
# Mobility Data Management & Exploration

## Ch. 06. Preparing for Mobility Data Exploration

**Nikos Pelekis & Yannis Theodoridis**

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

v.2014.05

*“The only source of knowledge is experience.”*
*Albert Einstein*

# Chapter outline

6.1. Mobility Data Warehousing

6.2. OLAP Analysis in Trajectory Data Cubes

6.3. Calculating Similarity between Trajectories

6.4. Summary

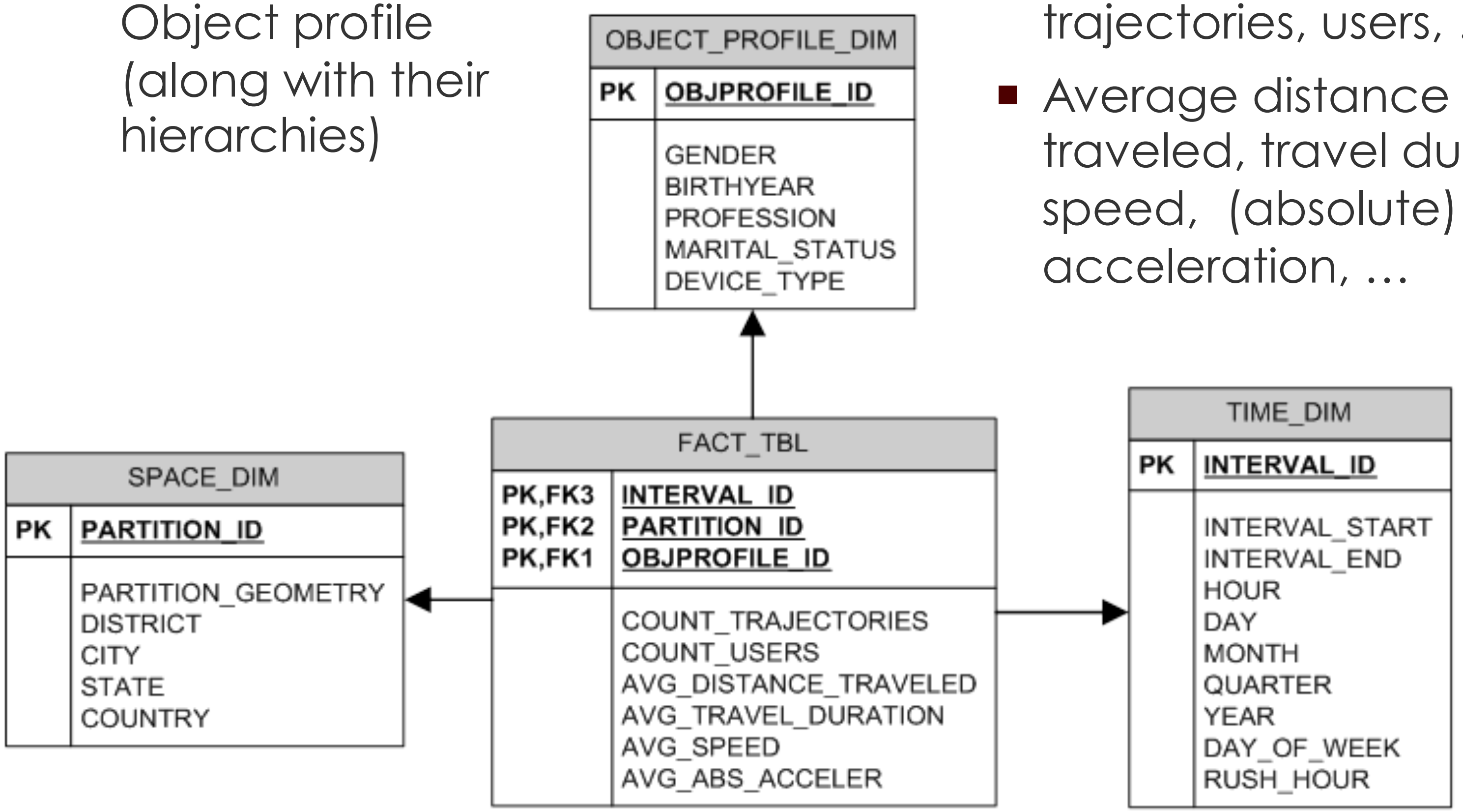
# 6.1.

# Mobility data warehousing

# Modeling trajectory data cubes

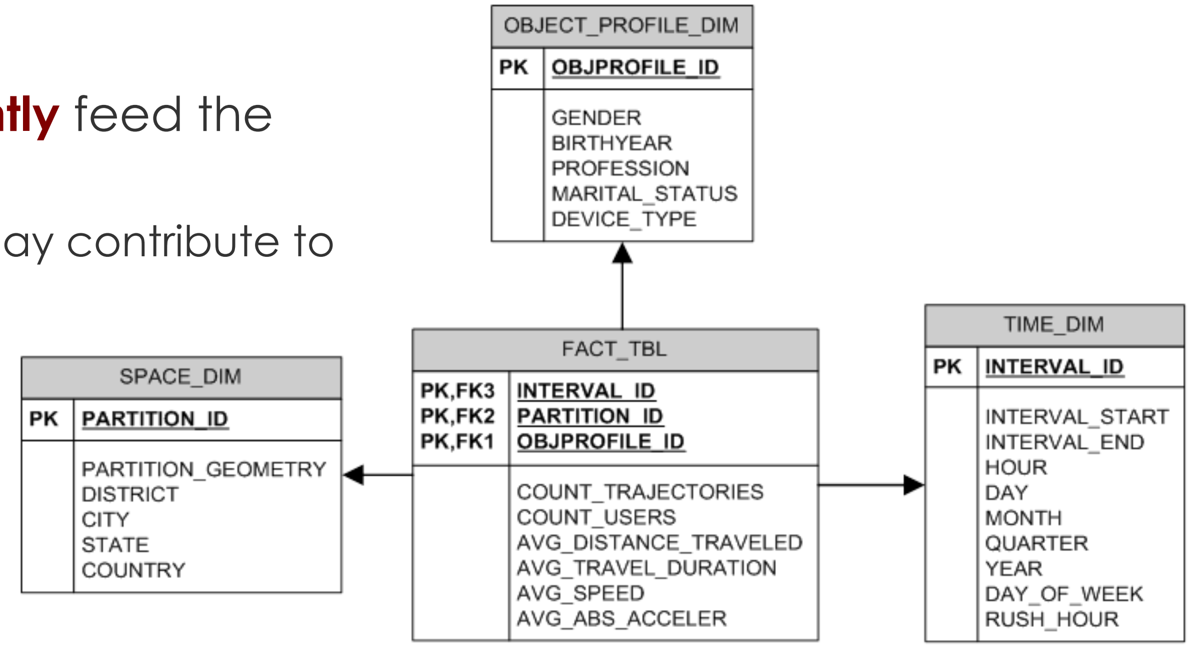
- **Dimensions** should include at least:
  - Space, Time, Object profile (along with their hierarchies)

- **Measures** should include at least:
  - Distinct count of trajectories, users, …
  - Average distance traveled, travel duration, speed, (absolute) acceleration, …



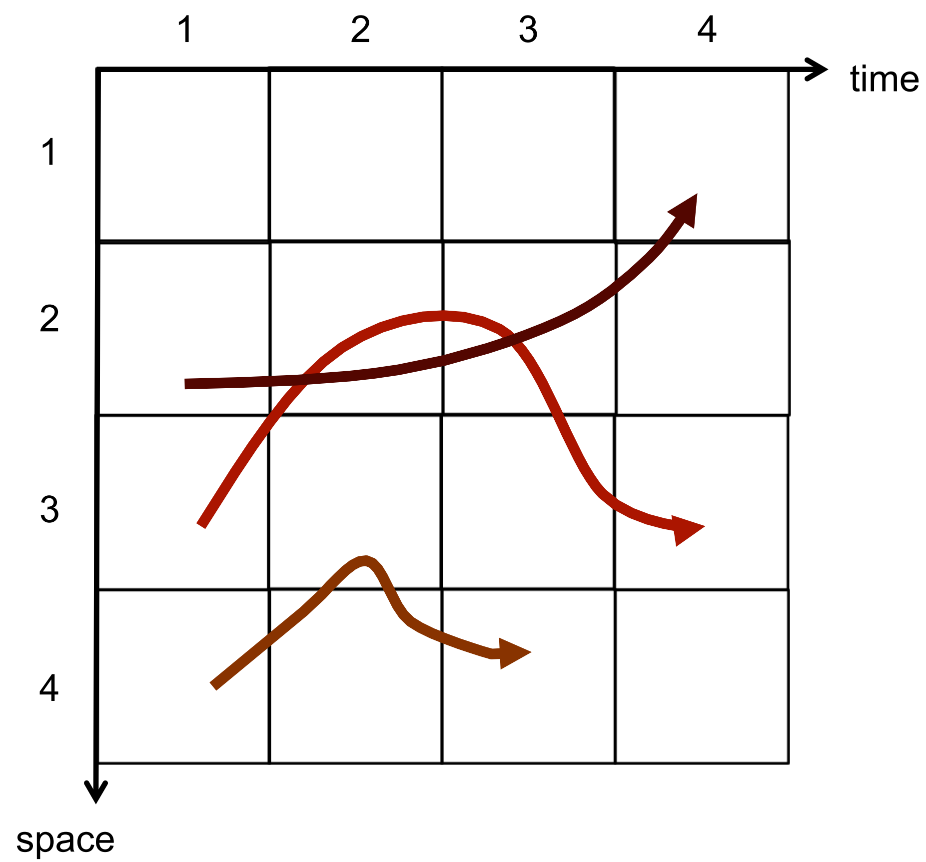
# Issues to be resolved

- During data cube design
  - The effect of spatial / temporal resolution in data cube size
  - An example: 1% of total extent (spatial resolution) X 10 min interval (temporal resolution) X 10 object profiles for 3 years history = **1.5 trillion records fact table** !!
  - (as usual) tradeoff between quality and usage of resources
- During ETL:
  - how to **efficiently** feed the fact table?
    - A trajectory may contribute to several cells



# Loading the data cube

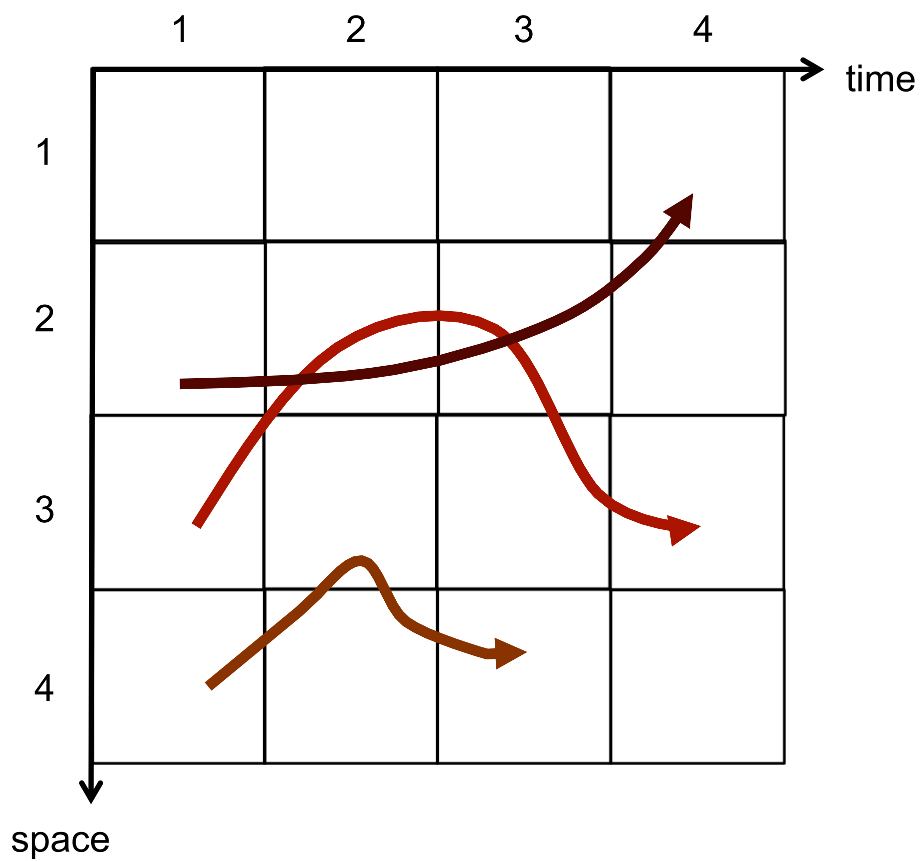
- Loading data into the dimension tables ➔ straightforward
- Loading data into the fact table ➔ complex, expensive
  - In order to calculate the measures of the fact table, we have to extract the sub-trajectories that fit into the base cells
  - cell- vs. trajectory-oriented approach



# Loading the data cube (cont.)

<u>Algorithm Cell-oriented-ETL-approach (COA)</u>
Input: trajectory database T, space partitioning S, time partitioning $\tau$
Output: measure matrix M
1. FOR each (spatiotemporal) base cell $c_{jk} = (s_j, \tau_k)$ in $S \times \tau$
2. Search for sub-trajectories in T that are contained in $c_{jk}$
3. Compute measures M[j,k]

<u>Algorithm Trajectory-oriented-ETL-approach (TOA)</u>
Input: trajectory database T, space partitioning S, time partitioning $\tau$
Output: measure matrix M
1. FOR each trajectory $T_i$ in T
2. Find the (spatiotemporal) base cell $c_{jk} = (s_j, \tau_k)$ in $S \times \tau$, $T_i$ is contained in
3. Compute measures M[j,k]

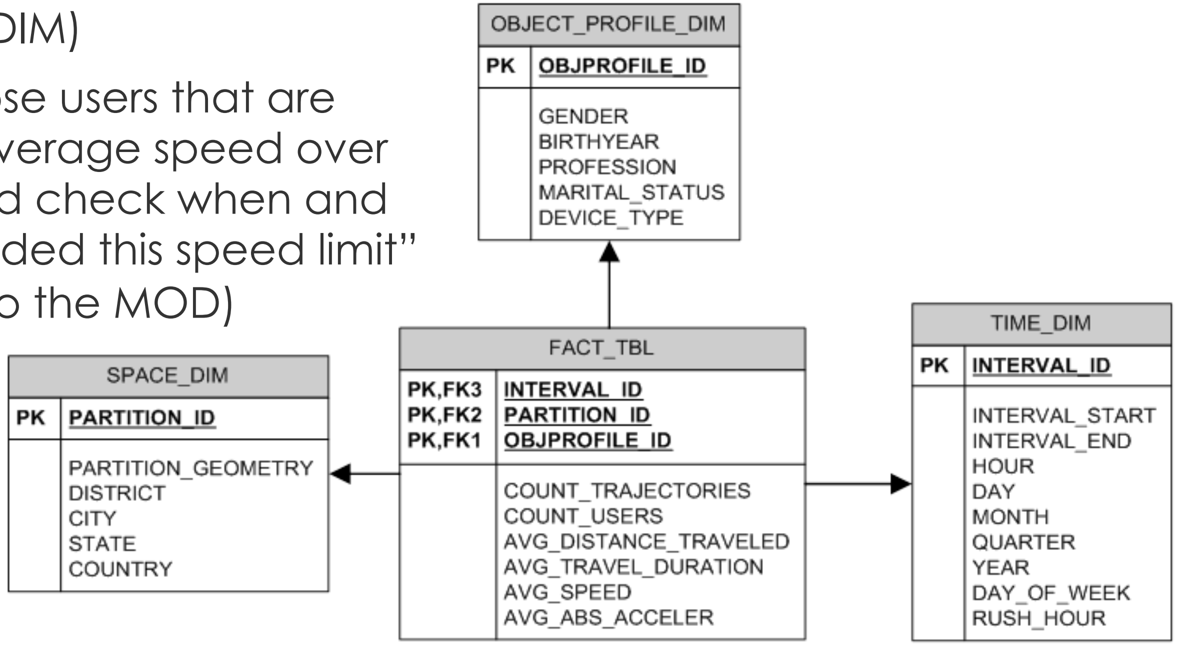


# 6.2.

# OLAP analysis in trajectory data cubes

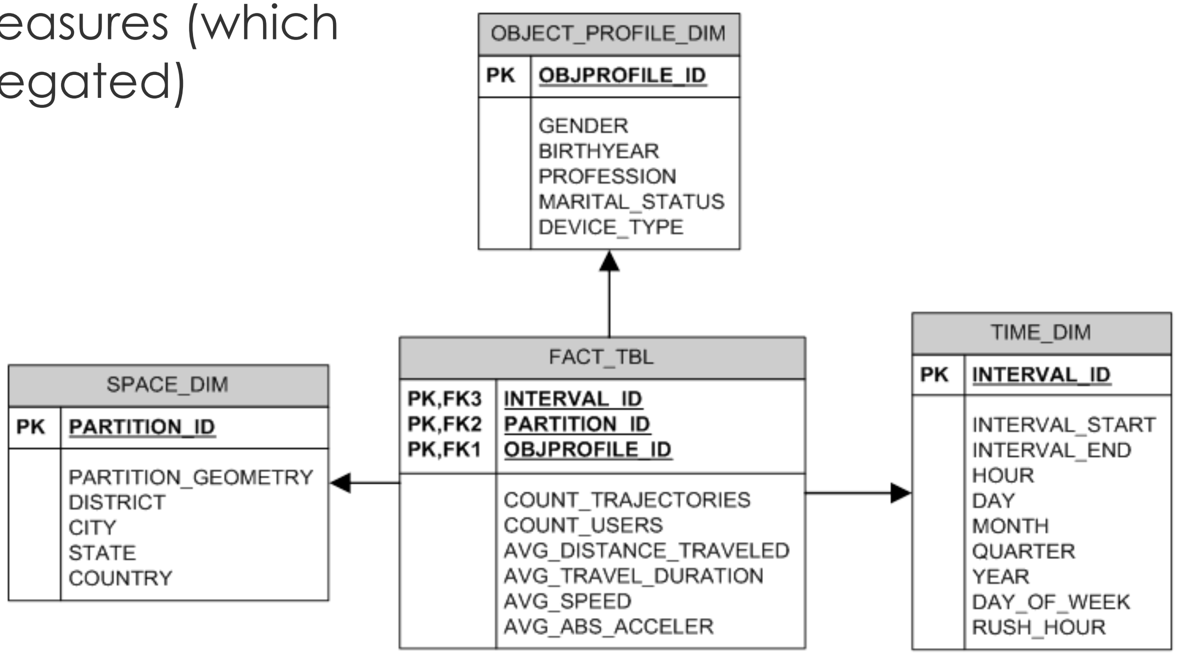
# Multi-dimensional (OLAP) analysis

- Typical OLAP operations: roll-up, drill-down, slice, cross-over
- Example of progressive analysis:
  - "Display the number of users and their average speed, for each space partition and per hour" (roll-up in table TIME_DIM)
  - "Then, focus on downtown area, night hours and young drivers, and display their average speed" (roll-up in table SPACE_DIM, slice in table SPACE_DIM, slice in table TIME_DIM, slice in table OBJECT_PROFILE_DIM)
  - "Then, retrieve those users that are 'responsible' for average speed over the speed limit and check when and where they exceeded this speed limit" (cross-over back to the MOD)



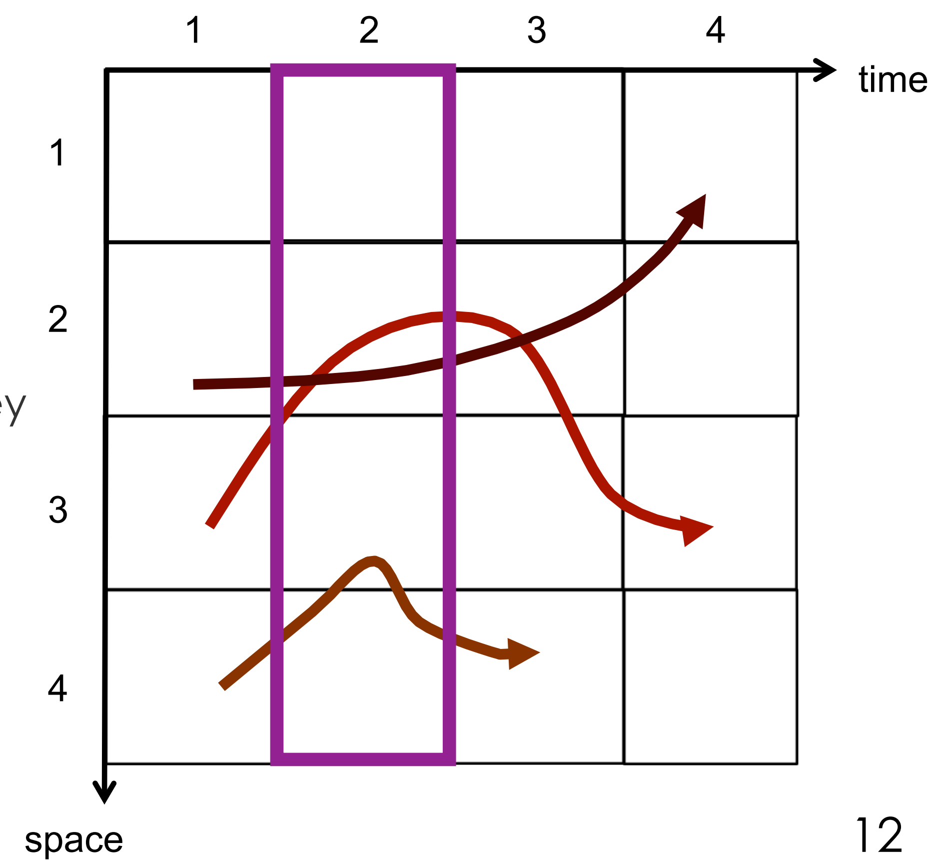
# Issues to be resolved during OLAP

- The problem (informally): a trajectory may contribute to several cells; what happens when rolling-up?
- The so-called "**distinct count problem**"
  - A trajectory may visit several cells or even the same cell multiple times
  - Hence, it contributes multiple times in the measures (which are then aggregated)



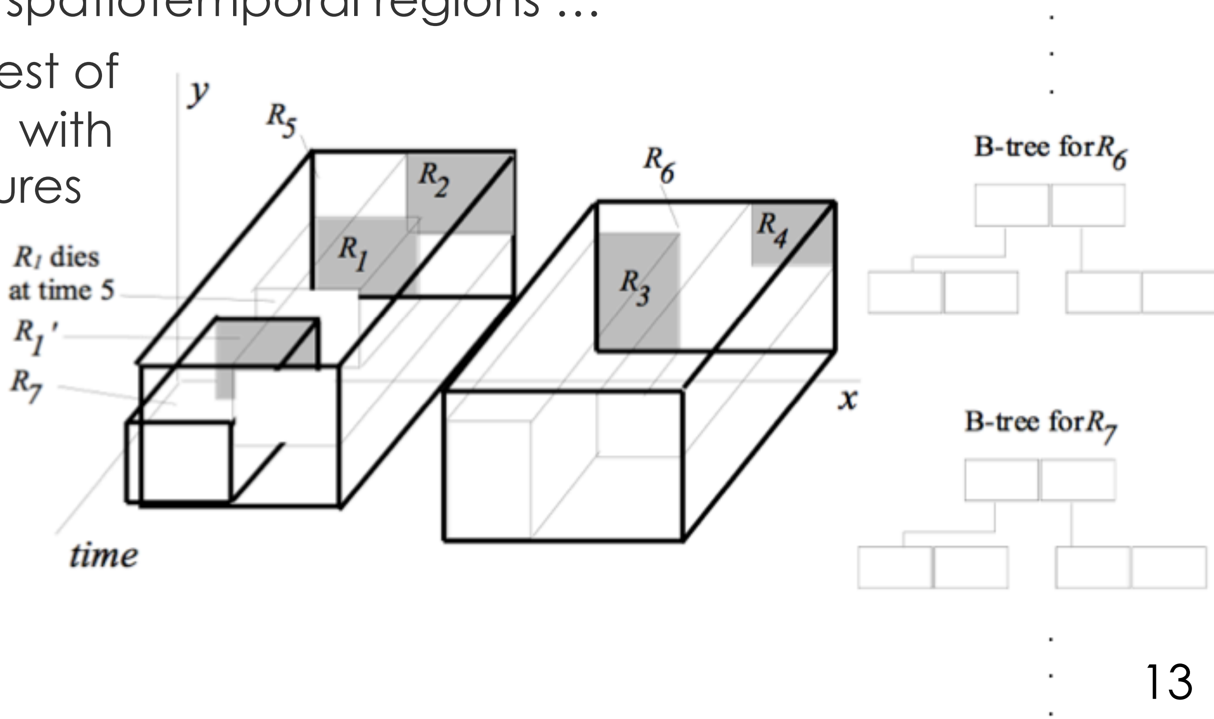
# The distinct count problem

- The problem (formally): Given a space partitioning S, a time partitioning τ, and a measure matrix M of size |S|×|τ|, the **distinct count problem** is to estimate as better as possible the resulting measure after aggregating in space and time due to a roll-up operation.
- Example: what is the number of trajectories at the union of cells $C_{i2}$, i = 1..4?
  - 3 instead of 4 (= 0+2+1+1)
- How to calculate this number?
  - Problem: we are not aware of the contributing trajectory ids since they are not stored in the data cube
- A (sub-optimal) solution: keep a note on the borders between base cells
  - In the above example, 4 – 1 = 3 !!



# Indexing for efficient OLAP

- For performance reasons, aggregate data could be stored in appropriate indexes.
- Main target: **window aggregate query**
- A proposal: **a3DRB-tree**
  - a 3D R-tree for the spatiotemporal regions …
  - … along with a forest of aggregate B-trees with the numeric measures of each region

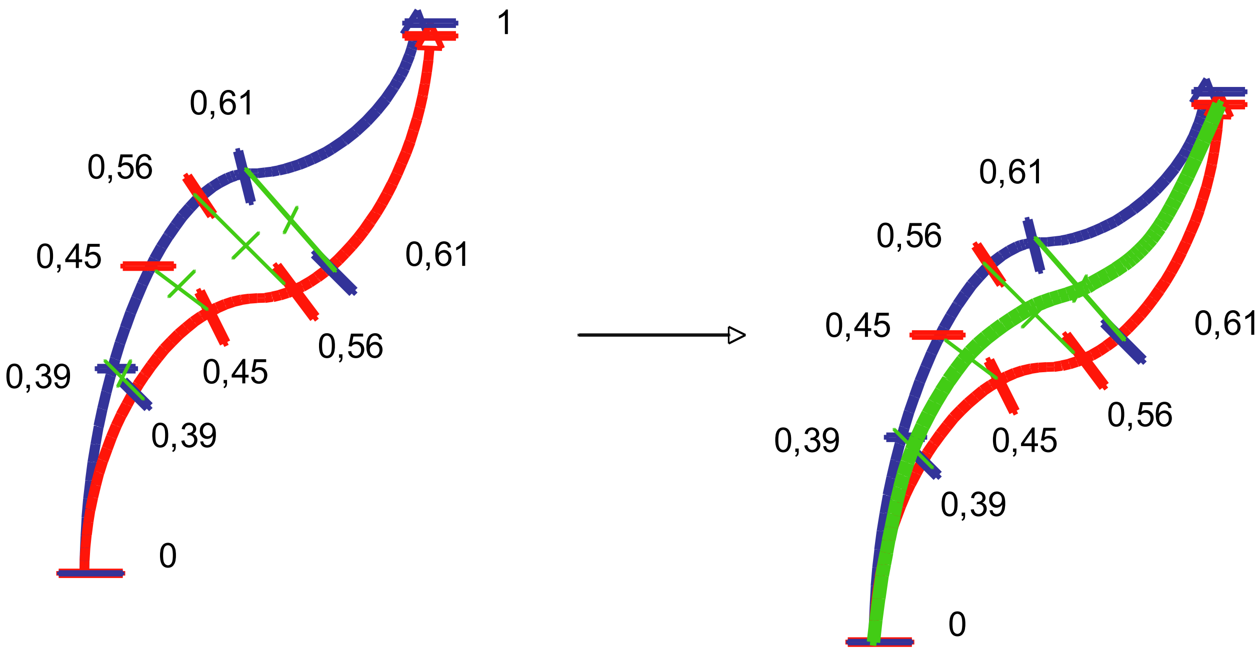


# 6.3.

# Calculating similarity between trajectories

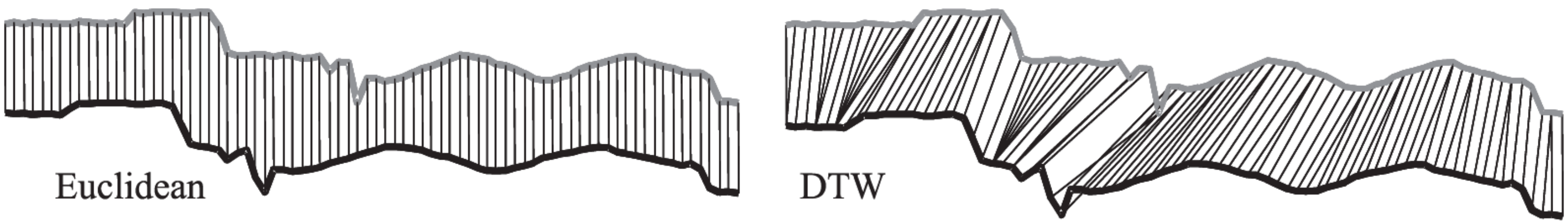
# Trajectory Similarity

- Key question: How do we measure **distance** or **(dis-) similarity** between two trajectories?
  - Not as simple as it sounds!
- A straightforward solution: (sum of) Euclidean distance(s) between respective points
  - The '**average**' trajectory can be calculated this way.



# Trajectory as a time-series

- Time-series similarity has been studied extensively
- Examples from the time-series domain
  - Euclidean distance, Chebyshev distance, Dynamic Time Warping (DTW), Longest Common SubSequence (LCSS), Edit Distance on Real sequences (EDR), Edit distance with Real Penalty (ERP), Swale, etc.




- However, trajectories are not identical to time-series! Both <u>where</u> and <u>when</u> are important

# Trajectory as a 3D polyline

- (extension of Euclidean distance) **DISSIM function**:


$$DISSIM(R,S) = \int_{t_1}^{t_n} L_2\big(R(t),S(t)\big)dt$$

  - … and its approximate computation:

$$DISSIM(R,S) = \sum_{k=1}^{n-1} \int_{t_k}^{t_{k+1}} L_2\big(R(t),S(t)\big)dt$$

$$DISSIM(R,S) \approx \frac{1}{2}\sum_{k=1}^{n-1} \Big(\Big(L_2\big(R(t_k),S(t_k)\big) + L_2\big(R(t_{k+1}),S(t_{k+1})\big)\Big) \cdot (t_{k+1} - t_k)\Big)$$

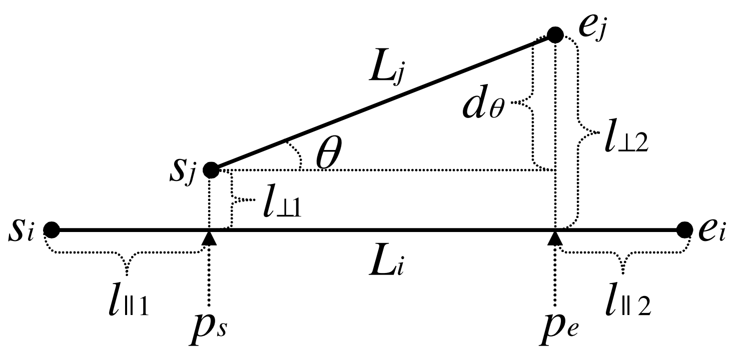
# Trajectory as a 3D polyline (cont.)

- The **Earth Movers Distance** (EMD)
  - weighted sum of two energies: translation $d_\perp(r_i, s_j)$ + rotation $d_\angle(r_i, s_i)$

$$d_\perp(r_i, s_j) = \sqrt{w_1 \cdot (x_r - x_s)^2 + w_1 \cdot (y_r - y_s)^2 - w_2 \cdot (t_r - t_s)^2}$$

$$d_\angle(r_i, s_j) = \min(|r_i|, |s_j|) \cdot |\theta|$$

- The **TRACLUS** approach:
  - weighted sum of three components (distances between directed segments): perpendicular $d_\perp$ + parallel $d_{||}$ + angular $d_\angle$



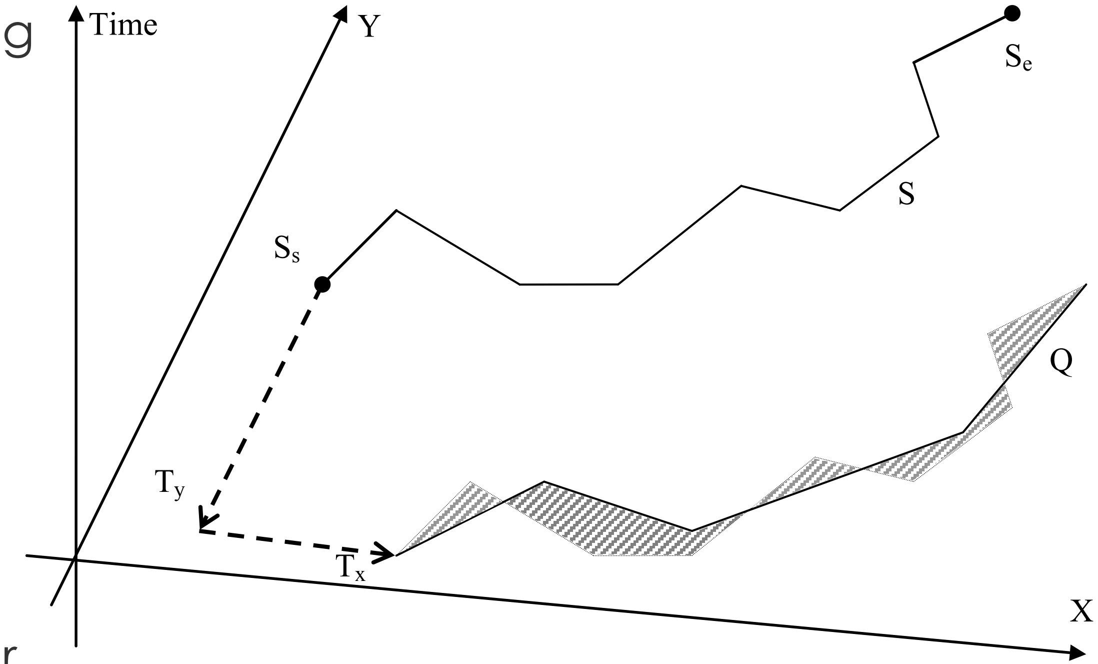

$$d_\perp = \frac{l_{\perp 1}^2 + l_{\perp 2}^2}{l_{\perp 1} + l_{\perp 2}}$$

$$d_\parallel = \mathrm{MIN}(l_{\parallel 1}, l_{\parallel 2})$$

$$d_\theta = \|L_j\| \times \sin(\theta)$$

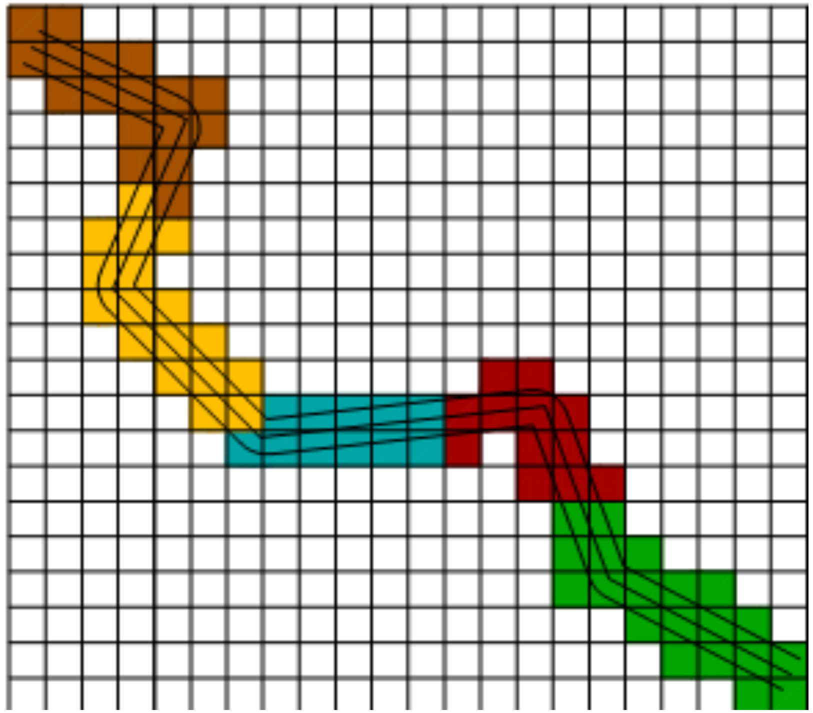
# Trajectory as a 3D polyline (cont.)

- The Locality In-between Polylines (**LIP**) distance function
  - Projects on the 2D space (assuming equal starting points)
  - Calculates the area in between the two (projected) routes
- LIP is meaningful when the two objects move (more or less) towards the same direction
  - Hence, it can be better applied on pairs of sub-trajectories of the original trajectories



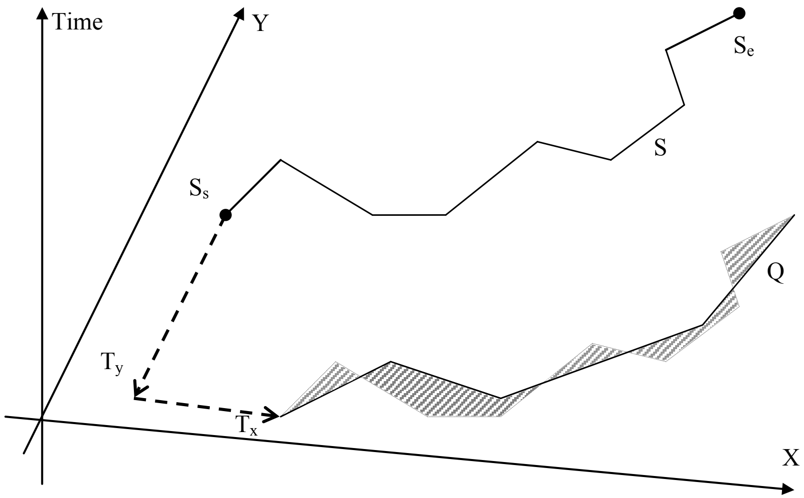
# Trajectory as a 3D polyline (cont.)

- The **uncertain regions** approach
  - Trajectories are transformed into sequences of cells (according to some partitioning of space and time)
- The distance between two uncertain regions could be
  - the minimum Euclidean distance between the regions' MBRs



# From an analyst point of view

- After all, why do we need a palette of different trajectory (dis-)similarity functions?
- Answer: in order to perform quite interesting analysis on MODs
- Examples:
  - "Find groups of trajectories that follow similar routes (i.e., projections of trajectories on 2-dimensional plane) during the same time interval (e.g. co-location and co-existence from 3 pm to 6 pm)" (spatiotemporal similarity)
  - "Find groups of trajectories by taking only their route into consideration (i.e., irrespective of time)" (time-relaxed spatial-only similarity)
  - "Find groups of trajectories that follow a given direction pattern (e.g. first NE and then W)" (derivative-based similarity)

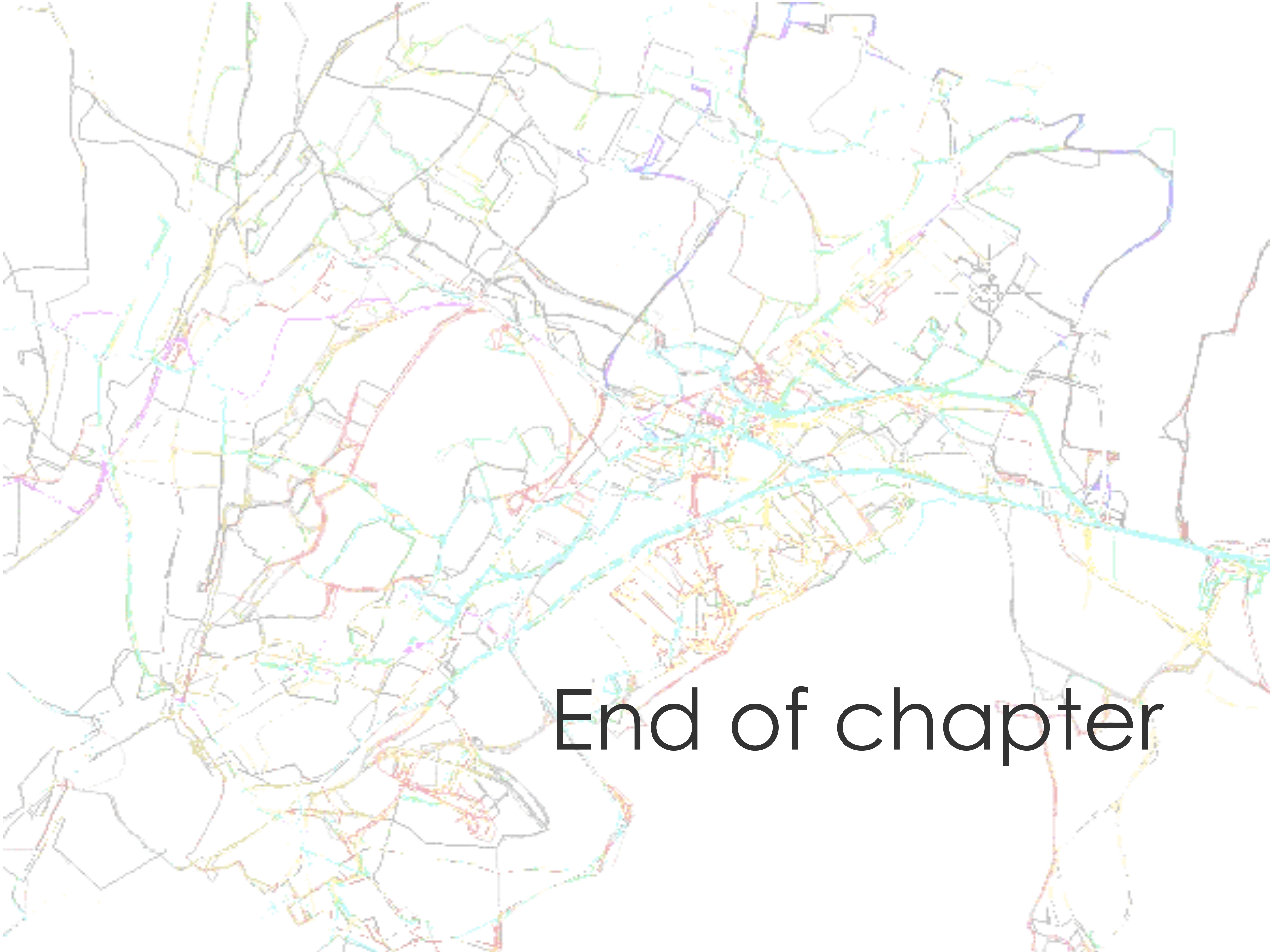


# 6.4.

## Summary

# Summarizing …

- On the way from data management to data exploration
- In this chapter, we presented:
  - How to design a mobility data warehouse and perform multi-dimensional (OLAP) analysis
  - Alternatives for calculating the (dis-)similarity between trajectories of moving objects



End of chapter