Welcome to

DS504/CS586: Big Data Analytics

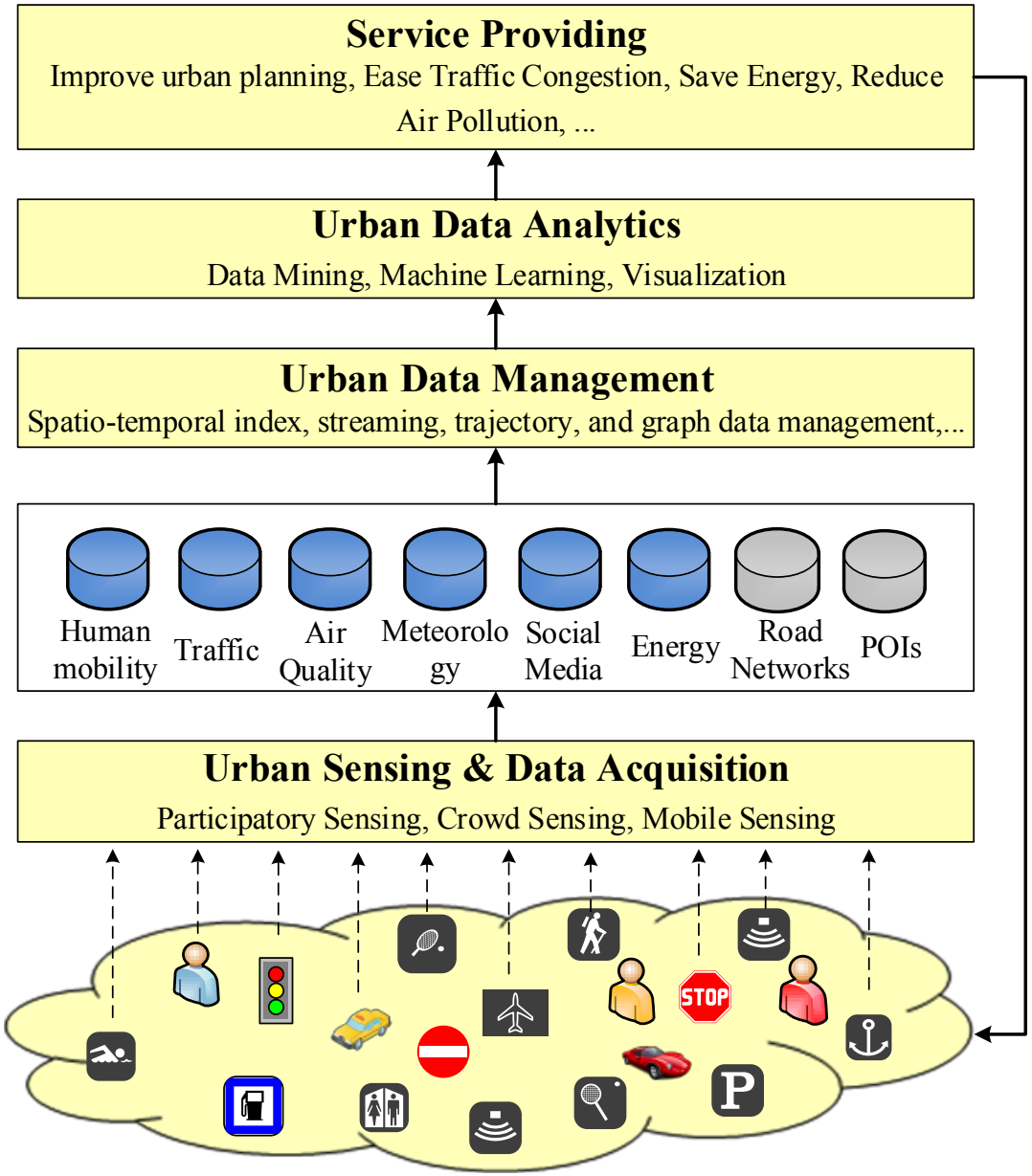
Data Management

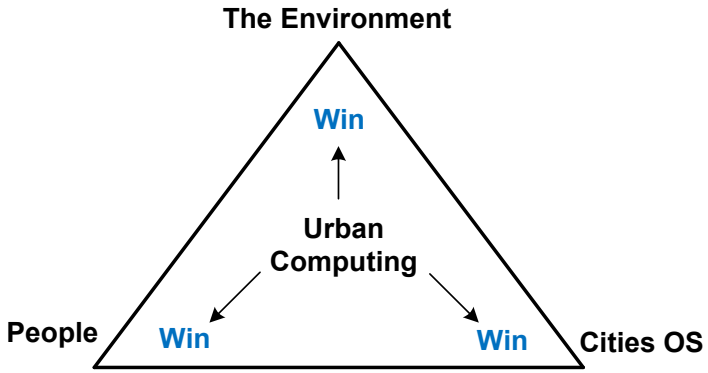
Prof. Yanhua Li

Time: 6:00pm –8:50pm R
Location: AK232
Fall 2016

First Grading for Assignment 2

- Summarizing the problem and solutions
- Critiques/comments/ideas



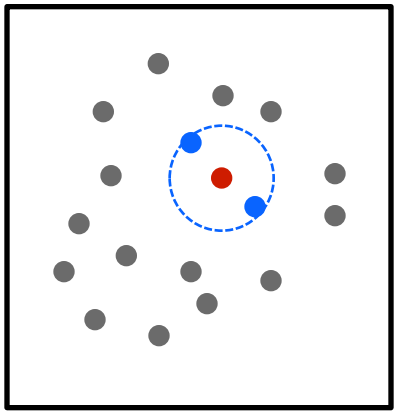


Tackle the Big challenges in Big cities using Big data!

Urban Computing: concepts, methodologies, and applications.
Zheng, Y., et al. *ACM transactions on Intelligent Systems and Technology*.

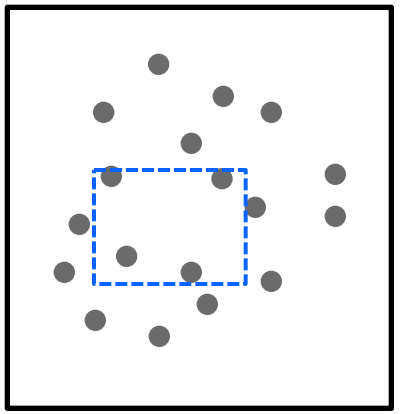
2D-Spatial Queries

K Nearest Neighbour (KNN) Queries



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

Region (Range) Query

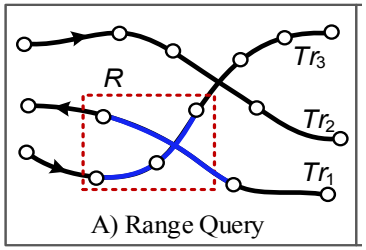


Ask for objects that lie partially or fully inside a specified region.

Trajectory Data Management

- Range queries

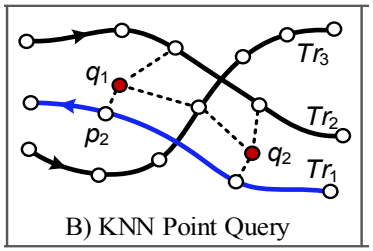
  E.g. Retrieve the trajectories of vehicles passing a given rectangular region *R* between 2pm-4pm in the past month



A) Range Query

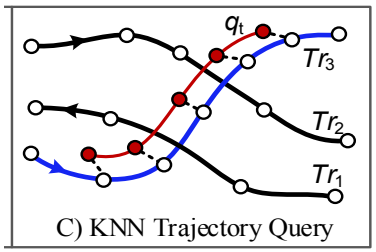
- KNN queries

  E.g. Retrieve the trajectories of people with the minimum aggregated distance to a set of query points
  Publications: [1][2] for a single point query, [3] for multiple query points



B) KNN Point Query

  E.g. Retrieve the trajectories of people with the minimum aggregated distance to a query trajectory
  Publications: Chen et al, SIGMOD05; Vlachos et al, ICDE02; Yi et al, ICDE98.



C) KNN Trajectory Query

[1] E. Frentzos, et al. Algorithms for nearest neighbor search on moving object trajectories. Geoinformatica, 2007

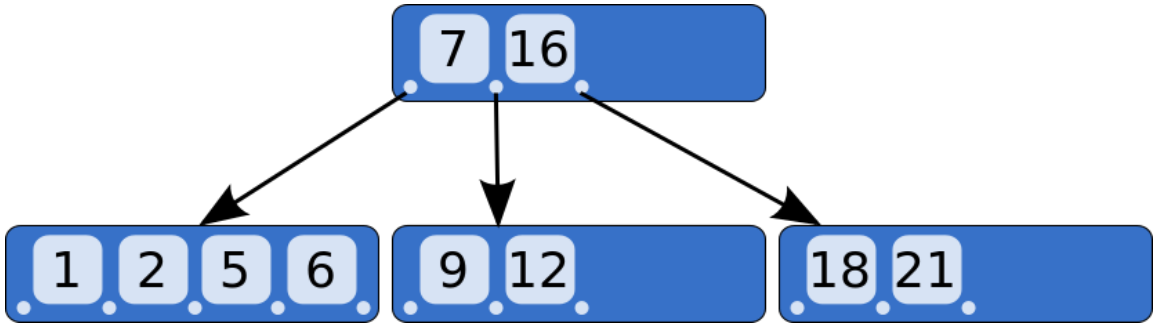
[2] D. Pfoser, et al. Novel approaches in query processing for moving object trajectories. VLDB, 2000.

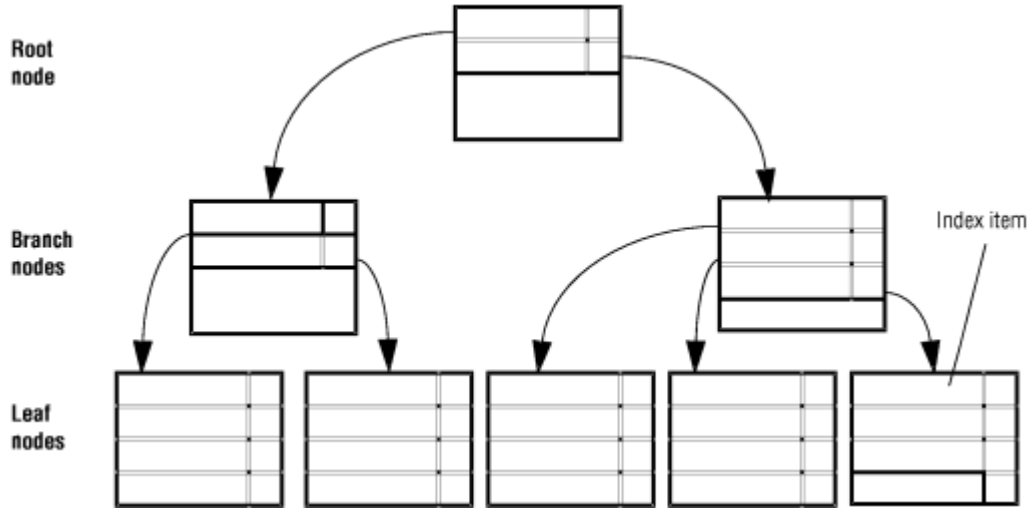
[3] Zaiben Chen, et al. Searching Trajectories by Locations: An Efficiency Study, SIGMOD 2010

Spatial/Temporal Indexing Structures

- Temporal Indexing (1-D data)
  - List index
  - B-tree
- Space Partition-Based Indexing Structures (2-D data)
  - Grid-based
  - Quad-tree

Full B-Tree Structure





B-Tree Index

- B-tree is the most commonly used data structures for indexing.
- It is fully dynamic, that is it can grow and shrink.

Three Types B-Tree Nodes

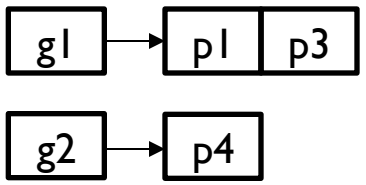
- Root node - contains node pointers to branch nodes.
- Branch node - contains pointers to leaf nodes or other branch nodes.
- Leaf node - contains index items

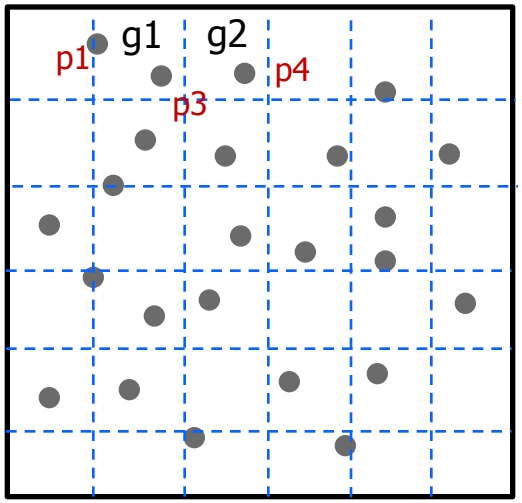
Spatial/Temporal Indexing Structures

- Temporal Indexing (1-D data)
  - B-tree
- Space Partition-Based Indexing Structures (2-D data)
  - Grid-based
  - Quad-tree

Grid-based Spatial Indexing

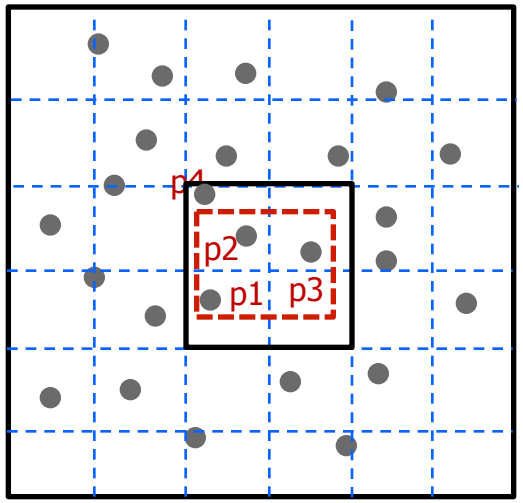
- Indexing
  - Partition the space into disjoint and uniform grids
  - Build an index between each grid and the points in the grid

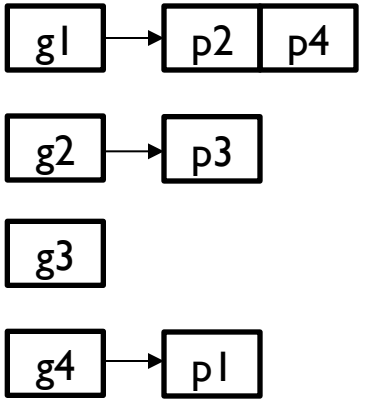




Grid-based Spatial Indexing

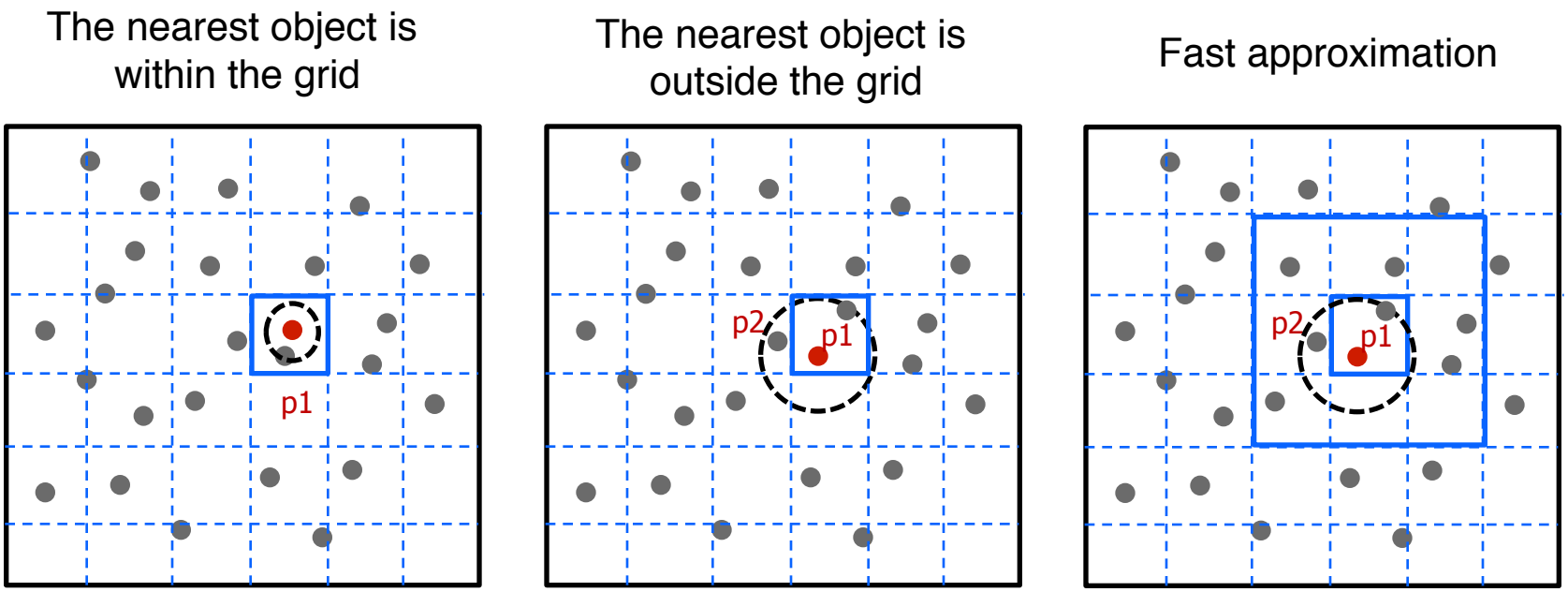
- Range Query
  - Find the girds intersecting the range query
  - Retrieve the points from the grids and identify the points in the range





Grid-based Spatial Indexing

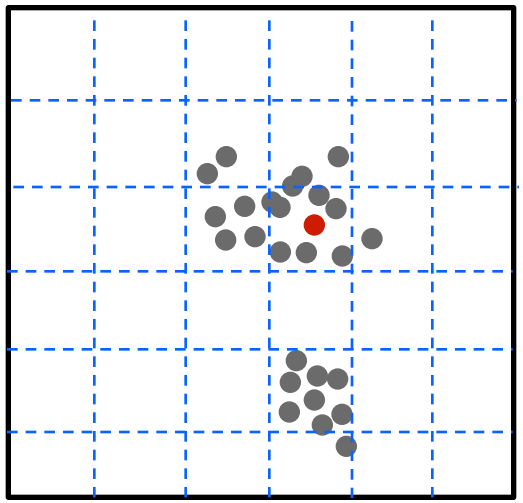
- Nearest neighbor query
  - Euclidian distance
  - Road network distance is quite different



Grid-based Spatial Indexing

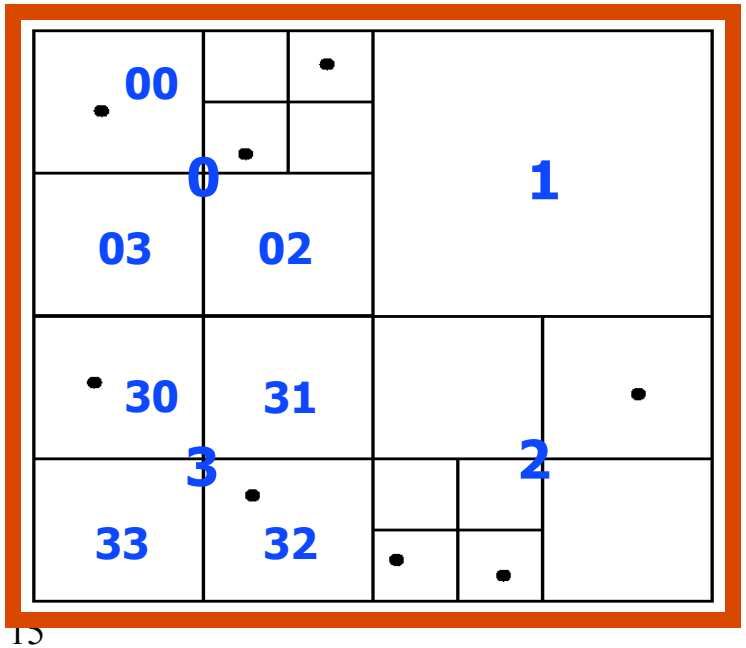
- Advantages
  - Easy to implement and understand
  - Very efficient for processing range and nearest queries
- Disadvantages
  - Index size could be big
  - Difficult to deal with unbalanced data

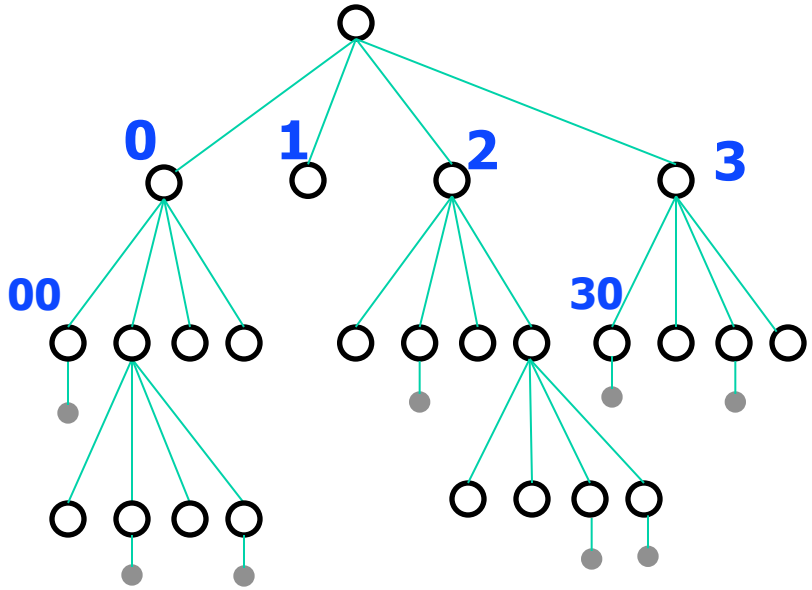
  - Think about what we discussed last time on the POI sampling and estimation.



Quad-Tree

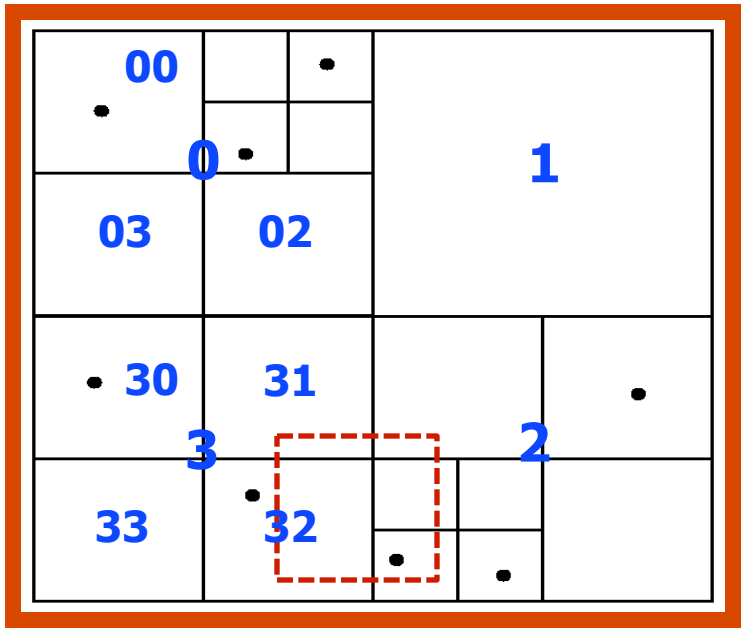
- **Indexing**
  - Each node of a quad-tree is associated with a rectangular region of space; the top node is associated with the entire target space.
  - Each non-leaf node divides its region into four equal sized quadrants
  - Leaf nodes have between zero and some fixed maximum number of points (set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

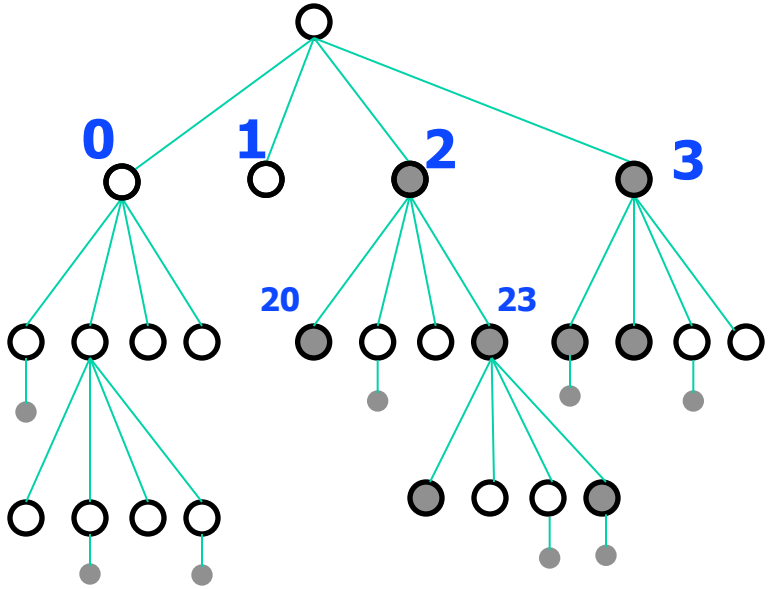




Quad-Tree

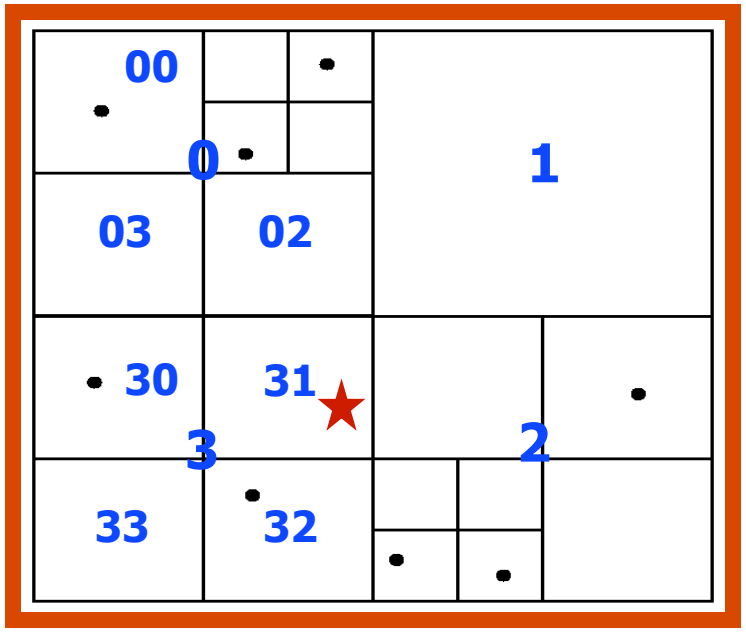
- Range query

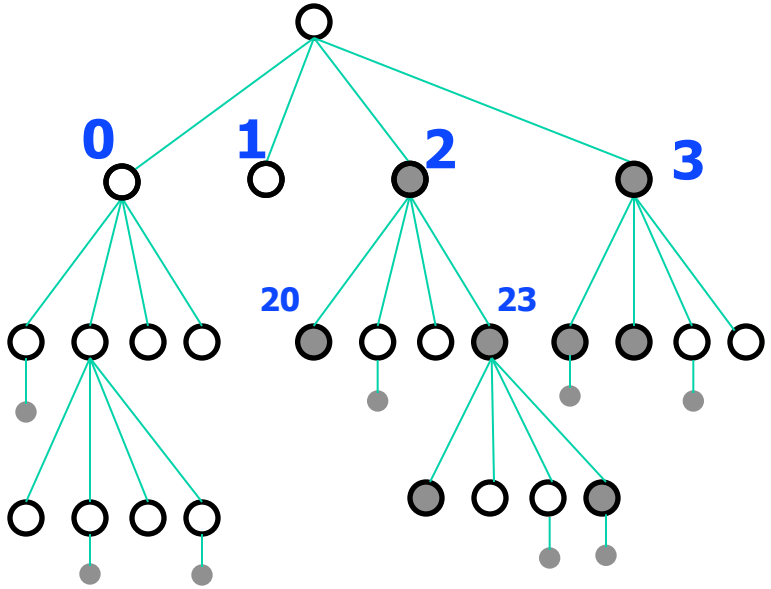




Quad-Tree

- Nearest Neighbour Query (hard)

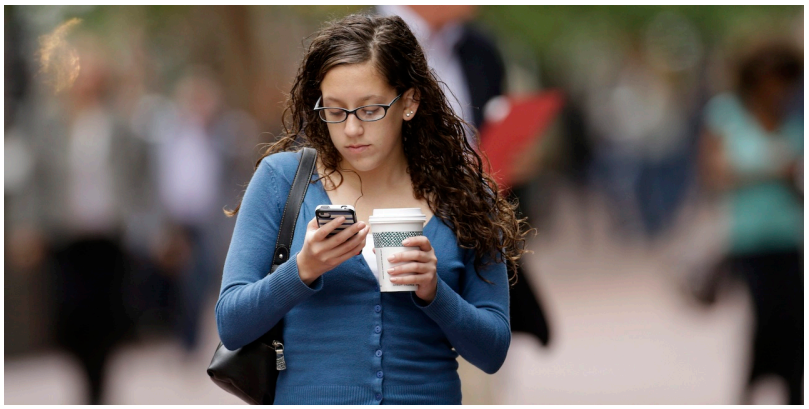




Sampling Big Trajectory Data

Big Trajectory Data in Urban Networks

Taxi GPS Trajectory



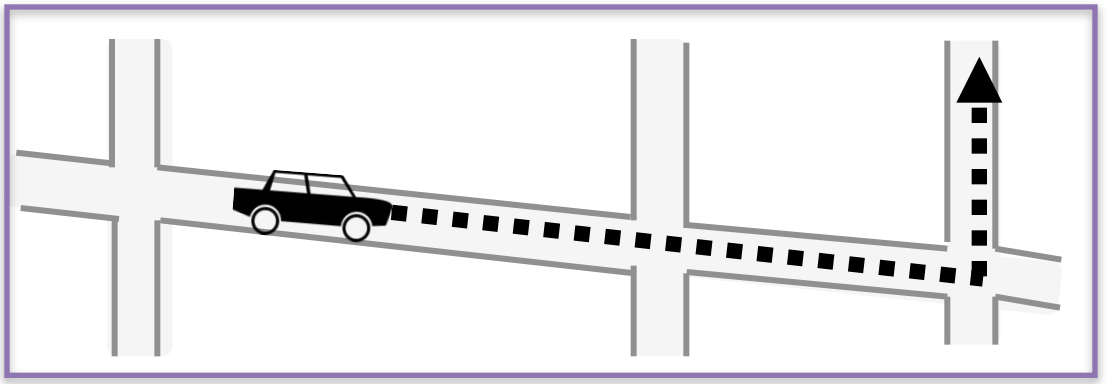
Mobile User Trajectory

- Urban roving sensors deliver big trajectory data.
  - Reveal moving patterns and urban issues.

Challenge

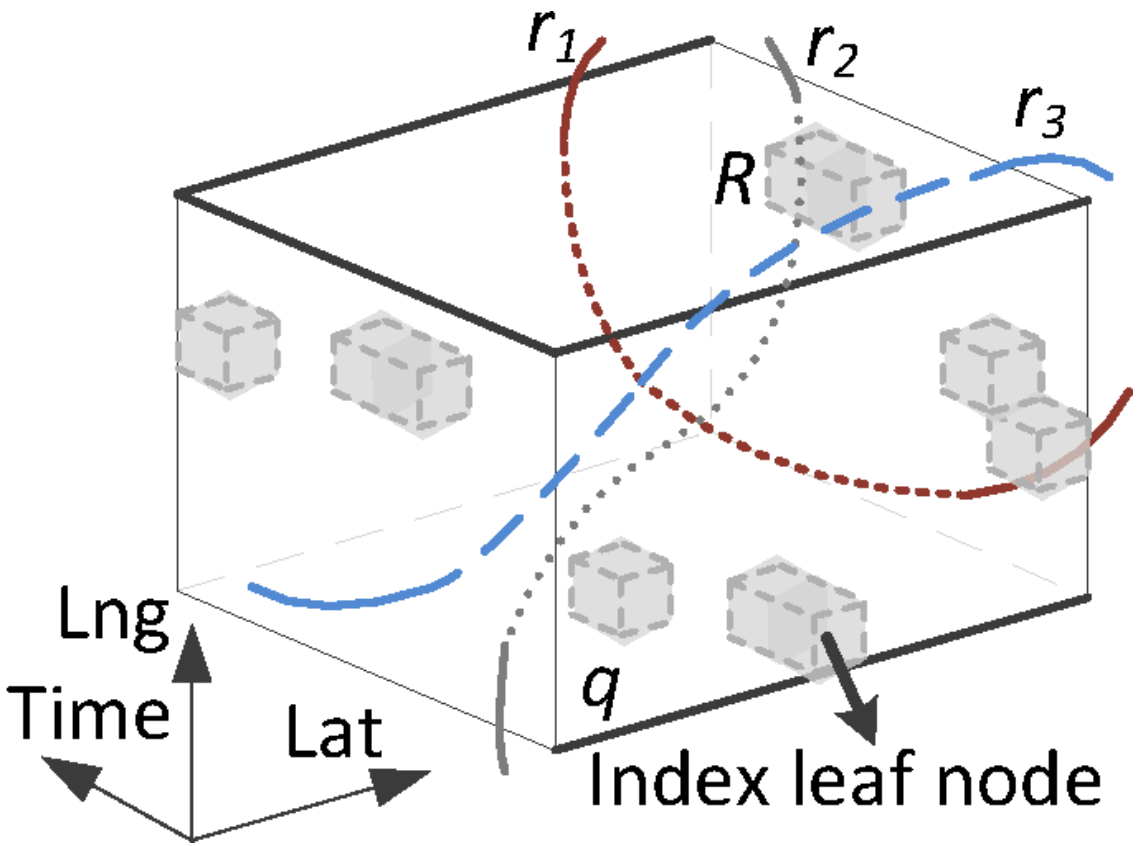
How to manage the big trajectory data to enable efficient query processing.

Trajectory Aggregate Query



- A trajectory aggregate query
  - Retrieves statistics of distinct trajectories passing a user-specified spatio-temporal region;
- *Examples,*
  - # of taxi trips with average speed of more than 5 miles per hour in New York City in 2014;
  - # of mobile users with iPhone in Hong Kong in 2013

Exhaustive Search



- r_i: a sequence of GPS points in (TID, Lat, Lng, Time)
- q: a trajectory aggregate query *with* N_q Trajectories
- Spatio-temporal indexing: B-tree, Quad-tree, etc,

Challenges with Big Trajectory Data

- Long responding time for large trajectory dataset
- In 2013, Shenzhen, China;

Mobility Data	788.6TB	6million users
Taxi GPS	1.58 TB	22,083 taxis
Bus GPS	1.34 TB	8,427 buses

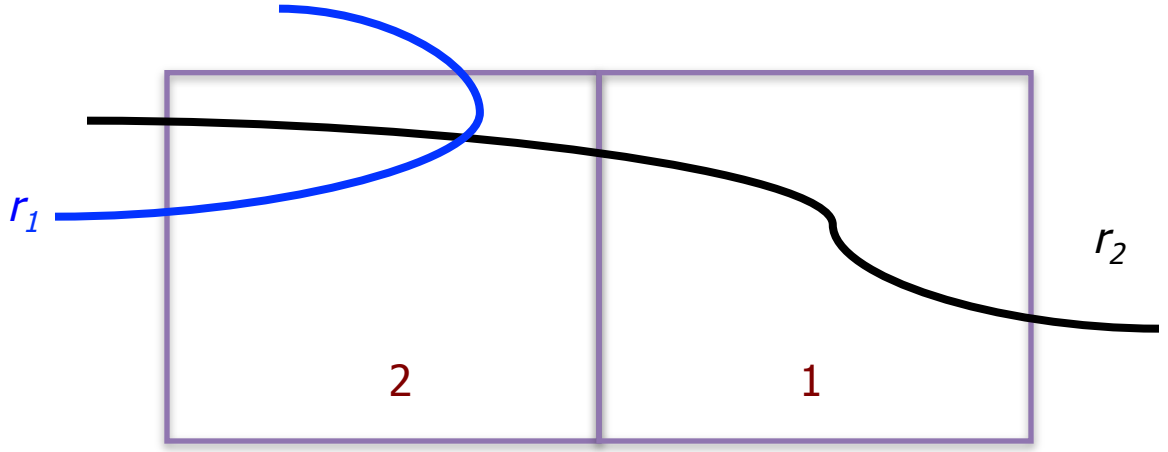
- Query: # of iPhone users and taxi/bus trips

iPhone Users	0.8 million
Taxi GPS	302 million trips
Bus GPS	1.38 billion trips

12 minutes to get the exact answers!

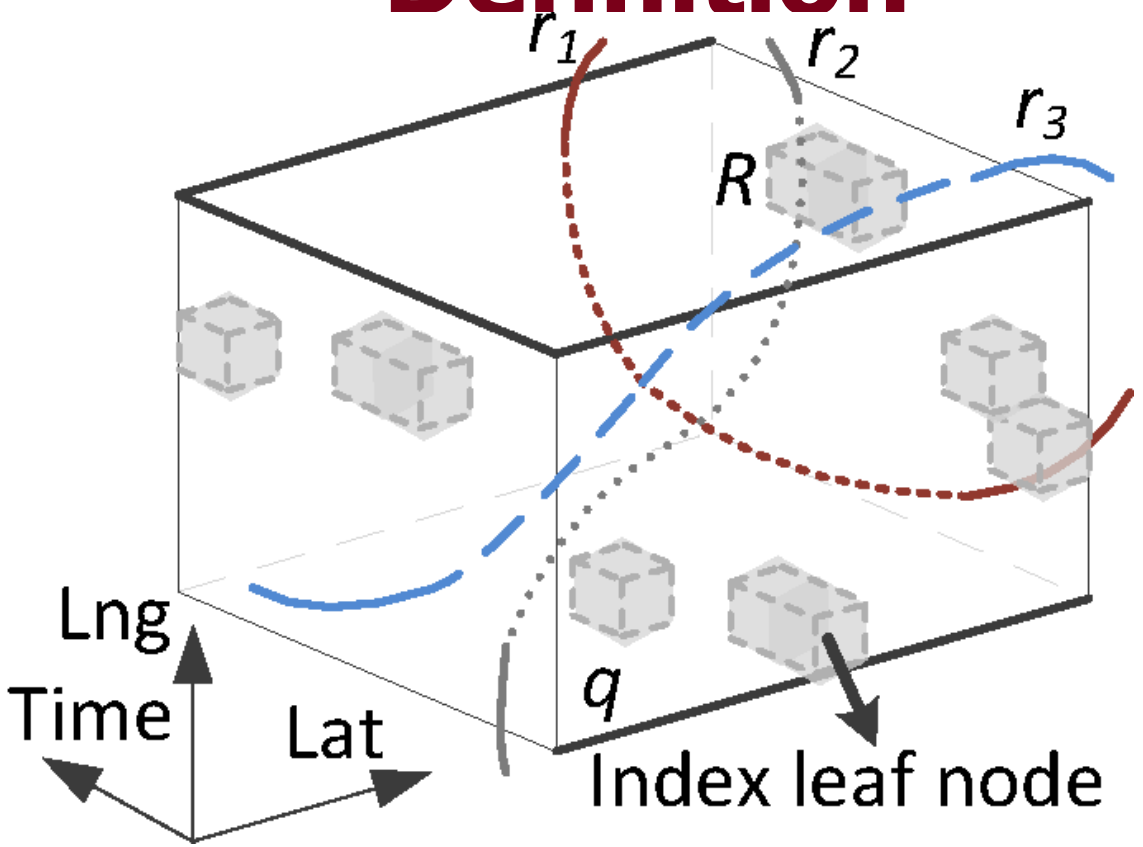
(System: A cluster of 3 machines with 24 Intel X5670 2.93GHz processors, 94GB memory.)

Key Challenges on Exact Answer



- A trajectory r_i may traverse multiple index leaf nodes
  - Cannot pre-compute and store the results on indices
  - Summing up two answers leads to over-counting

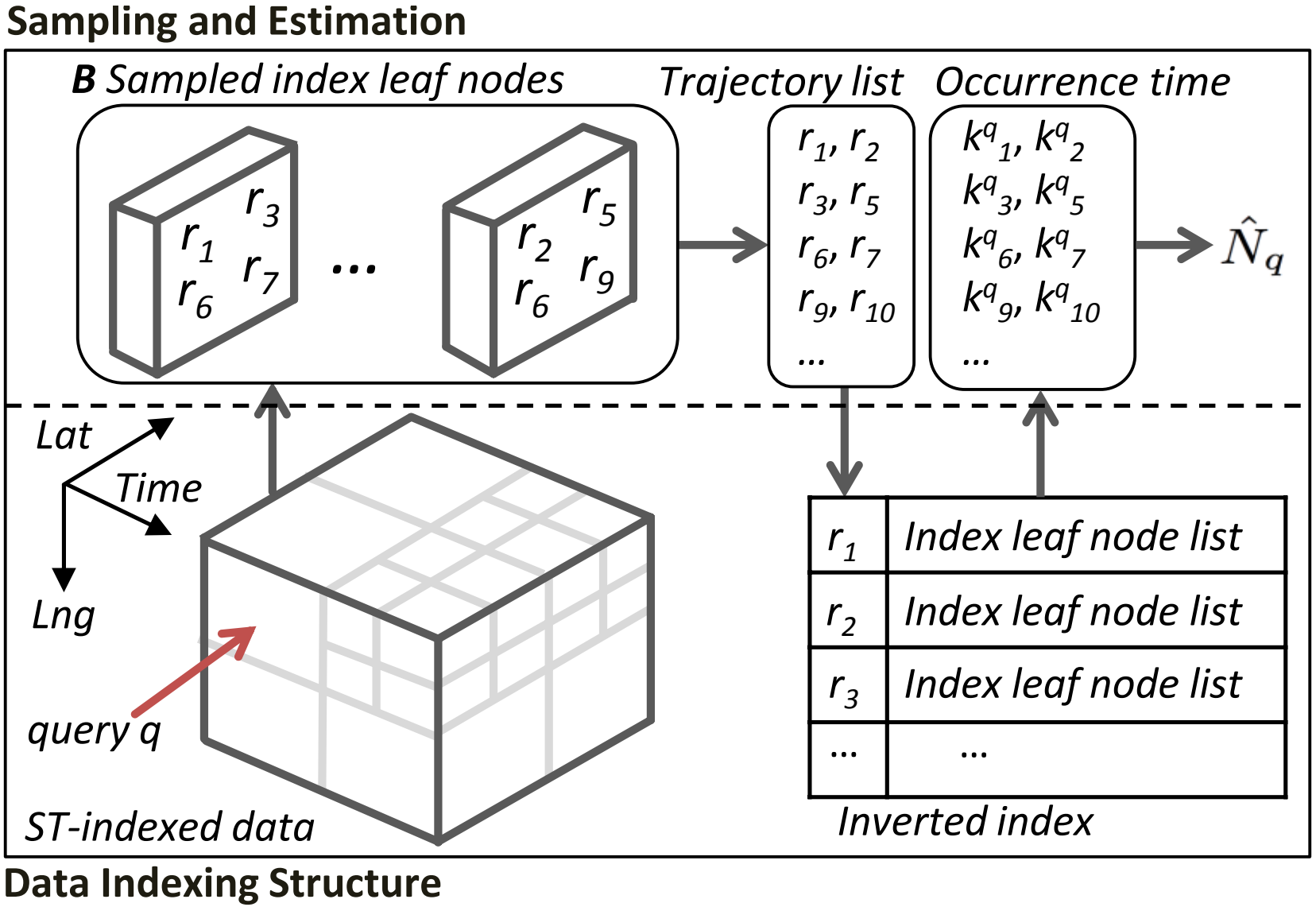
Motivation & Problem Definition



***q* covers n index leaf nodes**

How to sample *B* index leaf nodes to estimate # of trajectories in *q with a guaranteed error bound?*

Random Index Sampling



Random Index Sampling

- Stage 1: Sampling Stage:
  - Uniformly at random sample *B* index leaf nodes with replacement
- Stage 2: Estimation Stage: (Unbiased Estimator)

$$\hat{N}_q = \frac{n}{B}\sum_{t=1}^{B} f_q(\hat{R}_t^q) \qquad f_q(\hat{R}_t^q) = \sum_{r \in \hat{R}_t^q \wedge q}\{1/k_r^q\}$$

- Convergence analysis:

$$Pr(|\hat{N}_q - N_q| > \epsilon) \leq \alpha$$

when $B \geq \frac{\ln(2/\alpha)\delta^2 n^2}{2\epsilon^2}$ for any $\epsilon > 0$ and $0 < \alpha \leq 1$

δ *is the maximum number of trajectories in an index leaf node.*

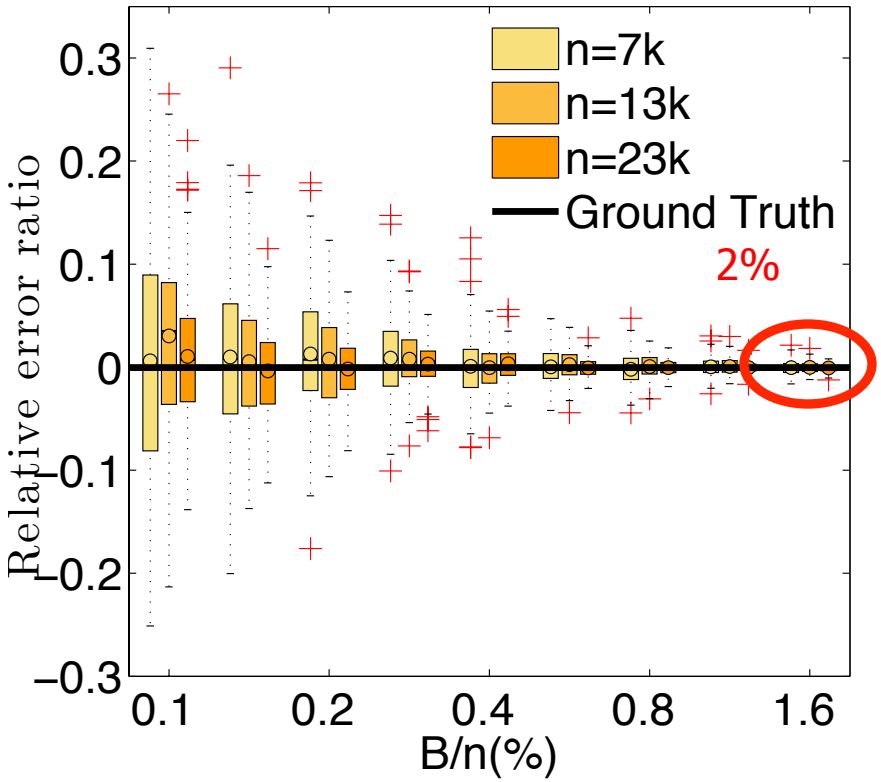
Evaluation

- Dataset: 3TB real human mobility data in a large city in eastern China

Statistics	Value
City Size	400 square miles
City Population Size	three million people
Duration	eight days at the end of 2010
Number of trajectories	**109,914 3G users**
# of spatio-temporal points	400 million (407, 040, 083)

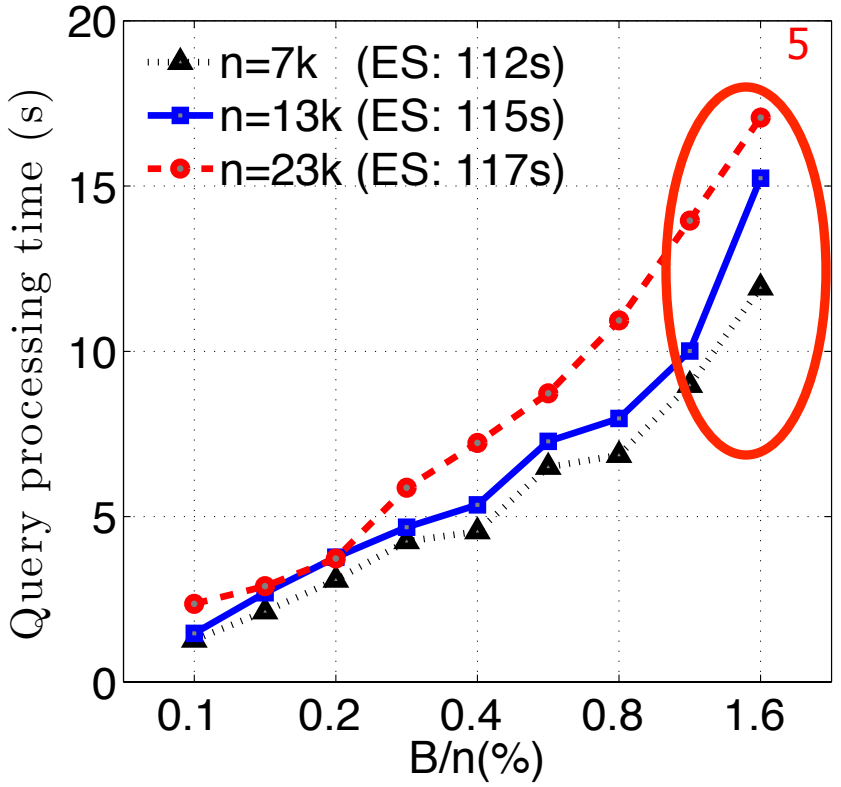
- Baseline Algorithm
  - Exhaustive search
- Evaluation metric
  - Relative error & Responding time

Evaluation Results



Relative error

Up to 2% relative error

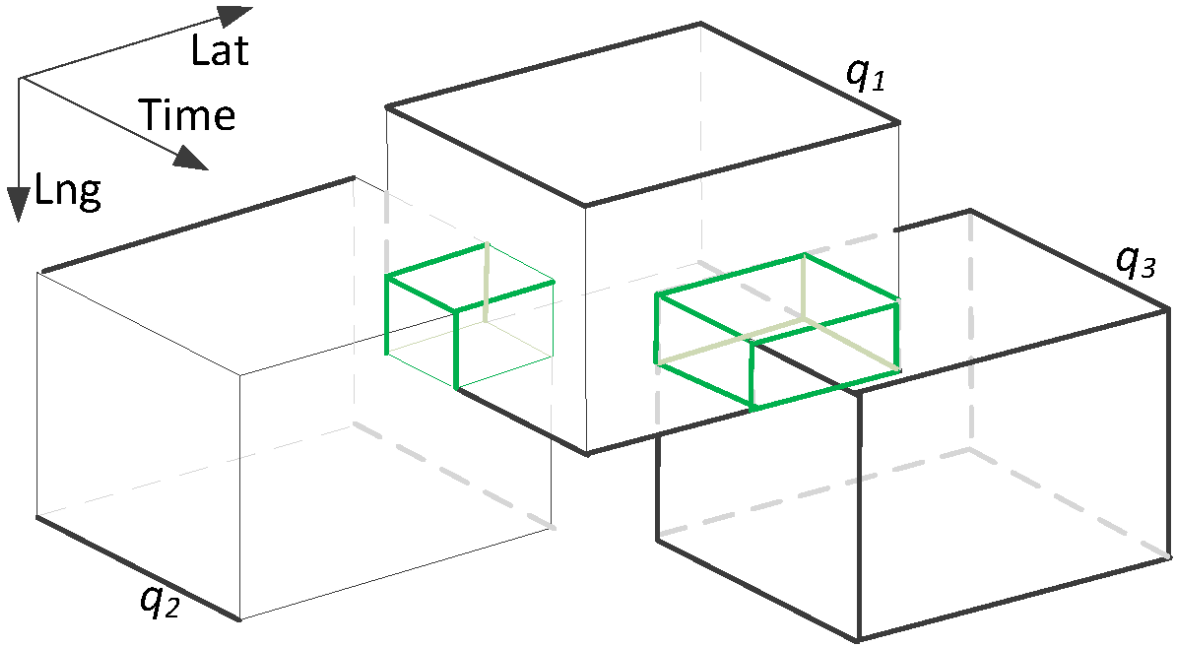


Processing time

5 times reduction

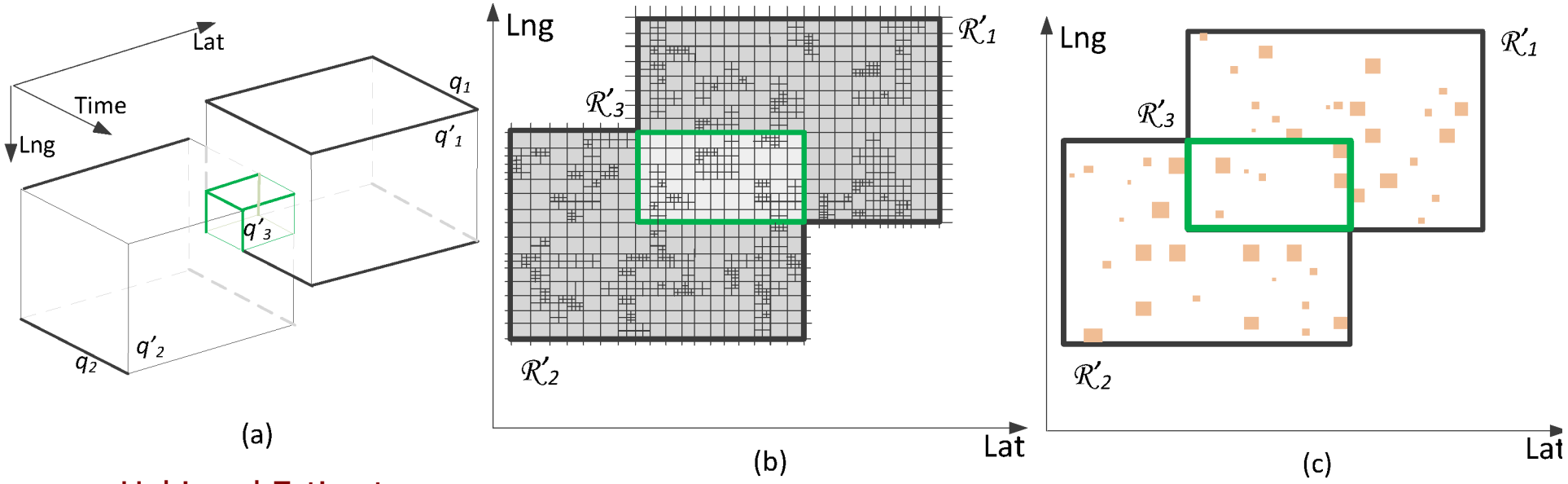
Concurrent Random Index Sampling

- Practical Issue:
  - A large number of concurrent aggregate queries



- Idea of Concurrent Random Index Sampling (CRIS):
  - Sampling Reuse
  - Stratified Sampling Technique

Concurrent Random Index Sampling



Unbiased Estimators:

$$\hat{N}_{q_1}^{CRIS} = \frac{n_1}{B_1}\left(\sum_{t=1}^{[B_1 n_1'/n_1]} f_{q_1}(\hat{R}_t^{q_1'}) + \sum_{t'=1}^{[B_1 n_3'/n_1]} f_{q_1}(\hat{R}_{t'}^{q_3'})\right)$$

$$\hat{N}_{q_2}^{CRIS} = \frac{n_2}{B_2}\left(\sum_{t=1}^{[B_2 n_2'/n_2]} f_{q_2}(\hat{R}_t^{q_2'}) + \sum_{t'=1}^{[B_2 n_3'/n_2]} f_{q_2}(\hat{R}_{t'}^{q_3'})\right)$$

Summary

- Approximate query processing
  - Single trajectory aggregate query
    - via Random Index Sampling (RIS)
  - Concurrent trajectory aggregate queries
    - via Concurrent Random Index Sampling (CRIS)

Any Comments & Critiques?

Weka

- 6 weeks

- Each week, 10+5 minutes for a short summary of what you learned.
  - As part of the oral evaluation of the class;
  - One or two team members to present, and the whole group got the same score

- At the last week, turn in your e-certificate
  - As part of the written evaluation of the class