An advanced course on

Mobility Data Management & Exploration

Yannis Theodoridis & Nikos Pelekis

InfoLab | University of Piraeus | Greece
infolab.cs.unipi.gr

Apr. 2013

Acknowledgments

- The content of this lecture series has been inspired by collaborative work done in the following EU projects:
  - FP7/SEEK (http://www.seek-project.eu), 2012-15
  - FP7/DATASIM (http://www.datasim-fp7.eu), 2011-14
  - ESF/COST-MOVE (http://move-cost.info), 2009-13
  - FP7/MODAP (http://www.modap.org), 2009-12
  - FP6/GeoPKDD (http://www.geopkdd.eu), 2005-09
- Also, special thanks to InfoLab members

Previous versions of this material

- 2012
  - MSc course @ KAUST, Jeddah, Saudi Arabia, Jun. 2012
  - PhD course @ Univ. Ghent, Belgium, Feb. 2012
- 2011
  - PhD course @ Univ. Trento, Italy, Nov. 2011
  - PhD course @ Univ. Aalborg, Denmark, Sep. 2011
  - MSc course @ KAUST, Jeddah, Saudi Arabia, Jun. 2011
  - PhD course @ Univ. Milano, Italy, May 2011
- 2010
  - PhD course @ Univ. Venice, Italy, Jun. 2010

More details at: http://infolab.cs.unipi.gr/

Introduction, Overview

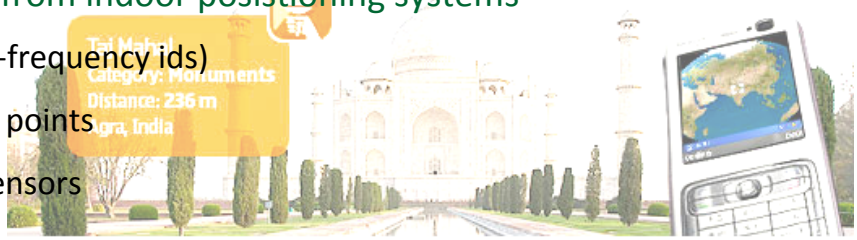
From digital mapping to mobile social networking – A tour on geospatial information management challenges

Mobile devices and services

- Large diffusion of mobile devices, mobile services and location-based services ➔ **location- and mobility-aware data**

Which data?

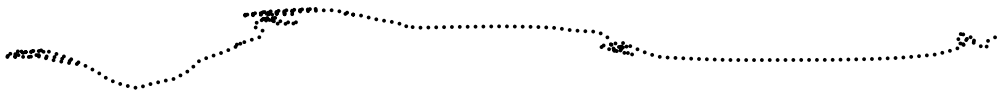
- Location data from mobile phones
  - i.e., cell positions in the GSM/UMTS network
- Location (and trajectory) data from GPS-equipped devices
  - Humans (pedestrians, drivers) with GPS-equipped smartphones
  - Vessels with AIS transmitters (due to maritime regulations)
- Location data from indoor posistioning systems
  - RFIDs (radio-frequency ids)
  - Wi-Fi access points
  - Bluetooth sensors



GPS Data

- Raw data: GPS recordings

```
objectID, trajectoryID, timestamp, longitude, latitude
201100024,1,2009-01-02 08:54:07,24.609728324369,38.013503319816
201100024,1,2009-01-02 08:54:25,24.6094016577037,38.0127699864845
201100024,1,2009-01-02 08:55:06,24.6086749910399,38.011116653155
201100024,1,2009-01-02 08:55:56,24.6076299910435,38.0092066531597
201100024,1,2009-01-02 08:56:16,24.6071983243782,38.0084733198281
201100034,1,2009-01-02 04:19:26,23.1092366579214,38.5853616531322
201100034,1,2009-01-02 04:19:36,22.9272199909328,38.8922416526431
201100034,1,2009-01-02 04:19:45,23.0359933243564,38.7788549861265
201100034,1,2009-01-02 04:19:55,22.9355449909622,38.868204986019
201100034,1,2009-01-02 04:20:05,23.0638616578755,38.6383849863914
...
```

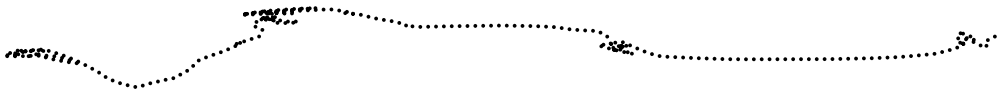


GPS Data

- Q: where is (24.6071983243782, 38.0084733198281) located ?
- A: in the short sea passage between Euboea and Andros islands, Greece

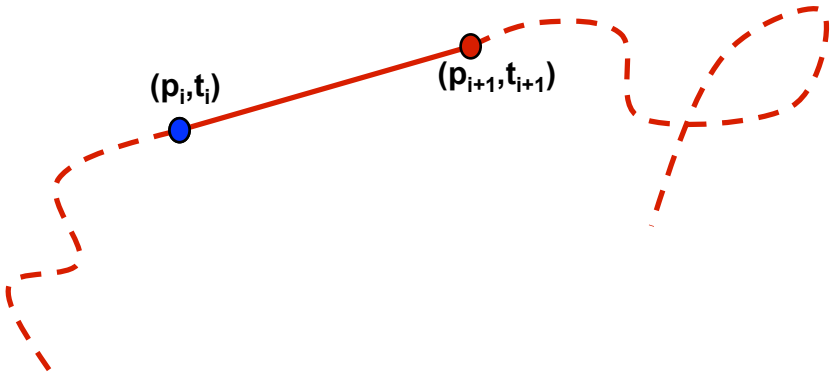
What is a (GPS-based) trajectory?

- A **trajectory** is a model for a motion path of a moving object (animal, car, human, …)
  - (due to discretization) a sequence of sampled time-stamped locations (p_i, t_i) where p_i is a 2D point (x_i, y_i) and t_i is the recording timestamp of p_i



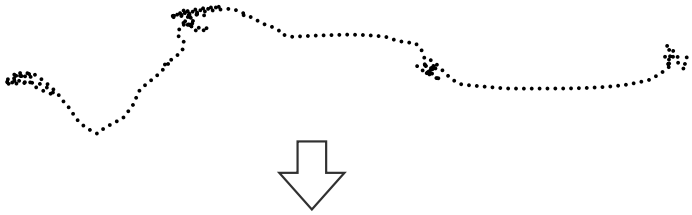
What is a (GPS-based) trajectory?

- A common representation in MOD is a **3D polyline** in the plane where vertices correspond to time-stamped locations (p_i, t_i)
  - and **linear interpolation** is assumed between (p_i, t_i) and (p_{i+1}, t_{i+1})

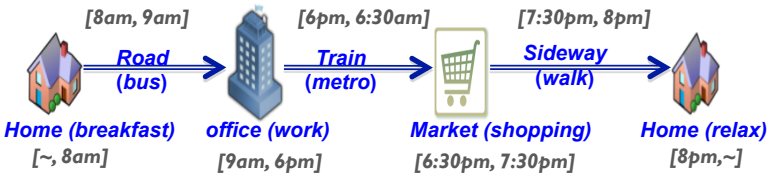


From "raw" to "semantic" trajectories

raw mobility data
sequence (x,y,t) points
e.g., GPS feeds



meaningful mobility tuples
<place, $time_{in}$, $time_{out}$, tags>



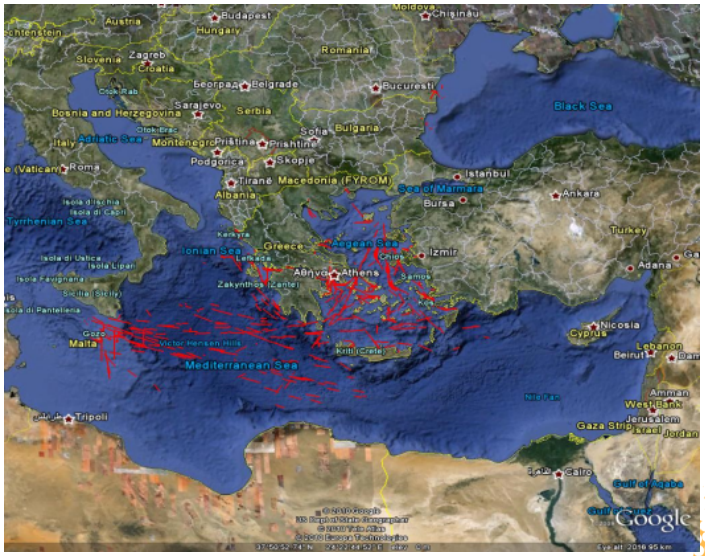
- **Semantic Trajectory: $T=\{e_{first},\ldots,e_{last}\}$**
- **Episode: e_i = (STOP | MOVE, t_{from}, t_{to}, *place, tag*)**

Examples of GPS trajectory data

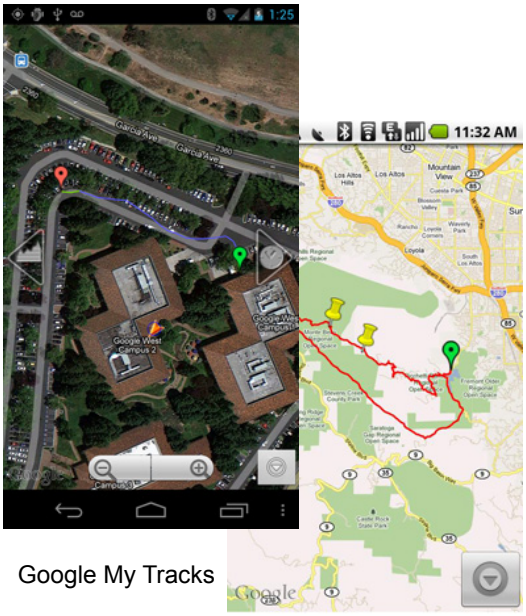
- "Milano dataset": vehicles moving in Milan
  - ~2M GPS recordings from 17241 distinct objects (7 days period) ➔ 214,780 trajectories

Examples of GPS trajectory data

- "IMIS3days" dataset: vessels sailing in Mediterranean sea
  - (only a small subset of the dataset at hand) ~4.5M GPS recordings from 1753 distinct objects (3 days period) ➔ 1503 trajectories



Examples of trajectory-based apps



Google My Tracks

RunKeeper

What can we do with / learn from mobility data ...

Vehicles datasets…

- (global) Traffic monitoring
  - How many cars are in the ring of the town?
  - Once an accident is discovered, immediately send alarm to the nearest police and ambulance cars

Vehicles datasets…

- (personalized) Location-aware queries
  - Where is my nearest Gas station?
  - What are the fast food restaurants within 3 miles from my location?
  - Let me know if I am near to a restaurant while any of my friends are there

Vessels datasets…

(requirements from Greek Maritime Conservation Agencies)

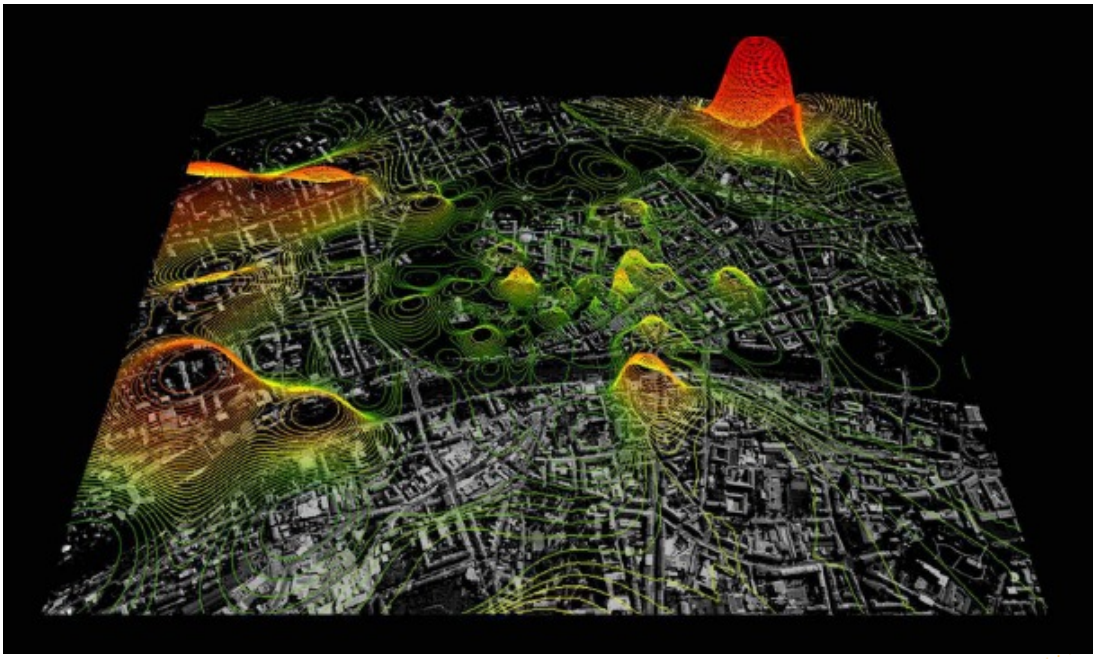
- Querying and mining trajectories:
  - Extract / draw the ship tracks (detailed vs. simplified)
  - Calculate average and minimum distance from shore; where and when
  - Calculate the number of ships in the vicinity of the ship (e.g. 10 n.m. radius)
  - Find whether (and how many times) a ship goes through narrow passages or biodiversity boxes
  - Calculate the number of sharp changes in direction
  - Find ships following typical routes vs. outliers

Vessels datasets…

Application- oriented analysis:

- **Improving safety**
  - Analyze the accuracy of data provided by base stations
- **Traffic optimization**
  - Calculate metrics from the traffic: traffic density, mean distance between ships, number of trajectories that are close to the 'optimal' departure – arrival path
    - Devise new sea routes to handle traffic increase
  - Measure the activity of each ship: number of intermediate stops
- **Environmental considerations**
  - Compare trajectories with environmental considerations (fuel consumption, noise pollution),

More ambitious: "Mobile Landscapes"
[Ratti et al. 2005]

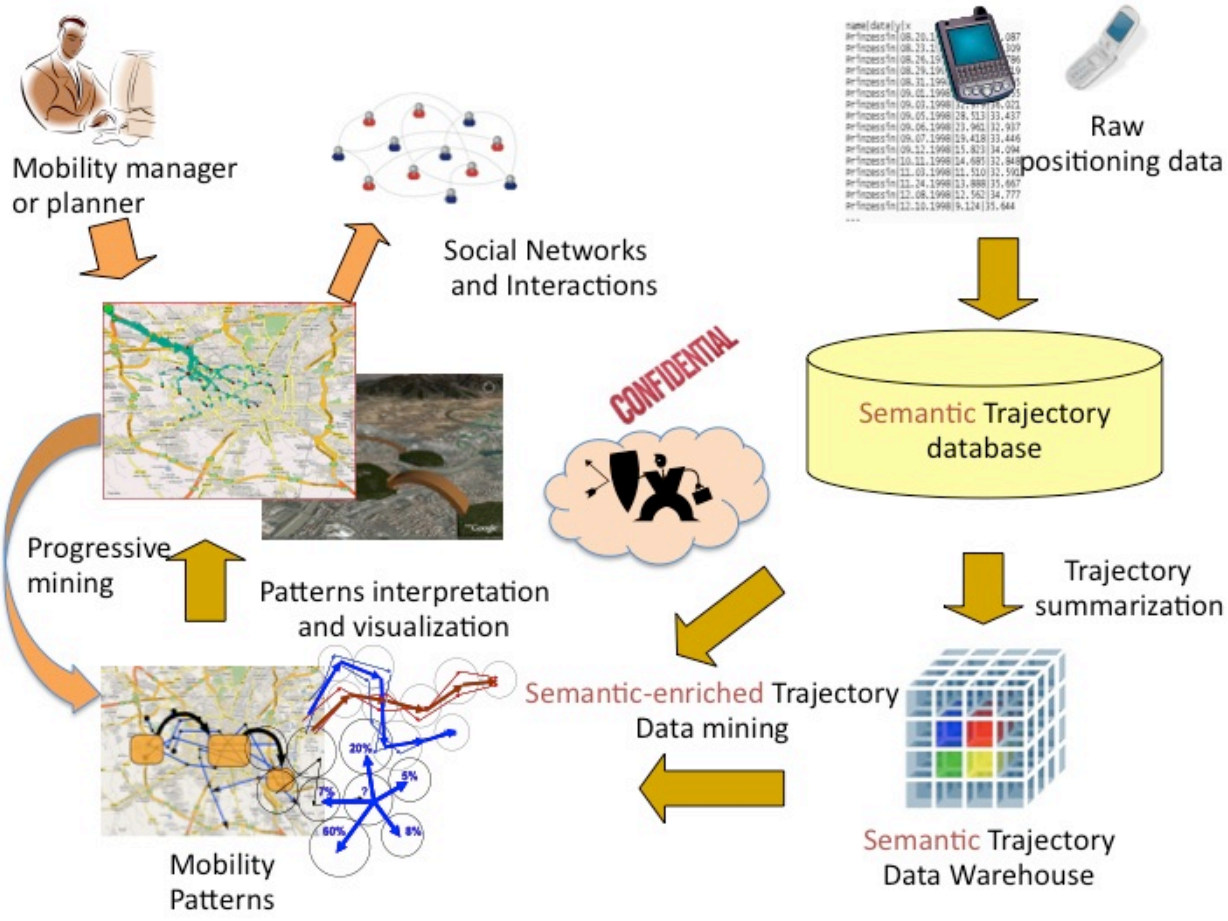


MIT senseable project: http://senseable.mit.edu/grazrealtime/

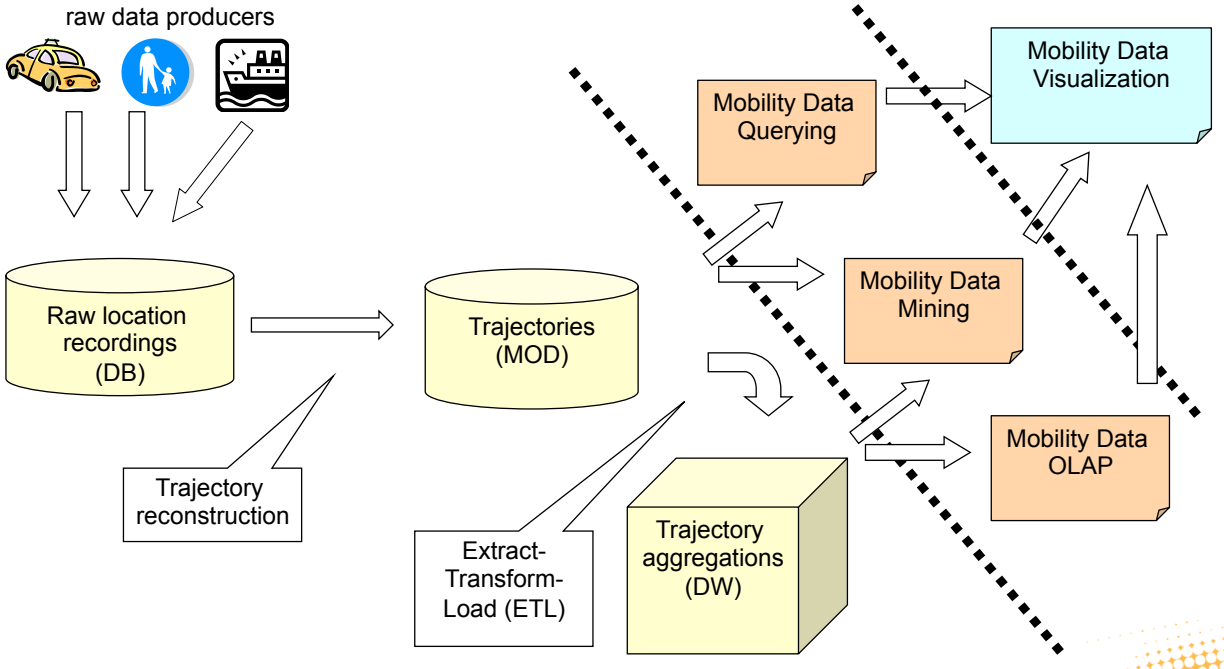
More ambitious: “Trajectory patterns”
[Giannotti et al. 2007]

EU GeoPKDD project: http://www.geopkdd.eu

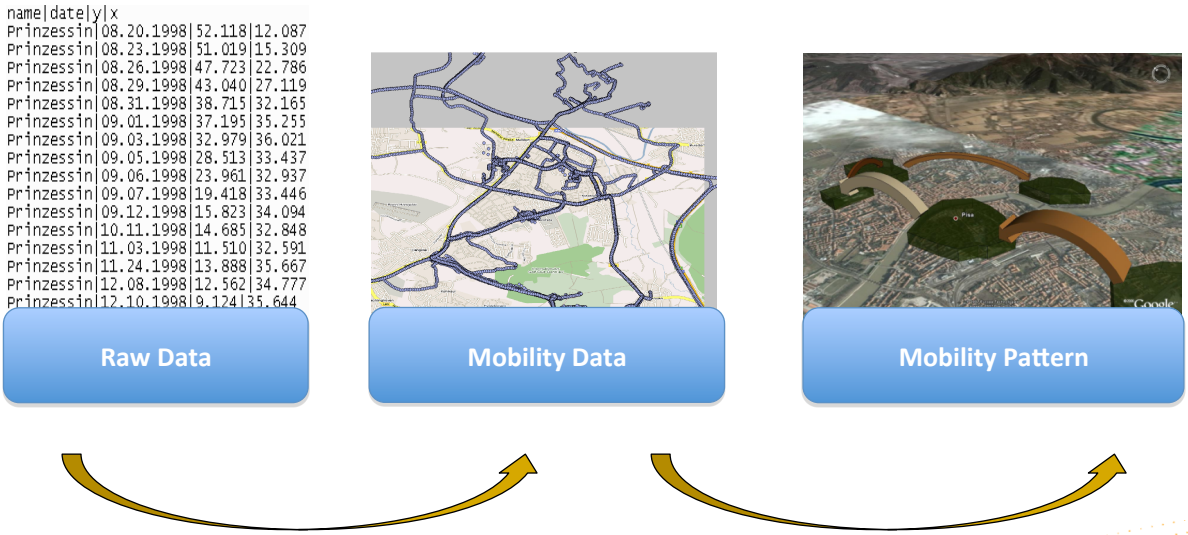
The big picture



The modules of our architecture



From data to information to knowledge



Key questions that arise

- How to **reconstruct a trajectory** from raw logs?
- How to **store trajectories** in a DBMS?
  - Is a trajectory simply a sequence of (x, y, t) tuples?
- What kind of **analysis** is suitable for mobility data?
  - In particular, trajectories of moving objects?
  - How does infrastructure (e.g. road network) affect this analysis?
- Which **patterns / models** can be extracted out of them?
  - Clusters, frequent patterns, anomalies / outliers, etc.
  - How to compute such patterns / models efficiently?
- How to **protect privacy / anonymity**?
  - trade-off between privacy protection and quality of analysis

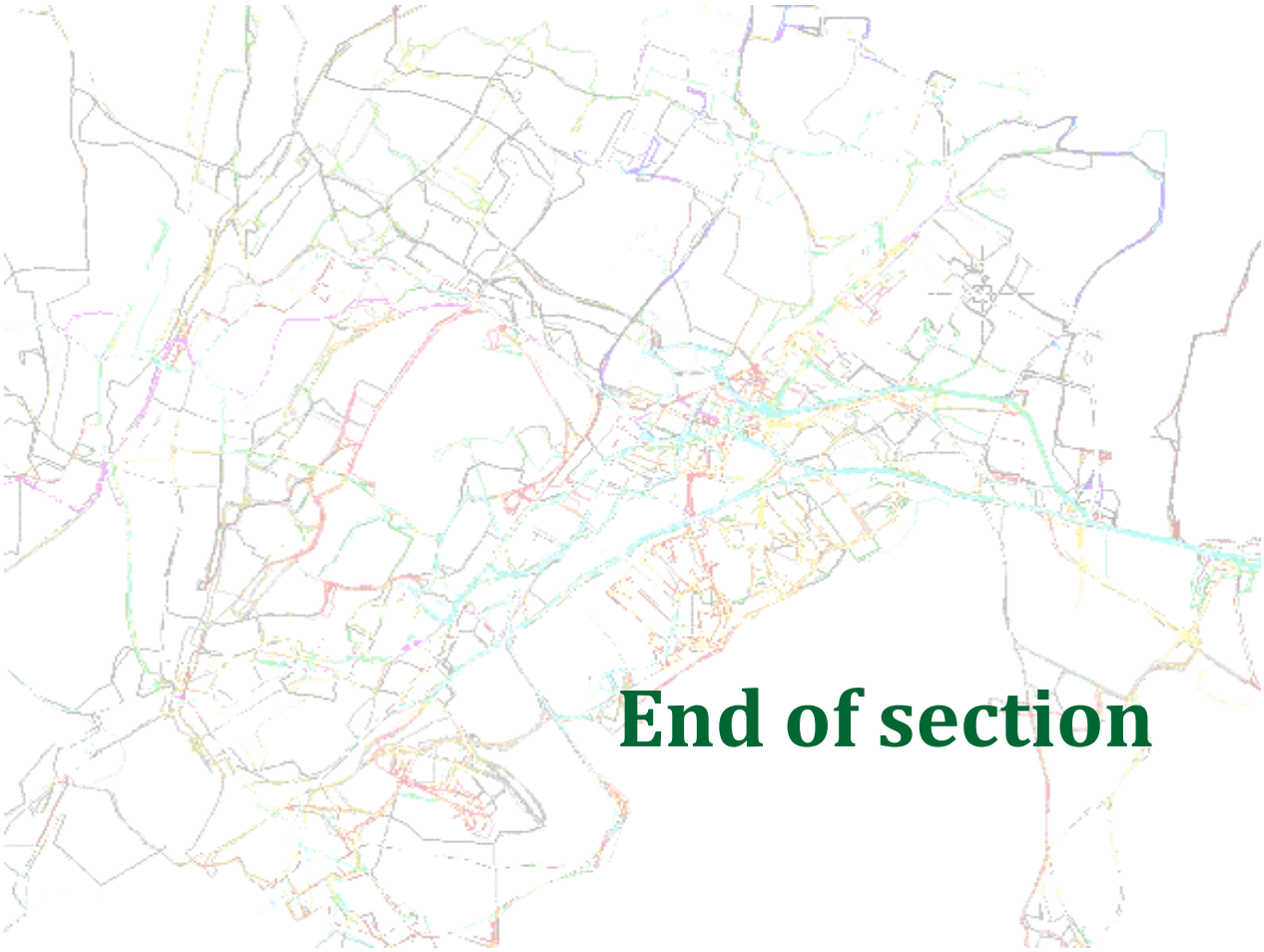
Course outline

- **I. Background** - GPS infrastructure; Spatial database management; Spatial OLAP and data mining
- **II. Mobility-aware applications and tools** - Location-based services and tools; Algorithms and operations for LBS
- **III. Mobility data management (storage and querying)** - Acquiring trajectories from raw data; Location-aware querying; Efficient trajectory indexing and storage in MODs
- **IV. Mobility data exploration (OLAP analysis and mining)** - Trajectory warehousing and OLAP; Mobility data mining and reasoning; Visual analytics for mobility data
- **V. Privacy aspects** - Preserving user traces' anonymity
- **VI. Outlook** - Open issues; Future Challenges

Reading list

Project overviews, manifesto papers, etc.

- Atzori, P. (2007) Privacy and anonymity in location and movement-aware data analysis – the GeoPKDD approach. Proceedings of ISI.
- Giannotti, F. and Pedreschi, D. (2008) Mobility, Data Mining, and Privacy: A Vision of Convergence. In Mobility, Data Mining and Privacy – Geographic Knowledge Discovery. Springer.
- Giannotti, F. et al. (2008) Mobility, Data Mining, and Privacy – the Experience of the GeoPKDD Project. Proceedings of PinKDD.
- Lopez, X. (2003) The Future of GIS: Real-time, Mission Critical, Location Services. Proceedings of Cambridge Conference.
- Nabian, N. et al. (2009) MIT GEOblog: A Platform for Digital Annotation of Space and Collective Community Based Digital Story Telling. Proceedings of IEEE-DEST.
- Ratti, C. et al. (2005) Mobile Landscapes: Graz in Real Time. Proceedings of LBS & TeleCartogrqaphy.

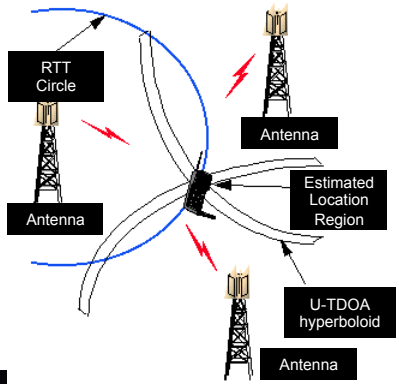


End of section

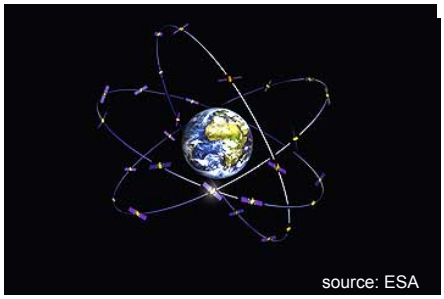
Background on Positioning technologies

Geo-positioning

- Positioning technologies (all standardized in early 2000's)
  - Using the mobile telephone network
    - Time of Arrival (TOA), UpLink TOA (UL-TOA)
  - Using information from satellites
    - Global Positioning System (GPS)
    - Assisted (A-GPS), Differential GPS (D-GPS)

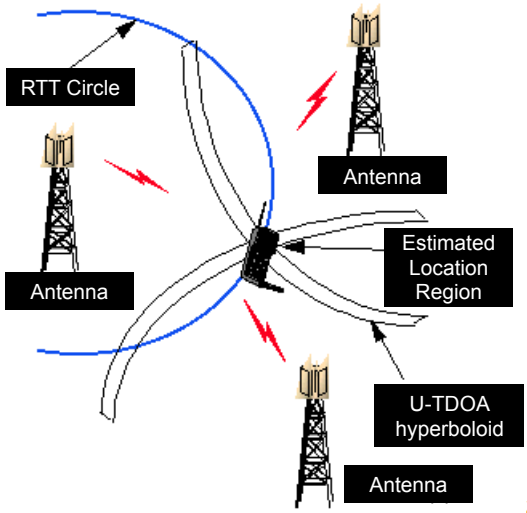


source: http://www.3gpp.org



Geo-positioning (cont.)

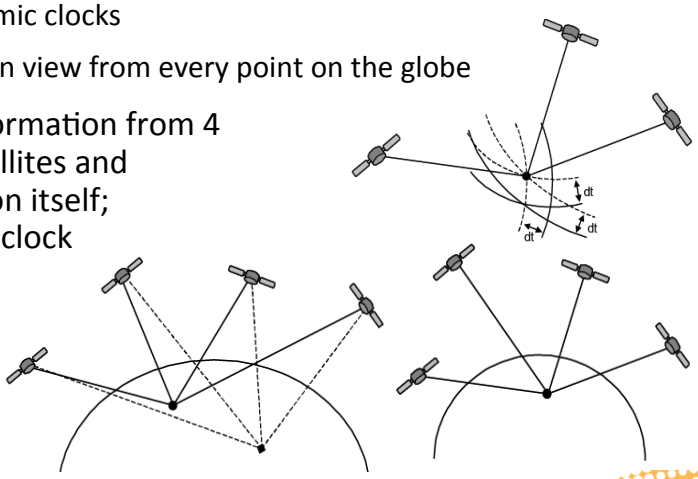
- Uplink - Time Difference of Arrival (U-TDOA)
  - At least 3 receivers (located together with antennas) get signals from a user's mobile, triangulate, and estimate its position
  - Accuracy: 30-120 m
  - Standardized by the 3GPP (3rd Generation Partnership Project)
  - Problem: Requires great investment in infrastructure



source: 3gpp.org

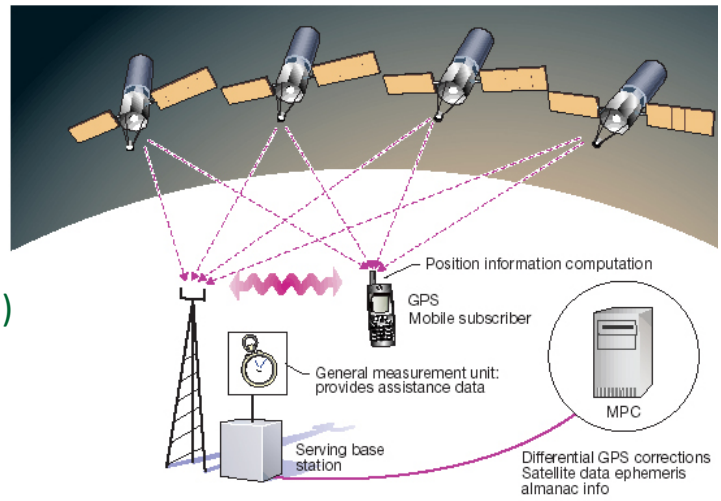
Satellite-supported positioning

- GPS (Global Positioning System)
  - Fully operational since 1994
  - 24-satellite constellation
    - monitored by 5 monitoring stations and 4 ground antennas; handled with (extremely precise) atomic clocks
    - At least 5 satellites are in view from every point on the globe
  - GPS receiver gathers information from 4 (or 3, the minimum) satellites and (a) triangulates to position itself; (b) fixes its (non-atomic) clock
  - Position accuracy: ~20m



Geo-positioning (cont.)

- Assisted GPS (A-GPS)
  - provides pre-calculated satellite orbits to the receiver
  - Accuracy 10-20 m
- Differential GPS (D-GPS)
  - accuracy down to 1m



source: (Swedberg, 1999)

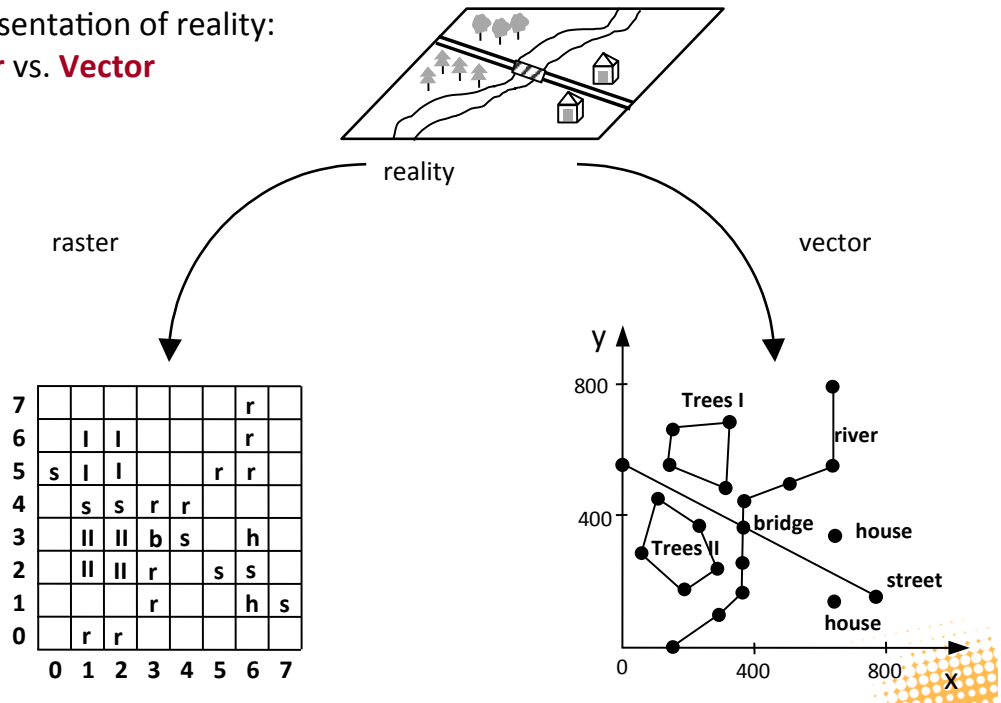
Geo-positioning (cont.)

- GPS competitors
  - Glonass (Russia) – currently, semi-operational
    - 24-satellite constellation; 1-10m accuracy
  - Galileo (EU) - fully operational by 2019
    - 30-satellite constellation; 1m accuracy
  - Beidou (China) - fully operational by 2020
    - 35-satellite constellation; 10m accuracy

Background on Spatial database management

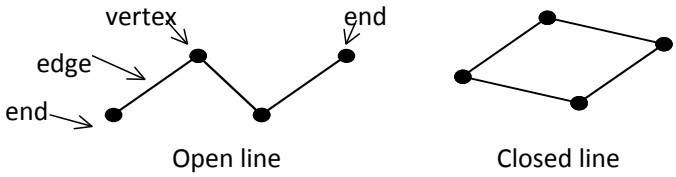
Geographical data models

- Representation of reality: **Raster** vs. **Vector**

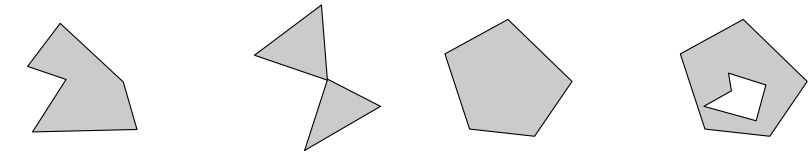


Vector model

- Geographical space = a set of entities
  - 0-d: points
  - 1-d: line segments, polylines, …

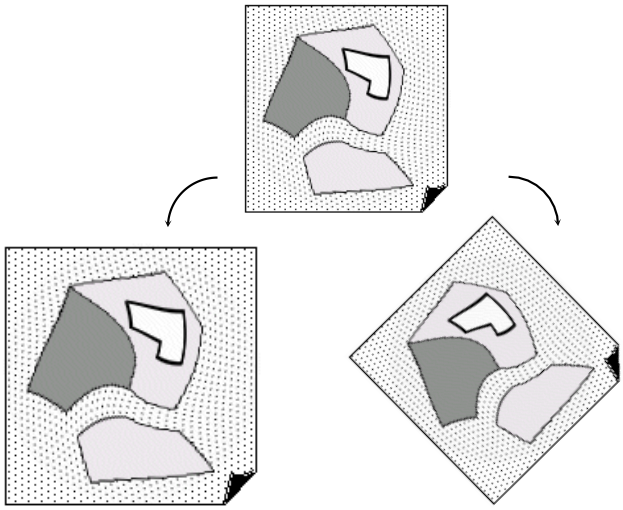


  - 2-d: polygons, polygons with holes, …



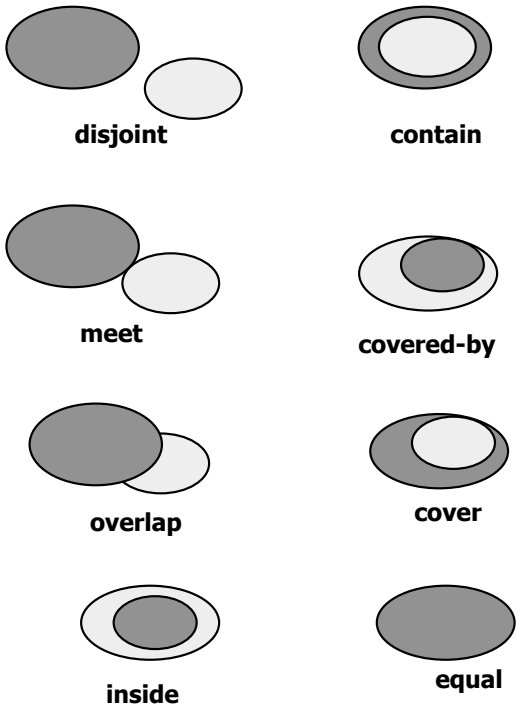
Spatial relationships

- Topological vs. directional relationships between spatial objects
  - Topological relationships are invariant to topological transformations
    - Shift, Rotation, Scaling



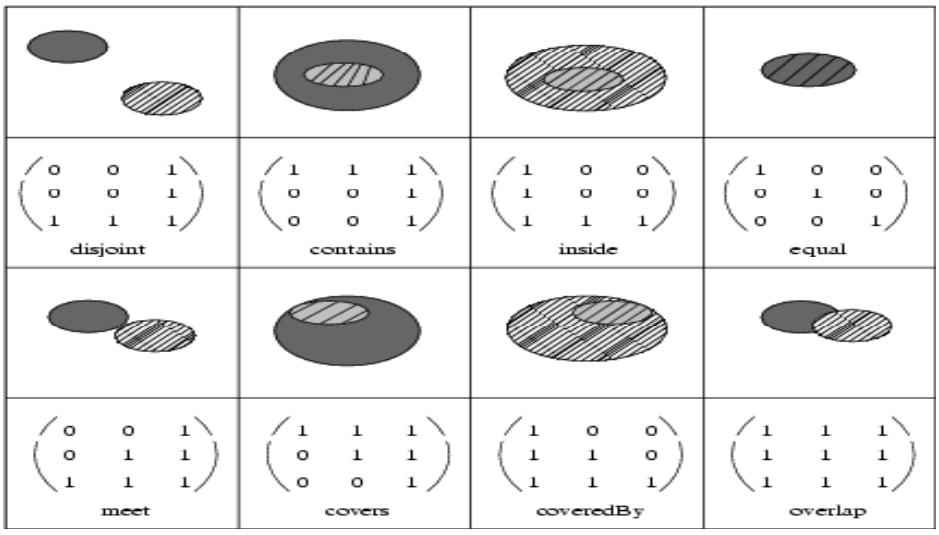
Topological relationships

- Egenhofer's 4- and 9-intersection model (Egenhofer and colleagues, 1989-93)
  - Based on the set intersections between objects' interior, boundary and exterior



Egenhofer's Nine-Intersection Model

$$\Gamma_9(A,B) = \begin{pmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B & A^\circ \cap B^- \\ \partial A \cap B^\circ & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^\circ & A^- \cap \partial B & A^- \cap B^- \end{pmatrix}$$

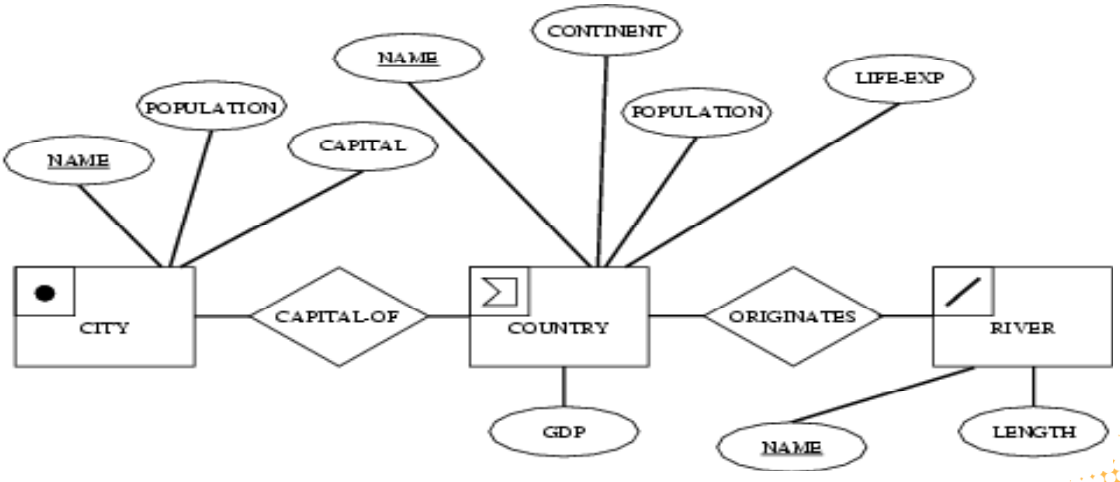


Example of a geo-DB

- Entities: countries, cities, rivers, etc.
  - Relationships between entities: capital-of-country, …
- How can we model and manage such information?
  - Country is of polygon shape
  - River is of polyline shape (?)
  - City is of point shape (?)
- Spatial operations:
  - area of a country, length of borderline between 2 countries, ...

Example: World Database

- At the conceptual level
  - 3 Entities: Country, City, River
  - 2 Relationships: capital-of, originates-in



Example: World Database (cont.)

- At the logical level: 3 relations (**Country**, **City**, **River**)

COUNTRY	Name	Cont	Pop (millions)	GDP (billions)	Life-Exp	Shape
	Canada	NAM	30.1	658.0	77.08	Polygonid-1
	Mexico	NAM	107.5	694.3	69.36	Polygonid-2
	Brazil	SAM	183.3	1004.0	65.60	Polygonid-3
	Cuba	NAM	11.7	16.9	75.95	Polygonid-4
	USA	NAM	270.0	8003.0	75.75	Polygonid-5
	Argentina	SAM	36.3	348.2	70.75	Polygonid-6

(a) Country

CITY	Name	Country	Pop (millions)	Capital	Shape
	Havana	Cuba	2.1	Y	Pointid-1
	Washington, D.C.	USA	3.2	Y	Pointid-2
	Monterrey	Mexico	2.0	N	Pointid-3
	Toronto	Canada	3.4	N	Pointid-4
	Brasilia	Brazil	1.5	Y	Pointid-5
	Rosario	Argentina	1.1	N	Pointid-6
	Ottawa	Canada	0.8	Y	Pointid-7
	Mexico City	Mexico	14.1	Y	Pointid-8
	Buenos Aires	Argentina	10.75	Y	Pointid-9

(b) City

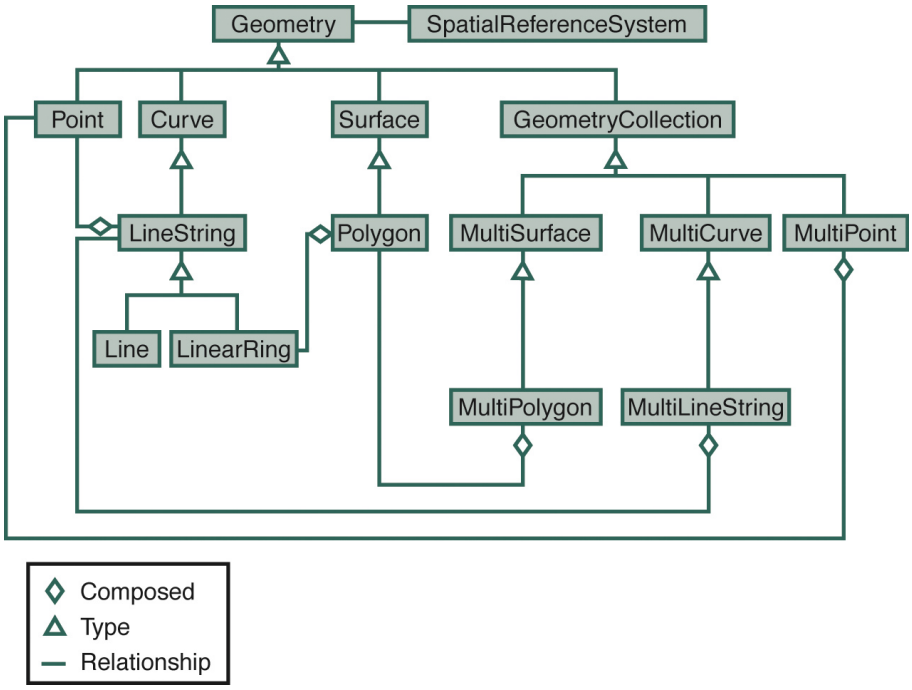
RIVER	Name	Origin	Length (kilometers)	Shape
	Rio Parana	Brazil	2600	LineStringid-1
	St. Lawrence	USA	1200	LineStringid-2
	Rio Grande	USA	3000	LineStringid-3
	Mississippi	USA	6000	LineStringid-4

(c) River

How do we implement spatial DBs

- An example Spatial DBMS: **PostgreSQL**
  - Geometric data types
    - **Point**
    - **Line segment** = 2 points
    - **Box** = 2 points
    - **Path** = sequence of points
    - **Polygon** = sequence of points
    - **Circle** = point + number (radius)
  - Spatial indexing techniques
    - GiST – generalized search trees
    - Special case: R-tree
  - Geometric functions and operators
    - A variety …

PostgreSQL geometric data types



PostgreSQL geometric operators

Operator	Description	Example
+	Translation	box '((0,0),(1,1))' + point '(2.0,0)'
-	Translation	box '((0,0),(1,1))' - point '(2.0,0)'
*	Scaling/rotation	box '((0,0),(1,1))' * point '(2.0,0)'
/	Scaling/rotation	box '((0,0),(2,2))' / point '(2.0,0)'
#	Point or box of intersection	'((1,-1),(-1,1))' # '((1,1),(-1,-1))'
#	Number of points in path or polygon	# '((1,0),(0,1),(-1,0))'
@-@	Length or circumference	@-@ path '((0,0),(1,0))'
@@	Center	@@ circle '((0,0),10)'
##	Closest point to first operand on second operand	point '(0,0)' ## lseg '((2,0),(0,2))'
<->	Distance between	circle '((0,0),1)' <-> circle '((5,0),1)'
&&	Overlaps?	box '((0,0),(1,1))' && box '((0,0),(2,2))'
&<	Does not extend to the right of?	box '((0,0),(1,1))' &< box '((0,0),(2,2))'
&>	Does not extend to the left of?	box '((0,0),(3,3))' &> box '((0,0),(2,2))'

PostgreSQL geometric operators (cont.)

Operator	Description	Example
<<	Is left of?	circle '((0,0),1)' << circle '((5,0),1)'
>>	Is right of?	circle '((5,0),1)' >> circle '((0,0),1)'
<^	Is below?	circle '((0,0),1)' <^ circle '((0,5),1)'
>^	Is above?	circle '((0,5),1)' >^ circle '((0,0),1)'
?#	Intersects?	lseg '((-1,0),(1,0))' ?# box '((-2,-2),(2,2))'
?-	Is horizontal?	?- lseg '((-1,0),(1,0))'
?-	Are horizontally aligned?	point '(1,0)' ?- point '(0,0)'
?|	Is vertical?	?| lseg '((-1,0),(1,0))'
?|	Are vertically aligned?	point '(0,1)' ?| point '(0,0)'
?-|	Is perpendicular?	lseg '((0,0),(0,1))' ?-| lseg '((0,0),(1,0))'
?||	Are parallel?	lseg '((-1,0),(1,0))' ?|| lseg '((-1,2),(1,2))'
~	Contains?	circle '((0,0),2)' ~ point '(1,1)'
@	Contained in or on?	point '(1,1)' @ circle '((0,0),2)'
~=	Same as?	polygon '((0,0),(1,1))' ~= polygon '((1,1),(0,0))'

PostgreSQL Examples

- A table of areas (zones)

```
CREATE TABLE zones (poly_id integer, name varchar(30), sector polygon);
INSERT INTO zones VALUES (1, 'PARK', '(479243, 4204000,
        477728, 4202750, 477559, 4202100, 476271, 4204750)' :: polygon);
```

- A table of points (locations)

```
CREATE TABLE locations (point_id integer, name varchar(30), pos point);
INSERT INTO locations VALUES (52, 'Freedom Sq.',
        '(476600, 4202800)' :: point);
```

PostgreSQL Examples

- Objects within distance from a given point

```
SELECT point_id,(pos<->Point '(475750, 4201500)') as distance
FROM locations
WHERE (pos <-> Point '(475750, 4201500)' ) <= 200
```

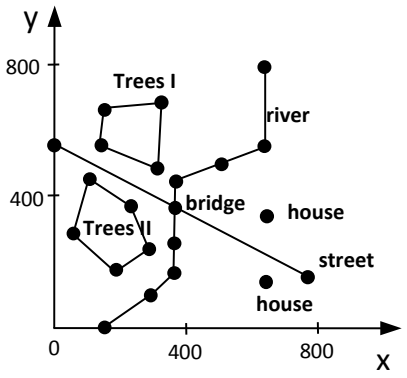
- Objects within a given region

```
SELECT point_id, name
FROM locations
WHERE (pos @ box '(476271, 4204000, 479243, 4204750)') = TRUE

SELECT point_id, name
FROM locations
WHERE (pos @ (SELECT sector FROM zones WHERE name = 'PARK'))
```

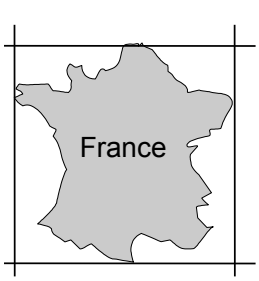

SDBMS physical level

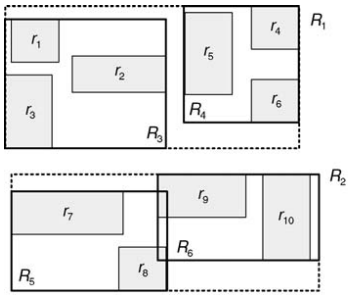
- Issue:
  - How to store and efficiently process this kind of information?
- Relational DBMS support traditional (alphanumeric, etc.) data types
  - Low complexity ➔ relational tables are efficient
  - Total ordering ➔ search trees (e.g. B^+-trees) for fast search
- Unfortunately, spatial objects
  - (a) are of high complexity and
  - (b) lack total ordering

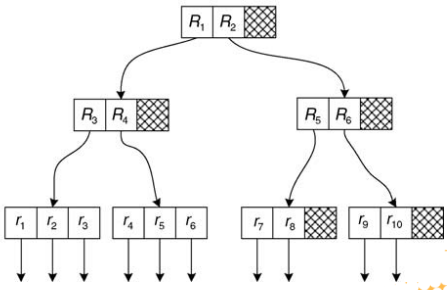


SDBMS physical level

- Unfortunately, spatial objects
  - (a) are of high complexity and (b) lack total ordering
- Nevertheless, can we do something?
  - Regarding (a): use spatial data approximations of low complexity
  - Regarding (b): adopt multi-dimensional search techniques

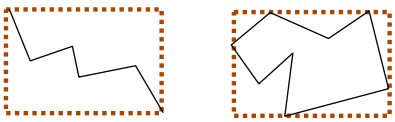


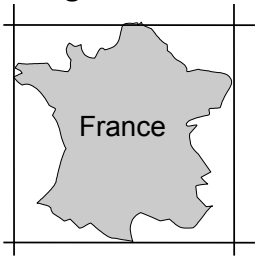




Spatial data approximations

- Minimum (Orthogonal) Bounding Rectangle (MBR)
  - MBR(obj) is the minimum orthogonal rectangle that covers obj





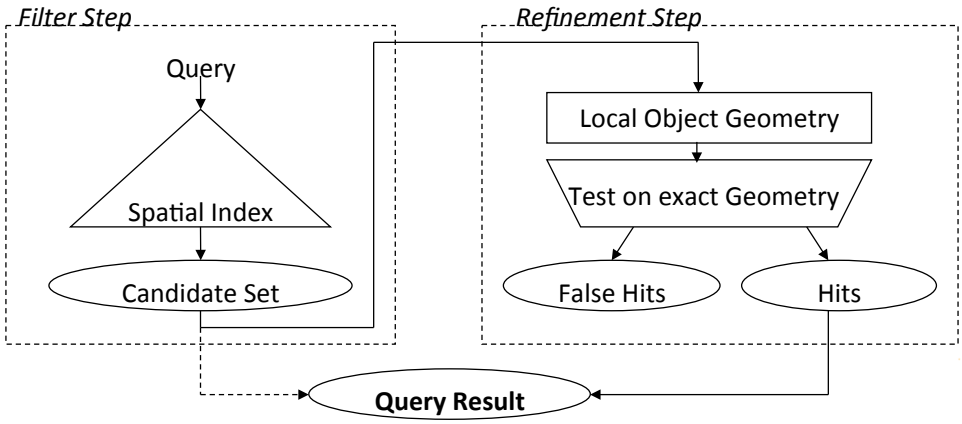
- Unfortunately, approximations are not identical to the original shapes they origin from ☹
  - Need for a **filter and refinement** procedure to support typical spatial queries

(parenthesis: “typical spatial queries”)

- Typical spatial queries are the following:
  - **Point** (D, p): find objects in dataset D covering point p
  - **Range** (D, r): find objects in dataset D that lie inside (or overlapping) region r
  - **NN** (D, p, k): find the (k-) object(s) in dataset D that lie nearest to point p
  - **SpatialJoin** (D1, D2): find all pairs (o1,o2) of objects in datasets D1, D2, that satisfy a spatial condition (usually, overlap)

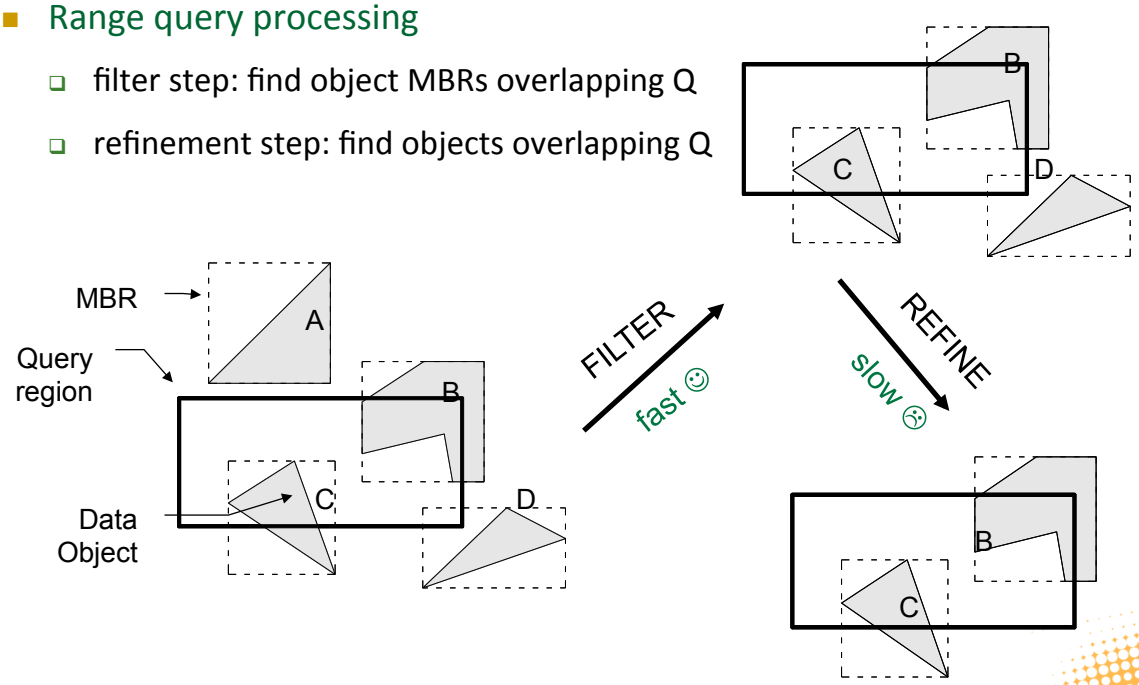
The Filter-Refinement procedure

- Processing a spatial query Q
  - Filter step: find a set S that contains (for sure) the answer set of Q using MBR approximations
  - Refinement step: find the exact answer set of Q by geometrically processing S



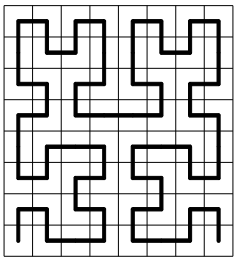
An example of filter – refinement

- Range query processing
  - filter step: find object MBRs overlapping Q
  - refinement step: find objects overlapping Q

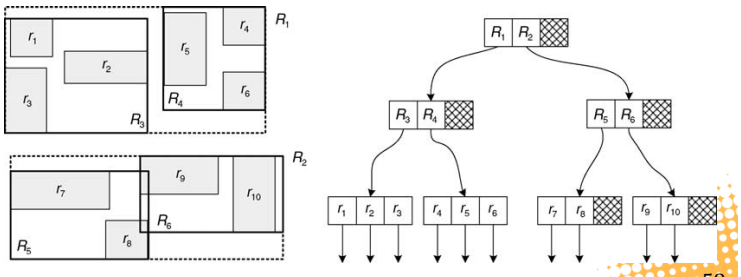


Indexing spatial objects

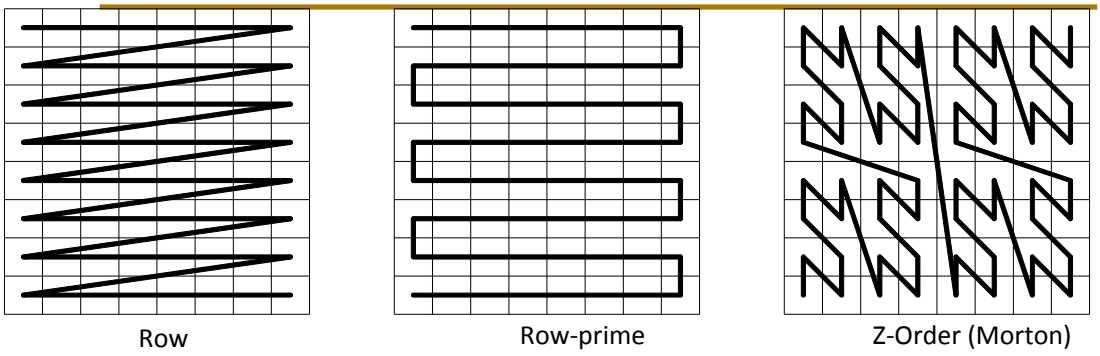
- Problem:
  - Cannot adopt "total ordering" in multi-dimensional space
    - If this was the case, we would have adopted the well-known B+-tree
- Solutions:
  - Adopt partial ordering (using space filling curves, e.g. Hilbert), transform 2D objects in 1D intervals, and exploit on e.g. B^+-trees, or
  - Invent novel spatial indexing techniques: R-trees, Quadtrees, etc.



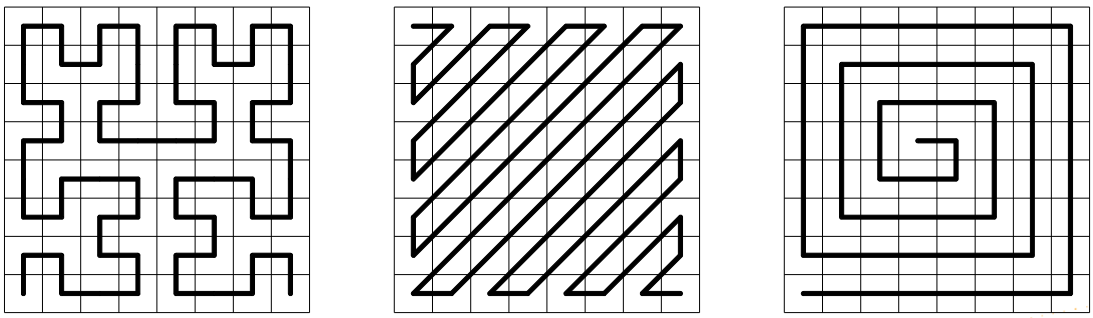
Hilbert curve



Space filling curves



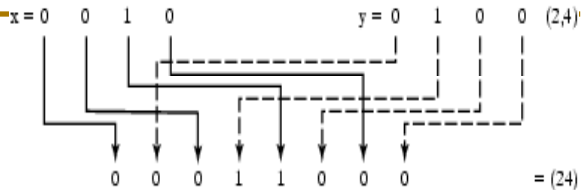
Row | Row-prime | Z-Order (Morton)

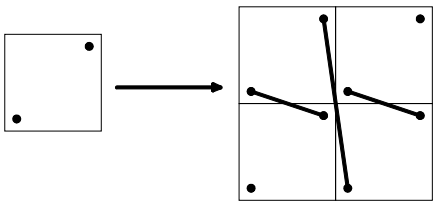


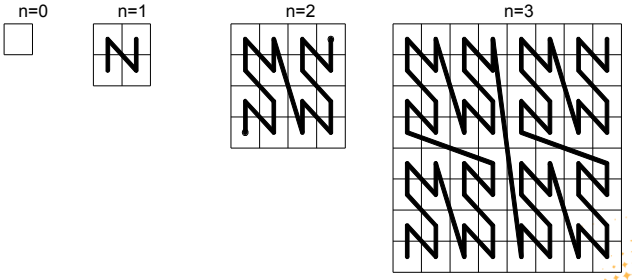
Hilbert | Cantor diagonal | Spiral

Z- vs. Hilbert curve

- Z-curve:
  - Produced by interleaving bits

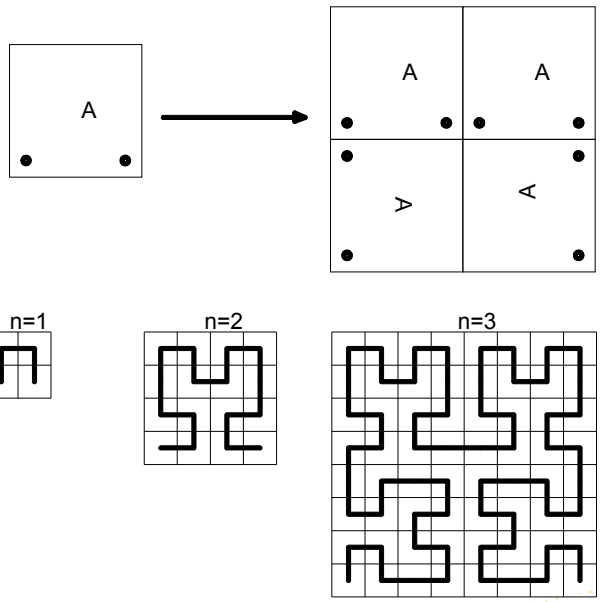






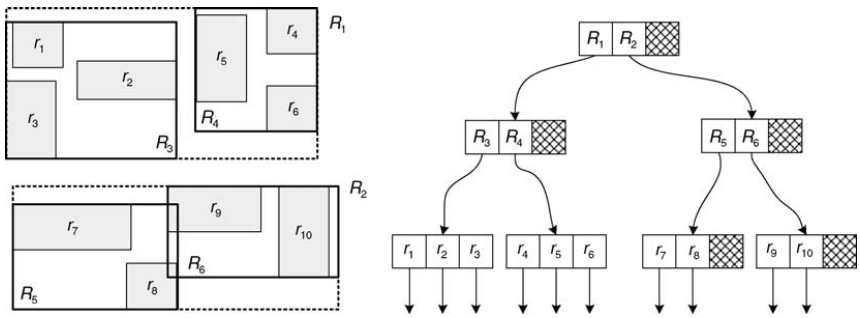
Z- vs. Hilbert curve

- Hilbert curve:
  - Produced by interleaving bits
  - … and rotating



R-tree (and family)

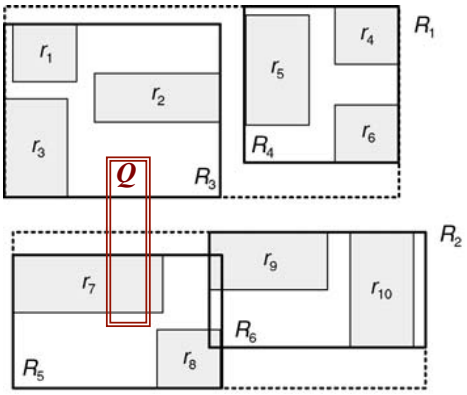
- The basic idea: extends B-tree to multi-dimensional space
- Basic properties:
  - Nodes correspond to disk pages; Balanced tree; Nodes consist of MBRs covering the entries of the lower level

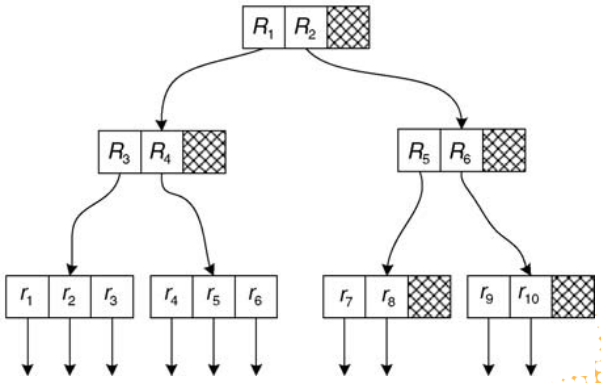


  - Implemented in Oracle, IBM DB2, PostgreSQL, etc.
  - Finds many applications: spatial, image, multimedia, time-series databases, OLAP, etc. (Manolopoulos et al. 2005)

Point / Range query processing in R-trees
[Guttman, 1984]

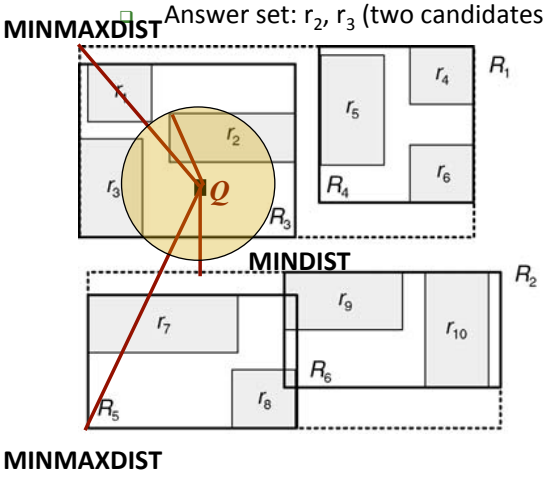
- Query window: Q
- Root level: Q overlaps R_1, R_2
- Depth-first propagation: node R_1 → node R_3, node R_2 → node R_5 → overlaps r_7
- Answer set: r_7

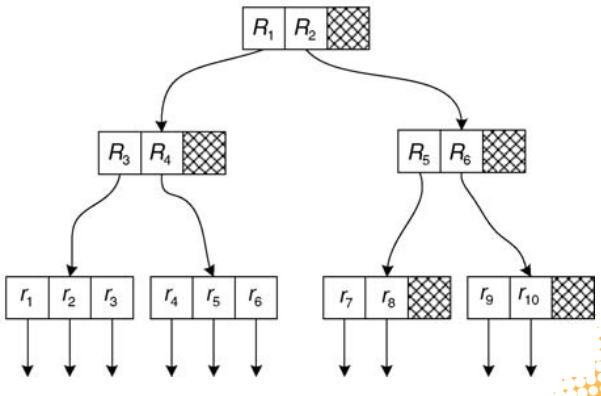




NN query processing in R-trees
[Roussopoulos et al. 1995]

- Query point: Q
- Root level: R_1, R_2 are candidates (at the moment…) for containing the answer
- Depth-first propagation (ask yourselves why…): node R_1 → node R_3 → r_2 and r_3 are candidate answers
  - side-effect: R_2 is pruned!
  - Answer set: r_2, r_3 (two candidates for the 1-NN !!)



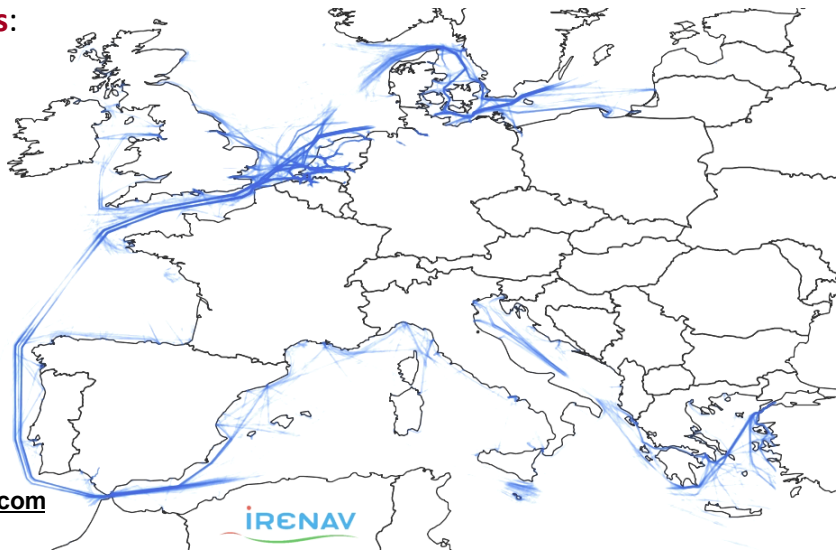


Spatial DB + Time = Spatio-temporal DB

- Including time dimension in spatial data is not straightforward (Koubarakis et al. 2003)
  - time is not simply a 3rd dimension (monotonicity, etc.)
- Adding motion in spatial objects (points, lines, regions)
  - Novel data types, e.g. “moving points” (Güting et al. 2000)
  - Spatio-temporal extensions of R-trees for indexing (Theodoridis et al. 1998)

An example of spatio-temporal DB

- Vessel traffic:
  - **Entities**: vessels, ports, coastlines, narrow passages, etc.
  - **Relationships** between entities: vessels’ scheduled trips (port-to-port)
  - **Spatial operations**: when does a vessel approach (pass through) a port (narrow passage), how close (and when) do two vessels approach each other,...

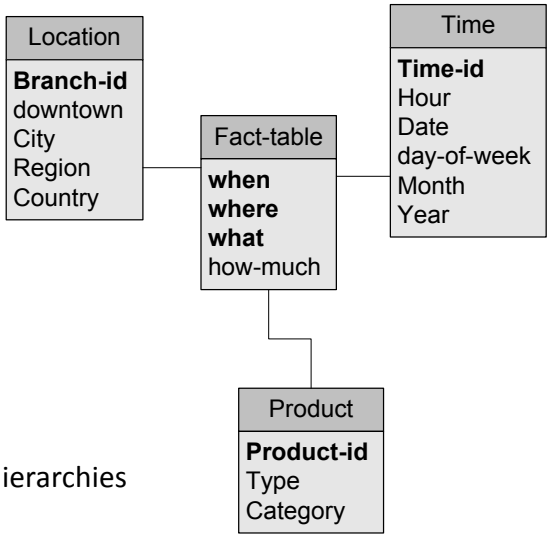


Live information: vesseltracker.com

Background on Spatial data exploration

Aggregating DB information: Data Cubes

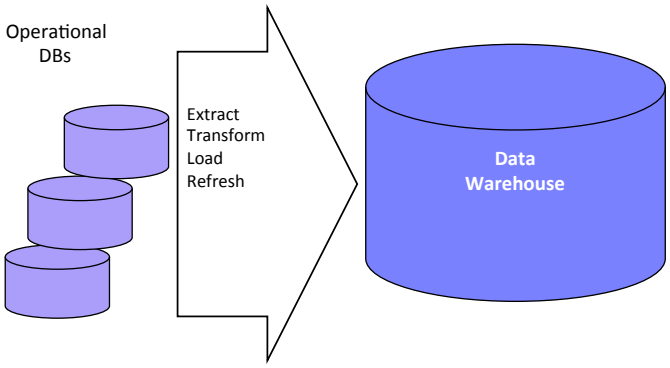
- Aggregated information from DBs is stored in **data cubes** [Gray et al. DMKD '97]
  - Feeded from DB via an Extract-Transform-Load (ETL) procedure
  - Technically, a collection of relations (if relational model is adopted)
- Typical structure: **star schema**
  - Several **dimension tables** with their hierarchies
  - One **fact table** with **measures**
  - Variation: constellation schema (more than one fact tables)



ETL example

- DB schema

```
product (product_ID,
   type, category)
location (branch_ID,
   downtown, city,
   region, country)
sales-transaction (
   timestamp, product_ID,
   branch_ID, units_sold,
   unit_price)
```

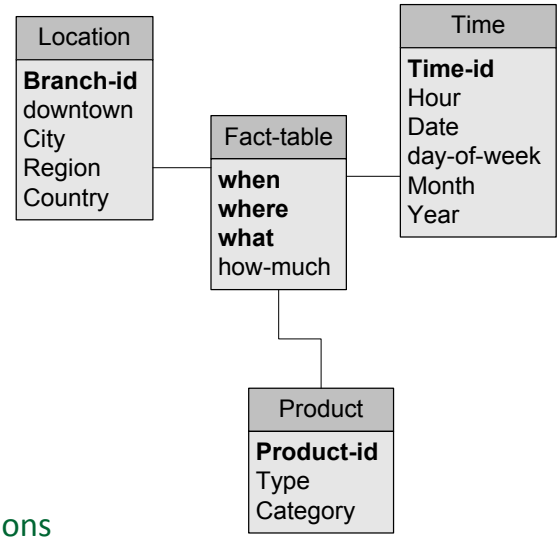


- ETL query

```
INSERT INTO sales
   ( SELECT datetime(timestamp) AS when,
            branch_ID AS where, product_ID AS what,
     sum(units_sold*unit_price) AS how-much
     FROM sales-transaction
     GROUP BY when, where, what
     HAVING how-much > 0 )
```


OLAP operations on data cubes

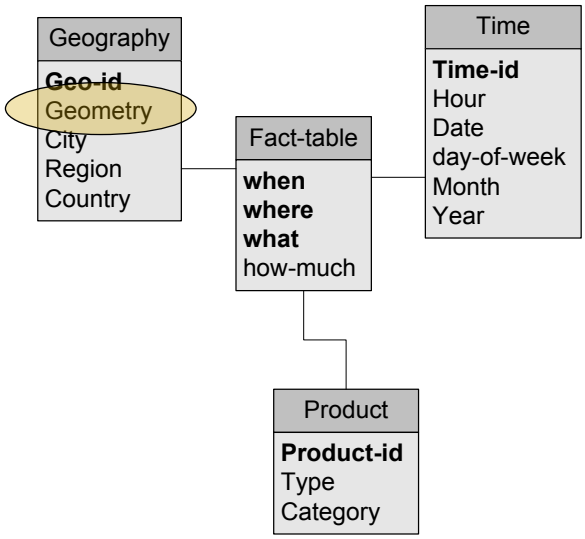
- A sequence of operations:
  - (**roll-up**) "What was the total turnover ("how-much" measure) per month and per city?"
  - (**slice**) "Especially in March, what was the turnover per city?"
  - (**drill-down**) "Especially in March, what was the turnover on weekdays vs. weekends?"
  - (**cross-over**) "Display the DB records that support the above result."



- Degree of efficiency of OLAP operations depends on the type of measures
  - distributive vs. algebraic vs. holistic

Data cubes for spatial data

- **Spatial data cubes**
  [Han et al. PAKDD'98]
  - Dimensions
    - Spatial (e.g. Geography) vs.
    - non-spatial /thematic (e.g. Time, Product)
  - Measures:
    - Numerical vs. Spatial



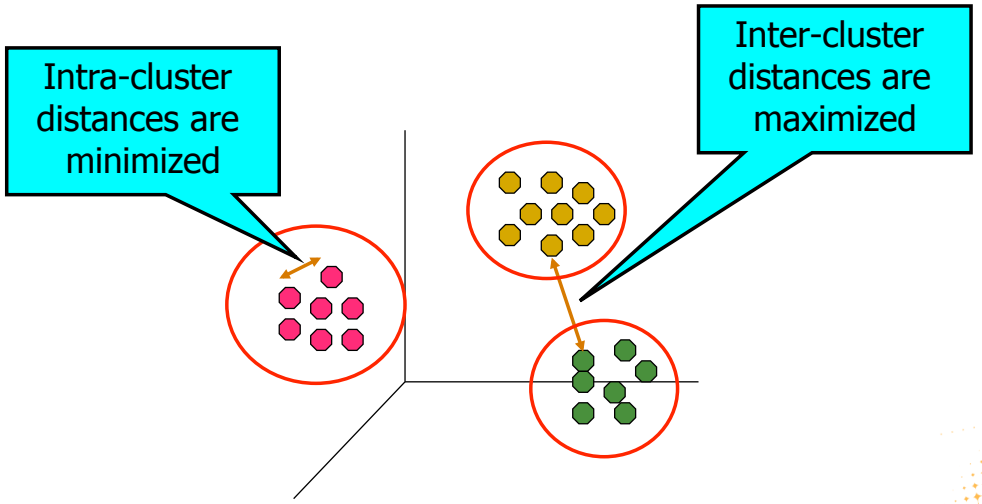
Cluster analysis (and outlier detection)

- The settings:
  - A dataset of entities $D = \{e_1, e_2, \ldots, e_N\}$
  - For each pair of entities, a distance $Dist(e_{ij})$ can be measured (hence, a NxN distance matrix is potentially formed)
    - (hopefully) the distance measure $Dist(e_{ij})$ should be a **metric**.
- The objective goal:
  - Partition entities of D into K groups (**clusters**), $G_1, \ldots, G_K$ with the following properties:
    - $\cup G_i = D$, $G_i \cap G_j = \varnothing$
    - The intra-cluster (inter-cluster) distance between entities is minimized (maximized, resp.), as better as possible

What is Cluster Analysis?

original slide from (Tan et al. 2004)

- Finding groups of objects such that:
  - the objects in a group will be similar (or related) to one another and
  - … different from (or unrelated to) the objects in other groups



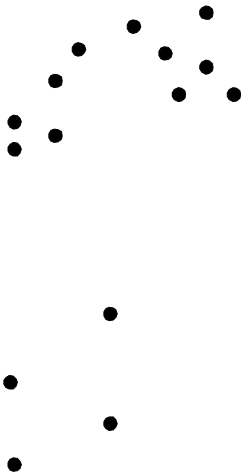
Types of Clusterings

original slide from (Tan et al. 2004)

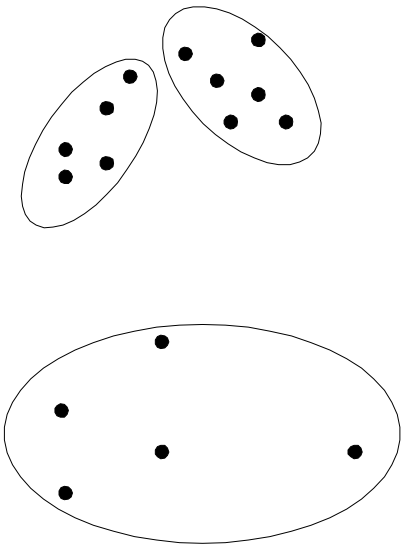
- A **clustering** is a set of clusters
- Important distinction between **hierarchical** and **partitional** sets of clusters
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree
- Partitional Clustering
  - A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset

Partitional Clustering

original slide from (Tan et al. 2004)



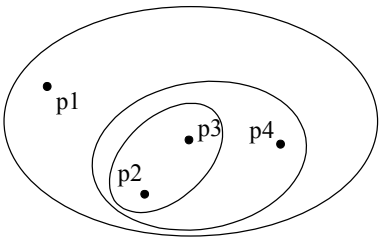
Original Points



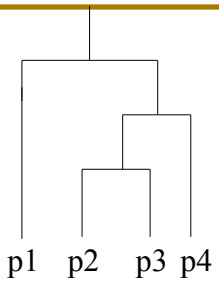
A Partitional Clustering

Hierarchical Clustering

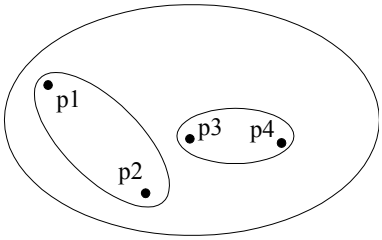
original slide from (Tan et al. 2004)



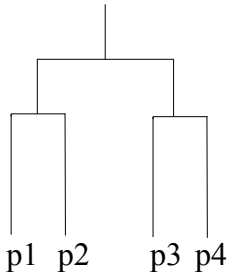
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Clustering Algorithms

original slide from (Tan et al. 2004)

- Partitional algorithm
  - K-means and its variants
- Hierarchical clustering
  - Agglomerative vs. divisive approaches
  - Issue: how to measure distance between two clusters?
- Density-based clustering
  - DBSCAN, OPTICS, etc.

K-means Clustering

original slide from (Tan et al. 2004)

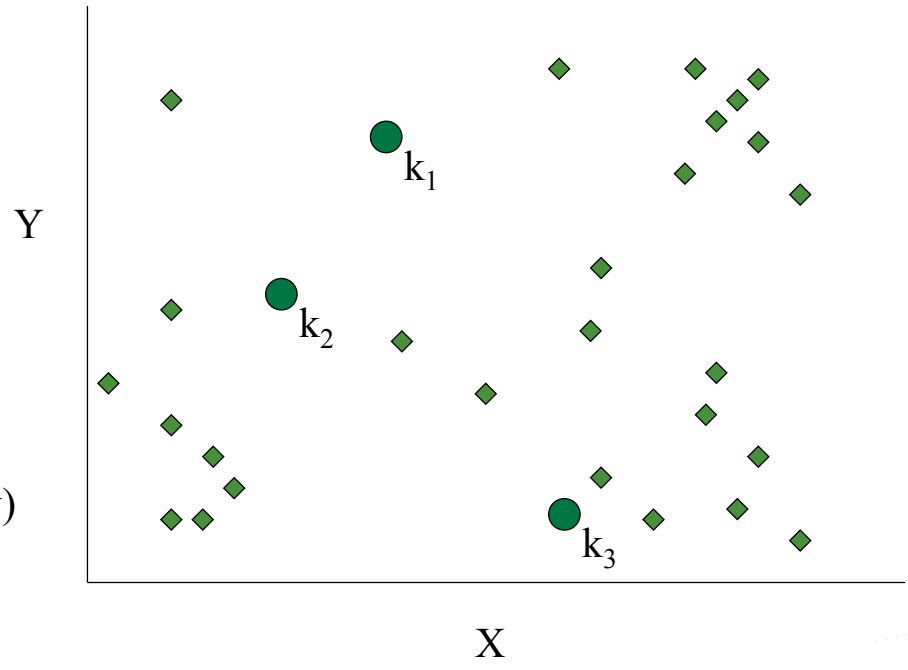
- Partitional clustering approach
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified
- The basic algorithm is very simple

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

K-means example, step 1

original slide from (Piatetsky-Shapiro 2003)

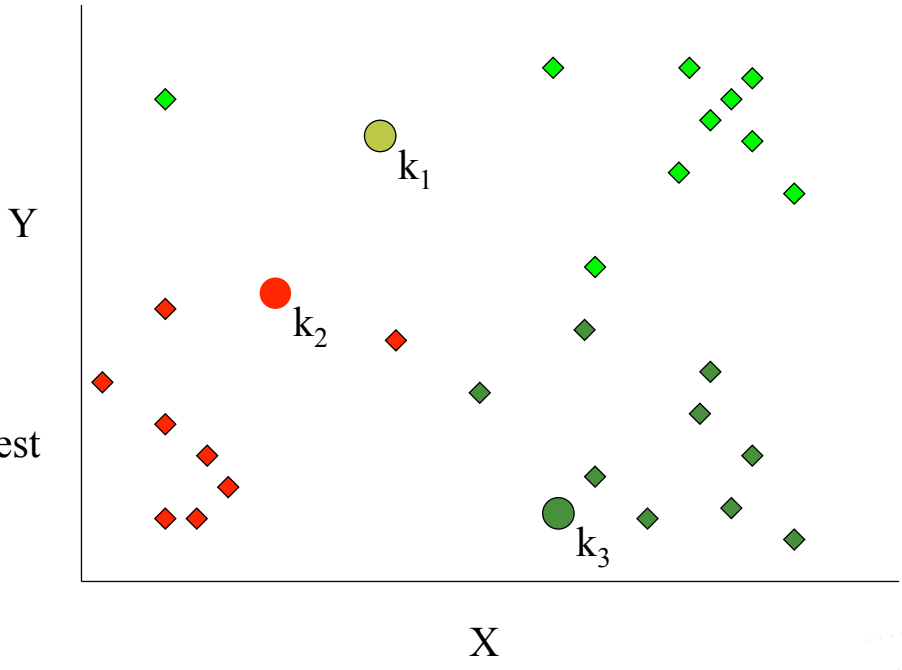
Pick 3 initial cluster centers (randomly)



K-means example, step 2

original slide from (Piatetsky-Shapiro 2003)

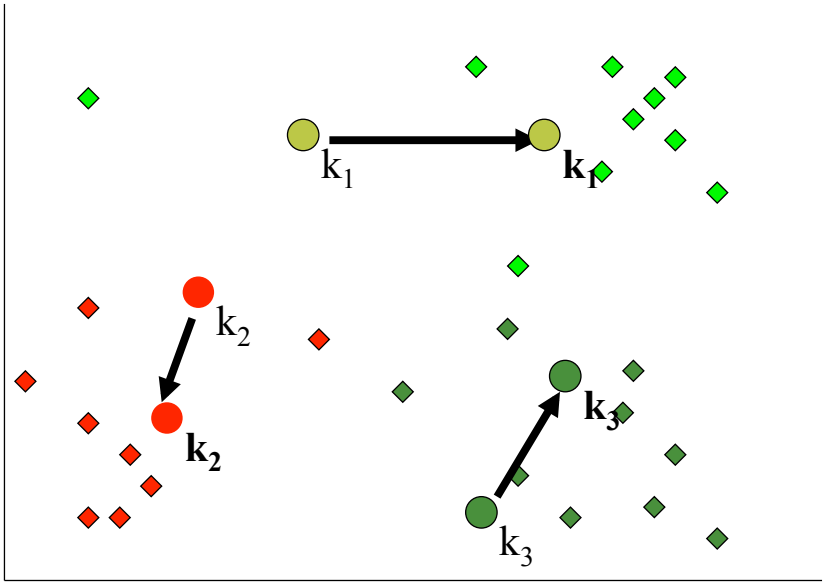
Assign each point to the closest cluster center



K-means example, step 3

original slide from (Piatetsky-Shapiro 2003)

Move each cluster center to the mean of each cluster

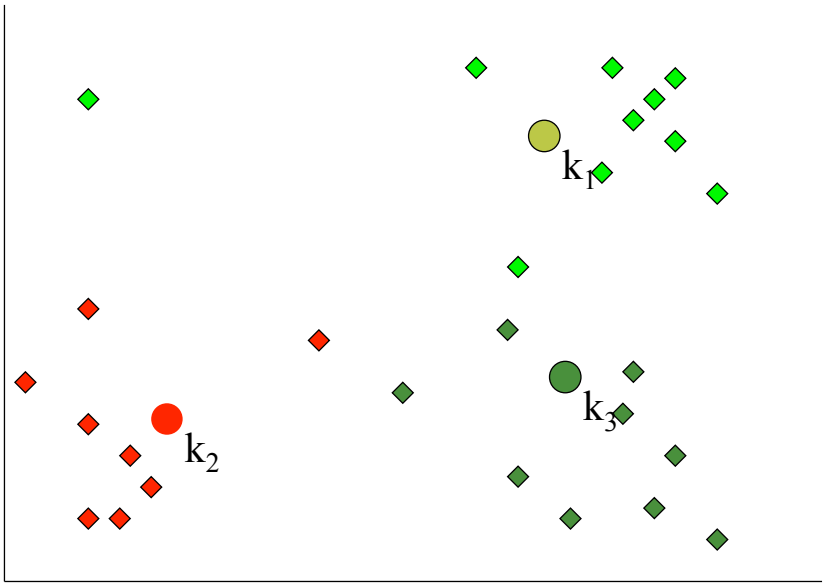


K-means example, step 4

original slide from (Piatetsky-Shapiro 2003)

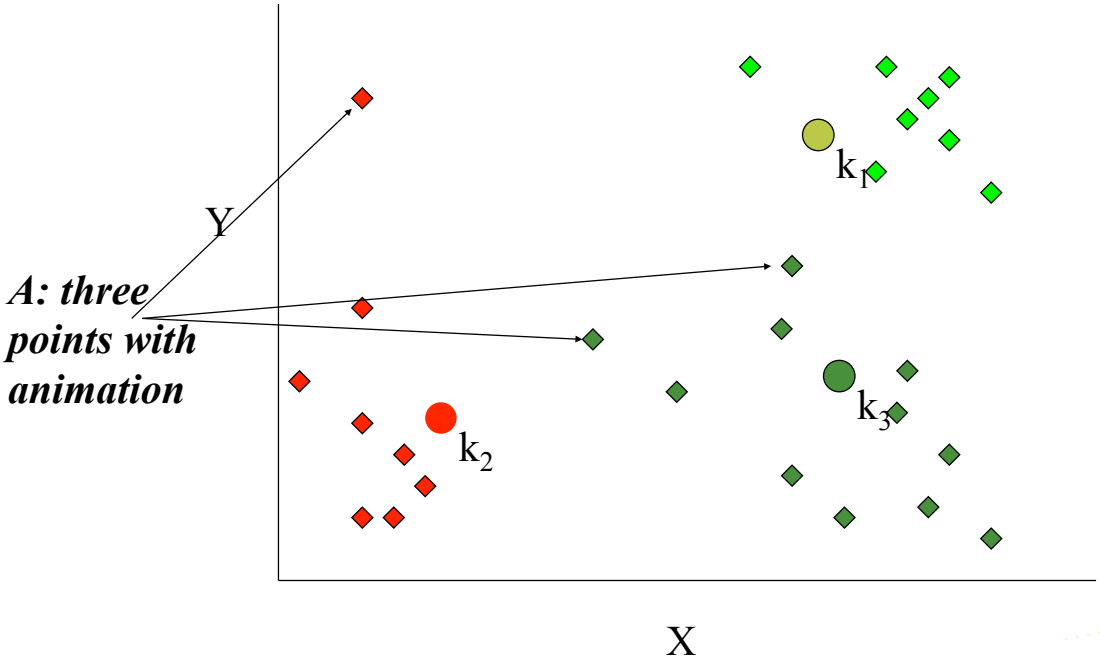
Reassign points closest to a different new cluster center

Q: Which points are reassigned?



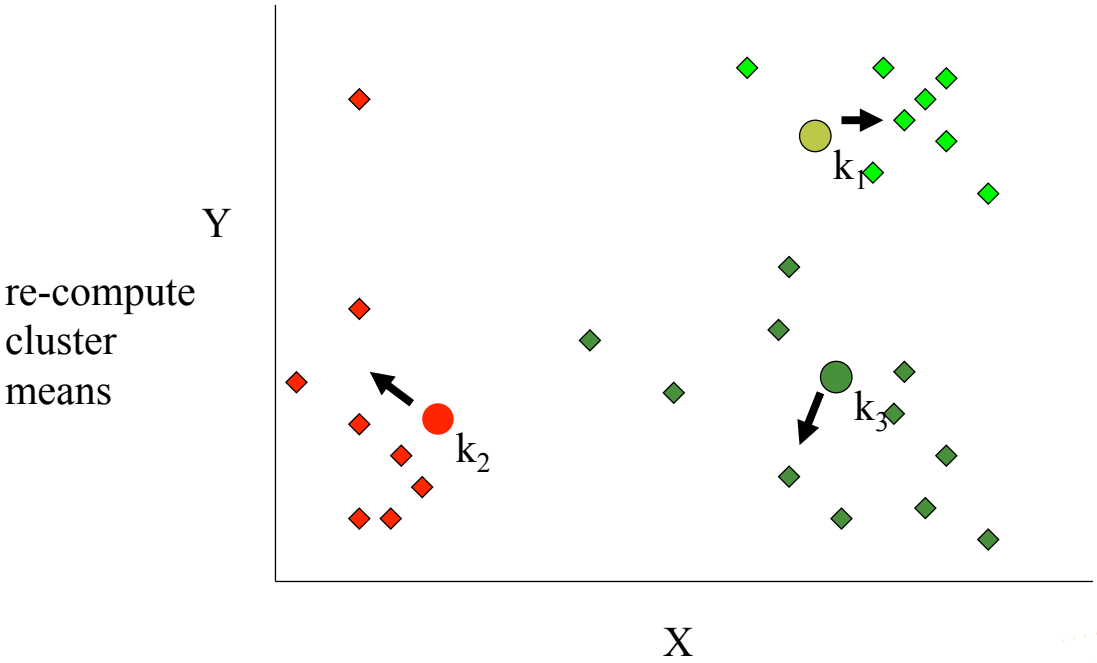
K-means example, step 4 …

original slide from (Piatetsky-Shapiro 2003)



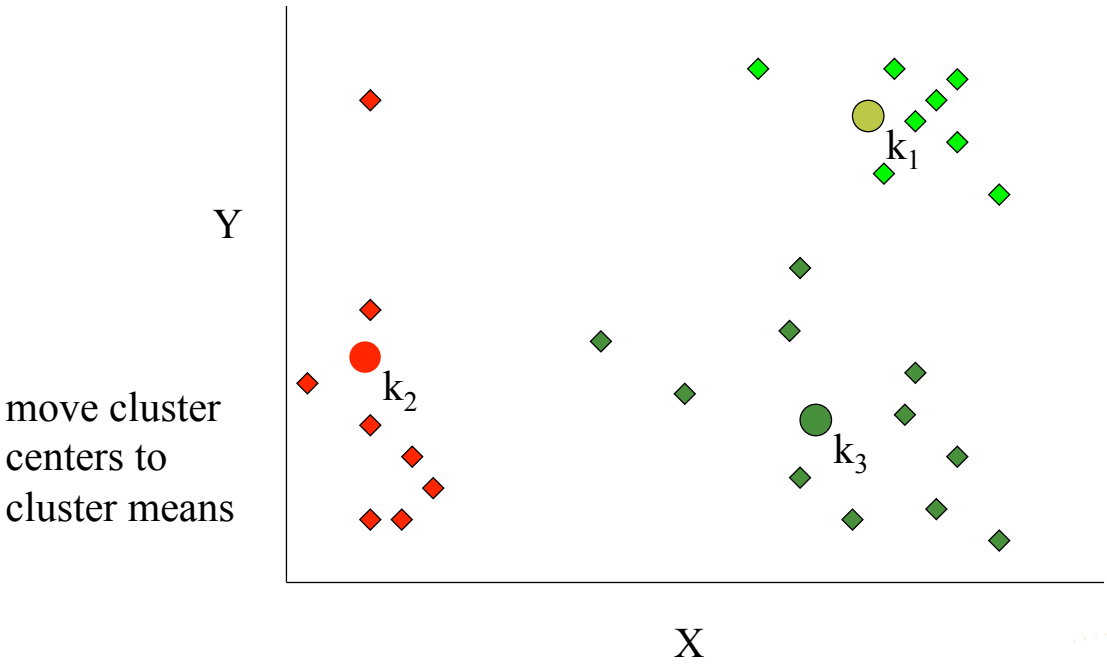
K-means example, step 4b

original slide from (Piatetsky-Shapiro 2003)



K-means example, step 5

original slide from (Piatetsky-Shapiro 2003)



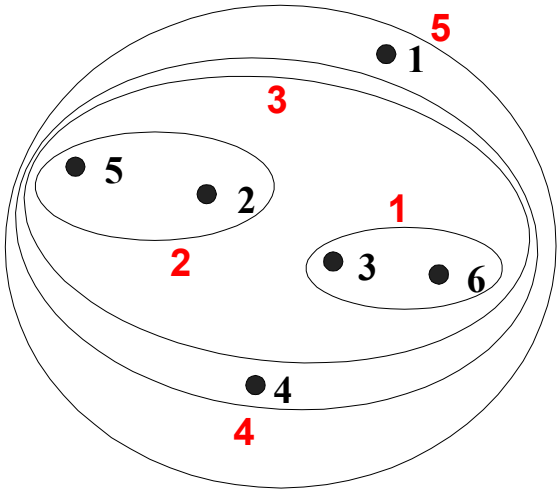
Hierarchical Clustering

original slide from (Tan et al. 2004)

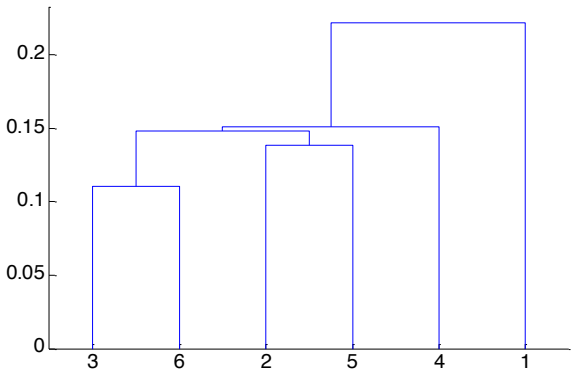
- Two main types of hierarchical clustering
  - **Agglomerative**:
    - Start with the points as individual clusters; At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - **Divisive**:
    - Start with one, all-inclusive cluster; At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

Hierarchical Clustering – an example

original slide from (Tan et al. 2004)



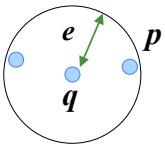
Nested Clusters



Dendrogram

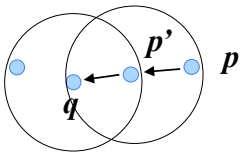
DBSCAN

- DBSCAN (Ester et al. KDD'96) is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
- The notion of **density reachability**
  - Directly Density-Reachable

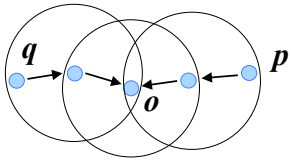


$m = 3$

  - Density-Reachable



  - Density-Connected

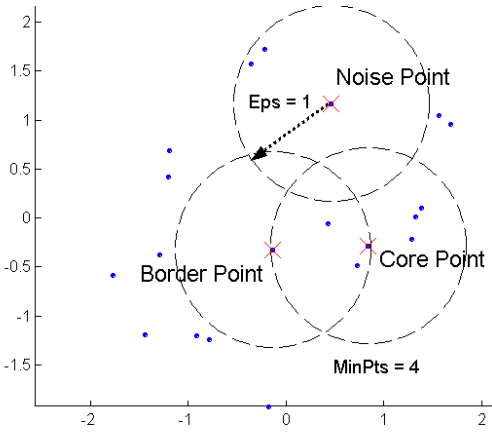


DBSCAN

- Core vs. Border vs. Noise points.

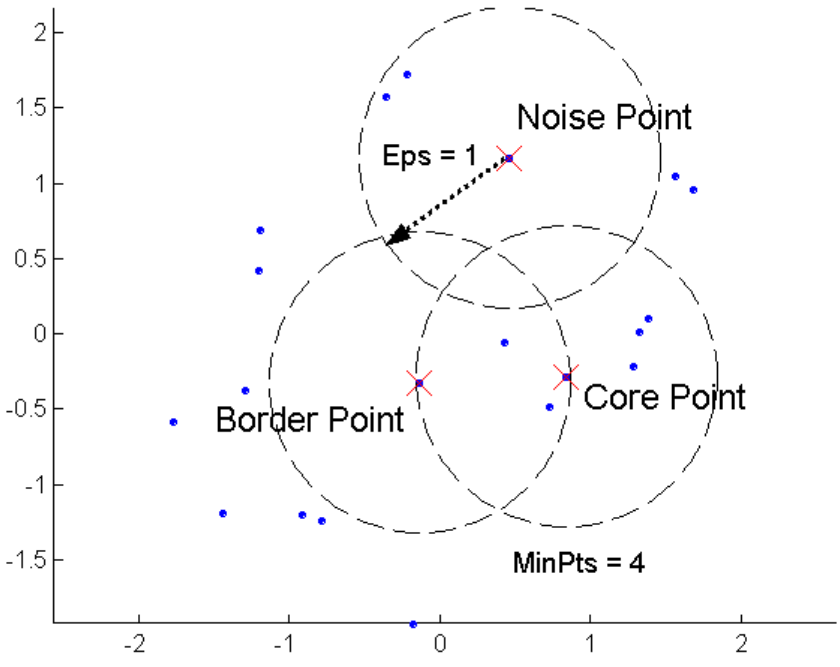
  A point is characterized as:

  - **core point**, if it has at least MinPts points within its Eps neighborhood
    - Core points are expected to be the cores of clusters
  - **border point**, if it is not a core point, but it lies in the neighborhood of a core point
    - Border points are expected to be included in the clusters of their cores
  - **noise point**, otherwise
    - Noise points are expected to be excluded from clusters (hence, outliers)



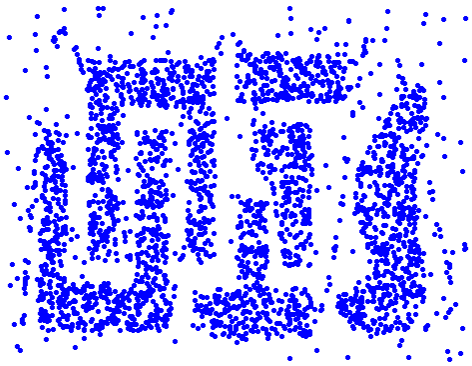
DBSCAN: Core, Border, and Noise Points

original slide from (Tan et al. 2004)

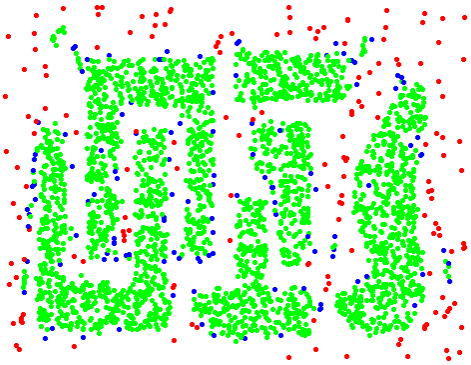


DBSCAN: Core, Border and Noise Points

original slide from (Tan et al. 2004)



Original Points



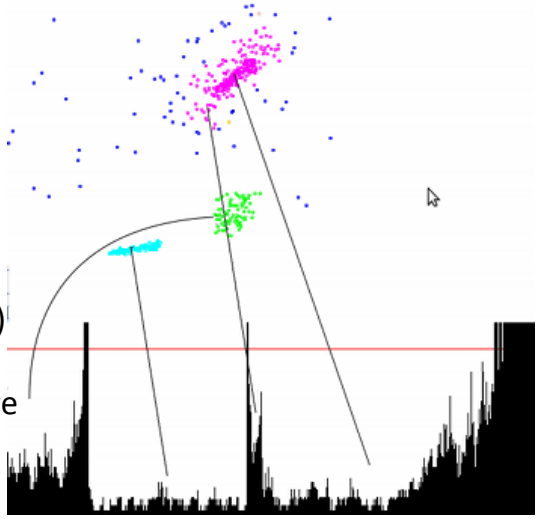
Point types: core, border and noise

Eps = 10, MinPts = 4

OPTICS: Ordering Points To Identify the Clustering Structure

- Addresses DBSCAN's major weakness:
  - the problem of detecting meaningful clusters in data of varying density
- OPTICS methodology
  - Points are (linearly) ordered according to their closeness
  - A special distance is stored ("**reachability**") for each point that represents the density to be accepted for a cluster in order to have both points belong to the same cluster

Reachability plot



$$\text{core-distance}_{\varepsilon,MinPts}(p) = \begin{cases} \text{UNDEFINED} & \text{if } |N_\varepsilon(p)| < MinPts \\ \text{distance to the } MinPts\text{-th point} & \text{otherwise} \end{cases}$$

$$\text{reachability-distance}_{\varepsilon,MinPts}(p,o) = \begin{cases} \text{UNDEFINED} & \text{if } |N_\varepsilon(o)| < MinPts \\ \max(\text{core-distance}_{\varepsilon,MinPts}(o), \text{distance}(o,p)) & \text{otherwise} \end{cases}$$

Conclusions

- Assumptions
  - Wireless networks infrastructures are the nerves of our territory
  - besides offering their services, they gather highly informative traces about human (animal, etc.) mobile activities
  - Ubiquitous computing infrastructure will further push this phenomenon
- Therefore,
  - Mobility data collections will be more and more popular …
  - … asking for effective and efficient **management** and **exploration**
    - … while, in parallel, taking **privacy issues** into consideration

Questions

Reading list

Positioning & tracking technologies

- Bajaj R. et al. (2002) GPS: Location-Tracking Technology. IEEE Computer, 35(4): 92-94.
- Bar-Noy, A. and I. Kessler (1993) Tracking Mobile Users in Wireless Communication Networks. IEEE/ACM Transactions on Information Theory, 39(6):1877-1886.
- Bulusu, N. et al. (2000) GPS-less Low Cost Outdoor Localization for Very Small Devices. IEEE Personal Communications Magazine, 7(5):28-34.
- Djunkic, G.M. and R.E. Richton (2001) Geolocation and Assisted GPS. IEEE Computer, 34(2):123-125.
- Hofman-Wellenhoff, B. et al. (1997) Global Positioning System: Theory and Practice, 4th ed. Springer.
- Kaplan, E. (1996) Understanding GPS Principles and Applications. Artech House.

Positioning & tracking technologies

- Mauve, M. et al. (2001) A survey on position-based routing in mobile ad hoc networks. IEEE Network Magazine, 15(6):0-39.
- Misra, A. et al. (2004) An Information-Theoretic Framework for Optimal Location Tracking in Multi-System 4G Networks. Proceedings of IEEE INFOCOM Conf.
- Porcino, D. (2001) Location of Third Generation Mobile Devices: A Comparison between Terrestrial and Satellite Positioning Systems. Proceedings of IEEE Vehicular Technology Conf.
- Ward, A. et al. (1997) A New Location Technique for the Active Office. IEEE Personal Communications, 4(5):42-47.
- Xiang, Z. et al. (2004) A Wireless LAN-based Indoor Positioning Technology. IBM Journal of Research and Development, 48(5/6):617-626.

Spatial/temporal data modeling

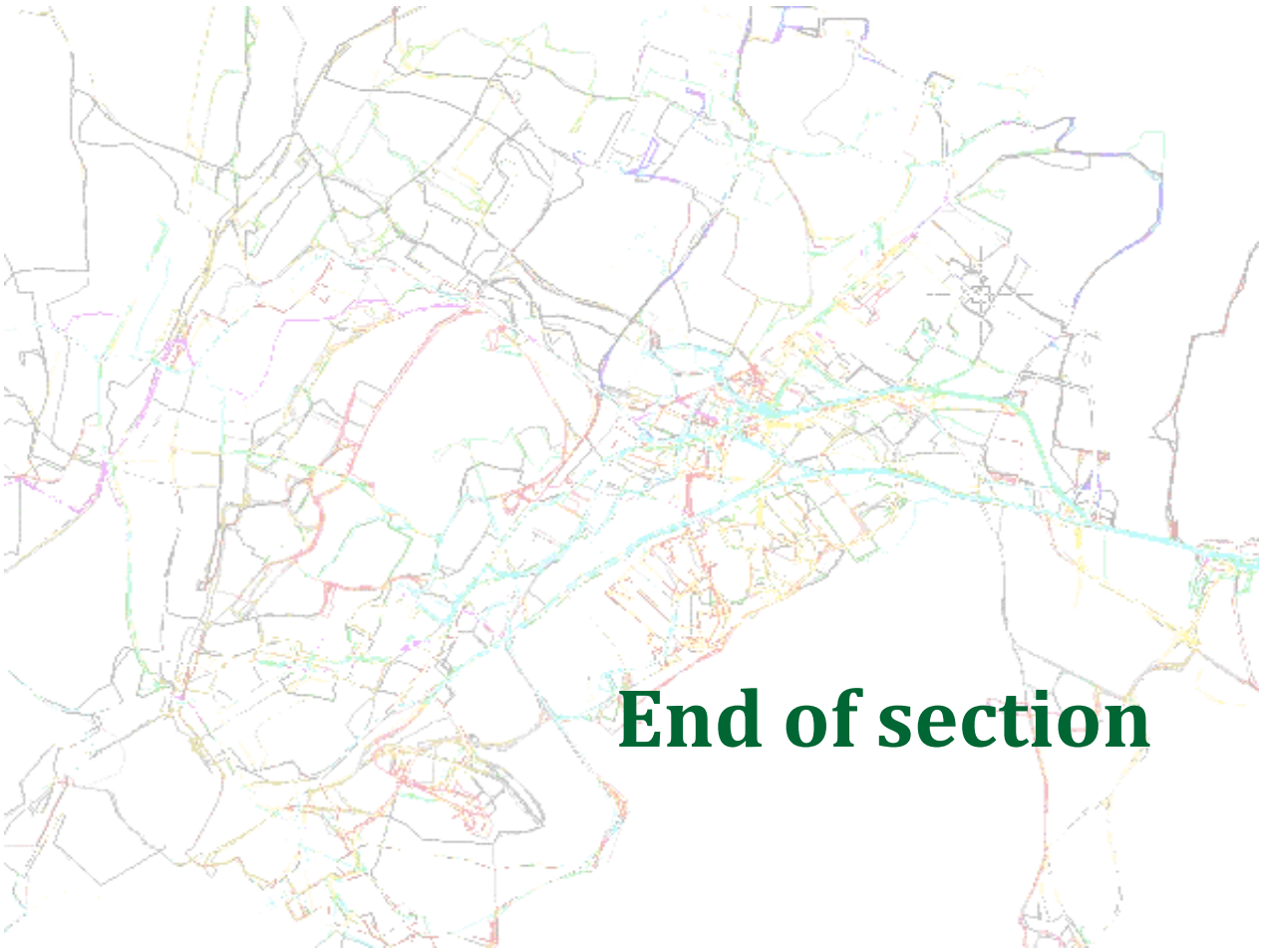
- Allen JF (1983) Maintaining knowledge about temporal intervals. Communications of the ACM, 11, 832-843.
- Egenhofer MJ (1989) A Formal Definition of Binary Topological Relationships. Proceedings of FODO Conference.
- Egenhofer MJ, Sharma J (1993) Topological Relations Between Regions in R^2 and Z^2. Proceedings of SSD Conference.

Spatial/temporal database management

- Güting, R.H. et al. (2000) A Foundation for Representing and Querying Moving Objects. ACM Transactions on Database Systems, 25(1):1-42
- Guttman, A. (1984) R-trees: A Dynamic Index Structure for Spatial Searching. Proceedings of ACM SIGMOD Conference.
- Koubarakis, M. et al. (2003) Spatio-Temporal Databases – the Chorochronos approach. Springer.
- Manolopoulos, Y. et al. (2005) R-trees: Theory and Applications. Springer.
- Roussopoulos, N. et al. (1995) Nearest Neighbor Queries. Proceedings of ACM SIGMOD.
- Theodoridis, Y. et al. (1998) Specifications for Efficient Indexing in Spatio-temporal Databases. Proceedings of SSDBM.

Online Resources

- Positioning technologies
  - 3GPP specifications, http://www.3gpp.org/specs/specs.htm
  - ETSI – European Telecommunication Standards Institute. http://www.etsi.org
  - Open GIS Consortium, OpenGIS® Location Services (OpenLS): Core Services, http://www.openls.org
  - Open Mobile Alliance (OMA), http://www.openmobilealliance.org
  - OpenPrivacy Initiative, http://www.openprivacy.org
  - Trimble: All About GPS, http://www.trimble.com/gps
- Spatio-temporal database management
  - ChoroChronos.org. A portal of datasets and algorithms for mobility data management. http://www.chorochronos.org



End of section