Non-Standard-Datenbanken und Data Mining

Probabilistic Spatio-Temporal Databases and Streams

Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Übersicht

- Semistrukturierte Datenbanken (JSON, XML) und Volltextsuche
- Information Retrieval
- Mehrdimensionale Indexstrukturen
- Cluster-Bildung
- Einbettungstechniken
- First-n-, Top-k-, und Skyline-Anfragen
- Probabilistische Datenbanken, Anfragebeantwortung, Top-k-Anfragen und Open-World-Annahme
- Probabilistische Modellierung, Bayes-Netze, Anfragebeantwortungsalgorithmen, Lernverfahren,
- Temporale Datenbanken und das relationale Modell, SQL:2011
- Probabilistische Temporale Datenbanken
- SQL: neue Entwicklungen (z.B. JSON-Strukturen und Arrays), Zeitreihen (z.B. TimeScaleDB)
- Stromdatenbanken, Prinzipien der Fenster-orientierten inkrementellen Verarbeitung
- Approximationstechniken für Stromdatenverarbeitung, Stream-Mining
- Probabilistische raum-zeitliche Datenbanken und Stromdatenverarbeitungsssysteme: Anfragen und Indexstrukturen, Raum-zeitliches Data Mining, Probabilistische Skylines
- Von NoSQL- zu NewSQL-Datenbanken, CAP-Theorem, Blockchain-Datenbanken

Acknowledgments

Presentation slides are largely taken from

Location-aware Query Processing and Optimization: A Tutorial

Mohamed F. Mokbel
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA
mokbel@cs.umn.edu

Walid G. Aref
Department of Computer Science, Purdue University, West Lafayette, Indiana, USA.
aref@cs.purdue.edu

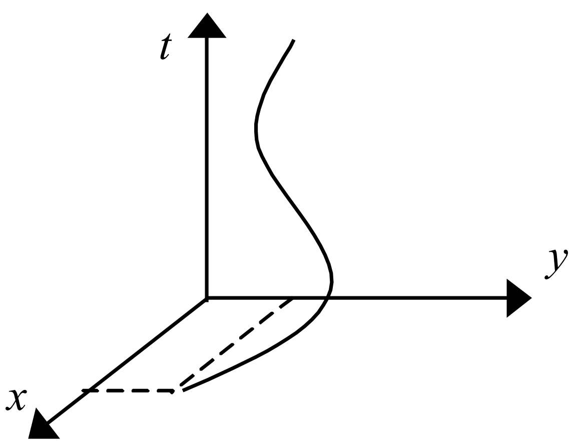
Some slides (indicated) were produced by George Kollios

Slides have been modified or extended. Faults are mine!

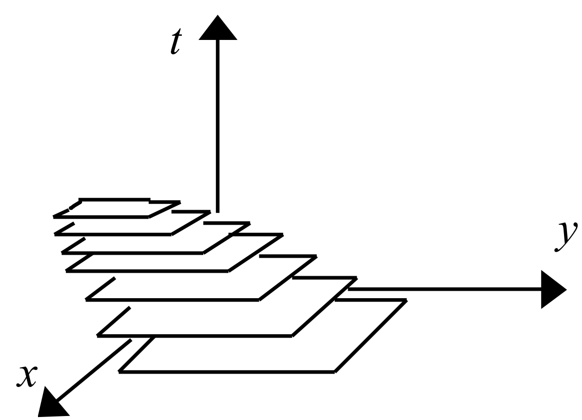
Mohamed F. Mokbel, Thanaa M. Ghanem, and Walid G. Aref. Spatio-temporal Access Methods. IEEE Data Engineering Bulletin, 26(2):40-49, June 2003.

Spatio-Temporal Objects

- Moving points (extent does not matter)
  - Each object is modeled as a point (e.g., moving vehicles in a GIS based transportation system)
- Moving regions (extent matters)
  - Each object is represented by an MBR, the MBR can change as the object moves (e.g., thunderstorm, noise)



(a)
a moving point



(b)
a moving and shrinking region

Location-aware Queries

Continuously report the number of cars on freeway 71-75

- *Type:* Range query
- *Time:* Present
- *Duration:* **Continuous**
- *Query:* Stationary
- *Objects:* Moving

What are my nearest McDonalds for the next hour?

- *Type:* Nearest-neighbor query
- *Time:* **Future**
- *Duration:* **Continuous** / **Snapshot**
- *Query:* Moving (reference rectangle)
- *Objects:* Stationary (McDonalds)

Send E-coupons to all cars that I am their nearest gas station

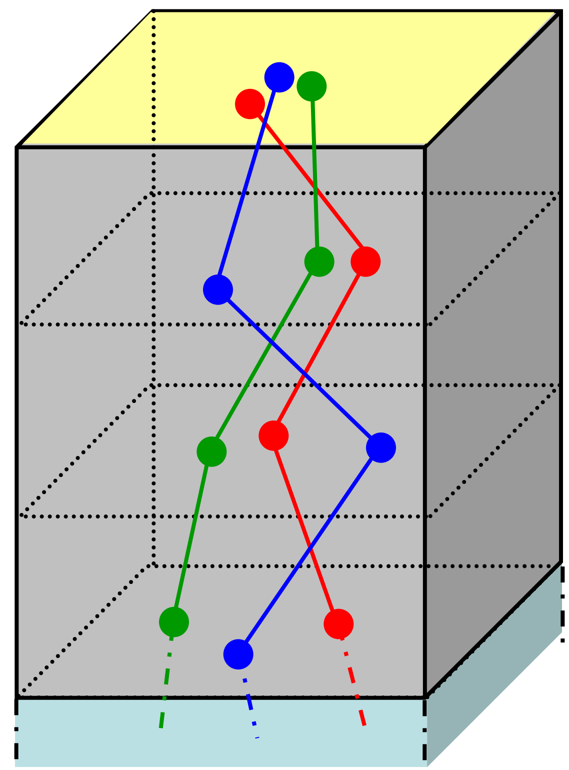
- *Type:* Reverse NN query
- *Time:* **Present**
- *Duration:* **Snapshot**
- *Query:* Stationary (gas station)
- *Objects:* Moving

What was the closest distance between Taxi A & me yesterday?

- *Type:* Closest-point query
- *Time:* **Past**
- *Duration:* **Snapshot**
- *Query:* Moving
- *Objects:* Moving

Snapshot Querying the Past

- Examples:
  - **Temporal** Dimension:
    What was the location of a certain object from 7:00 AM to 10:00 AM yesterday?
  - **Spatial** Dimension:
    Find all objects that were in a certain area at 7:00 AM yesterday
  - **Spatio-temporal** Dimension:
    Find all objects that were close to each other from 7:00 AM to 8:00 AM yesterday

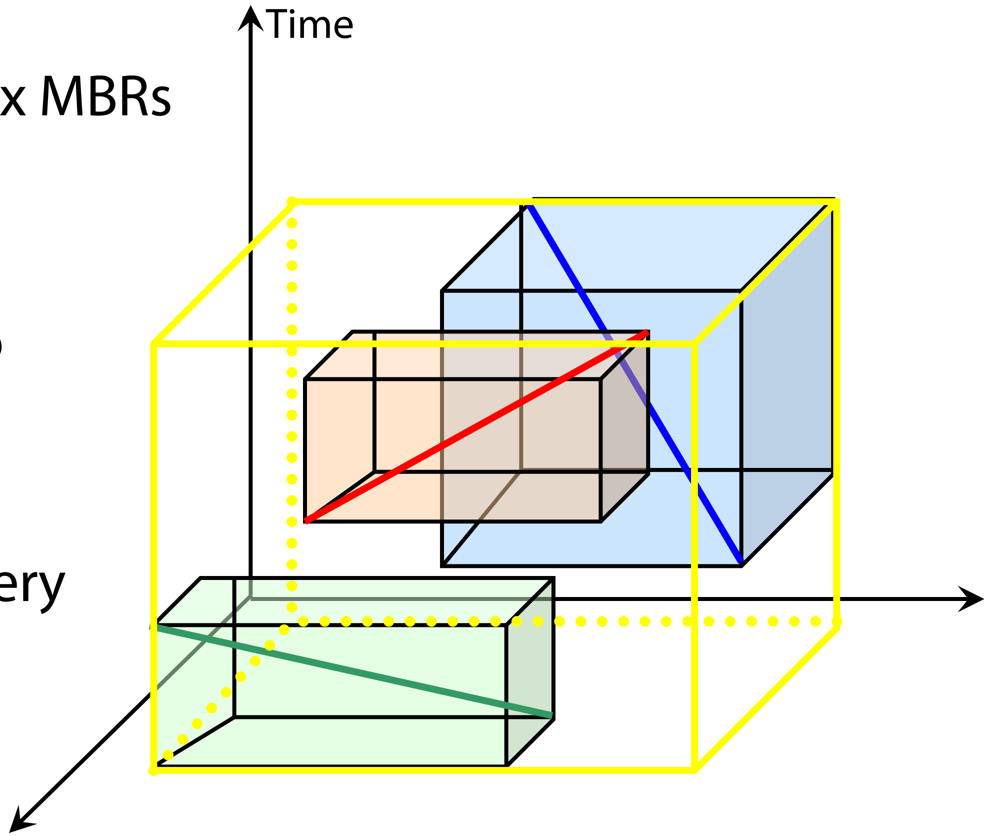


- Features:
  - Large number of historical trajectories
  - Persistent read-only data
  - Query spatial and/or temporal dimensions

Indexing the Time Dimension

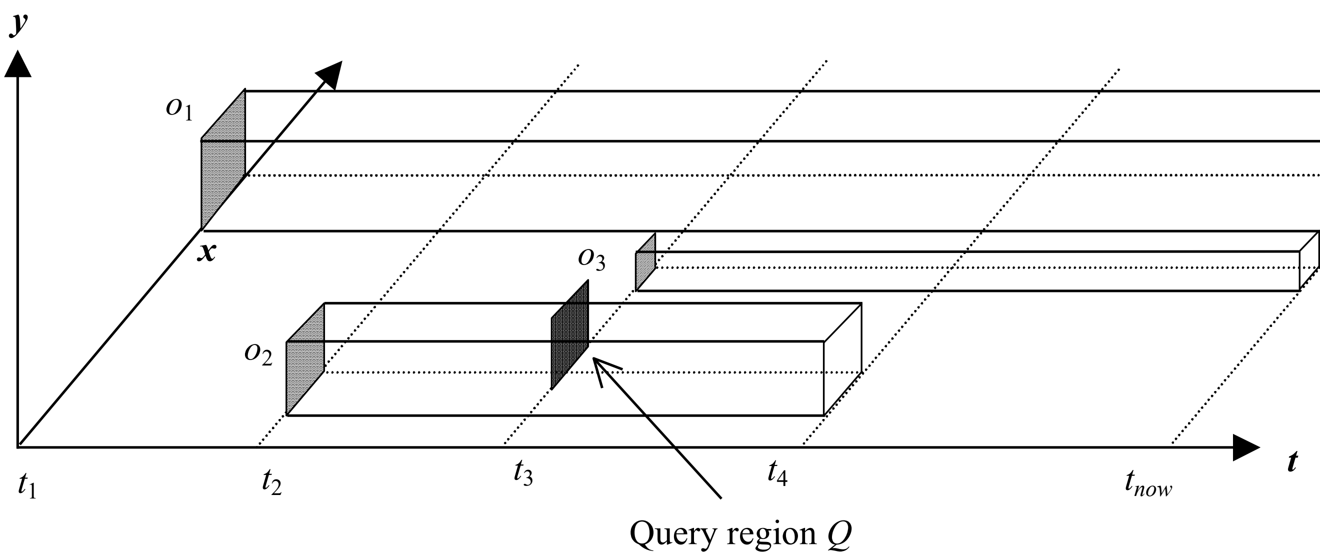
Historical trajectories are represented by their three-dimensional Minimum Bounding Rectangle (MBR)

- 3D R-tree can be used to index MBRs



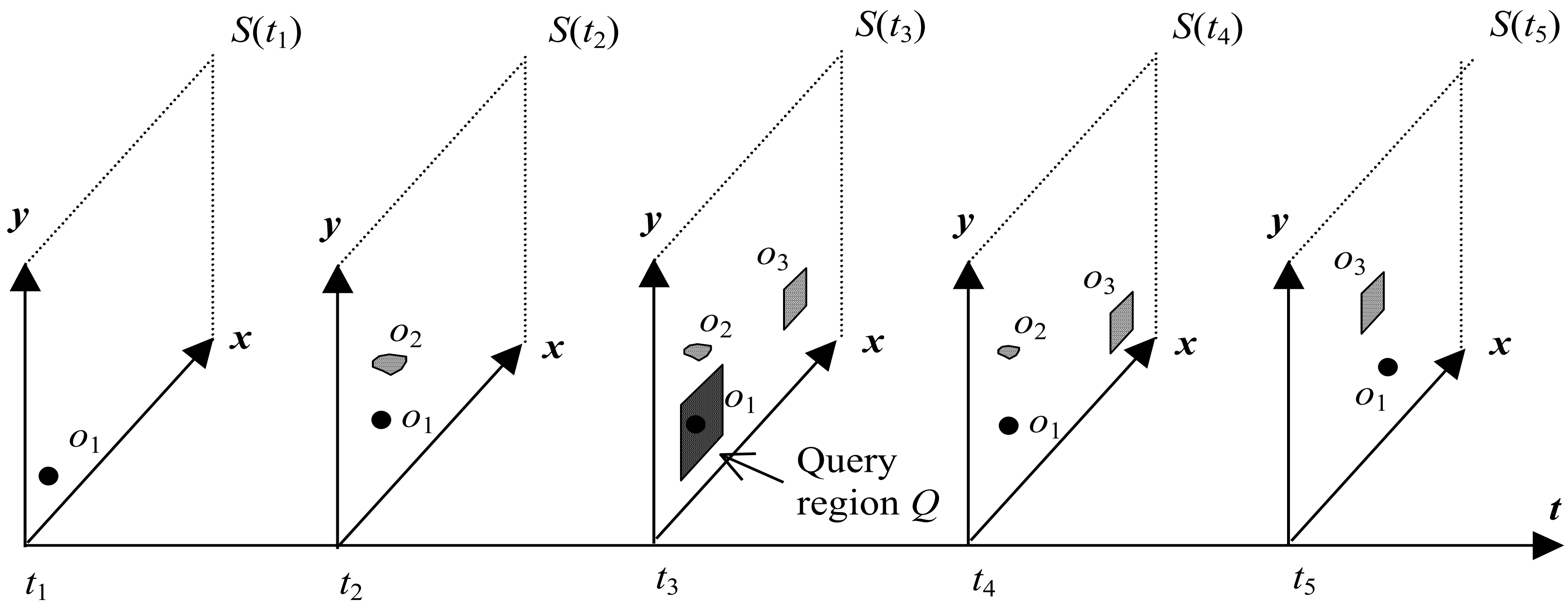
- Technique simple and easy to implement
- Does not scale well
- Does not provide efficient query support for snapshot queries (aka timestamp queries)

3D R-Tree



Objects are somewhere in the gray rectangular regions.

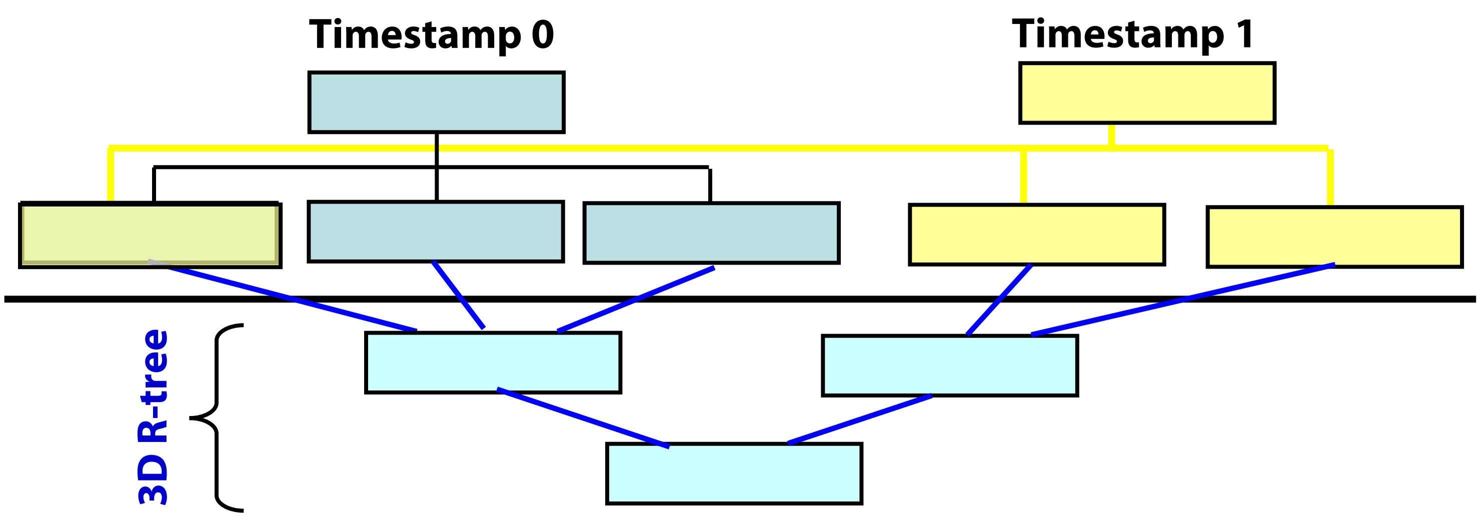
Modeling Evolution: Historical R-Trees



Snapshot query

Multi-Version Index Structures (MVR-Trees)

- Maintain an R-tree for each time instance (aka historical r-tree, HR-tree)
- R-tree nodes that are not changed across consecutive time instances are linked together (remove redundancies: MVR-tree)



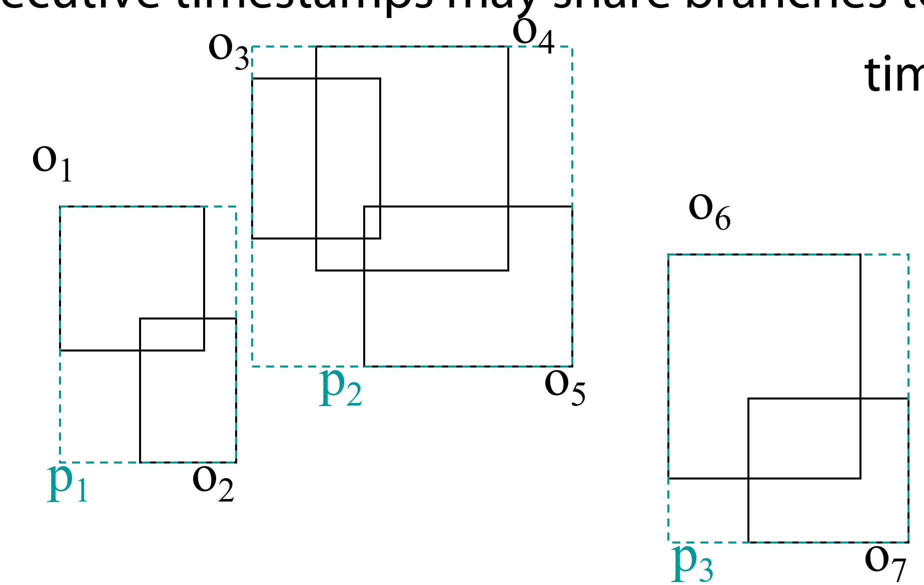
- A multi-version R-tree can be combined with a 3D-R-tree to support interval queries (combination is called MV3R-Tree)

Yufei Tao and Dimitris Papadias. MV3R-Tree: A Spatio-temporal Access Method for Timestamp and Interval Queries. In Proc. VLDB-01, pp. 431-440, 2001

Historical R-trees (HR-trees)

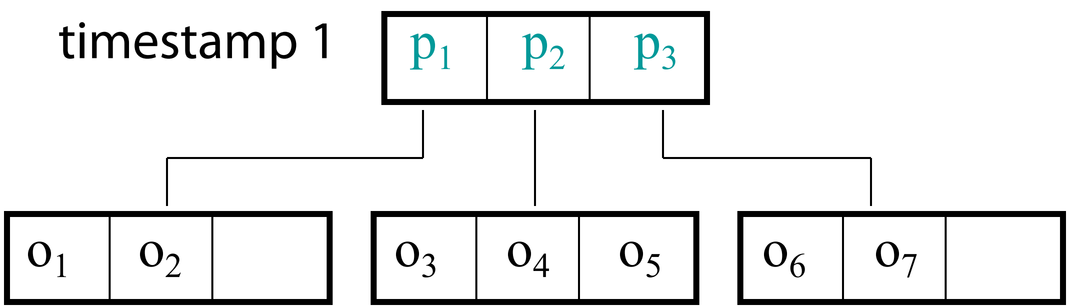
An R-tree is maintained for each timestamp in history.

Trees at consecutive timestamps may share branches to save space.



timestamp 1

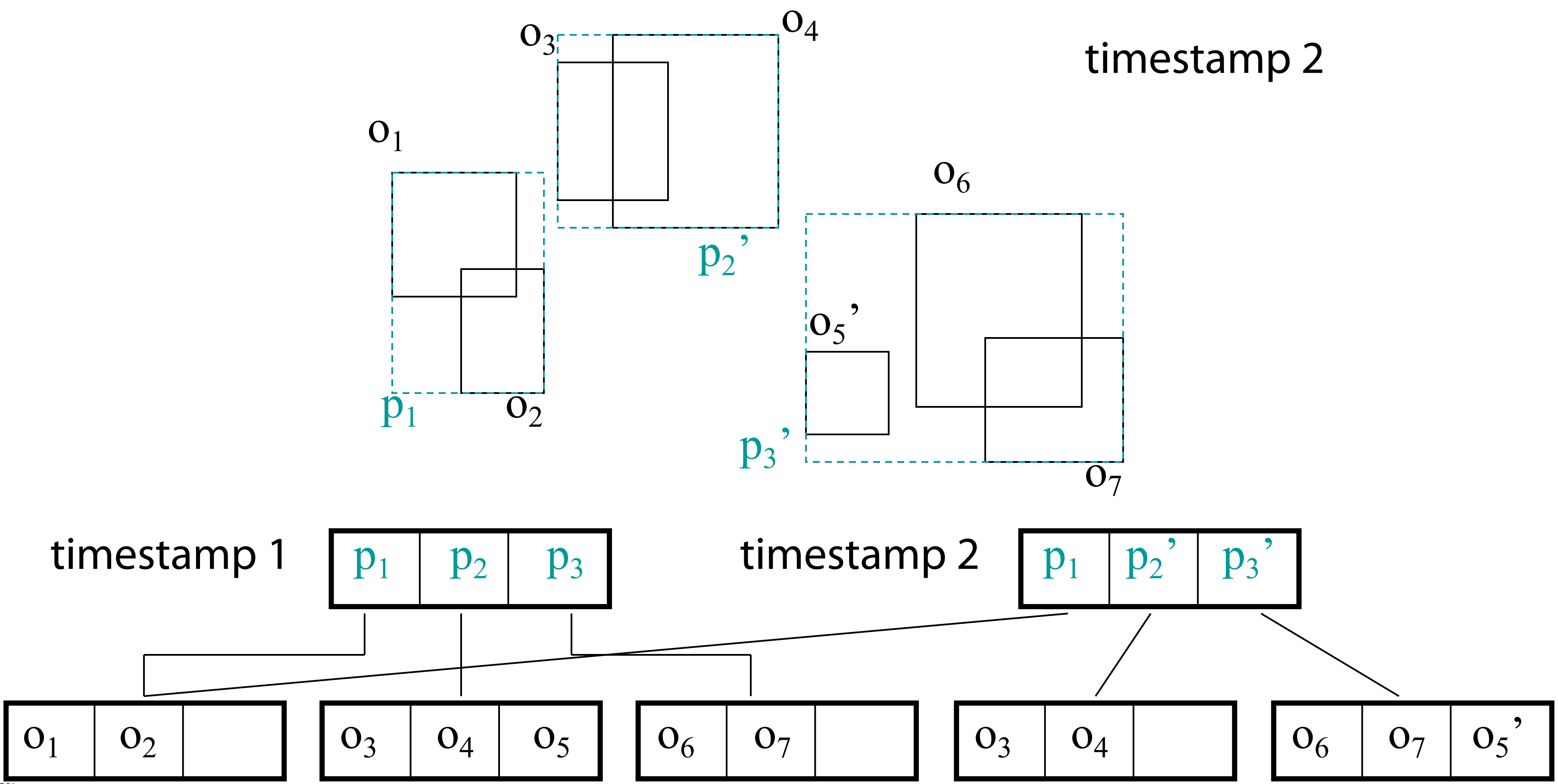
timestamp 1



Historical R-trees

An R-tree is maintained for each timestamp in history.

Trees at consecutive timestamps may share branches to save space.

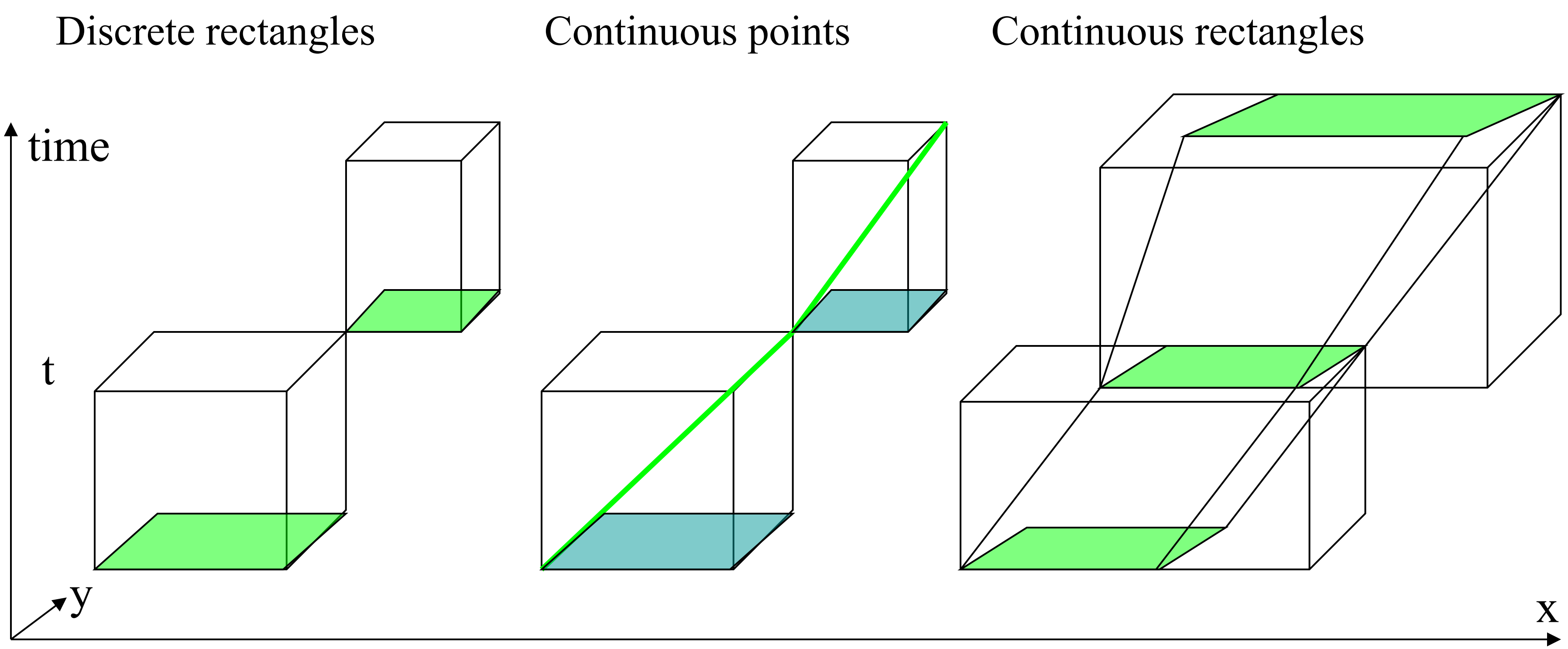


Building a 3D R-tree on the Leaves of the MVR-tree

- Size of the 3D R-tree is much smaller than a complete 3D R-tree as the number of leaf nodes is significantly lower than the number of actual objects.
- Long interval queries can be processed with auxiliary 3D R-trees

Rectangles

Problem of indexing any type of moving objects can be reduced to indexing discrete rectangles



Optimization

- If N objects move with linear functions of time:
- Minimize total volume by splitting in equidistant points
- Given K splits you can decide the best splits in O(K log N) time.

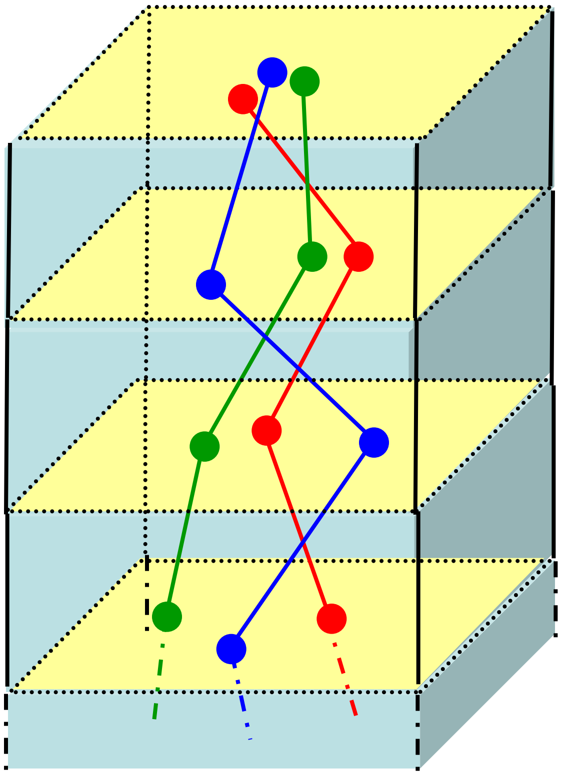
Yufei Tao and Dimitris Papadias. MV3R-Tree: A Spatio-temporal Access Method for Timestamp and Interval Queries. In Proc. VLDB-01, pp. 431-440, **2001**

Querying the Present

- Time is always NOW
- Example Queries:
  - Find the number of objects in a certain area
  - What is the current location of a certain object?

- Features:
  - Continuously changing data
  - Real-time query support is required
  - Index structures should be update-tolerant

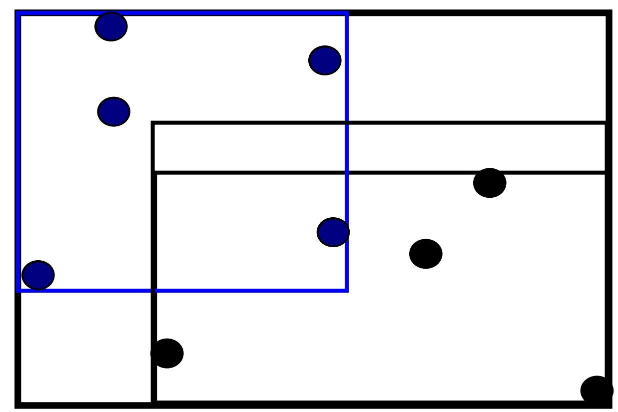
- Present data is always accessed through continuous queries

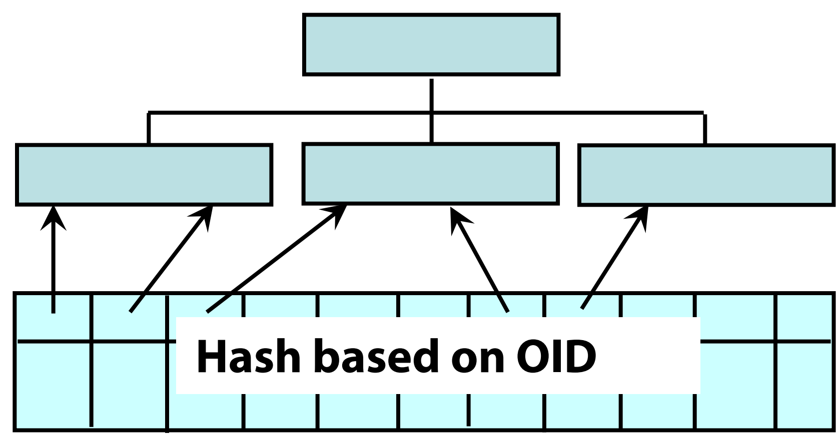


Updating Index Structures

- Traditional R-tree updates are *top-down*
- Updates translated to delete and insert transactions

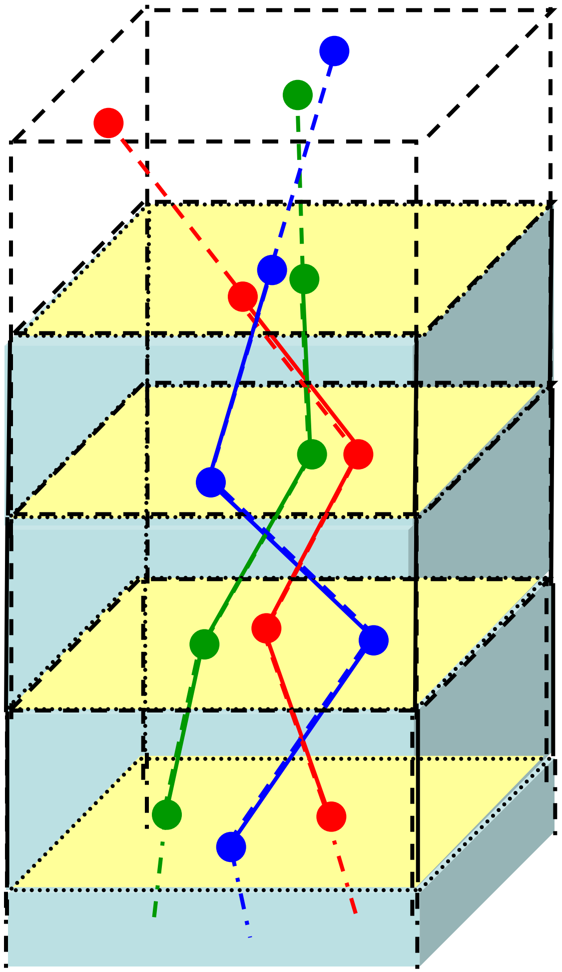
- To support frequent updates:
  - Updates can be managed "inline" without the need for deletion or insertions
  - *Bottom-up* approaches through auxiliary index structures to locate the object identifier





Querying the Future

- Examples:
  - *What will my nearest restaurant be after 30 minutes?*
  - *Does my path conflict with any other cars for the next hour?*
- Features:
  - Predict the movement through a velocity vector
  - Prediction could be valid for only a limited time horizon in the future



Example: Location Prediction

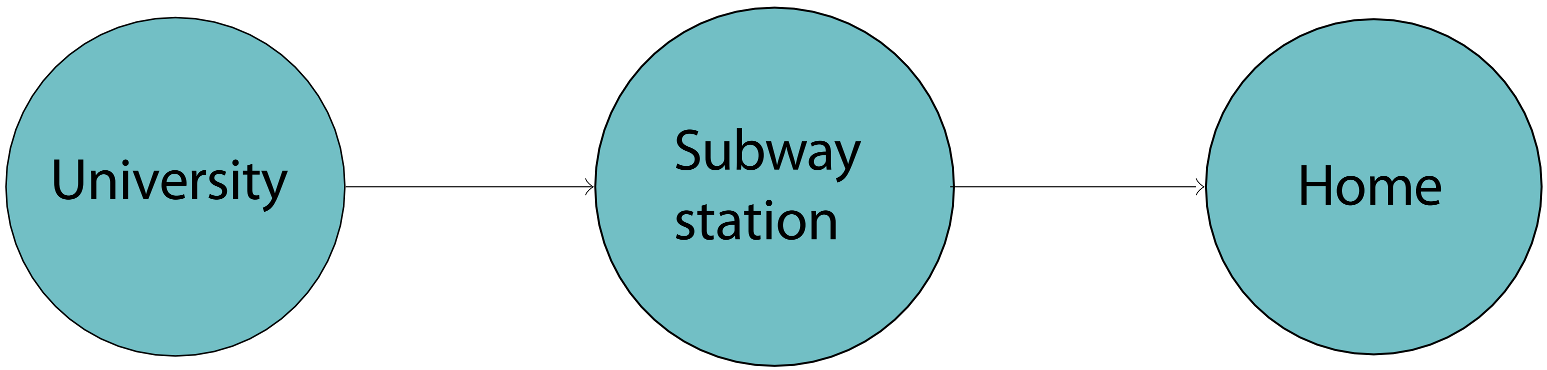
Location prediction seems to be a simple task in some cases:

Jonas Lüthke. Location Prediction Based on Mobility Patterns in Location Histories. Master thesis, TU Hamburg-Harburg, **2013**

https://www.ifis.uni-luebeck.de/~moeller/publist-sts-pw-and-m/source/papers/2013/luethke13.pdf

Location Prediction - Approach

Location prediction seems to be a simple task in some cases:



Previous observations can enable an educated guess

Example: Location History Data

Cabspotting data set:

- GPS coordinates collected from 563 cabs in San Francisco over 30 days
- Interval between measurements < 60seconds
- Ten taxis selected for testing (with regard to measurement density, measurement errors)

- Spatial probability distribution could be estimated from this (e.g., GMM)
- *Spatiotemporal* probability distribution is needed

Delay Embedding

Embed location time series in $2m$-dimensional space using a delay v:

- Time series is iteratively sampled using delay time v
- Every m subsequent locations are combined into one vector (*delay vector*)

Starting from each location x_n, combine x_n with m subsequent locations if they were observed at a time interval v

$$\boldsymbol{x}_n = (x_n^1, x_n^2) \text{ location data points, index } n \in \{1, \ldots, N\}$$

$$\boldsymbol{\delta}_n = [x^1_{n-(m-1)}, x^2_{n-(m-1)}, x^1_{n-(m-2)}, x^2_{n-(m-2)}, \ldots, x^1_n, x^2_n]$$

For example: $m = 2 : \boldsymbol{\delta}_n = [x^1_{n-1}, x^2_{n-1}, x^1_n, x^2_n]$

Note: Prior sampling with delay v is omitted for simplification

Delay Embedding – Benefits

- Euclidean distance is a measure for similarity between subsequences
- Similar subsequences are close in embedding space
- Density is a measure for likelihood of a subsequence
- Mobility patterns can be extracted in terms of density

Prediction Approach

Learn mobility patterns from large amount of history data:

- Delay embedding to map mobility patterns to density
- Density estimation based on embedding space

$$P(X_t = x, X_{t-1}, \dots, X_{t-(m-1)})$$

- Derive conditional distribution

$$P(X_t = x | X_{t-1}, \dots, X_{t-(m-1)}) = \alpha \; P(X_t = x, X_{t-1}, \dots, X_{t-(m-1)})$$

Predict location given the last $m-1$ locations (current context):

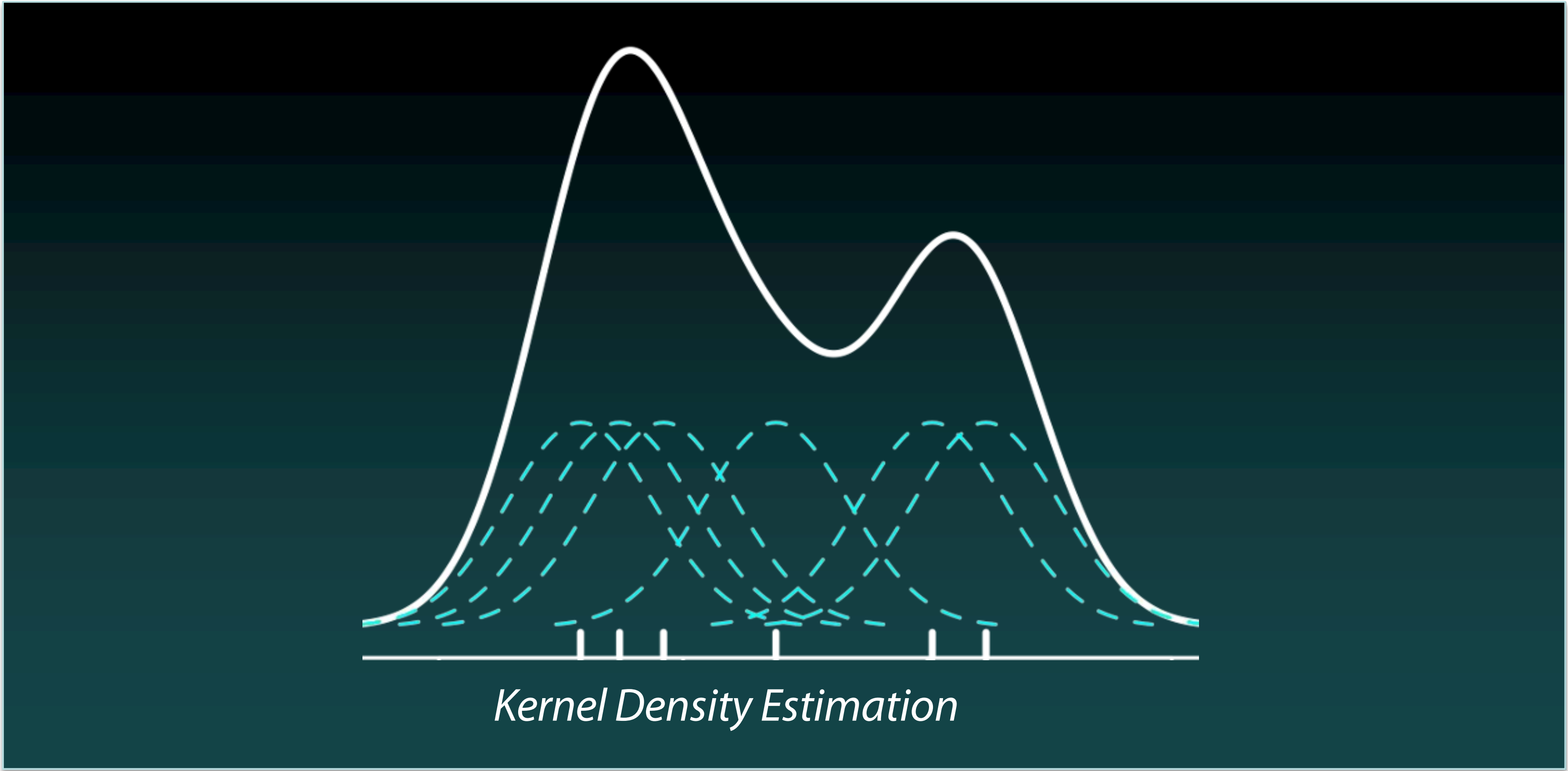
- Maximization of probability density to obtain most likely location (MLL problem)

$$x^* = \underset{x}{\operatorname{argmax}} \, P(X_t = x, X_{t-1}, \dots, X_{t-(m-1)})$$

Assuming $(m-1)$-th order Markov process

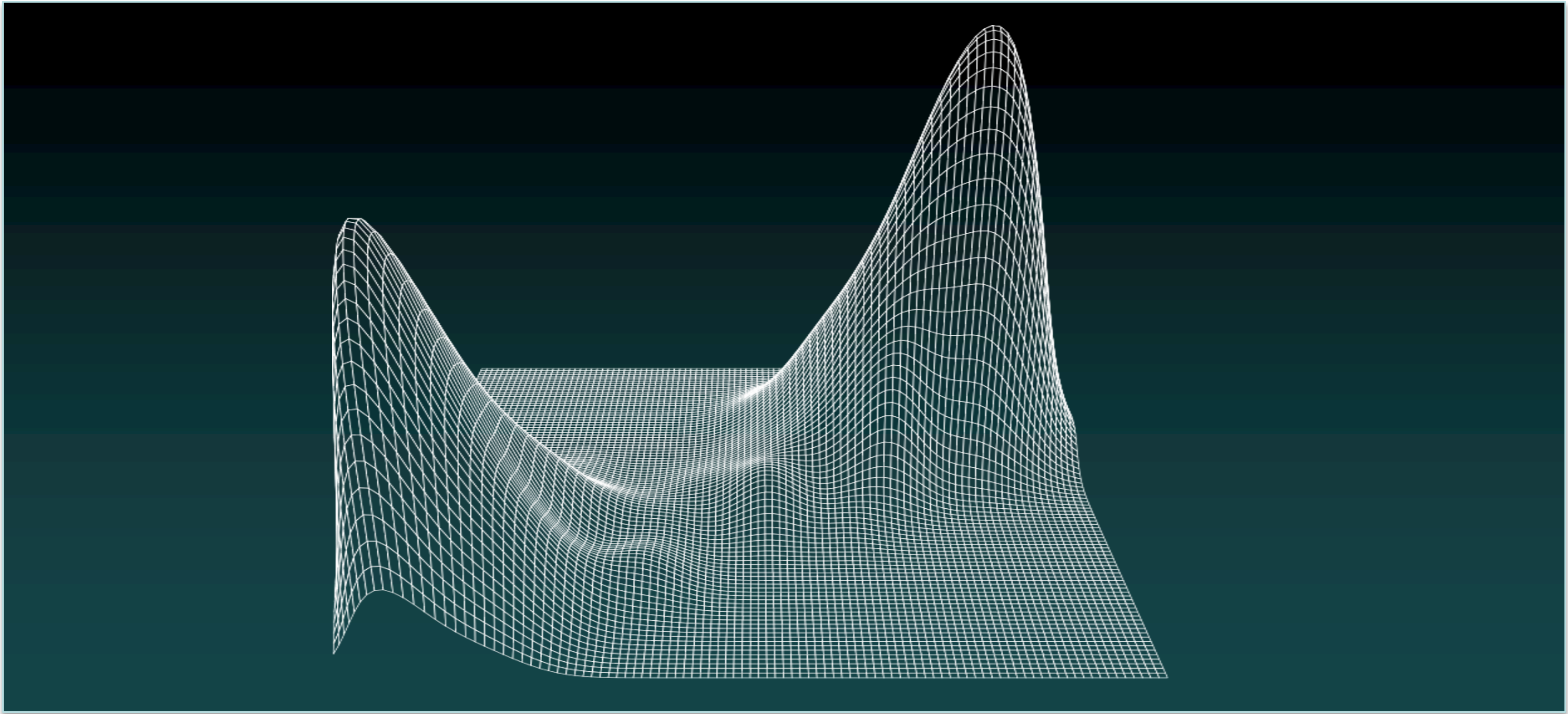
What about m=2?

Density Estimation



Optimization problem:
Minimize distance between estimated and unknown underlying distribution (AMISE, asymptotic mean integrated square error)

Gaussian Mixture Models



$$P(\mathbf{x}) = \sum_{m \in M} \omega_m \, N(\mathbf{x} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$$

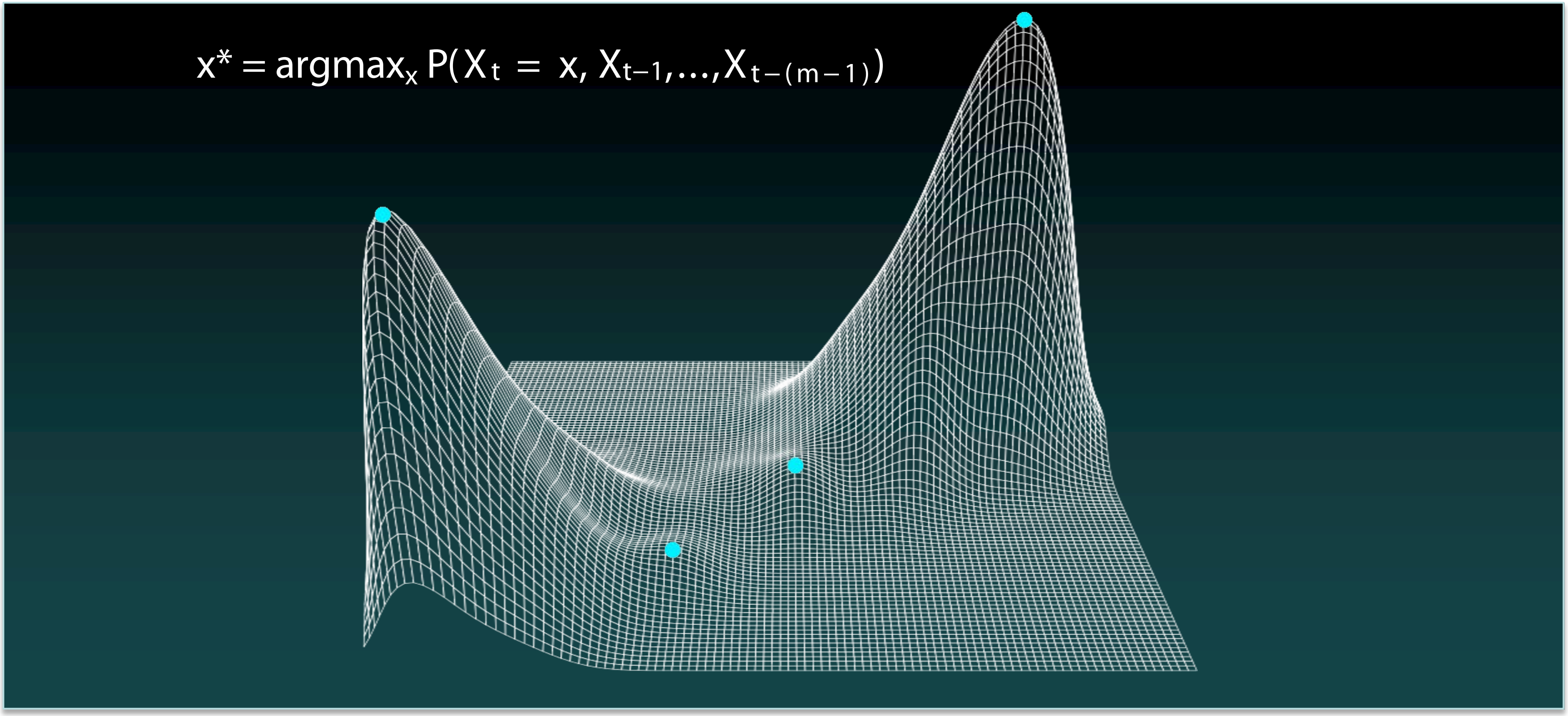
Online Kernel Density Estimation

- Incremental - can be updated as new data arrives
- Uses compression to keep memory footprint small

Christoph Heinz, Kernel Density Estimation over Data Streams, Dissertation Philipps-Universität Marburg, **2007**

Matej Kristan, Aleš Leonardis, and Danijel Skočaj. 2011. Multivariate online kernel density estimation with Gaussian kernels. Pattern Recogn. 44, 10-11, 2630-2642, **2011**

Solving MLL: Mode Finding

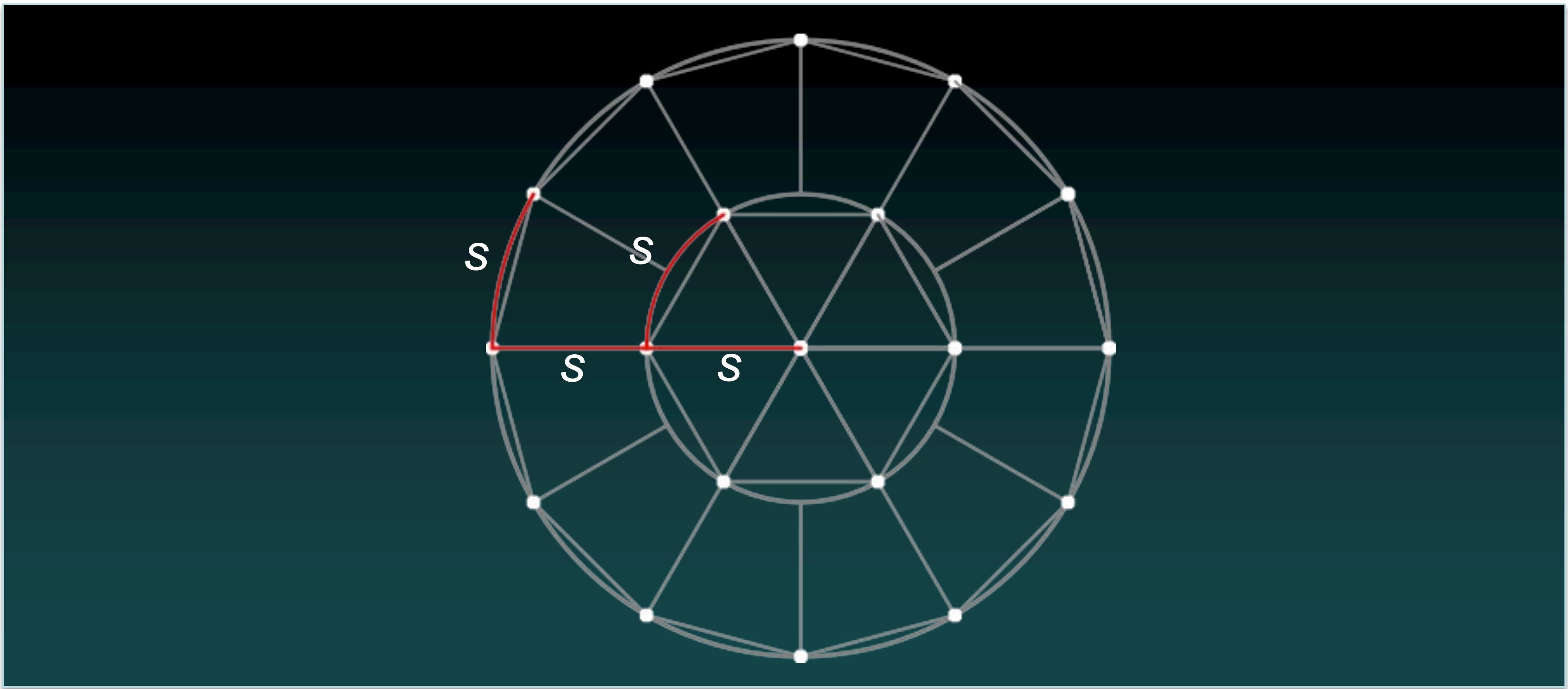


- Use hill-climbing search to find position of maximum
- Starting points?

Miguel Á. Carreira-Perpiñán. 2000. Mode-Finding for Mixtures of Gaussian Distributions. IEEE Trans. Pattern Anal. Mach. Intell. 22, 11, 1318-1323, **2000**

Starting Points for Maxima Search

- Define search region around last observed location
- If radius large enough, all relevant maxima are found

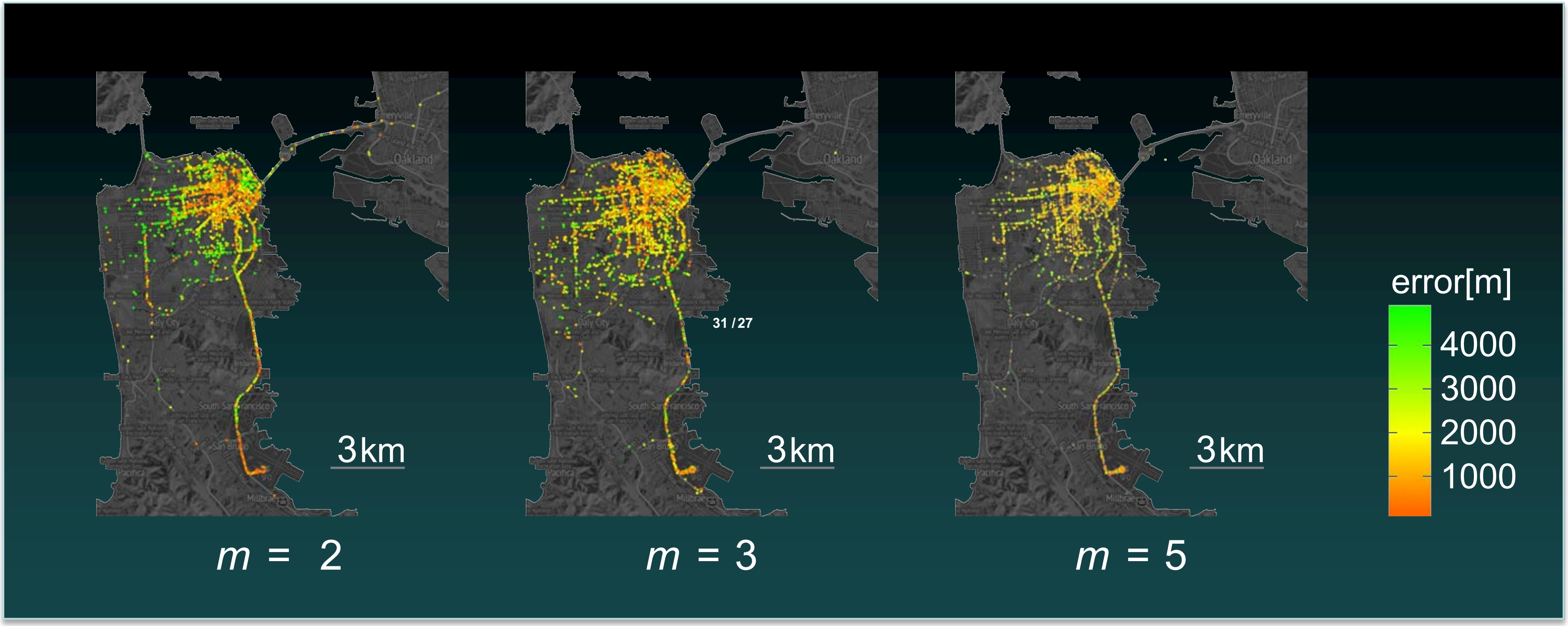


Summary - Prediction

- Delay embedding:
  Map mobility patterns to density
- Density estimation:
  Assigns probability to each possible location sequence
- Mode finding:
  Searches the most likely future location

Test Results

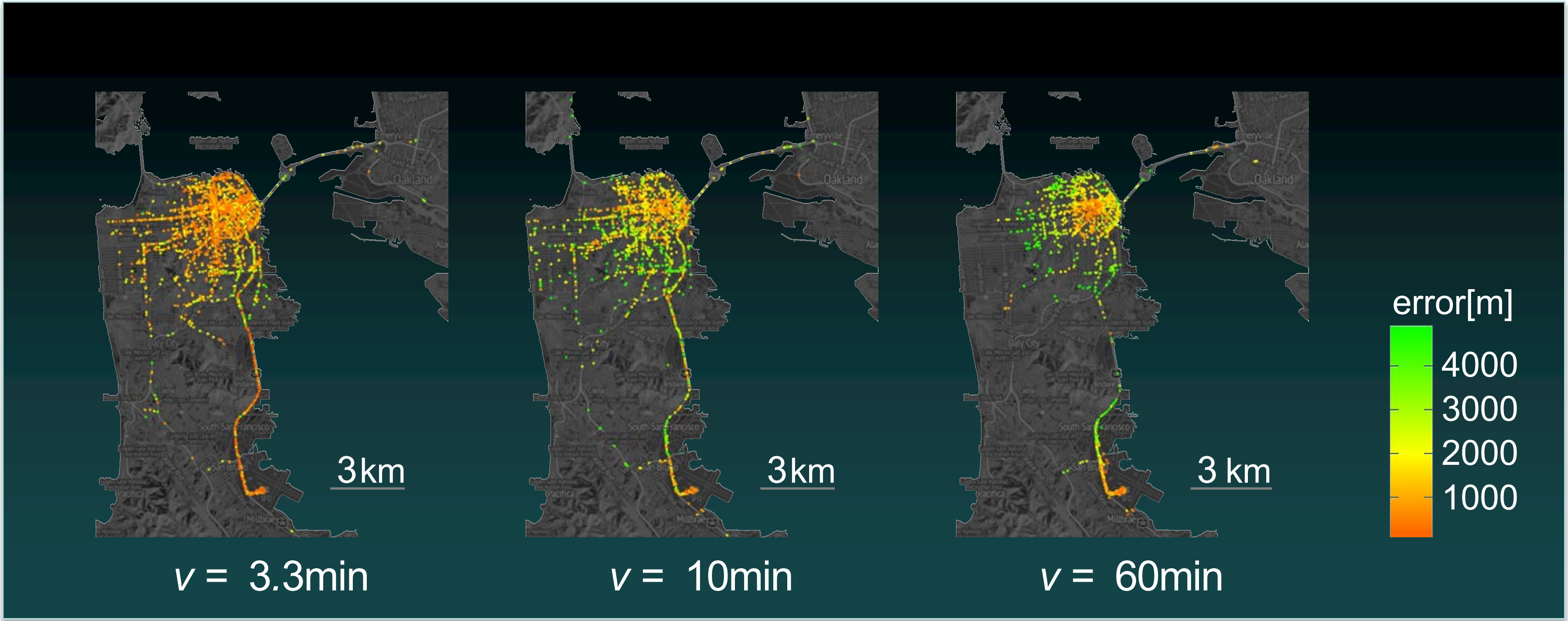
Varied m, fixed $v = 6$min:



Accurate predictions are more uniformly distributed for $m = 3$ and $m = 5$.

Test Results

Varied v, fixed $m = 3$:



Accurate predictions are increasingly clustered as v increases.

Test Result Analysis

- Algorithm is based on *sequential correlation* in data (delay embedding)
- Locations in taxi data only correlated if part of same trip
- For each trip the client defines new destination
- Recurring similar location sequences only observed when limiting time span to average trip time
- Else prediction falls back to $m = 2$

Similar Approaches:

- Song et al. - Markov predictor
- Scellato et al. - Nonlinear predictor

L. Song, D. Kotz, R. Jain, and X. He, Evaluating location predictors with extensive with mobility data, In Proc. IEEE Computer and Communications Societies, pp. 1414-1424, **2004**

S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. T. Campbell, NextPlace: a spatio- temporal prediction framework for pervasive systems, In: Proc. Pervasive Computing, **2011**

Duality Transformation: Avoid 3D-Rtrees?

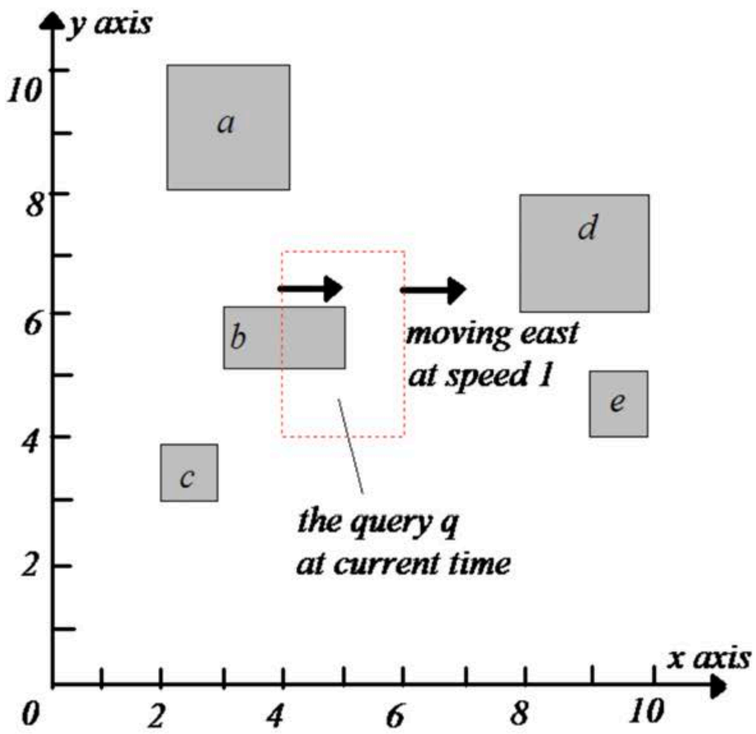
- A linear trajectory in two-dimensional space can be transformed into a point in another *dual* two-dimensional space
- Trajectory: $x(t) = vt + a$ ➔ Point: (v,a)
- Embedding in more dimensions
- All queries will need to be transformed into the dual space

Non-Standard-Datenbanken und Data Mining

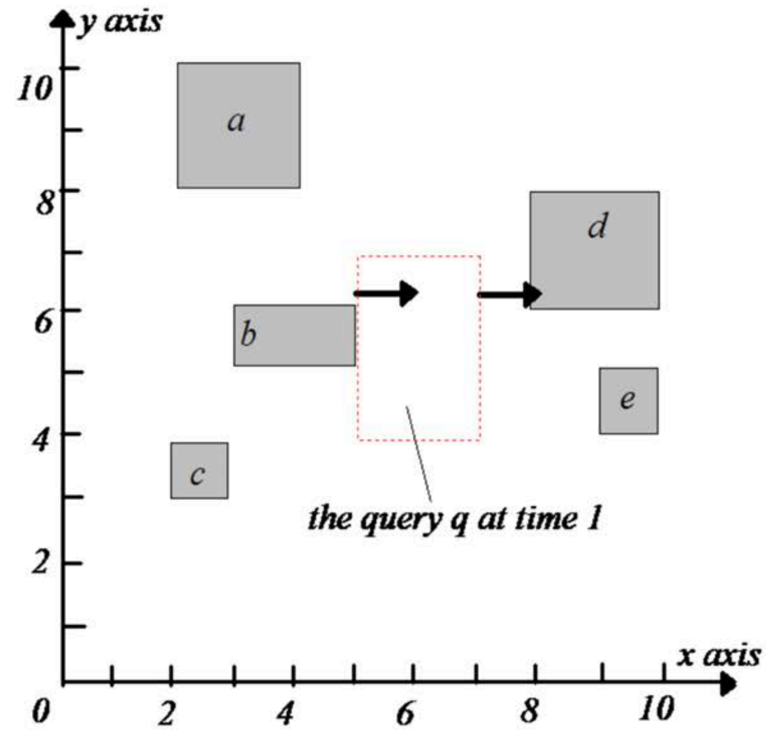
Probabilistic Spatio-Temporal
Databases and Streams

Prof. Dr. Ralf Möller
Universität zu Lübeck
Institut für Informationssysteme

Time Parameterized Queries



- Result={b}
- Conventional Query



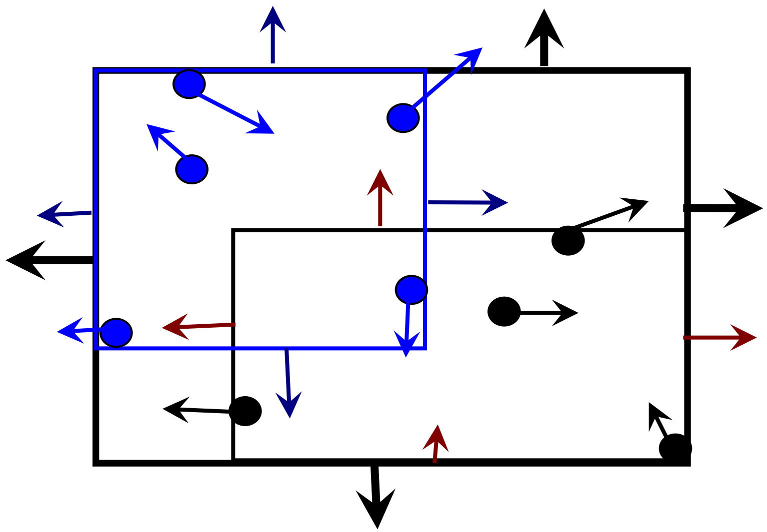
- At time 1 b would be the nearest neighbor, after that time the results expire and d would be the new nearest neighbor
- Time Parameterized Query

Time Parameterized queries (TP queries)

- Whenever a query is issued, a TP returns:
  - Actual result that satisfies the corresponding spatial query.
  - Validity period/expiration time of the result.
  - Change that cause the expiration of the results
- Can be used for prediction

Time-Parameterized Data Structures

- The Time-Parameterized R-tree (TPR-tree) consists of:
  - Minimum bounding rectangles (MBR)
  - Velocity bounding rectangles (VBR)



- A bounding rectangle with MBR & VBR is guaranteed to contain all its moving objects as long as they maintain their velocity vector

- Optimization: Minimize area of the bounding rectangle

- Time-Parameterized Bounding Rectangles (TPBRs) for answering TP queries

Indexing Past, Present, and Future

- A unified index structure for both past, present, and future data
- Makes use of the partial-persistent R-tree for past data and the TPR-tree for current and future data

Katerina Raptopoulou, Michael Vassilakopoulos, and Yannis Manolopoulos.. Efficient processing of past-future spatiotemporal queries. In Proc. ACM Symposium on Applied Computing (SAC '06). ACM, pp. 68-72, **2006**

Outline

- Location-aware Environments
- Location-aware *Snapshot* Query Processing
- Location-aware *Continuous* Query Processing
- Scalable Execution of Continuous Queries
- Location-aware Query Optimizer
- Uncertainty in Location-aware Query Processing

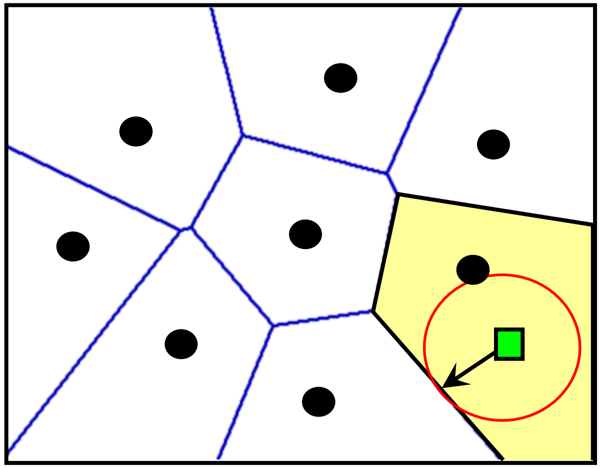
Approaches

- *Straightforward Approach*
  - Abstract the continuous queries to a series of snapshot queries evaluated periodically (and possibly incrementally)
- *Result Validation*
- *Result Caching*
- *Result Prediction*
- *Incremental Evaluation*

Result Validation

- Associate a *validation* condition with each query answer

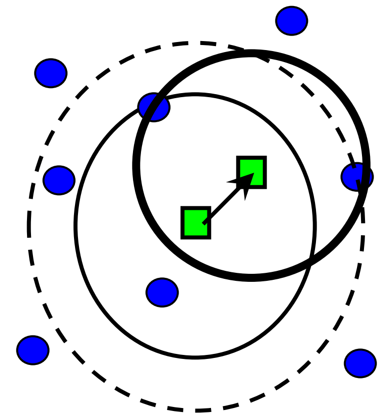
- *Valid time (t):*
  - The query answer is valid for the next *t* time units
- *Valid region (R)*
  - The query answer is valid as long as you are within a region *R*



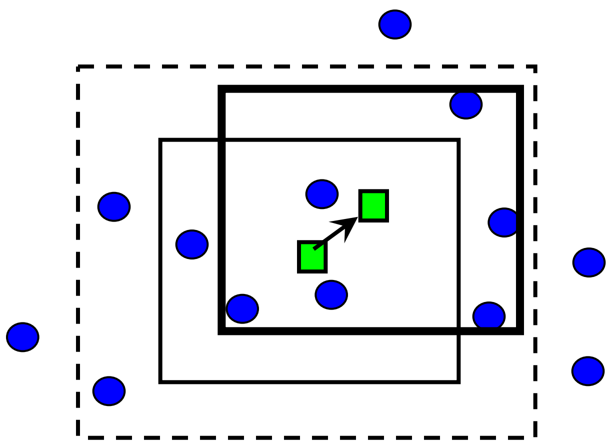
- It is challenging to maintain the computation of valid time/region for querying *moving objects*
- Once the associated validation condition expires, the query will be *reevaluated*

Caching the Result

- *Observation:* Consecutive evaluations of a continuous query yield very similar results



- *Idea:* Upon evaluation of a continuous query, retrieve more data that can be used later
- *K-NN query*
  - Initially, retrieve more than k

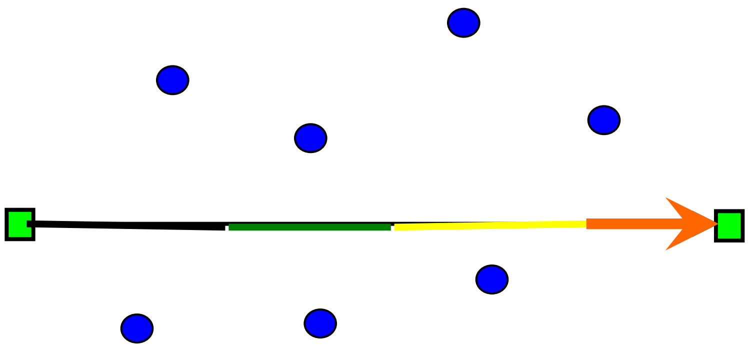


- *Range query*
  - Evaluate the query with a larger range
- How much do we need to pre-compute?
- How do we do re-caching?

Predicting the Result

- Given a future trajectory movement, the query answer can be pre-computed in advance

- The trajectory movement is divided into N intervals, each with its own query answers A_i



Nearest-Neighbor Query

- The query is evaluated once (as a snapshot query). Yet, the answer is valid for longer time periods

- Once the trajectory changes, the query will be reevaluated

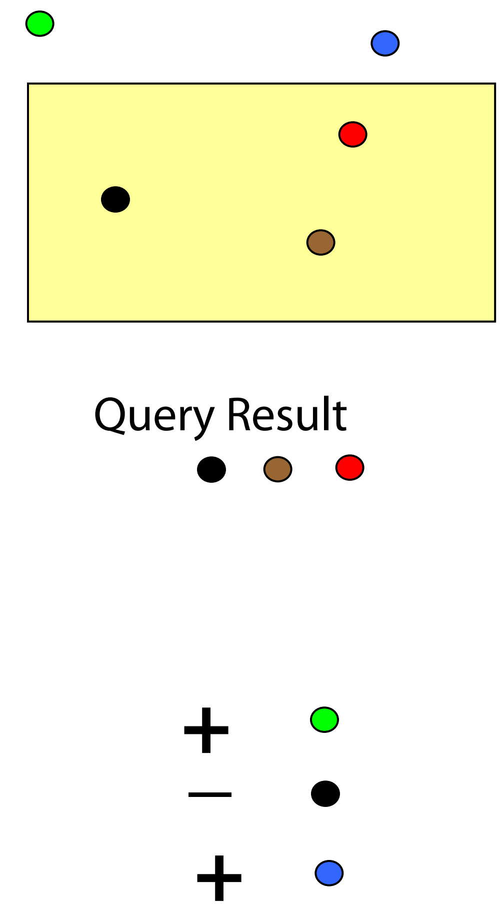
Incremental Evaluation

- The query is evaluated only once. Then, only the *updates* of the query answer are evaluated

- There are two types of updates. *Positive* and *Negative* updates

- Only the objects that cross the query boundary are taken into account

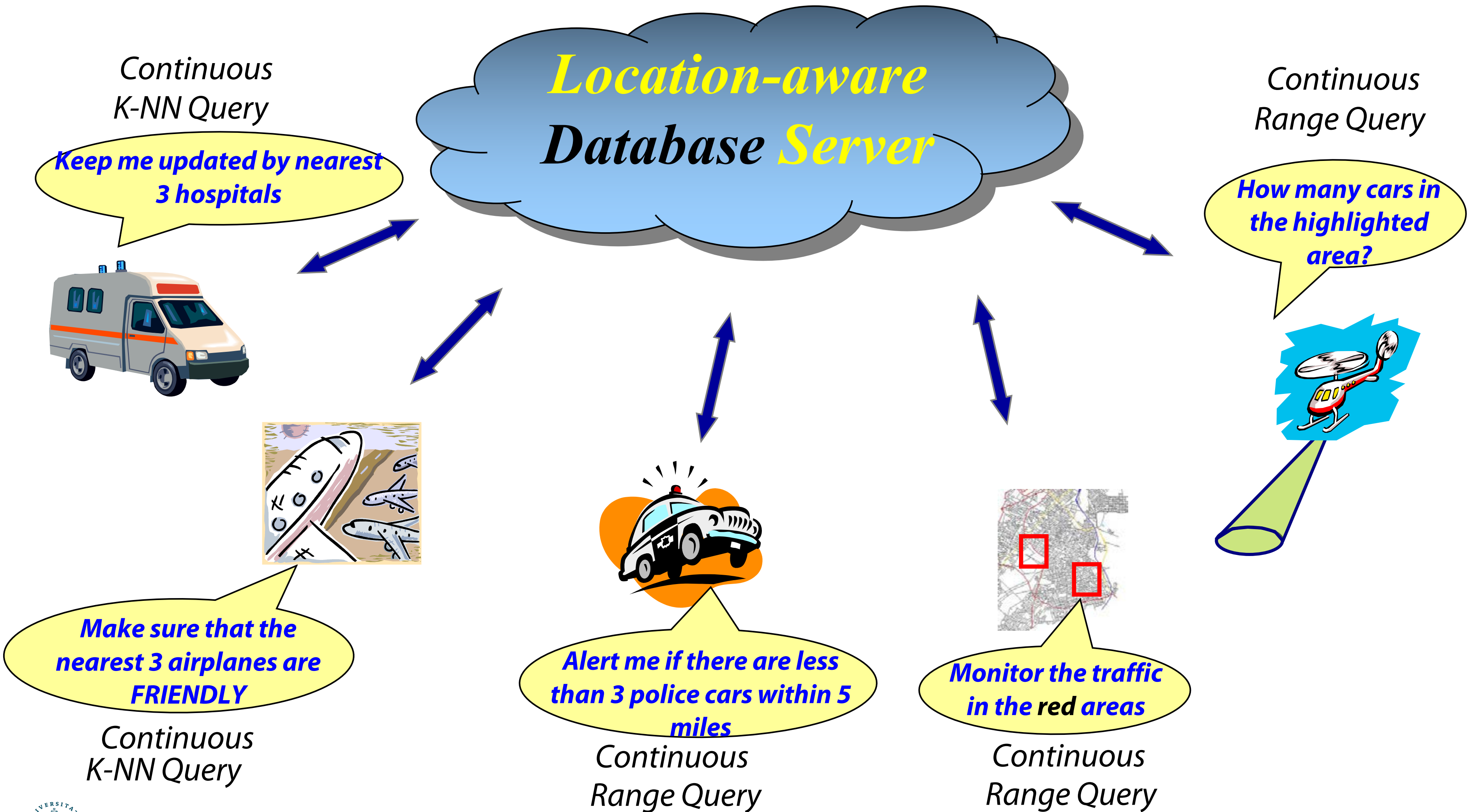
- Need to continuously listen for notifications that someone crosses the query boundary



Outline

- Location-aware Environments
- Location-aware *Snapshot* Query Processing
- Location-aware *Continuous* Query Processing
- Scalable Execution of Continuous Queries
  - Location-aware Centralized Database Systems
  - Location-aware Distributed Database Systems
  - Location-aware Data Stream Management Systems
- Location-aware Query Optimizer
- Uncertainty in Location-aware Query Processing

Queries as Data – Motivation



Main Concepts

Continuous queries last for long times at the server side

- → While a query is active in the server, other queries will be submitted
- ❑ *Shared execution among multiple queries*

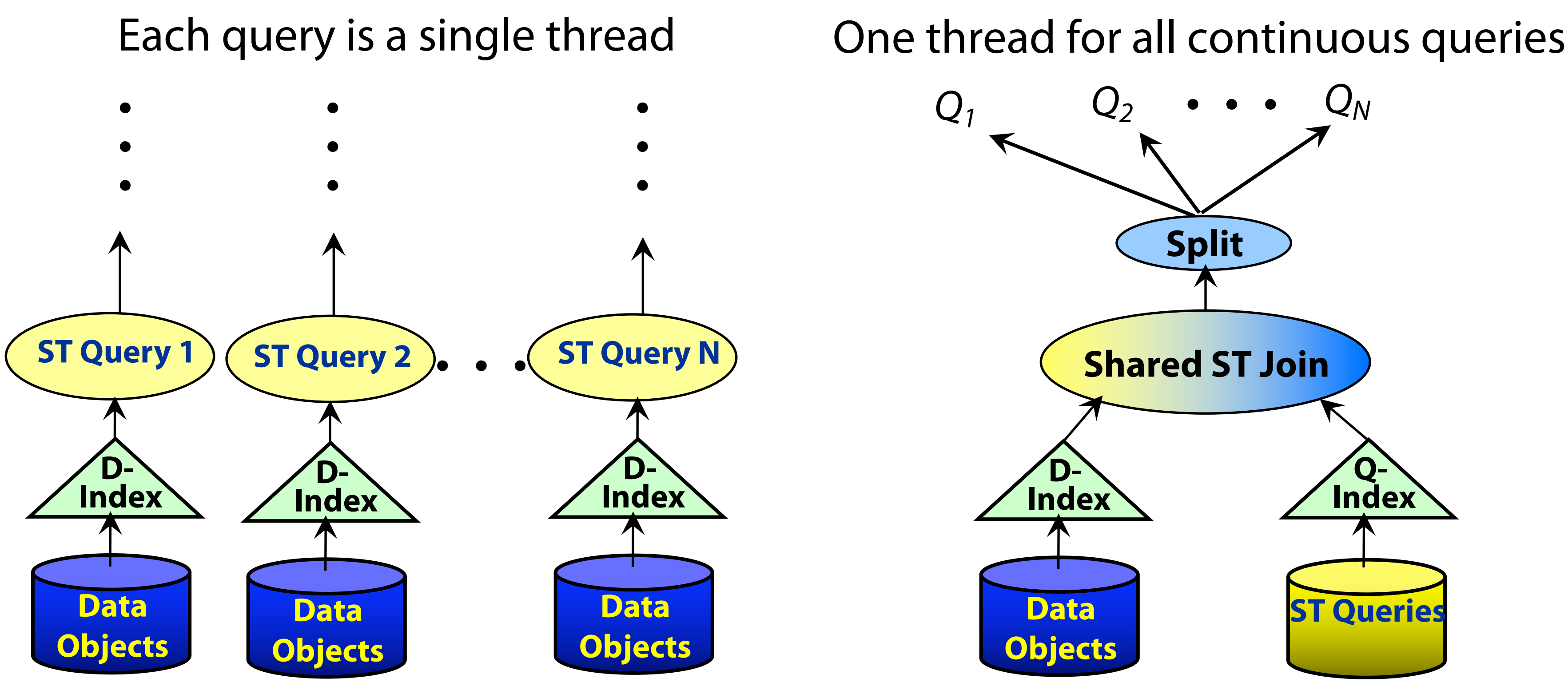
Should we index data OR queries?

- → Data and queries may be stationary or moving
- → Data and queries are of large size
- → Data and queries arrive to the system with very high rates
- ❑ *Treat data and queries similarly*

Queries are coming to data OR data are coming to queries?

- → Both data and queries are subjected to each other
- ❑ *Join data with queries*

Main Concepts (Cont.)

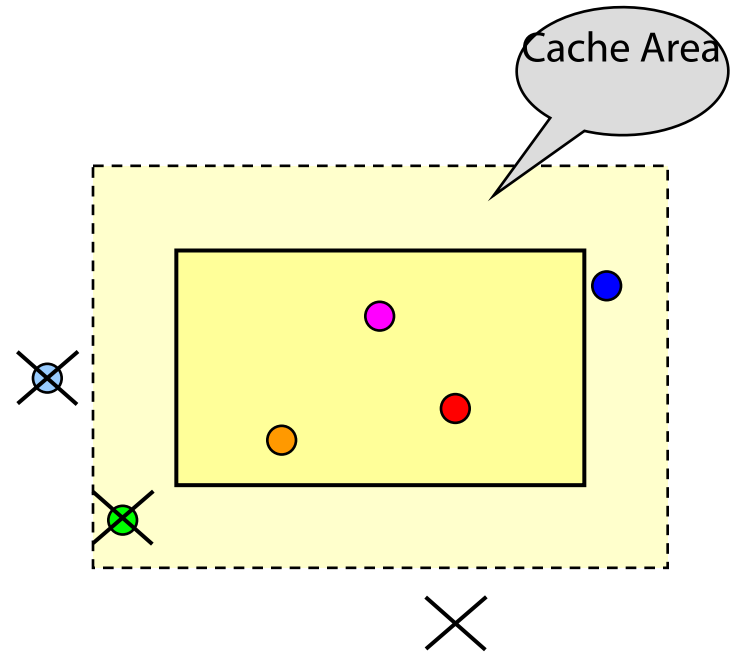


- Evaluating a large number of concurrent continuous spatio-temporal queries is abstracted as a *spatio-temporal join* between moving objects and moving queries

Location-aware Data Stream Management Systems

- Only *significant* objects are stored in-memory

- An object is considered *significant* if it is either in the query area or the cache area



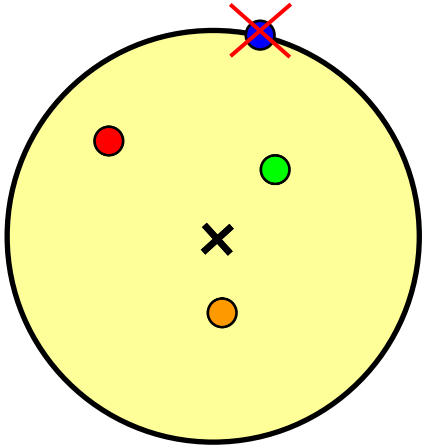
- Due to the query and object movements, a stored object may become *insignificant* at any time

- Larger cache area indicates more storage overhead and more accurate answer

Location-aware Data Stream Management Systems (Cont.)

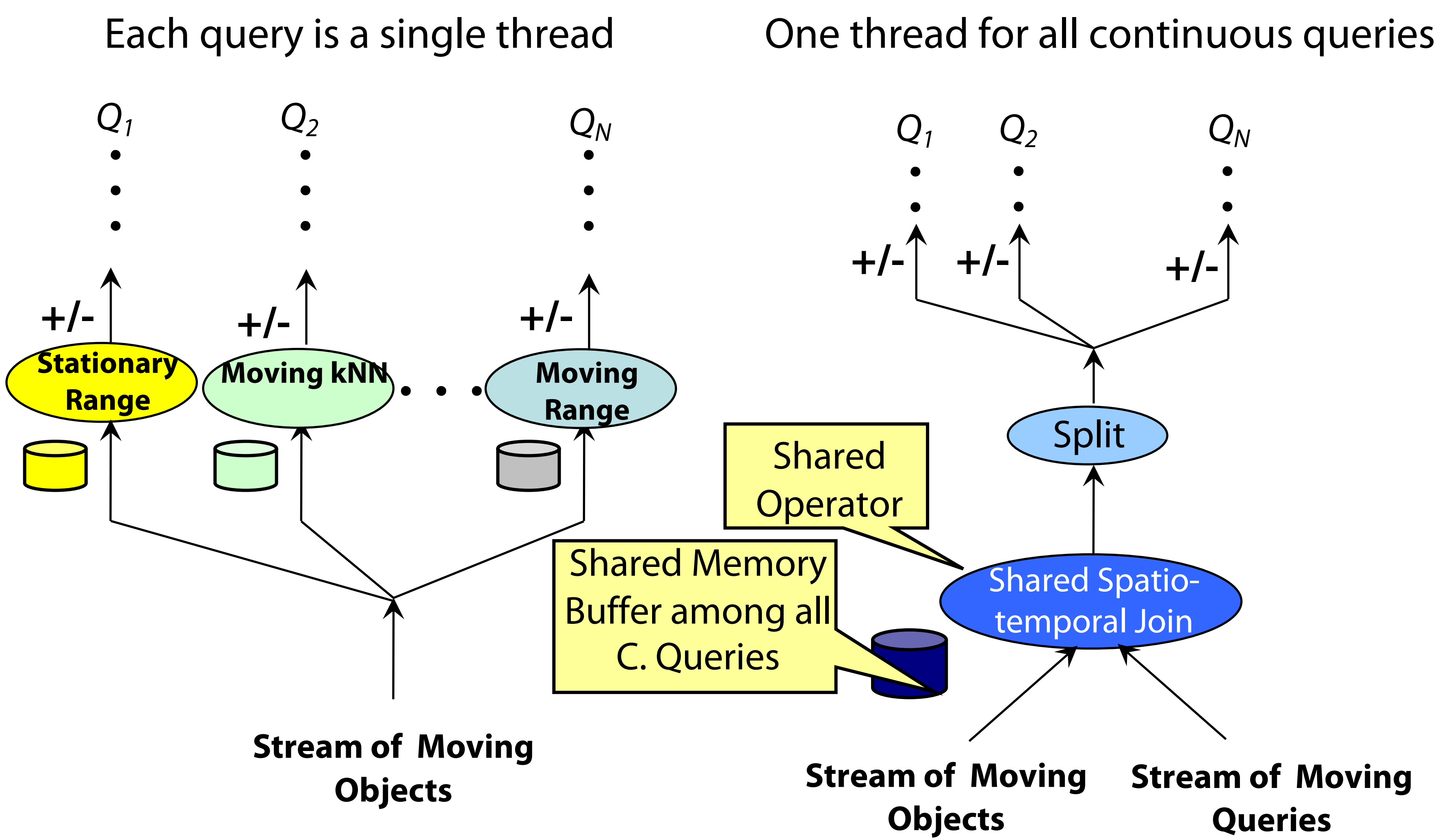
- The first *k* objects are considered an initial answer
- *K*-NN query is reduced to a circular range query

However, the query area may shrink or grow



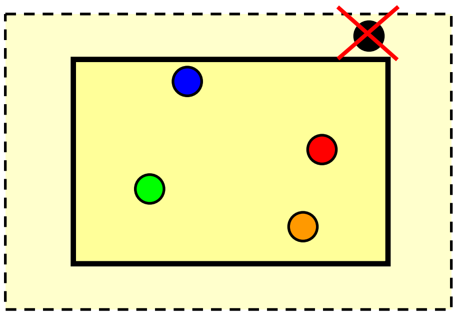
$K = 3$

Location-aware Data Stream Management Systems (Cont.)

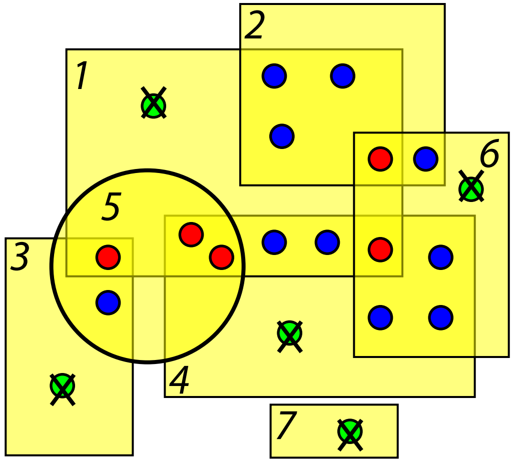


Location-aware Data Stream Management Systems (Cont.)

- Query Load Shedding
  - Reduce the cache area
  - Possibly reduce the query area
  - *Immediately* drop *insignificant* tuples
  - Intuitive and simple to implement



- Object Load Shedding
  - Objects that satisfy less than *k* queries are *insignificant*
  - *Lazily* drop *insignificant* tuples
  - *Challenge I:* How to choose *k?*
  - *Challenge II:* How to provide a lower bound for the query accuracy?



K = 2

Tutorial Outline

- Location-aware Environments
- Location-aware *Snapshot* Query Processing
- Location-aware *Continuous* Query Processing
- Scalable Execution of Continuous Queries
- Location-aware Query Optimization
- Uncertainty in Location-aware Query Processing

Location-aware Query Optimization

- Spatio-temporal pipelinable query operators
  - Range queries
  - Nearest-neighbor queries

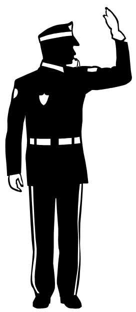
- Selectivity estimation for spatio-temporal queries/operators
  - Spatio-temporal histograms
  - Sampling

- Adaptive query optimization for continuous queries

Spatio-temporal Query Operators

- Existing Approaches are Built on Top of DBMS (at the Application Level)

Continuously report the ***trucks*** in this area



Scalar functions
(Stored procedure)

Only produce objects in the

The performance of scalar functions is limited

Engine

SELECT *O. ID*
FROM *Objects O*
WHERE *O.type* = ***truck***
INSIDE ***Area A***

Database Engine

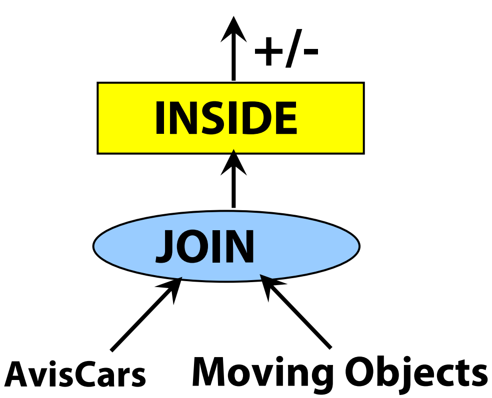
Spatio-temporal Operators

Spatio-temporal Query Operators

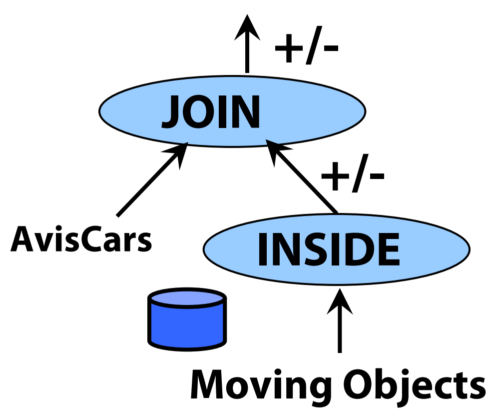
- *"Continuously report the Avis cars in a certain area"*

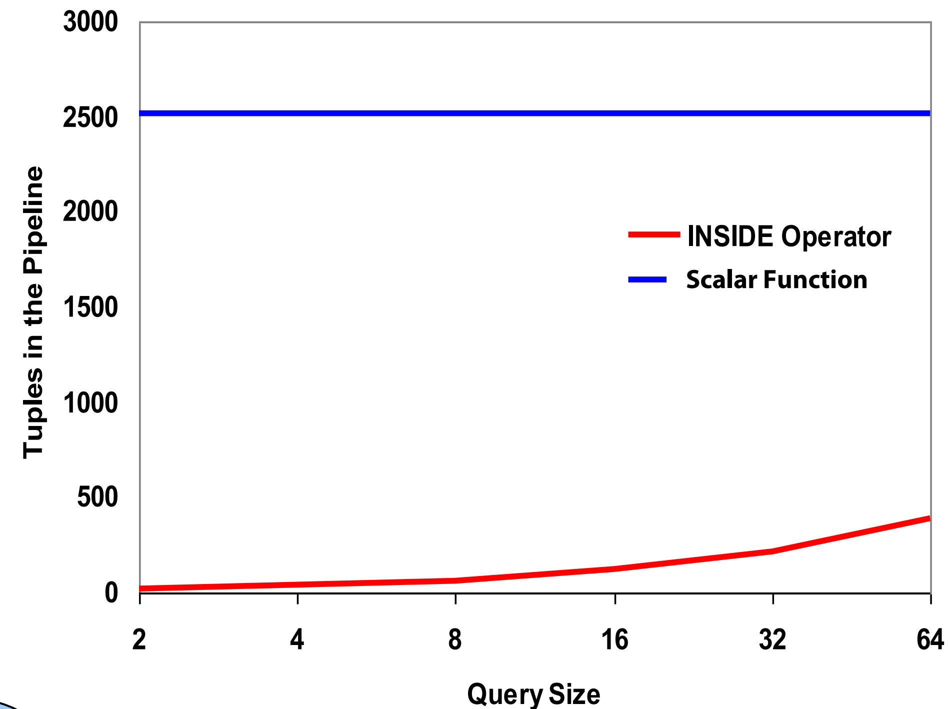
SELECT *M.ObjectID*
FROM *MovingObjects M, AvisCars A*
WHERE *M.ID = A.ID*
INSIDE *RegionR*

Scalar Function



Spatio-temporal Operators

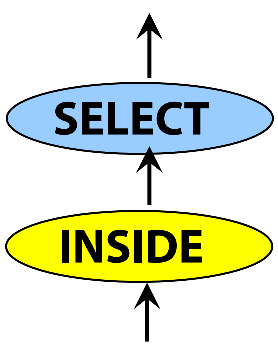


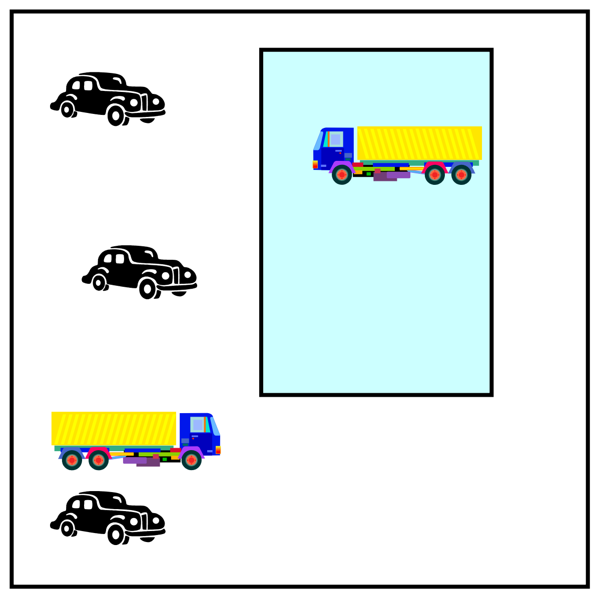


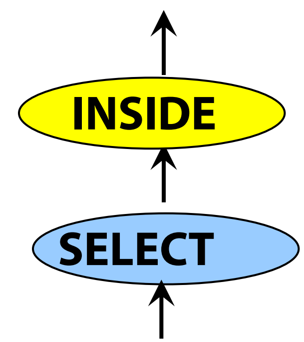
Spatio-temporal Selectivity Estimation

- Estimating the selectivity of spatio-temporal operators is crucial in determining the best plan for spatio-temporal queries

SELECT *ObjectID*
FROM *MovingObjects M*
WHERE *Type = Truck*
INSIDE *Region R*

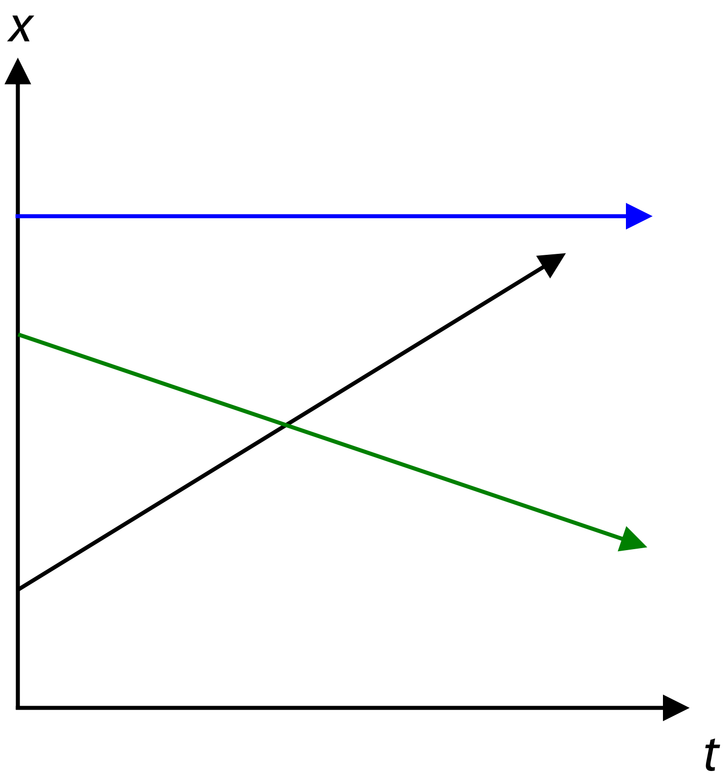


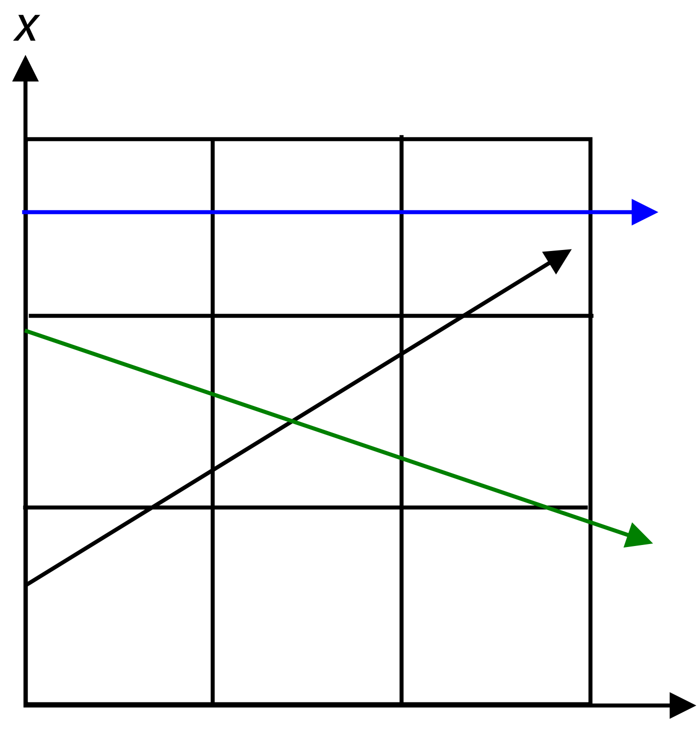




Spatio-temporal Histograms

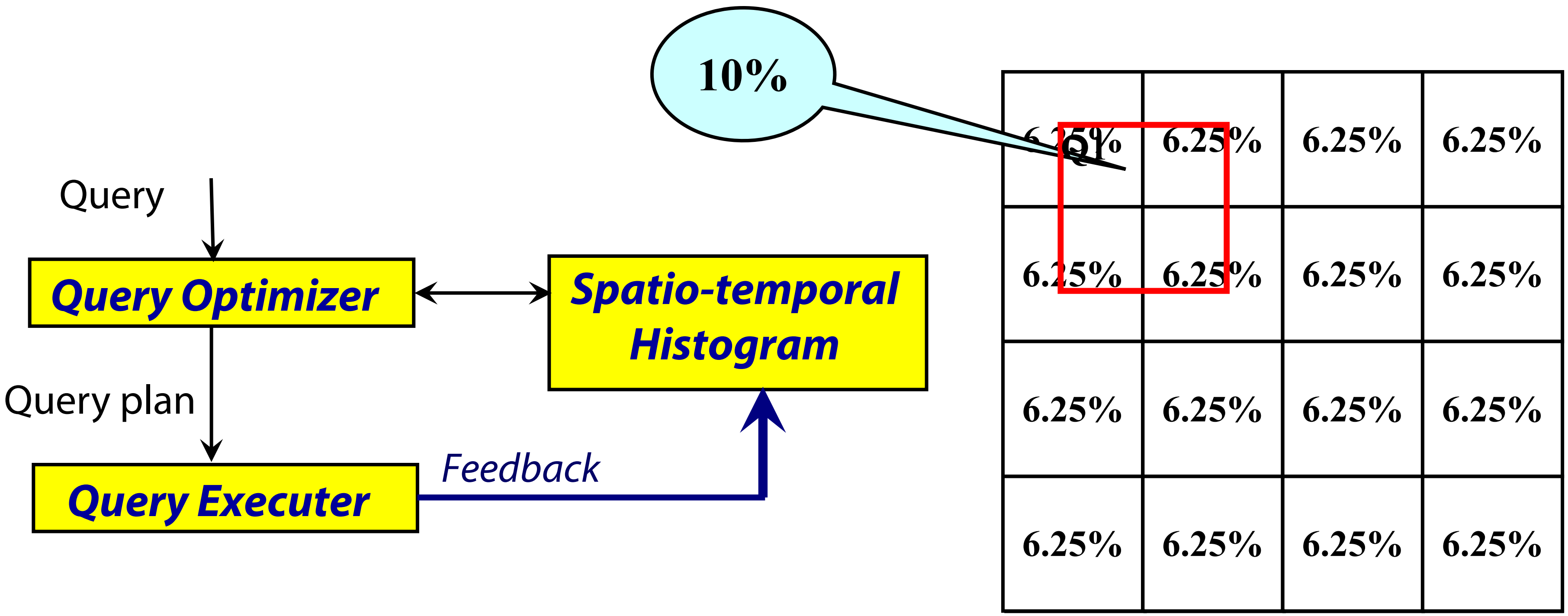
- Moving objects in D-dimensional space are mapped to 2D-dimensional histogram buckets





Spatio-temporal Histograms with Query Feedback

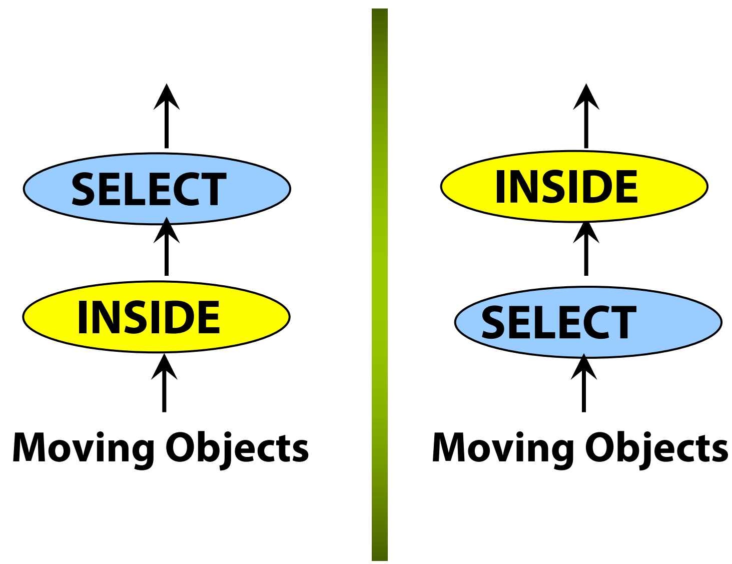
- Estimating the selectivity of spatio-temporal operators is crucial in determining the best plan for spatio-temporal queries



Adaptive Query Optimization

- Continuous queries last for long time (hours, days, weeks)
  - ➔ Environment variables are likely to change
  - ➔ The initial decision for building a query plan may not be valid after a while
- Need continuous optimization and ability to change the query plan:
  - ➔ Training period: Spatio-temporal histogram, periodicity mining
  - ➔ Online detection of changes

```
SELECT ObjectID
FROM   MovingObjects M
WHERE  Type = Truck
INSIDE Region R
```



Non-Standard-Datenbanken und Data Mining

Probabilistic Spatio-Temporal Databases and Streams

Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

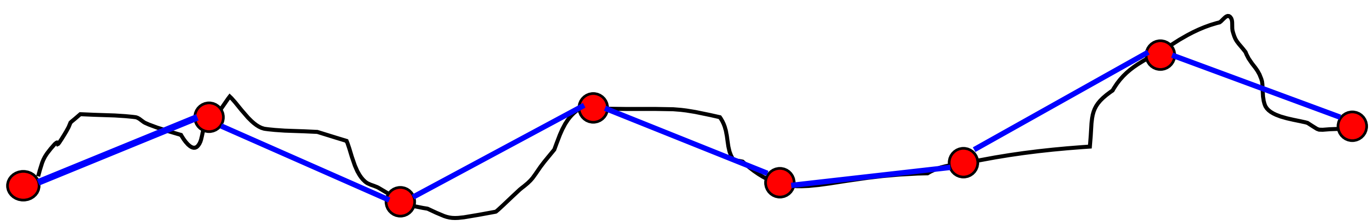
Uncertainty in Moving Objects

- Location information from moving objects is inherently inaccurate

- Sources of uncertainty:
  - Sampling. A moving object sends its location information once every *t* time units. Within any two consecutive locations, we have no clue about the object's exact location
  - Reading accuracy. Location-aware devices do not provide the exact location
  - Object movement and network delay. By the time that a certain reading is received by the server, the moving object has already changed its location

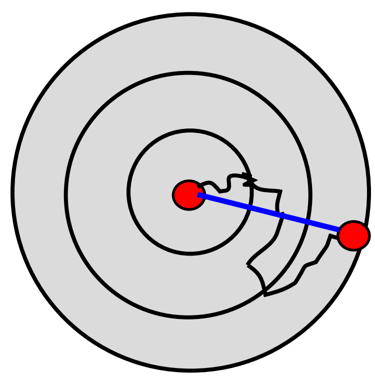
Uncertainty in Moving Objects

- Historical data (Trajectories)



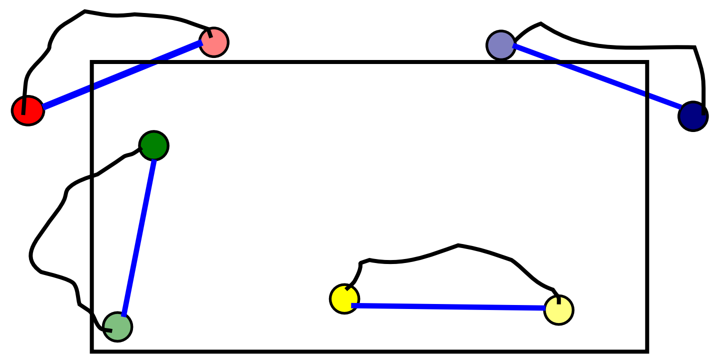
- Current data

$T_0+\epsilon_0$

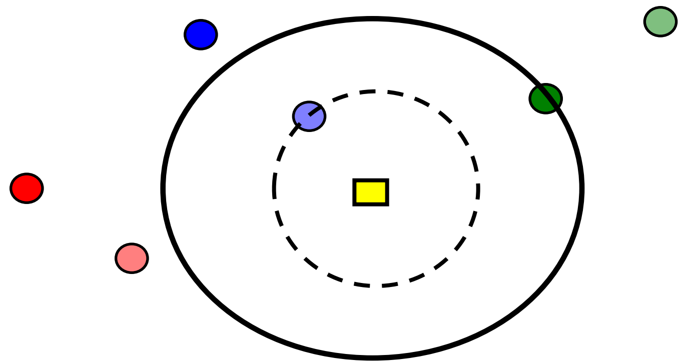


Error in Query Answer

- Range Queries



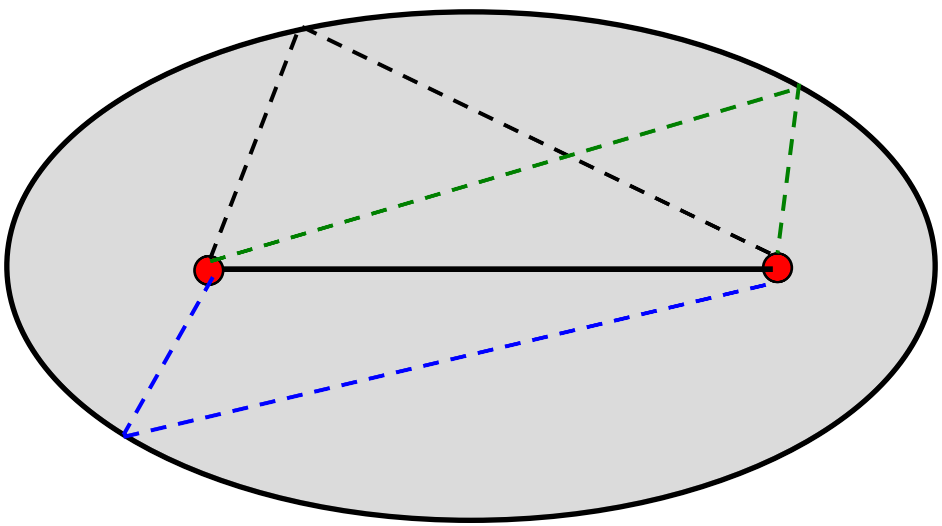
- Nearest Neighbor Queries



Representing Uncertain Data using Ellipses

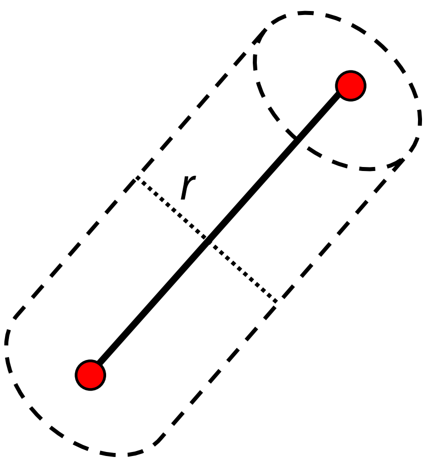
- Given :
  - Start point
  - End point
  - Maximum possible speed ➔ Maximum traveling distance S

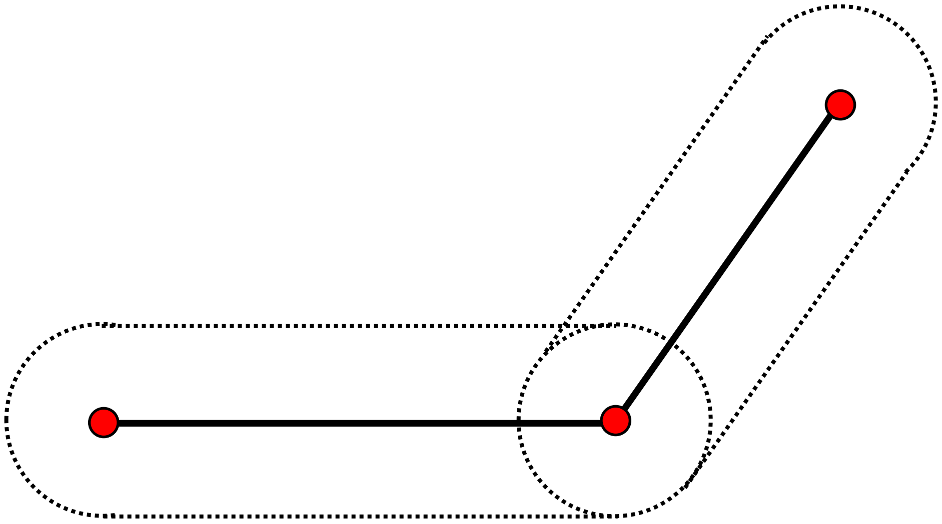
- If S is greater than the distance between the two end points, then the moving object may have deviated from the given route



Representing Uncertain Data using Cylinders

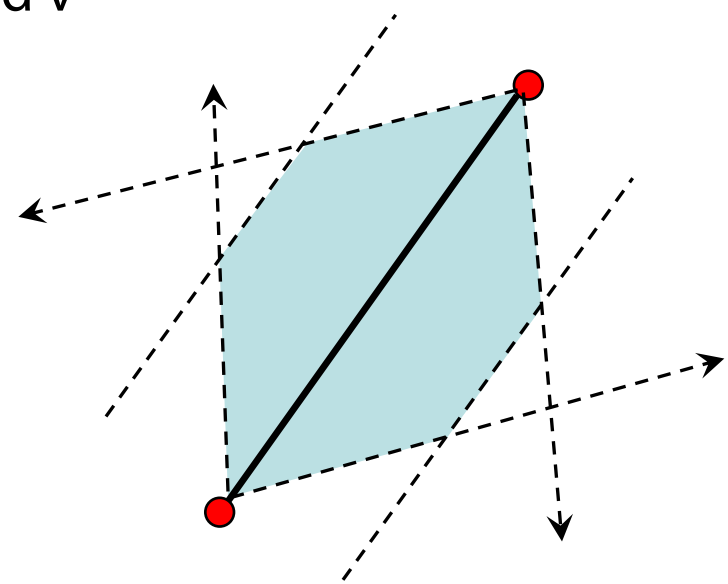
- Given:
  - Start and end points
- Constraint:
  - An object would report its location only if it is deviated by a certain distance r from the predicted trajectory





Representing Uncertain Data in Road Networks

- Given:
  - Start and end points
- Constraints :
  - Deviation threshold r
  - Speed threshold v

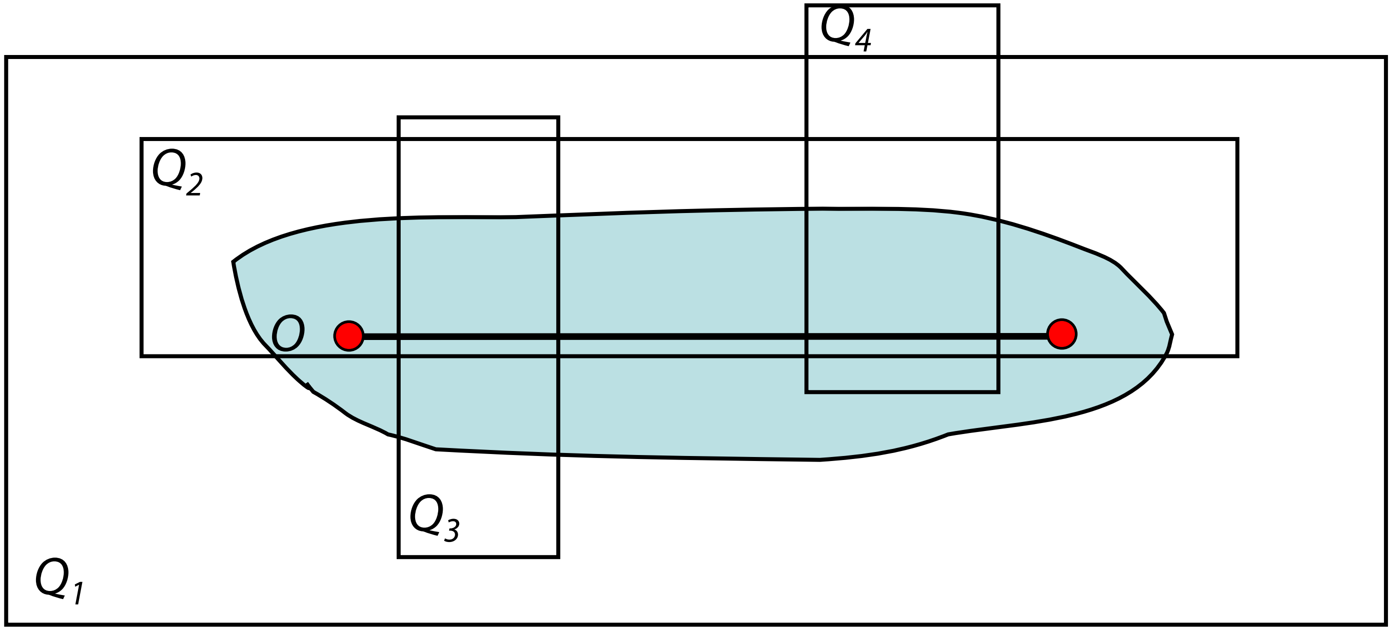


Querying Uncertain Data Uncertain Keywords

- KEYWORDS:
  - Probability: *possibly, definitely*
  - Temporal: *sometimes, always*
  - Spatial: *somewhere, everywhere*

- *Examples:*
  - *What are the objects that are possibly sometimes within area R at time interval T?*
  - *What are the objects that definitely passed through a certain region?*
  - *Retrieve all the objects that are always inside a certain region*
  - *Retrieve all the objects that are sometimes definitely inside region R*

Querying Uncertain Data Uncertain Keywords (Cont.)



- *Object O is definitely always in Q_1*
- *Object O is possibly always in Q_2*
- *Object O is definitely sometimes in Q_3*
- *Object O is possibly sometimes in Q_4*

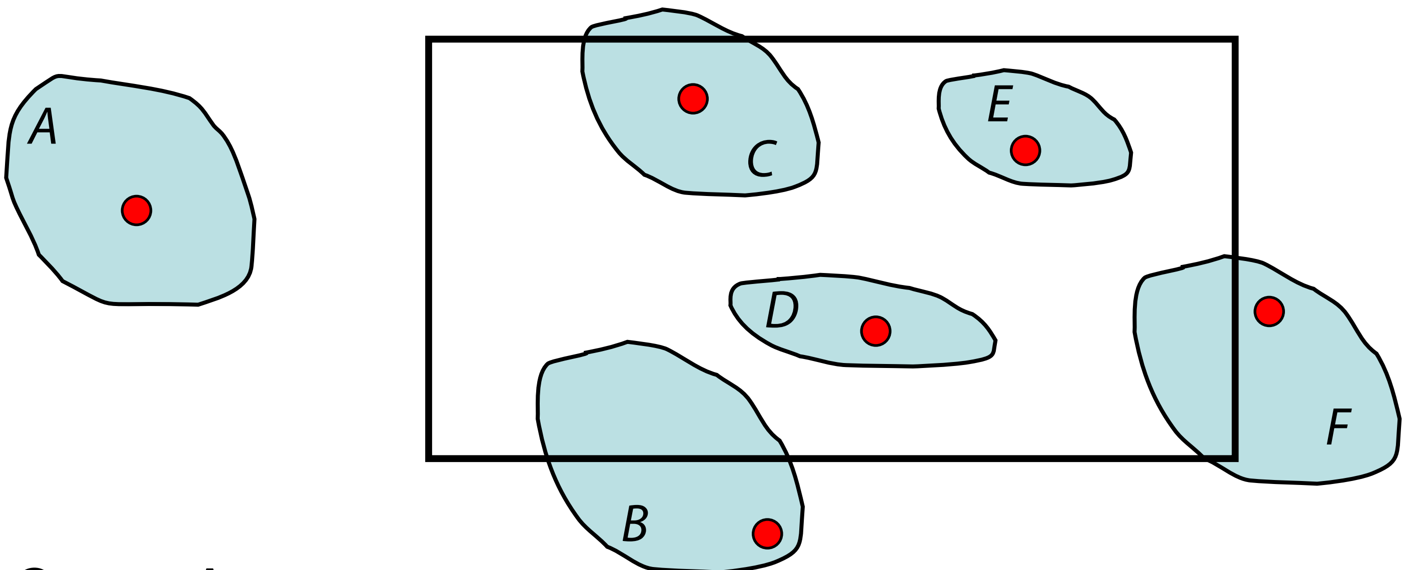
Querying Uncertain Data Probabilistic Queries

- With each query answer, associate a probability that this answer is true

- The answer set of a query Q is represented as a set of tuples <ID, p> where ID is the tuple identifier and p is the probability that the object ID belongs to the answer set of Q

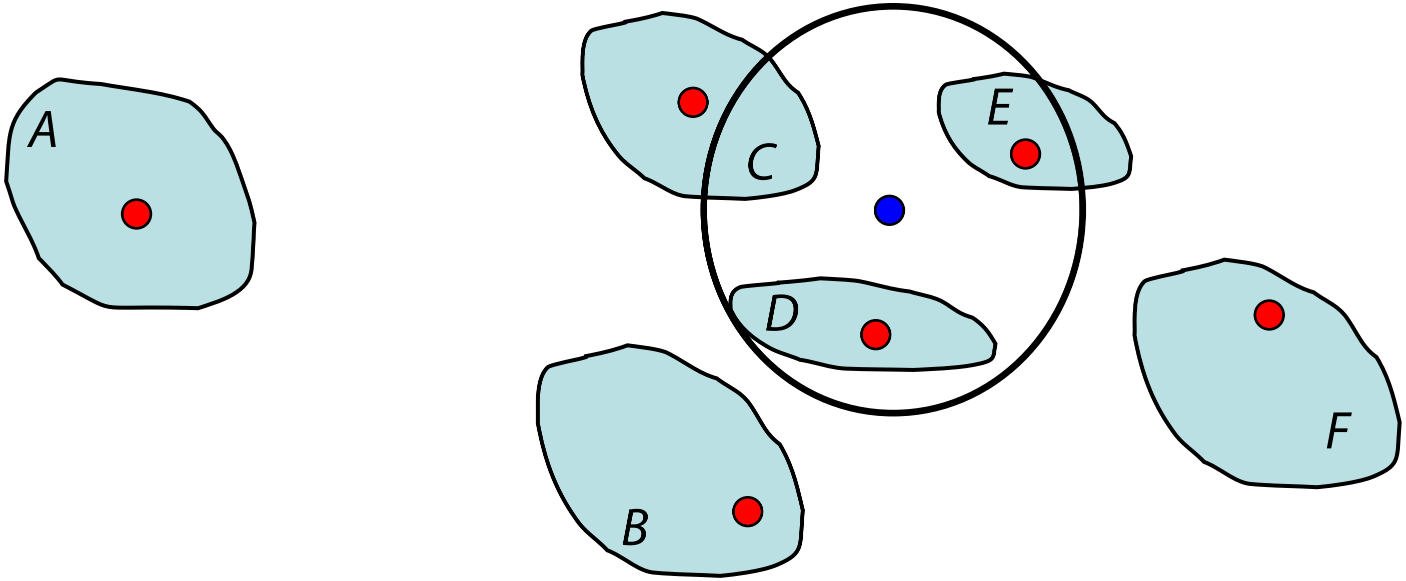
- Assumptions:
  - Objects can lie anywhere uniformly within their uncertainty region

Querying Uncertain Data Probabilistic Range Queries



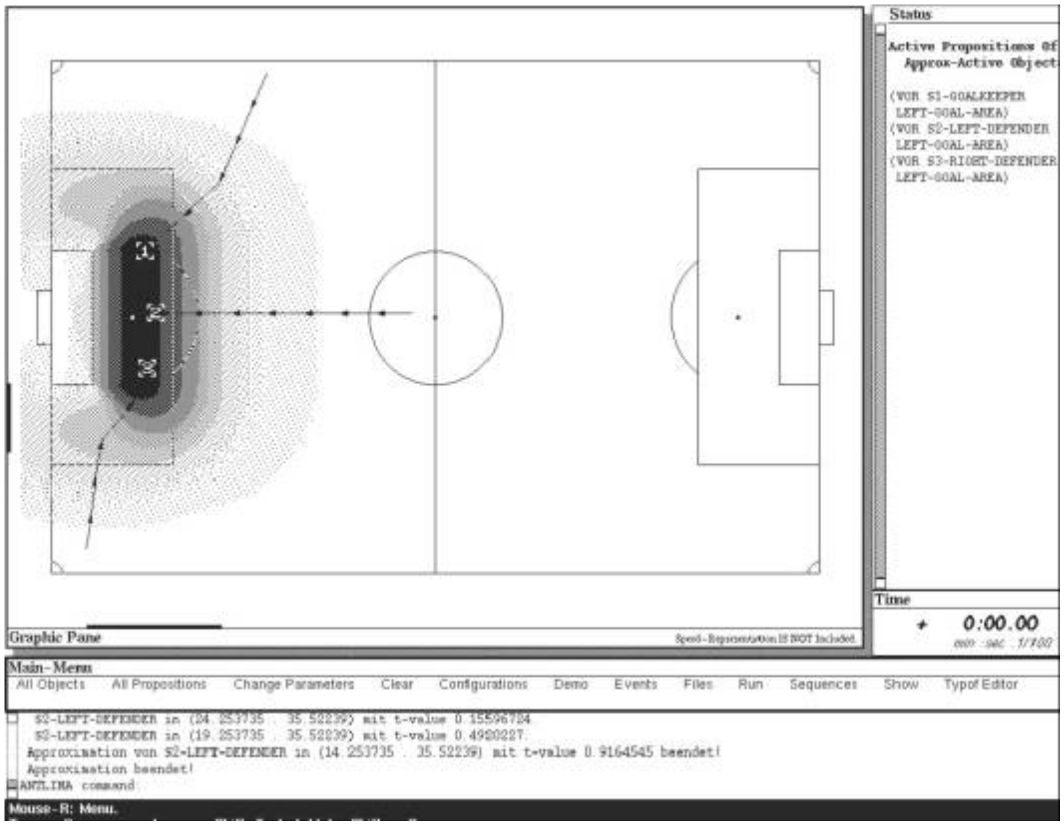
- Query Answer:
  - (B, 50%)
  - (C, 90%)
  - D
  - E
  - (F, 30%)

Querying Uncertain Data Probabilistic NN Queries



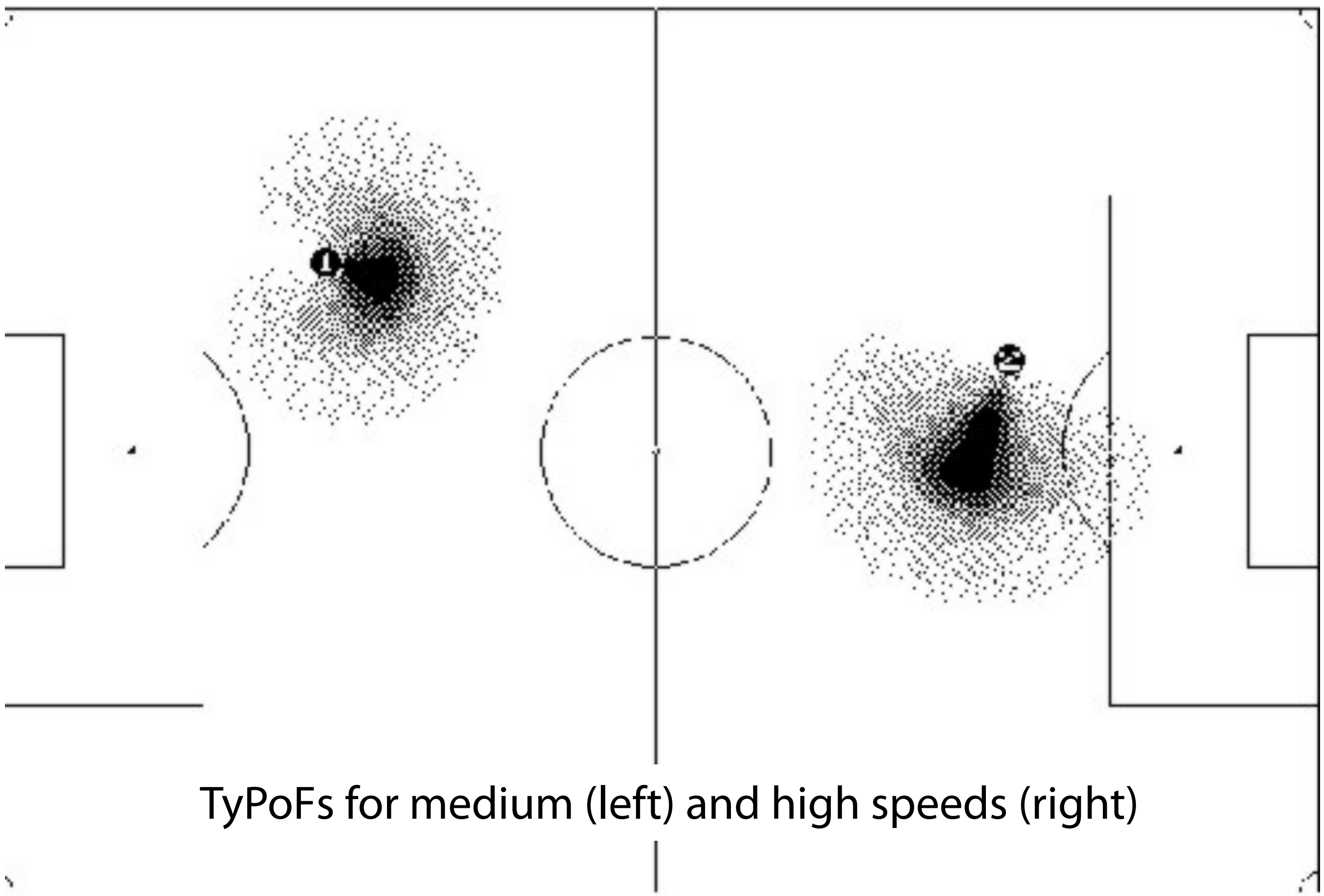
- Query Answer (k=1):
  - (C, p_1)
  - (D, p_2)
  - (E, p_3)

Typicality Potential Fields (TyPoFs)



‘Spieler vor dem Strafraum’

Typicality Potential Fields



TyPoFs for medium (left) and high speeds (right)

J.R.J. Schirra: Bildbeschreibung als Verbindung von visuellem und sprachlichem Raum – Eine interdisziplinäre Untersuchung von Bildvorstellungen in einem Hörermodell. Dissertation. Infix, St. Augustin, **1994**

Recap: Skyline Queries

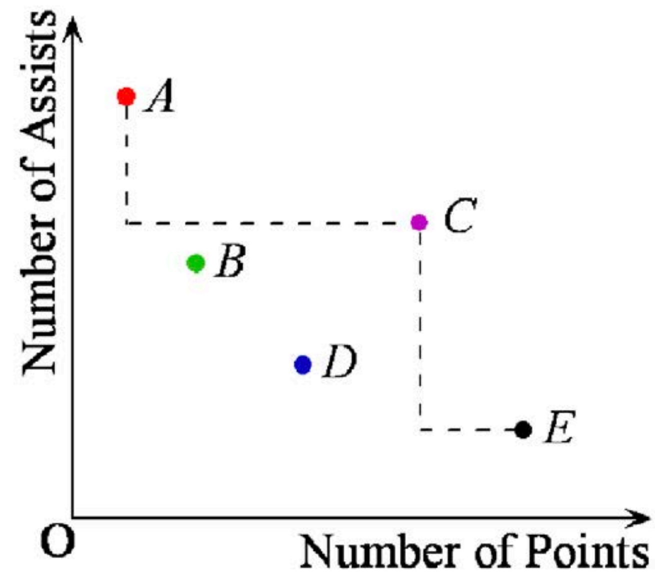
- Numeric space $D = (D_1, \ldots, D_n)$, larger values more preferable

- Two points, u dominates v ($u \succ v$), if
  - $\forall D_i \ (1 \leq i \leq n),\ u.D_i \geq v.D_i$
  - $\exists D_j \ (1 \leq j \leq n),\ u.D_j > v.D_j$

- Given a set of points S,

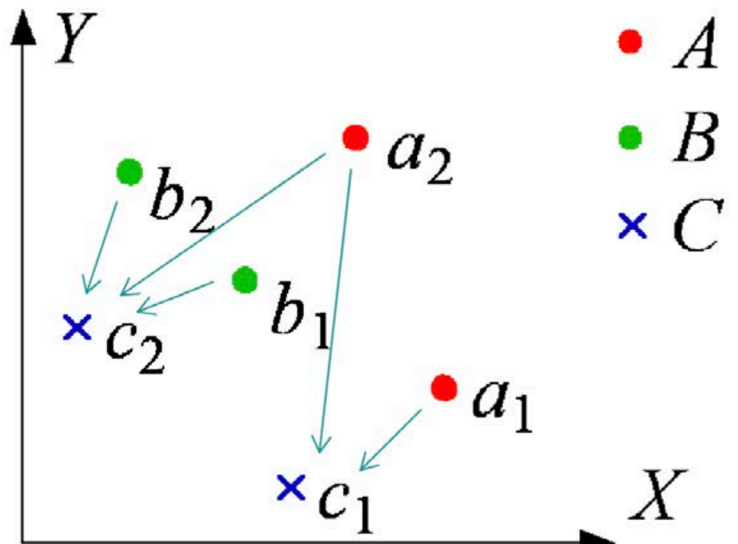
  Skyline = {u | u ∈ S and u is not dominated by any other point}

- Example:
  $C \succ B$, $C \succ D$ skyline = {A, C, E}



Skylines on Uncertain Data

- Limitations of conventional methods
  - Aggregates may be misled by outliers
  - Data distribution is not captured

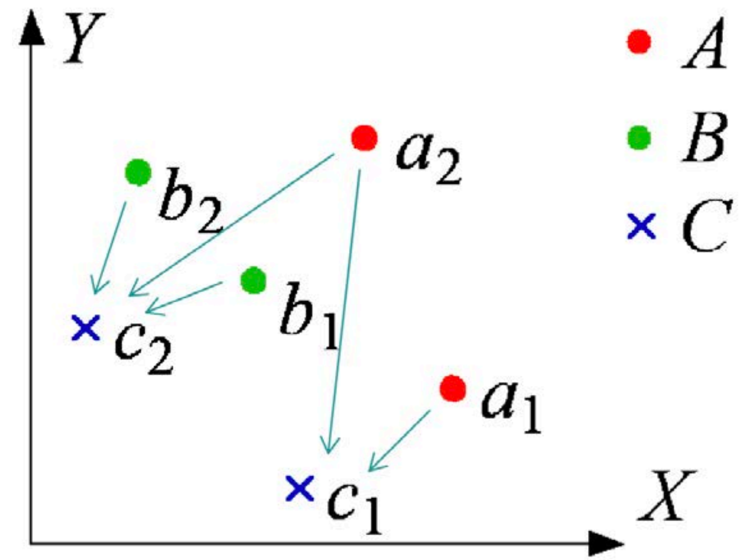


- Probabilistic skylines
  - Objects vs. instances
  - An instance has a probability to represent the object
  - An object has a probability to be in the skyline

Jian Pei, Bin Jiang, Xuemin Lin, and Yidong Yuan. 2007. Probabilistic skylines on uncertain data. In Proc. VLDB '07, 15–26, **2007**.

A Probabilistic Skyline Model

- A set of objects $S = \{A, B, C\}$, instances a_i, b_i, c_i of each with probability 0.5 to appear
- Probabilistic Dominance
  - $\Pr(A \succ C) = 3/4$
  - $\Pr(B \succ C) = 1/2$
  - $\Pr((A \succ C) \vee (B \succ C)) = 1$

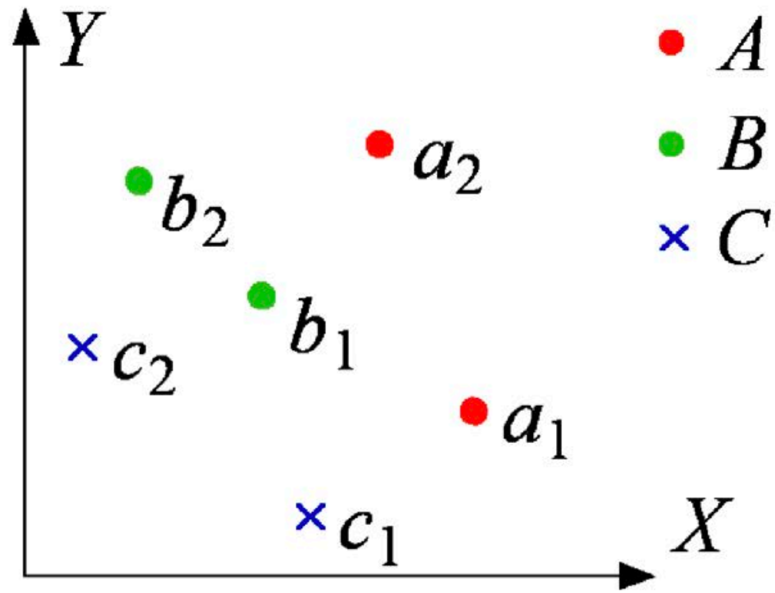


$\Pr(C \text{ is in the skyline}) \neq (1 - \Pr(A \succ C)) \times (1 - \Pr(B \succ C))$

Probabilistic dominance $\not\Longleftrightarrow$ Probabilistic skyline

Skyline Probabilities

- **Possible world**: $W = \langle a_i, b_j, c_k \rangle$ (i, j, k = 1 or 2)
  - $Pr(W) = 0.5 \times 0.5 \times 0.5 = 0.125$, $\sum_{W \in \Omega} Pr(W) = 1$
- $SKY(\langle a_1, b_1, c_1 \rangle) = \{a_1, b_1\}$
  - Objects A and B are in $SKY(\langle a_1, b_1, c_1 \rangle)$
- B is in the skyline of possible worlds $\langle a_1, b_1, c_1 \rangle$, $\langle a_1, b_1, c_2 \rangle$, $\langle a_1, b_2, c_1 \rangle$, and $\langle a_1, b_2, c_2 \rangle$
  - $Pr(B) = 4 \times 0.125 = 0.5$
- $Pr(A) = 1$, $Pr(C) = 0$



Problem Statement

- Skyline probability: $Pr(U) = \sum_{U \in SKY(W)} Pr(W)$
- For object: $Pr(U) = \frac{1}{|U|} \sum_{u \in U} \prod_{V \neq U} (1 - \frac{|\{v \in V \mid v \succ u\}|}{|V|})$
- For instance: $Pr(u) = \prod_{V \neq U} (1 - \frac{|\{v \in V \mid v \succ u\}|}{|V|})$
- $Pr(U) = \frac{1}{|U|} \sum_{u \in U} Pr(u)$
- p-skyline = {U | Pr(U) ⩾ p} for a given threshold p

Try to reduce
V candidates

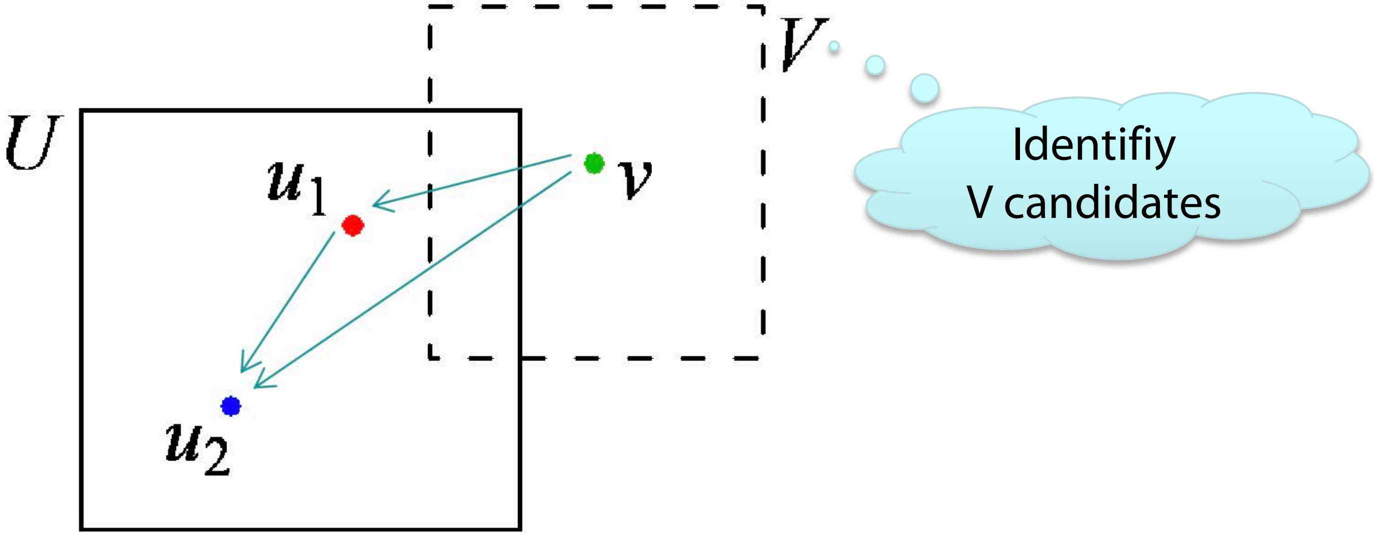
Probabilistic Skyline Computation

- Iteration: Bounding-Pruning-Refining
- Bounding
  - Bound $Pr(u)$: lower bound $Pr^-(u)$ and upper bound $Pr^+(u)$
  - Bound $Pr(U)$: $Pr(U) = \frac{1}{|U|}\sum_{u \in U} Pr(u)$
- Pruning
  - In p-skyline if lower bound $Pr^-(U) \geq p$
  - Not in p-skyline if upper bound $Pr^+(U) < p$
- Refining
  - Bottom-up method
  - Top-down method

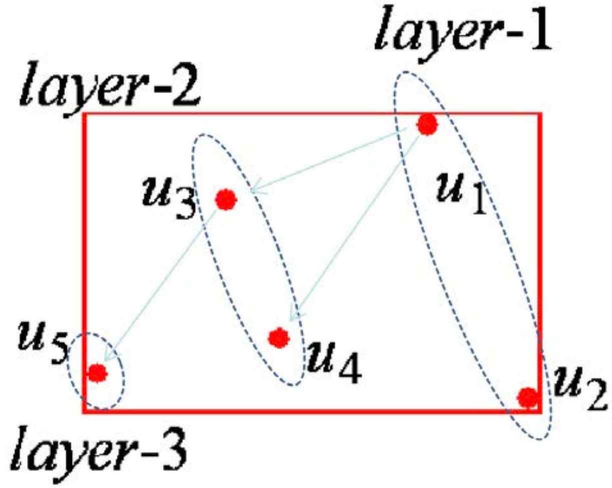
J. Pei, M. Hua, Y. Tao, and X. Lin: Query Answering Techniques on Uncertain and Probabilistic Data, Tutorial, SIGMOD 2008

The Bottom-Up Method

- Sort instances of an object according to dominance relation such that their skyline probabilities are in descending order
- Partial order relation (use topological sorting)
- Two instances u_1 and $u_2 \in U$,
  if $u_1 \succ u_2$ then $Pr(u_1) \geq Pr(u_2)$

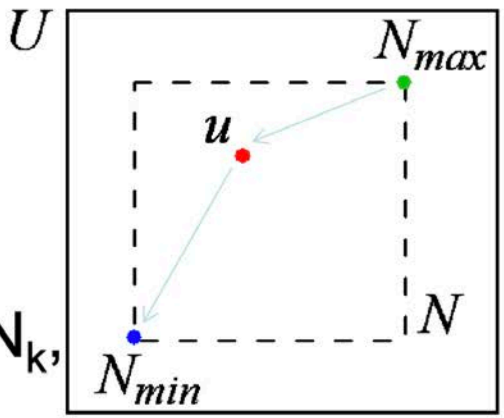


The Layer Structure



- layer-1: skyline of all instances
- layer-k ($k > 1$): skyline of instances except those at
  layer-1, …, layer-(k-1)
- $\forall$ u at layer-k : $\exists$ u′ at layer-(k-1) :
  $u' \succ u$ and $Pr(u') \geq Pr(u)$
- $\max\{Pr(u) \mid u \text{ is at layer-}(k-1)\} \geq \max\{Pr(u) \mid u \text{ is at layer-}k\}$
- Bounding example
  - $\max\{Pr(u1), Pr(u2)\} \geq \max\{Pr(u3), Pr(u4)\} \geq Pr(u5)$

The Top-Down Method



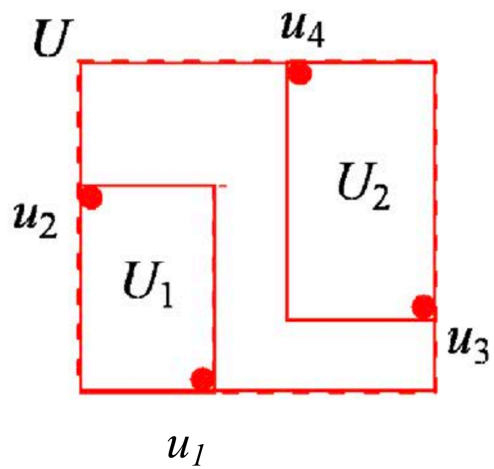
- For instances u_1 and $u_2 \in U$, if $u_1 \succ u_2$, then $Pr(u_1) \geq Pr(u_2)$
  - N is a subset of instances of U, $\forall u \in N$, $Pr(N_{max}) \geq Pr(u) \geq Pr(N_{min})$
- Object U has k partitions $N_1, \ldots, N_k$,

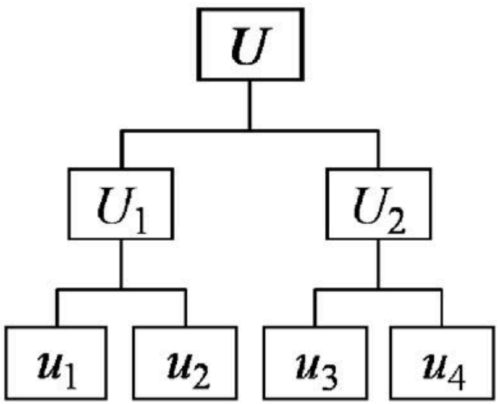
$$\frac{1}{|U|}\sum_{i=1}^{k}|N_i|\cdot Pr(N_{i,max}) \geq Pr(U) \geq \frac{1}{|U|}\sum_{i=1}^{k}|N_i|\cdot Pr(N_{i,min})$$

- Build a partition tree for each object to organize partitions

Partition Tree

- Binary tree





- Growing one level of the tree in each iteration
  - Choose one dimension in a round-robin fashion
  - Each leaf node is partitioned into two children nodes, each of which has half of instances
- Bound $Pr(N_{max})$ and $Pr(N_{min})$ of a partition N

Summary

- Location-aware Environments
- Location-aware *Snapshot* Query Processing
- Location-aware *Continuous* Query Processing
- Scalable Execution of Continuous Queries
- Location-aware Query Optimizer
- Uncertainty in Location-aware Query Processing