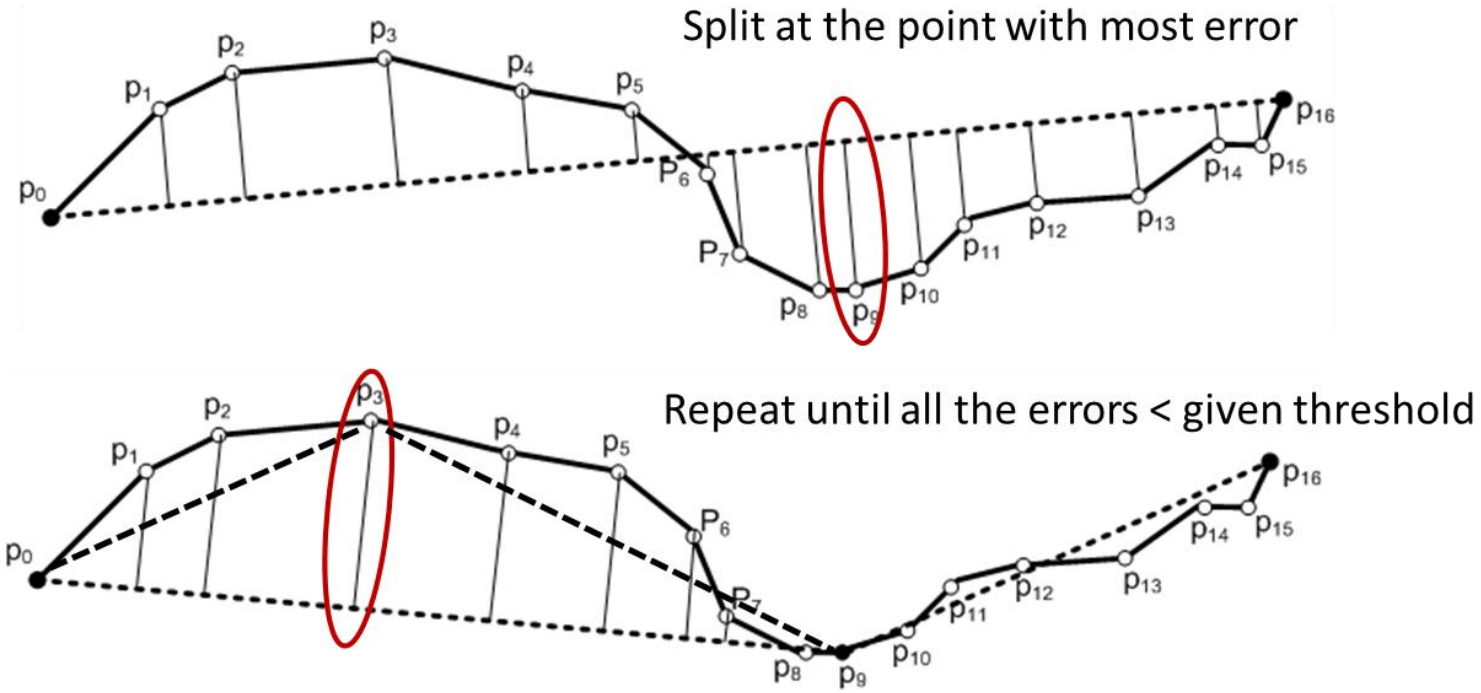
Trajectory Compression

- Two categories of compression
  - Offline compression (a.k.a. batch mode)
    - Reduces the size of trajectory after the trajectory has been fully generated
    - Applications: content sharing sites, such as Everytrail and Bikely
  - Online compression
    - Compressing a trajectory instantly as an object travels
    - Applications: traffic monitoring and fleet management
    - Algorithms: Sliding Window, Open Window, and safe area-based methods
- Two distance metrics
  - Perpendicular Euclidean Distance
  - Time Synchronized Euclidean Distance



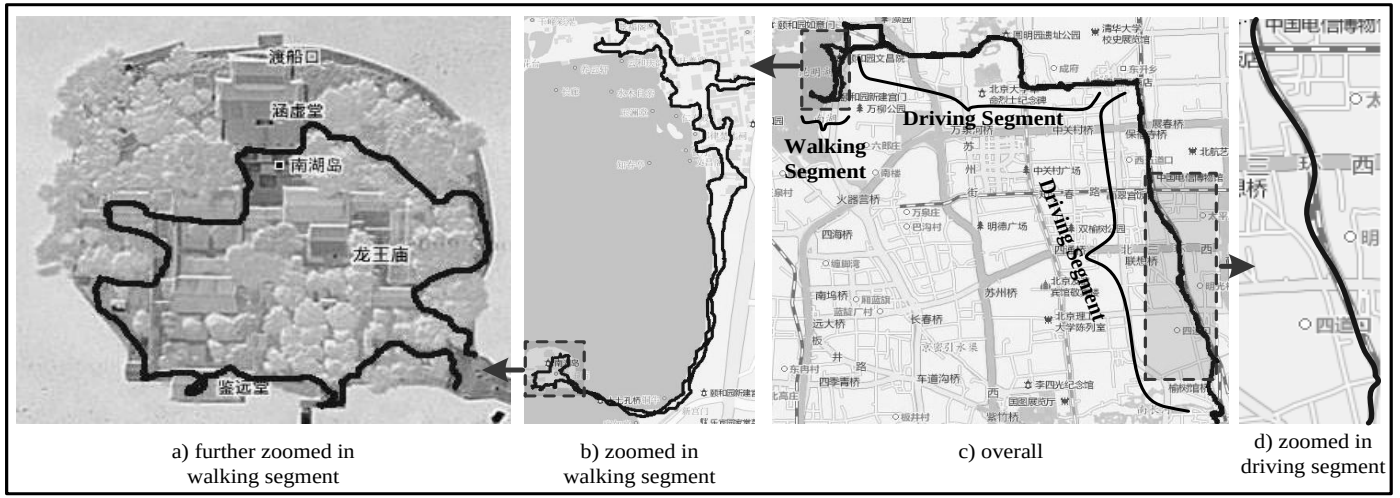
Offline Trajectory Compression

- Example



Semantic Meaning-Based Methods (TS)

- Keep the semantic meanings of a trajectory
- Some special points would be more significant
  - a user stayed, took photos, or changed direction greatly



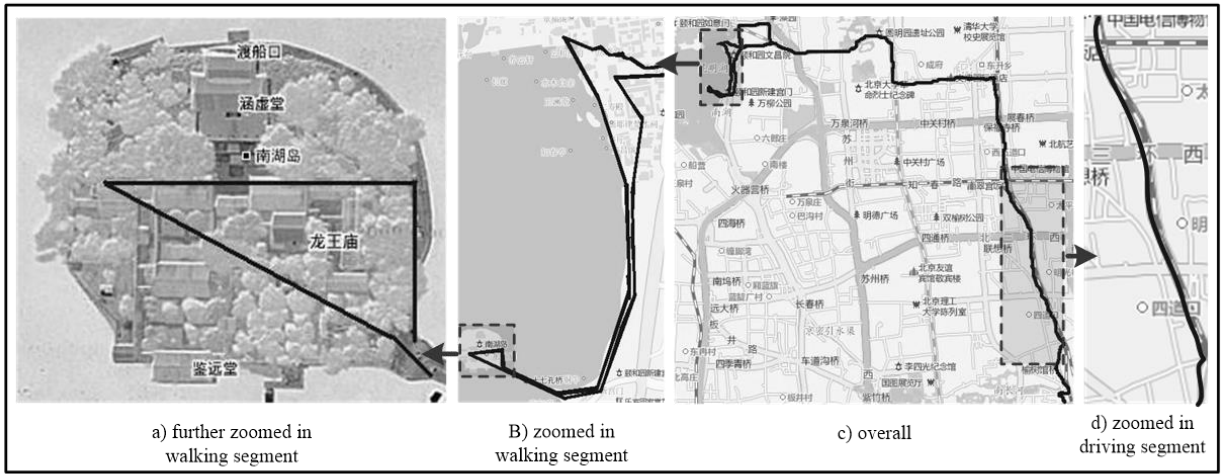
a) further zoomed in walking segment

b) zoomed in walking segment

c) overall

d) zoomed in driving segment

Douglas-Peucker algorithm

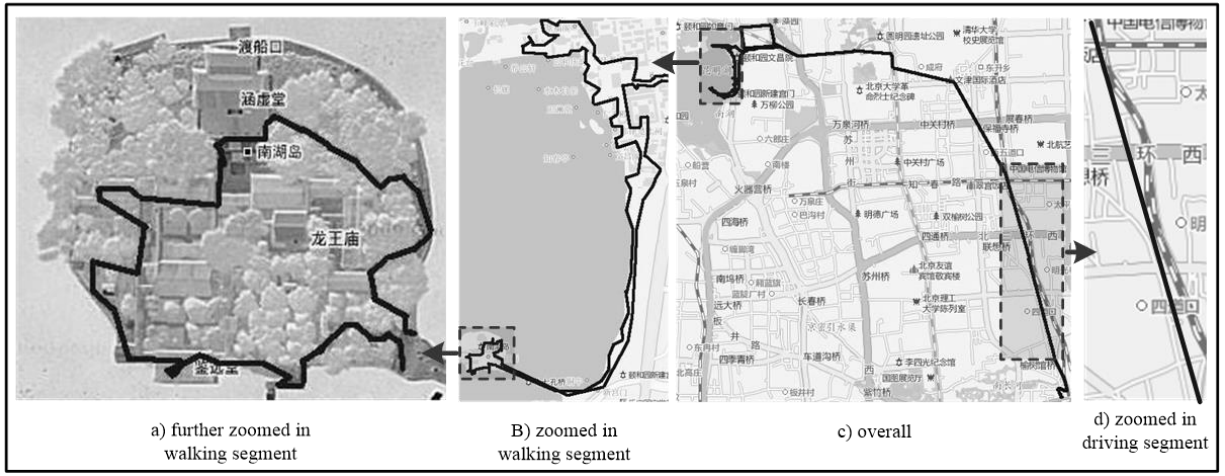


a) further zoomed in walking segment

B) zoomed in walking segment

c) overall

d) zoomed in driving segment



a) further zoomed in walking segment

B) zoomed in walking segment

c) overall

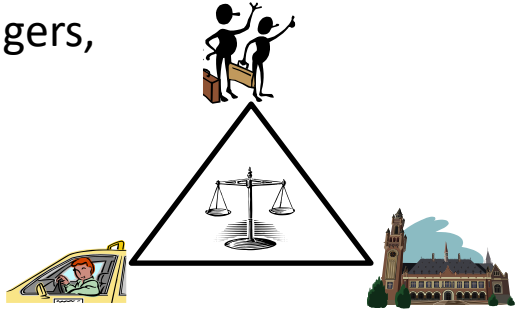
d) zoomed in driving segment

T-Share: A Large-Scale Dynamic Taxi Ridesharing Service

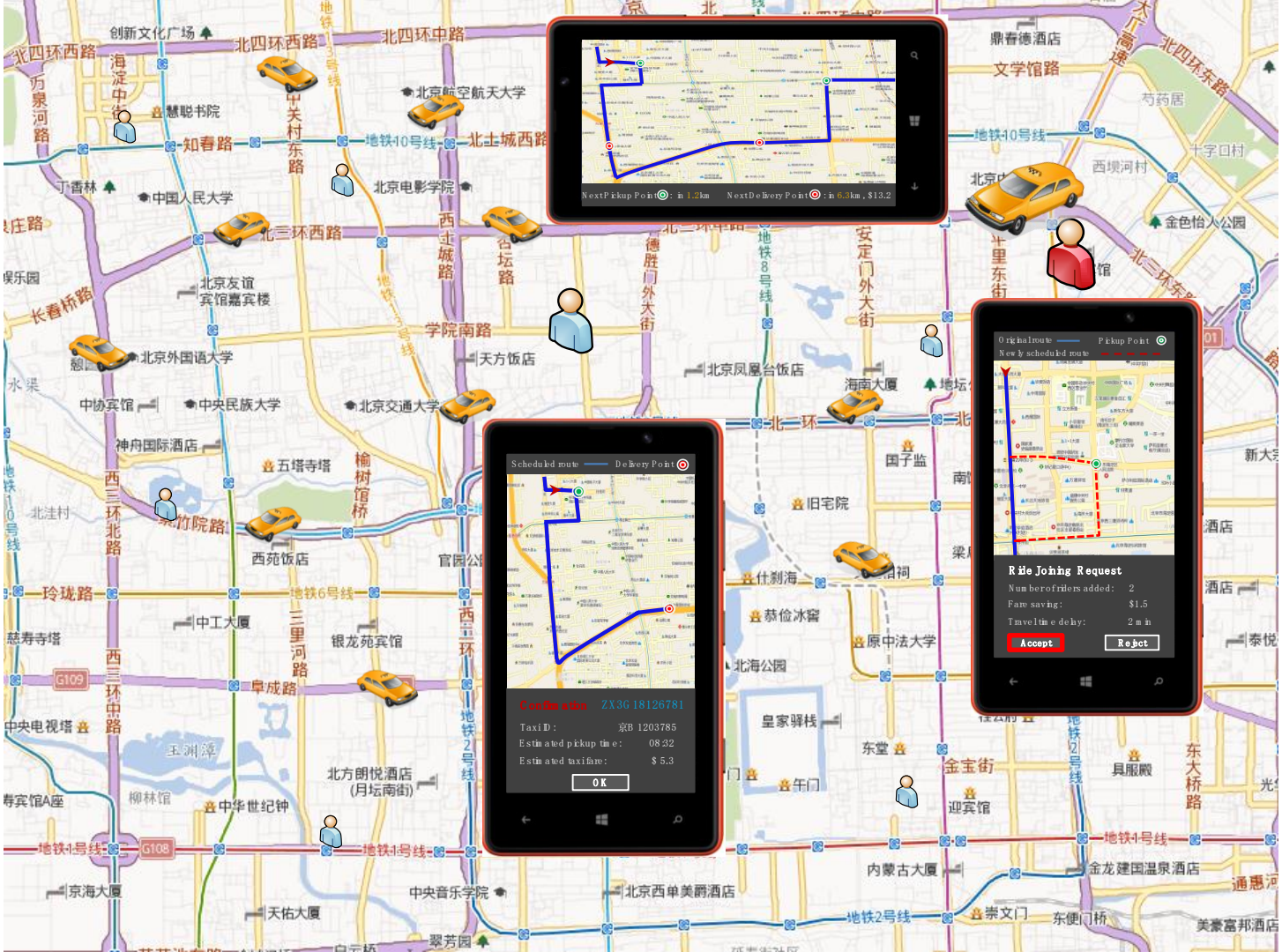
ICDE 2013 Best Paper Runner-Up Award

Difficult to take a taxi!

- A social problem among passengers, taxi drivers and government

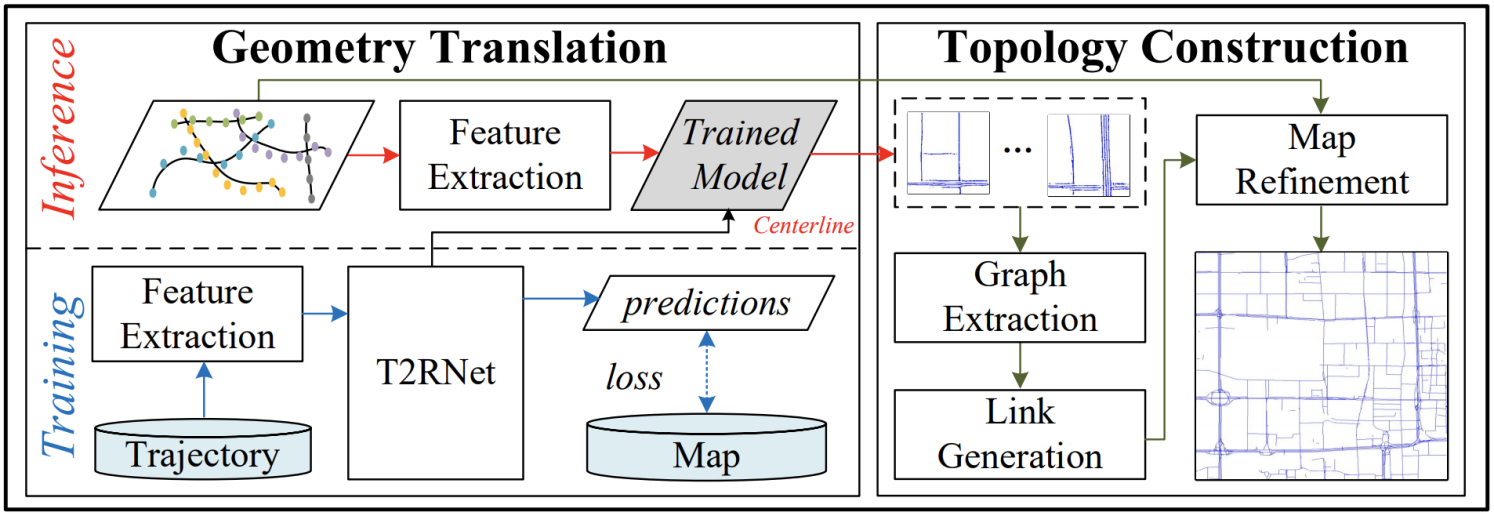


- Possible Solutions
  - Increasing taxis？
  - Taxi dispatching？？

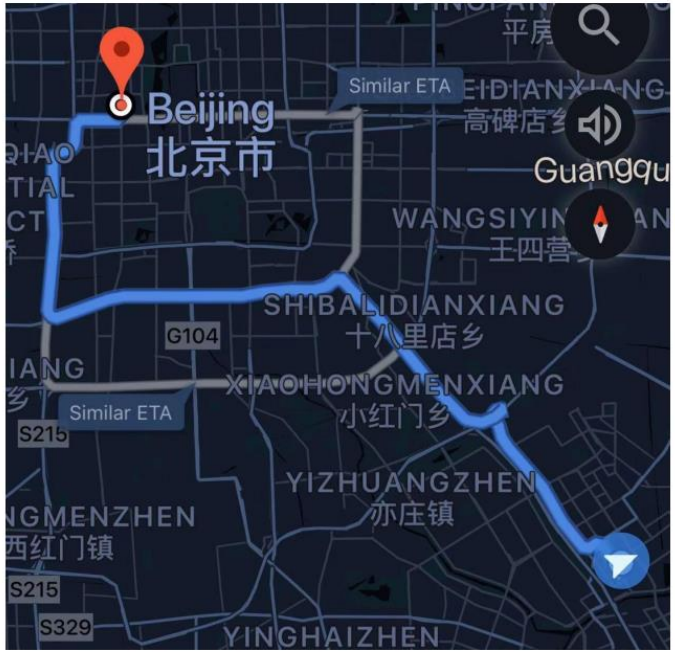

NextPickup Point : in 1.2km
NextDelivery Point : in 6.3km , $13.2
Scheduled route
Delivery Point
Taxi ID :
京B 1203785
Estimated pickup time :
08:32
Estimated taxi fare:
$ 5.3
OK
Original route
Pickup Point
Newly scheduled route
Ride Joining Request
Number of riders added:
2
Fare saving:
$1.5
Travel time delay:
2 min
Accept
Reject

Learning to Generate Maps from Trajectories

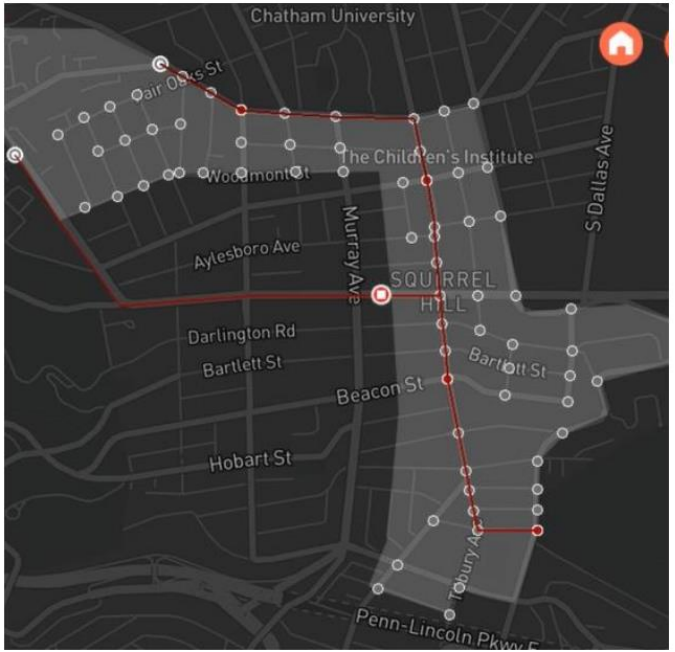
AAAI 2020



Increasing Demands of Accurate & Updated Maps

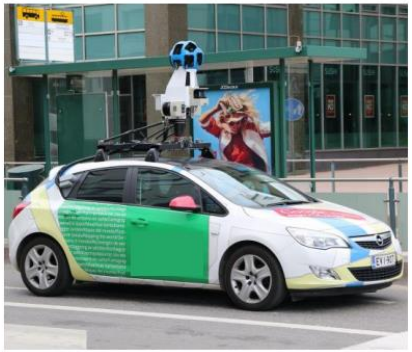


Navigation



Route Planning & Real-time Scheduling

Traditional Map Collection Methods

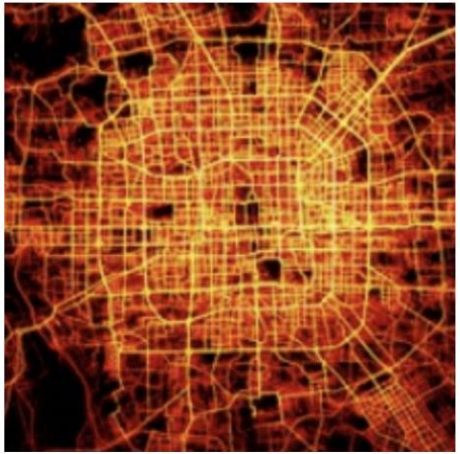


On-field Study

Low Spatial Coverage

Dynamic Traffic Status

Crowdsourcing GPS Trajectories

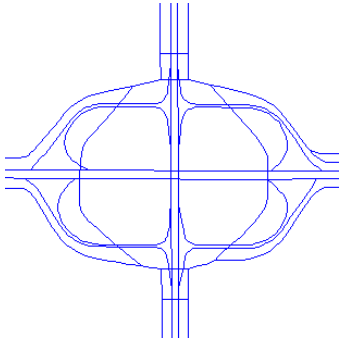


Massive Moving Objects

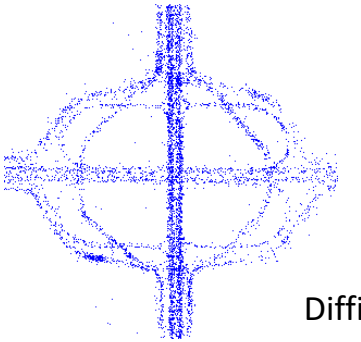
GPS Devices

Real-time City-wide Map Information

Challenges

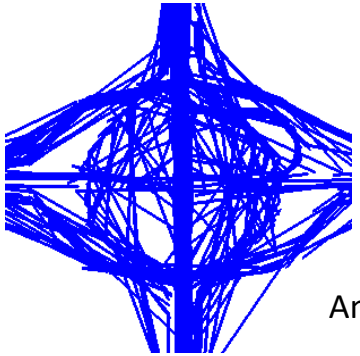


Ground Truth Map



Positioning Error (~15 meters)

Difficult to Distinguish Spatially Close Roads

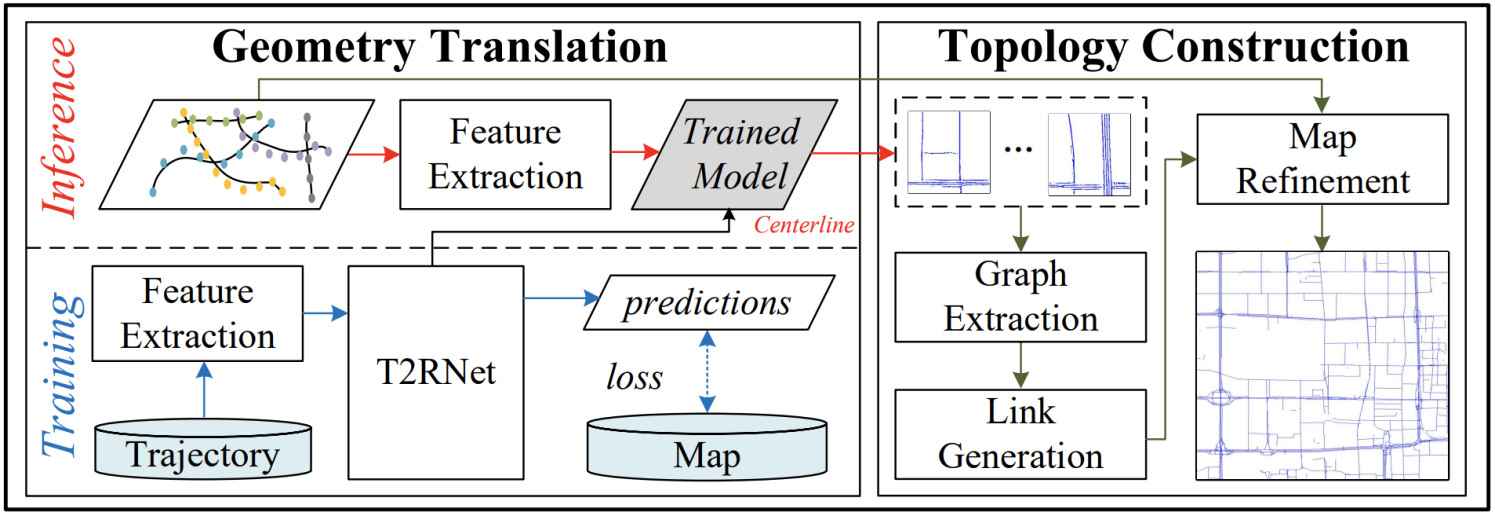


Low Sampling Rate (30s/pt)

Ambiguous Underlying Traversing Routes

Framework

- Two-stage solution (AI + DB)



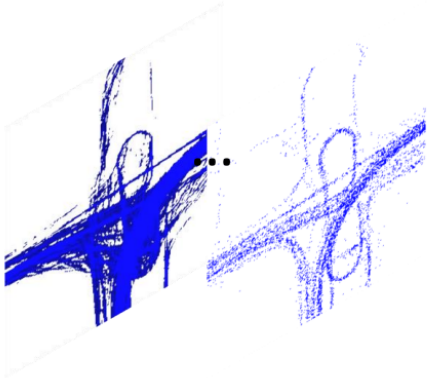
AAAI-20, New York

Ruan, Liang et al., Learning to Generate Maps from Trajectories. AAAI 2020.

Geometry Translation

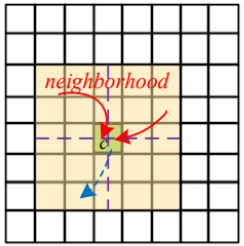
- Feature Extraction
  - For each $I \times J$ Region Tile

Spatial View ($\mathbb{R}^{11\times I\times J}$)



Point, Line, Speed, Direction (8 channels)

Transition View ($\mathbb{Z}_2^{2\times T\times T\times I\times J}$)



1	1
0	0

0	0
1	0

(a) Transitions S/E at c. (b) Local Incoming Matrix. (c) Local Outgoing Matrix.

Each grid cell has an incoming and an outgoing matrix

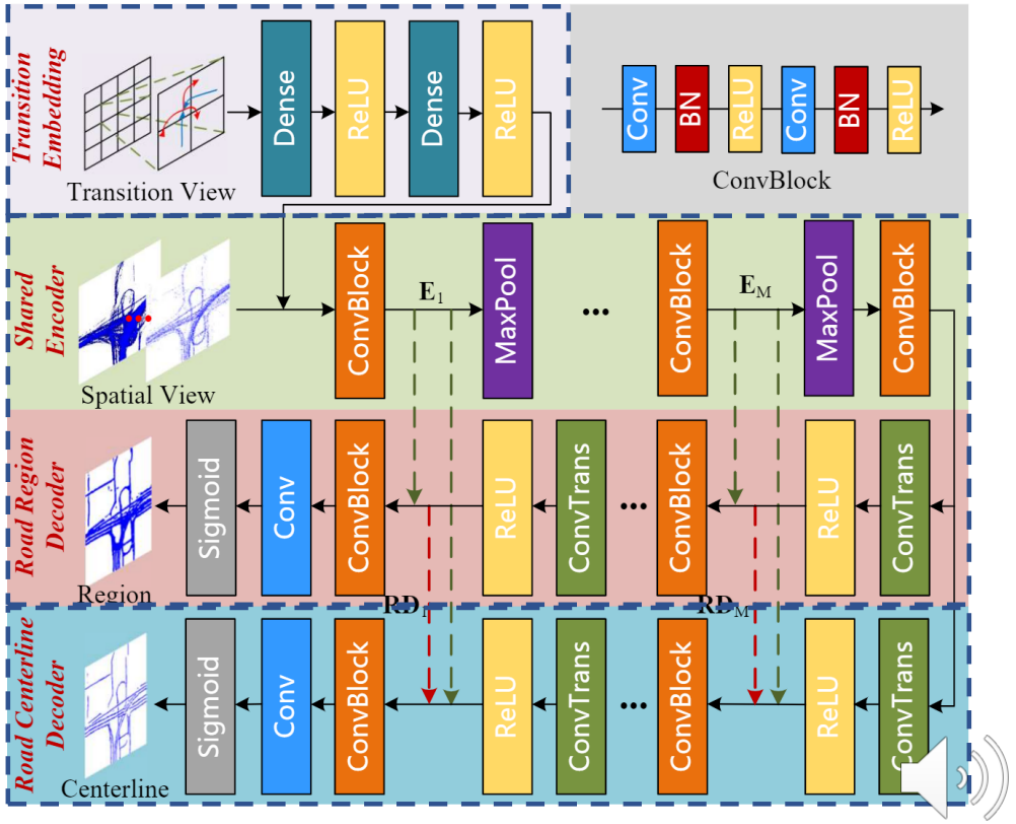
Geometry Translation

- T2RNet
  - Transition Embedding
  - Shared Encoder
  - Road Region Decoder
  - Road Centerline Decoder
- Optimization
  - Dice Loss [Milletari F, et al. 2016]

$$\mathcal{L}_{Dice}(\hat{\mathbf{Y}}, \mathbf{Y}) = 1 - \frac{2\sum_i^I \sum_j^J \hat{Y}_{ij} Y_{ij} + \epsilon}{\sum_i^I \sum_j^J \hat{Y}_{ij} + \sum_i^I \sum_j^J Y_{ij} + \epsilon}$$

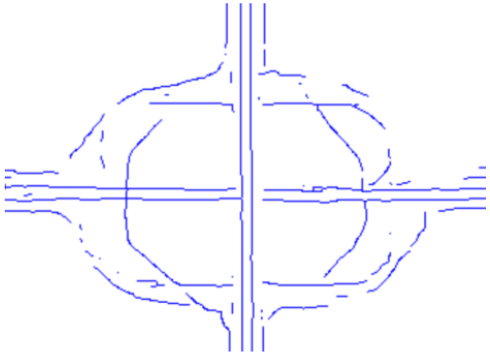
  - Multi-task Loss

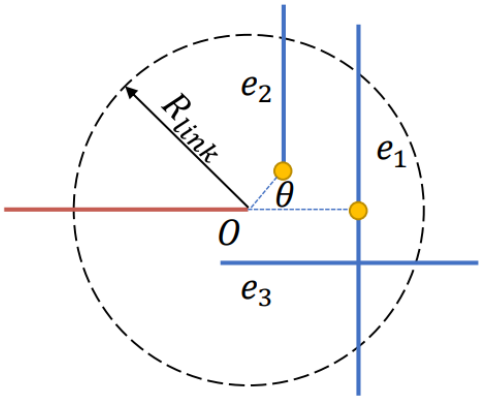
$$\mathcal{L}(\boldsymbol{\theta}) = (1-\lambda)\mathcal{L}_{Dice}(\hat{\mathbf{Y}}_c, \mathbf{Y}_c) + \lambda\mathcal{L}_{Dice}(\hat{\mathbf{Y}}_r, \mathbf{Y}_r)$$



Topology Construction

- Graph Extraction
  - Merge predicted tiles
  - Extract road segments
- Link Generation
  - For each dead end



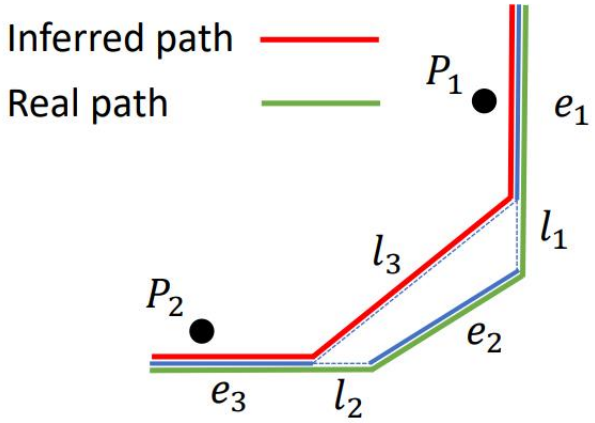


- Case 1: intersects another edge on the extension
- Case 2: has smooth transition to the closest dead end of another edge

Topology Construction

- Map Refinement
  - Perform trajectory map matching on the linked map [Yuan J, et al. 2010]
  - Remove edges and links with low support

If the map matching is directly applied…



Proposed solution

$$dist(P) = \sum_{e_i \in P} \omega(e_i) \cdot len(e_i)$$

$$\omega(e_i) = \begin{cases} \alpha, & e_i \text{ is a generated link} \quad \alpha > 1 \\ 1, & e_i \text{ is a predicted edge} \end{cases}$$

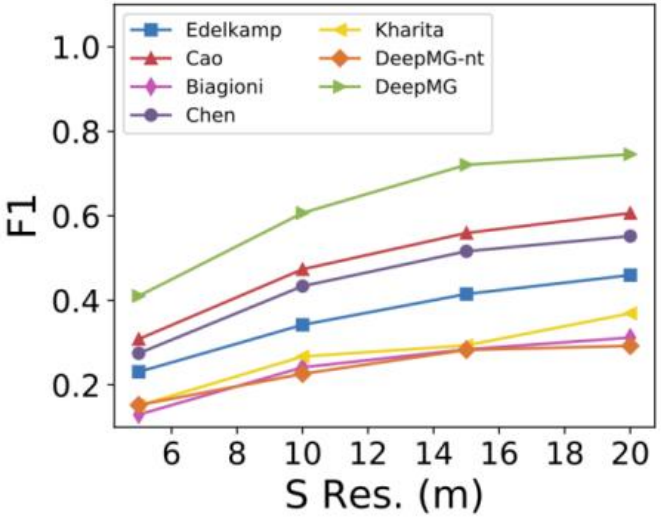
Evaluation

- Datasets
  - Trajectory
    - <oid, timestamp, latitude, longitude>
  - Map
    - Node: <latitude, longitude>
    - Edge: <start_node, end_node>

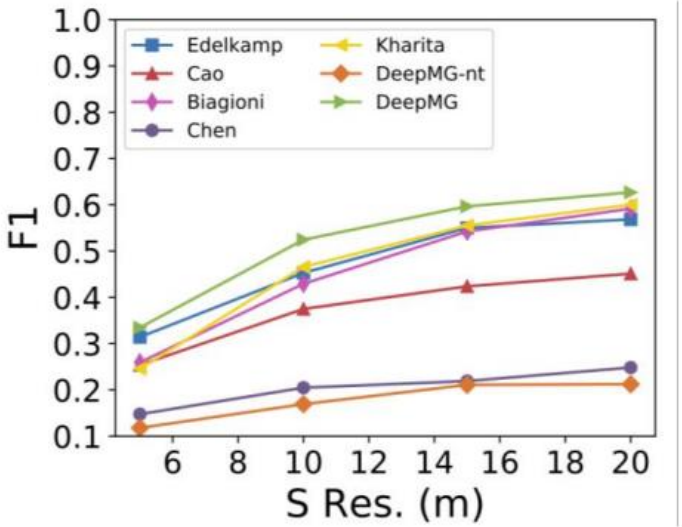
Dataset	TaxiBJ	TaxiJN
#Days	30	30
#Vehicles	500	70
Sampling Rate	~30s	~3s
Size (km^2)	16×16 (5×5)	16×26 (5×5)
#Points	3.1M (304K)	5.7M (322K)
#Trajectories	66,124 (13,462)	29,556 (3,954)
Roads (km)	2,772 (284)	2,048 (123)

- Evaluation Metrics
  - Topological F1 [Biagioni, et al. 2012]
    - Repeat N times
      a. Select a random starting location
      b. Find reachable area within a maximum radius
      c. Compare generated map with GT using F1
    - Report the average F1 score

Quantitative Comparison

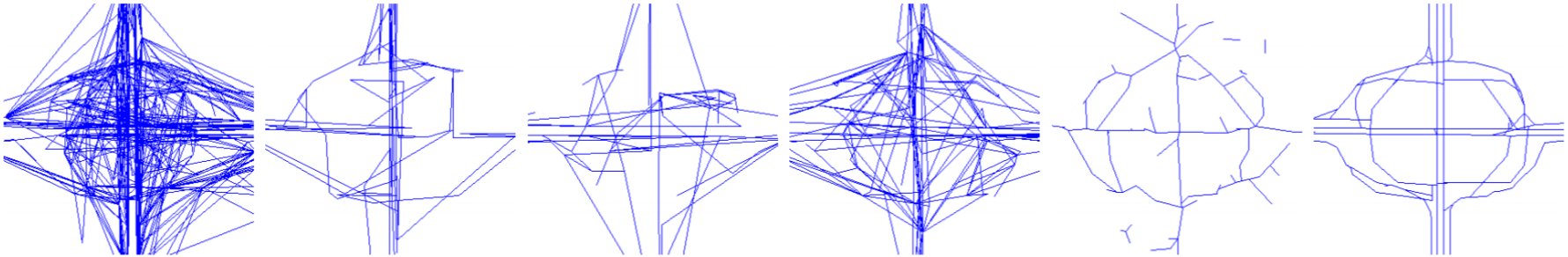


TaxiBJ (32.3% improvement)



TaxiJN (6.5% improvement)

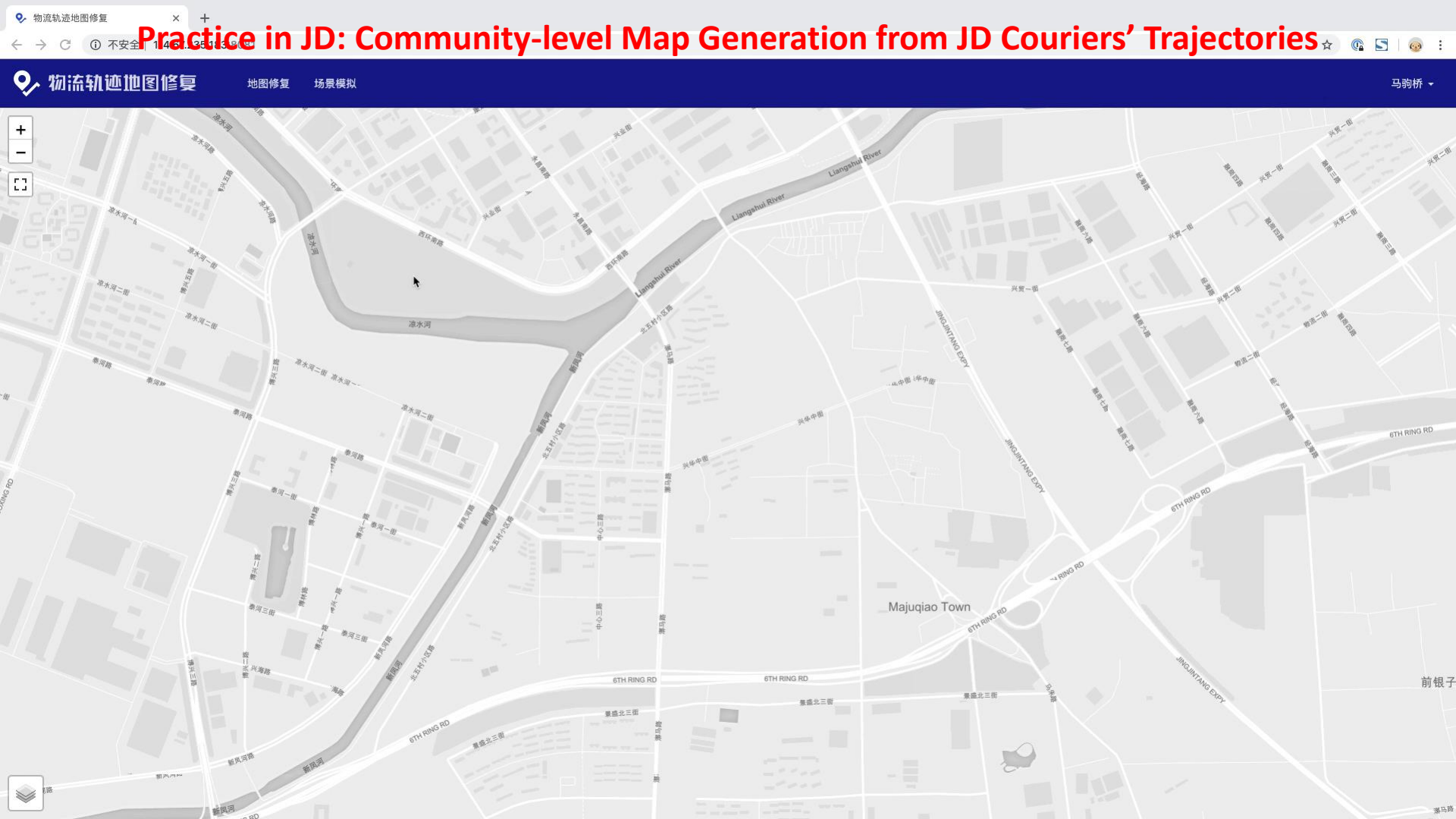
Case Study



(a) Edelkamp. (b) Chen. (c) Kharita. (d) Cao. (e) Biagioni. (f) DeepMG.

Figure 5: Seven algorithm at a roundabout near Gongzhufen area in TaxiBJ.

Practice in JD: Community-level Map Generation from JD Couriers' Trajectories

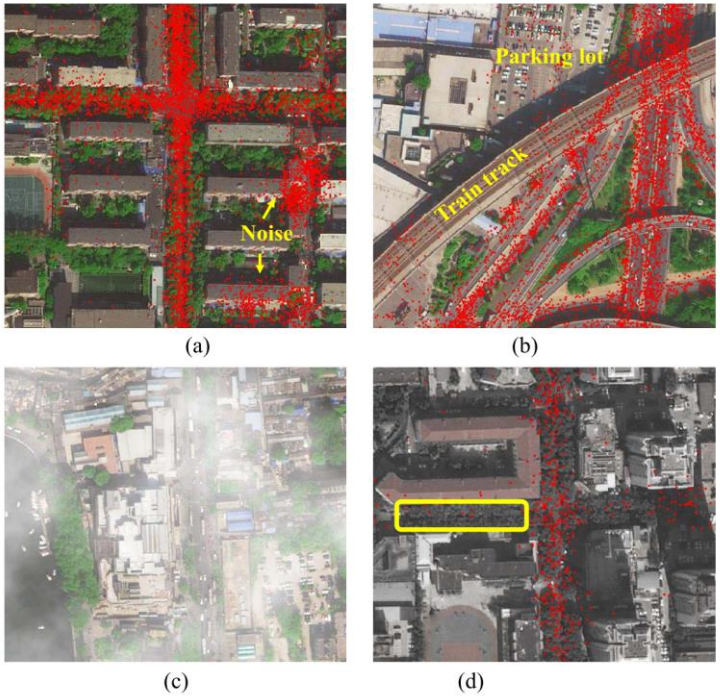


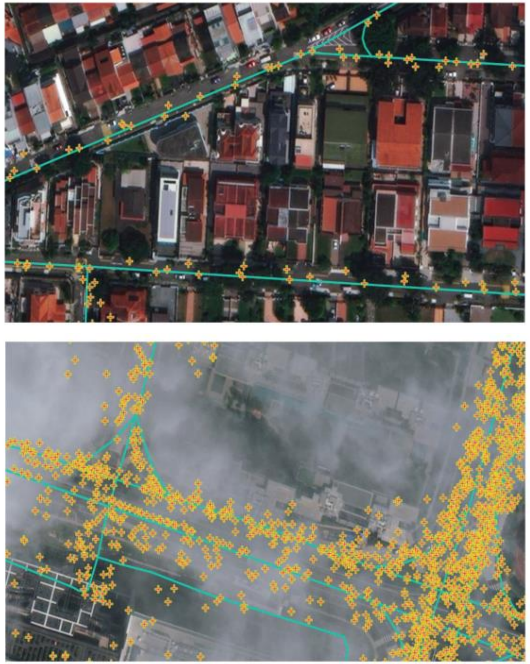
Conclusion

- A valuable but challenging task
  - Position errors
  - Low sampling rate
- Our method
  - Geometry translation (T2RNet)
    - Convolutional network: learns the structure of the road network
    - Auxiliary task: helps the centerline inference
  - Topology construction (Link + Prune)
    - Trajectories as transition evidences
- Results
  - Superior than traditional methods
  - More effective on low-sampling rate datasets

Related Work

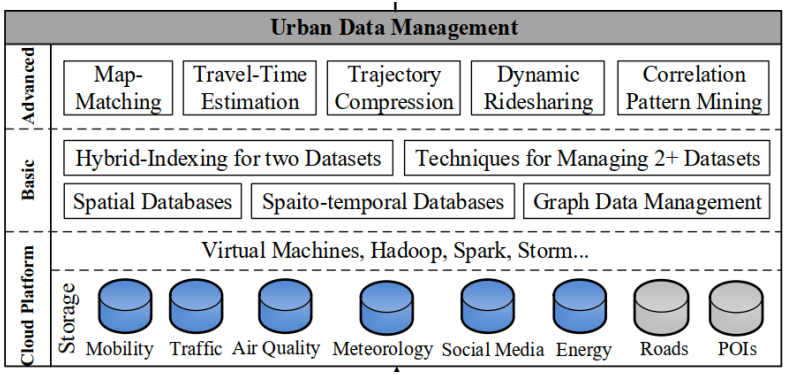
- GPS trajectories + Satellite Images



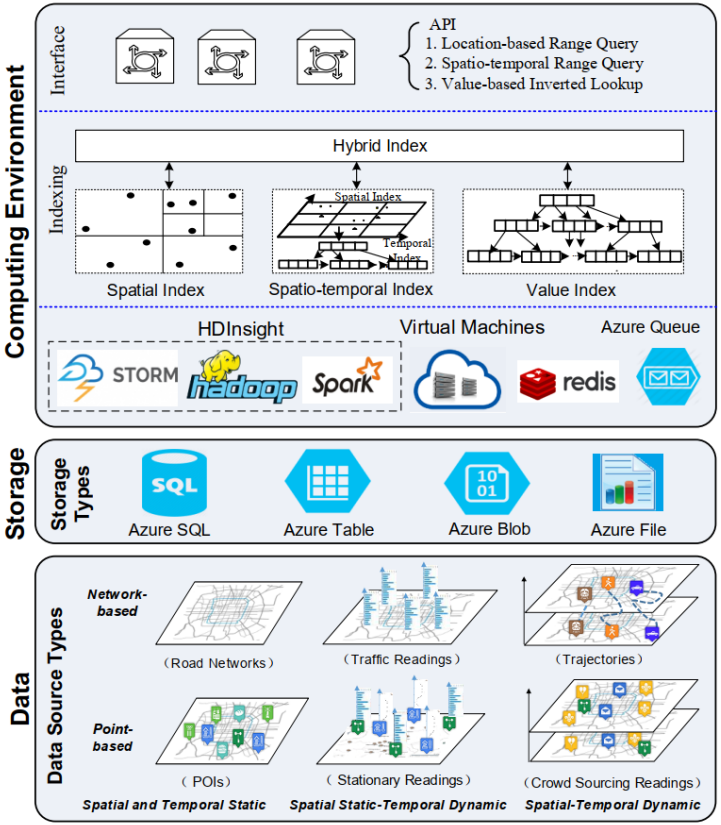


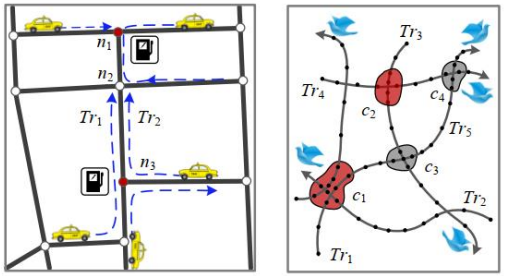
Urban Data Management

- Basic Data management Techniques
  - Spatial Databases
  - Spatio-Temporal Databases
- Advanced applications
  - Map Matching
  - Trajectory compression
  - Large-Scale Dynamic Ridesharing
- Managing Spatio-Temporal Data on the Cloud

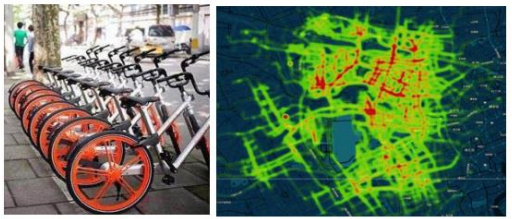


Cloud Computing for ST Data





Y. Li, J. Bao, Y. Li, Z. Gong, **Y. Zheng**.
Mining the Most Influential k-Location Set From Massive Trajectories.
IEEE Transactions on Big Data. 2017



J. Bao, T. He, S. Ruan, Y. Li, and Y. Zheng et al.
Planning bike lanes based on Sharing-bike's trajectories,
KDD 2017

Presented Papers

Spatio-Temporal Data Sesning and Management	1	Transformer-based Map Matching Model with Limited Ground-Truth Data using Transfer-L	Link	TRC	2022	McGill University	Jiahui LIANG
	2	TraSS: Efficient Trajectory Similarity Search Based on Key-Value DataStores	Link	ICDE	2022	JD	Gangyong ZHU
	3	Learning to Generate Maps from Trajectories	cdn.aaai.org/ojs/5435/5	AAAI	2020	Beijing Institute of Technology	Weilin RUAN

Q&A

Thanks!

CityMind Lab

amazon Microsoft Tencent Alibaba.com 京东 JD.COM CAINIAO 菜鸟