

# Introduction: Big Data

**Yannis Kotidis**

*kotidis@aueb.gr*

Professor, Department of Informatics  
Athens University of Economics and Business

# About me



3

- Professor, Dept of Informatics, Athens University of Economics and Business
  - URL: <http://pages.cs.aueb.gr/~kotidis/>
  - Email: [kotidis@aueb.gr](mailto:kotidis@aueb.gr)
  - Office: *A516, Πτέρυγα Αντωνιάδου, Κεντρικό Κτήριο ΟΠΑ*
  
- Interests: Big Data Systems/Algorithms, Data Summaries/Compression, Streams, Graphs, Data Warehousing/Analytics, Data Mining

# Recent Projects/Student Theses

(<http://pages.cs.aueb.gr/~kotidis/index.html>)

4



DeLorean: Compression, Indexing and Analysis Techniques for Time-Series Management



EasyFlinkCEP: Complex Event Processing using user-defined patterns



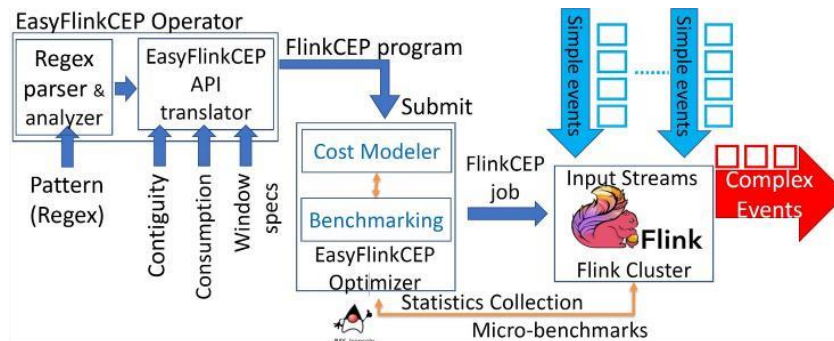
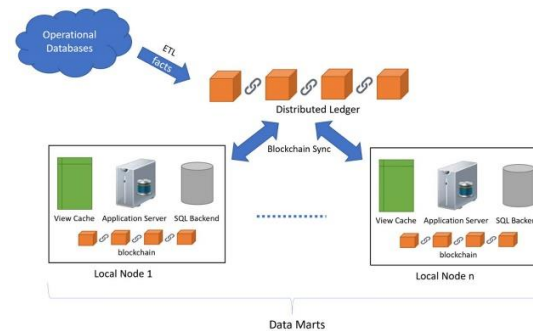
Smart-Views: Decentralized OLAP View Management using Blockchains



RECAST: Real time management of Complex Streams



Graph exploration, partitioning, augmentation



# Course Overview

- Discuss techniques for extracting knowledge from data
  - ▣ Preliminaries: implications of big data, working with different data types (categorical, ordinal, numerical, hierarchical, vectors, sets, strings, time series, graphs), data streams, hashing, graph embeddings
  - ▣ Basket data analysis: association rules discovery and usage, practical considerations
  - ▣ Dimensionality reduction techniques/feature selection (SVD)
  - ▣ Nearest neighbor search, locality sensitive hashing
  - ▣ Link analysis (Page Rank, simRank and other basic graph metrics)
  - ▣ Stream Processing – real time analytics
  - ▣ Clustering

# Key outcomes

- Gain proficiency in data mining techniques, including but not limited to association rules, nearest neighbor search, clustering, and classification.
- Learn to work with & mine different types of data:
  - ▣ High dimensional, graph, infinite (streams), big-data
- Develop practical intuition on applying these techniques to real datasets using contemporary programming frameworks
  - ▣ Examples in python, spark, neo4j

# Course Material

9

- Slides, sample code (python notebooks) & datasets
  - ▣ All material are already available on the class portal!
    - disclaimer: slides may get refreshed before each class
- Portal: for class assignments, email (preferred) for communication
- Suggested reading
  - ▣ *Mining of Massive Datasets*, Jure Leskovec, Anand Rajaraman, Jeff Ullman, <http://www.mmids.org/>

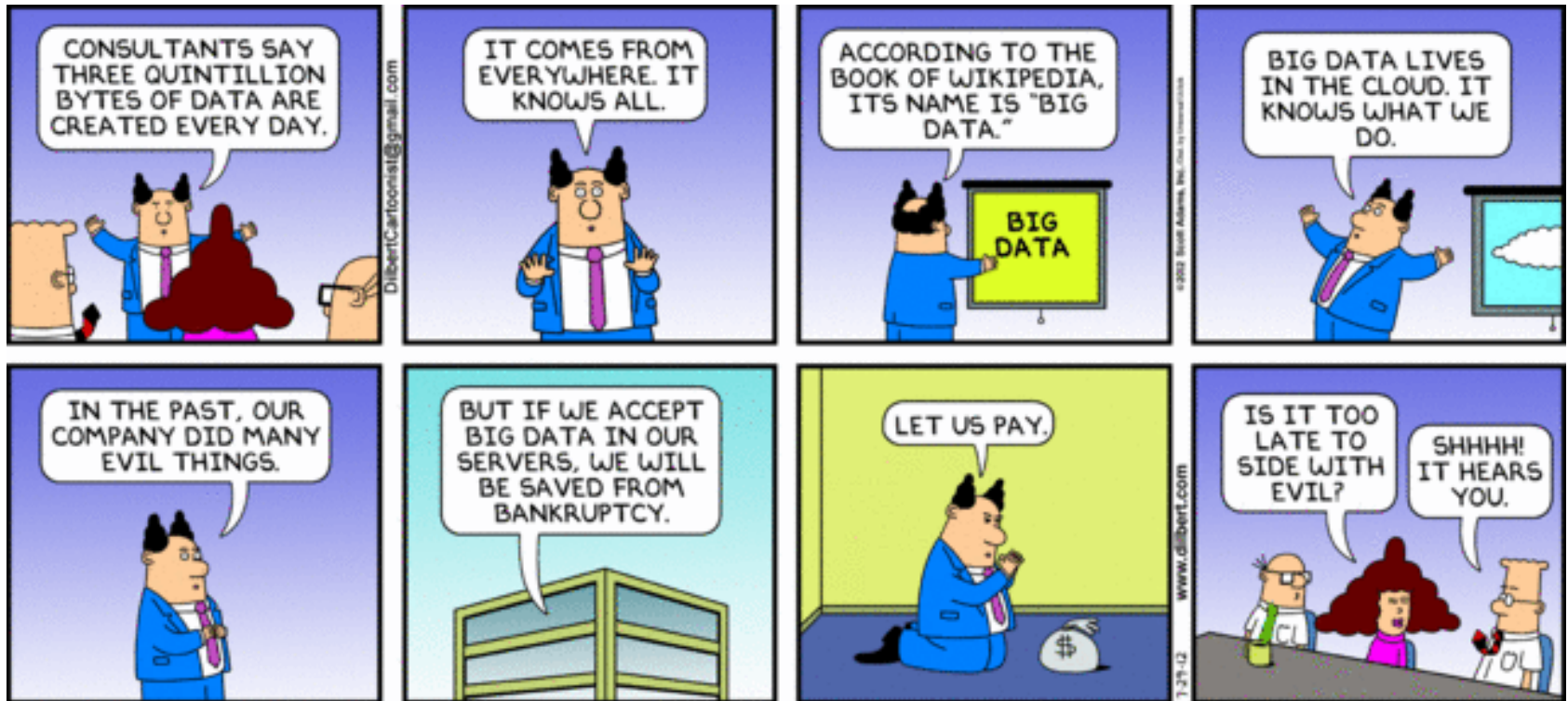
# Grades

10

- 50% assignments – 50% final – TBD
  
- Two hands-on assignments (programming) - TBD
  - ▣ Ioanna Filippidou will be responsible for the assignments
  
- The final examination will assess both theoretical understanding and practical problem-solving skills based on the topics covered in class sessions.

# Big Data is everywhere and knows it all!

21





# Data is the new oil in the 21<sup>st</sup> century

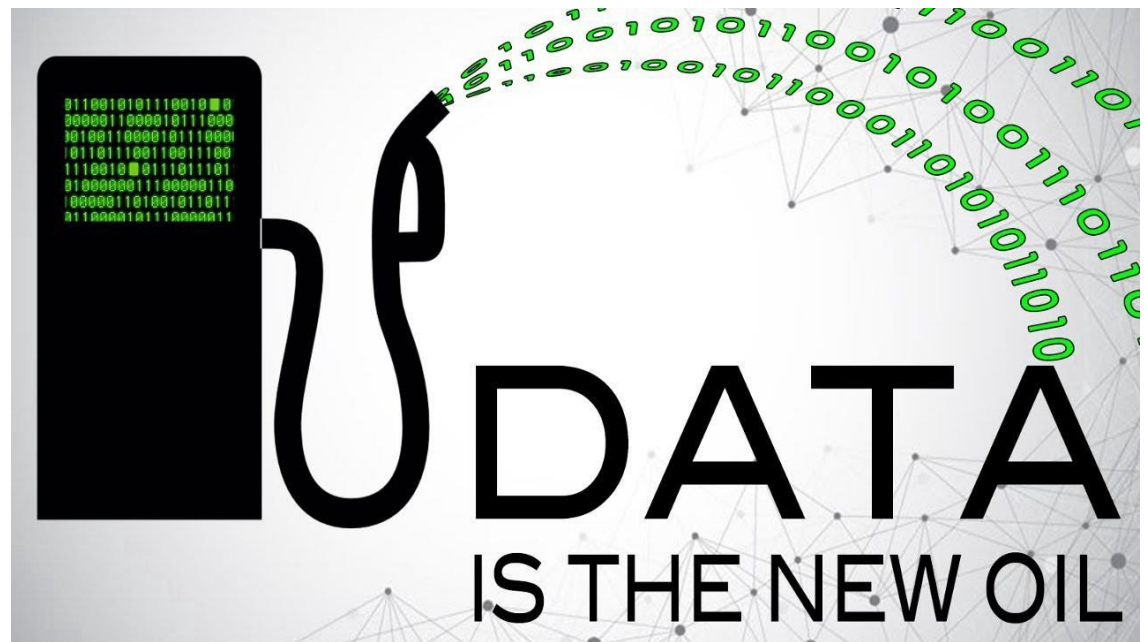
22

- **Industrial revolution:** abundant fossil fuels, and technological advances launched an era of accelerated change that continues to transform human society.
- **Information revolution:** data drives the information economy in much the same way that oil has fuelled the industrial economy

# Data is the new oil in the 21<sup>st</sup> century

23

- Data (such like oil) needs to be
  - Found,
  - Extracted,
  - Cleansed,
  - Refined,
  - Analyzed,
  - Distributed



<https://medium.com/@adeolaadesina/data-is-the-new-oil-2947ed8804f6>

# Data is no oil!

24

- Oil is **finite** - data is infinite and reusable
- Via its use **oil degrades** into a form that is of no use anymore while data is transformed into knowledge
- Common ground: **Data is an asset** and by using that data, it opens huge opportunities for the business

# Data is an asset, not a burden!

25

## Huge advances in our ability to collect data

- Web interactions (e.g. yahoo has 2PB of web data)
- Social Media (tweets, Facebook, etc.)
- Scientific experiments (e.g. NASA EOSDIS archives over 1 PB/year)
- Business Applications (e.g. Bank/Credit Card transactions)

## Gather whatever data you can whenever possible

- Data will have value in the future
- ...even for purposes not envisioned!



# Digital Data Explosion

26

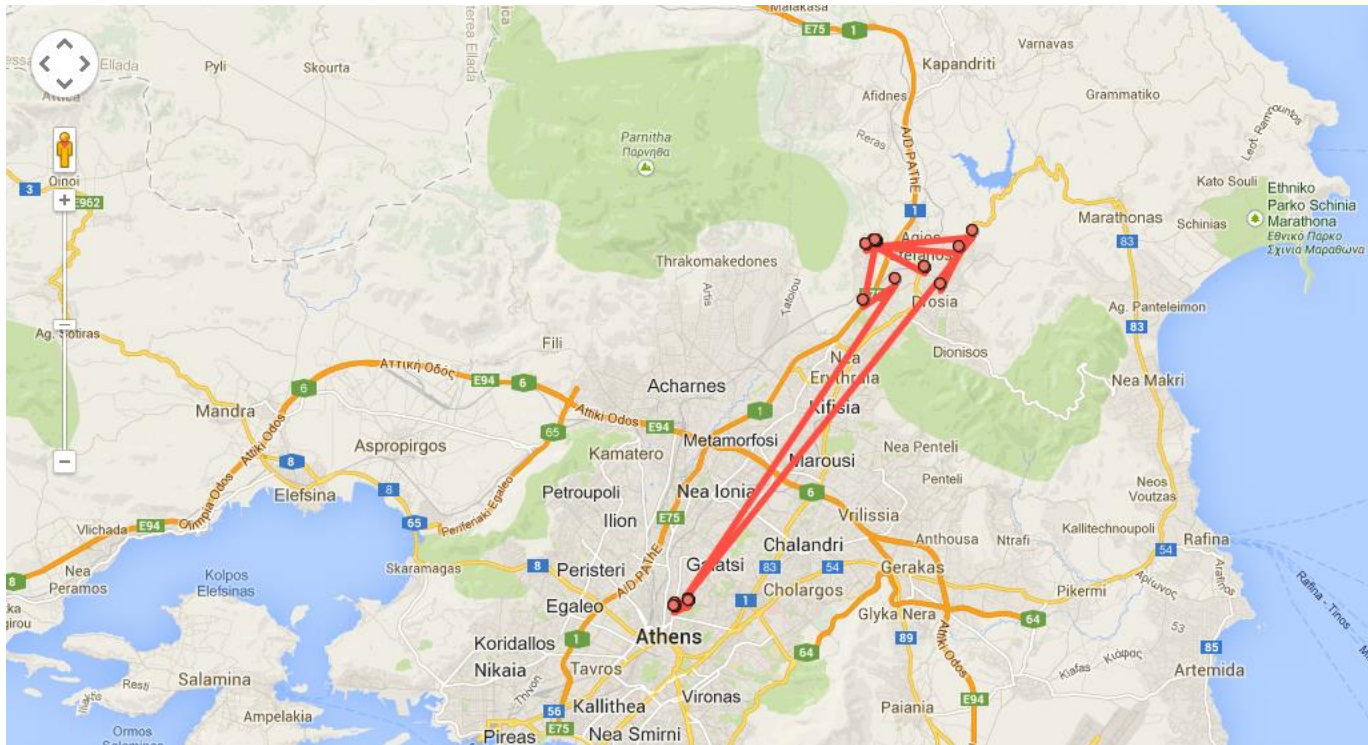
- Everything we see, read, hear, write, and measure can now be in a digital form!!
- Every few days we generate more data than we did from the dawn of civilization until 2000
  - ▣ Multimedia, scientific, **sensor**, etc. data is becoming prevalent

**How to use this data to drive new growth?**

# Location history based on cellular connections



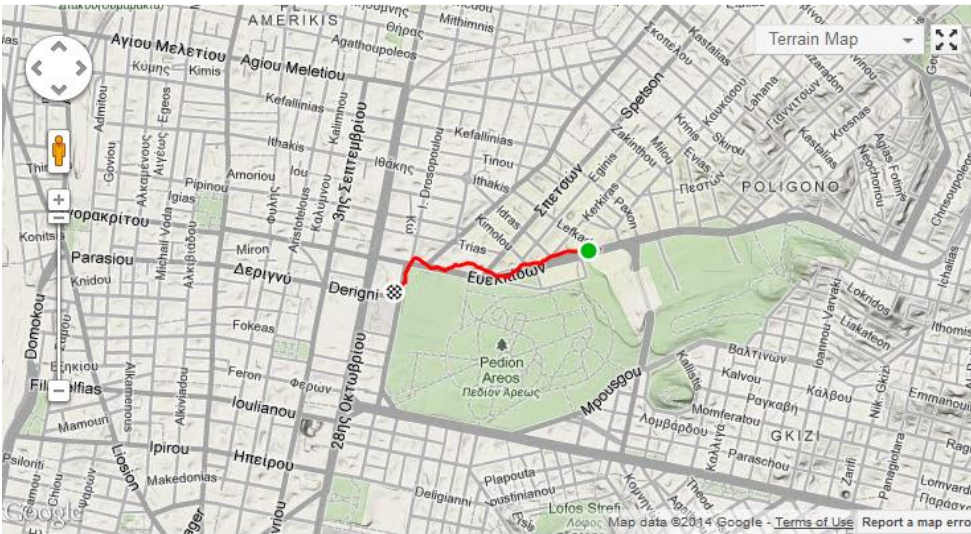
27



# Phone GPS tracking

28

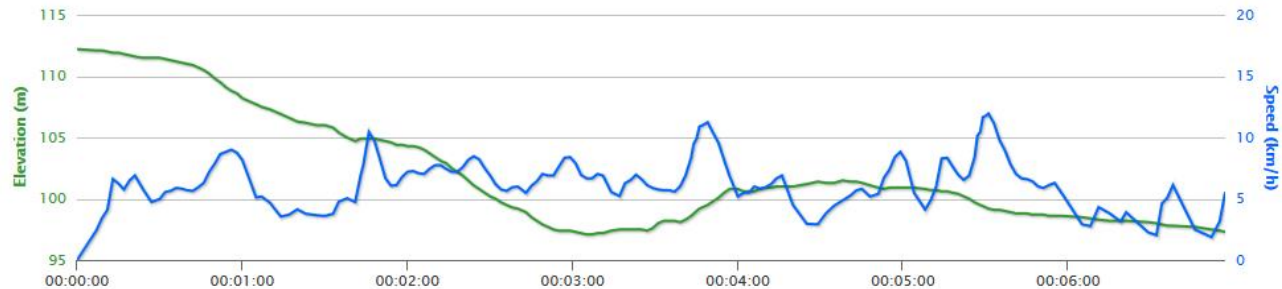
0.7 km 0m 6:57 10:06/km 82  
Distance Elevation Moving Time Avg Pace Calories



Elevation

Performance

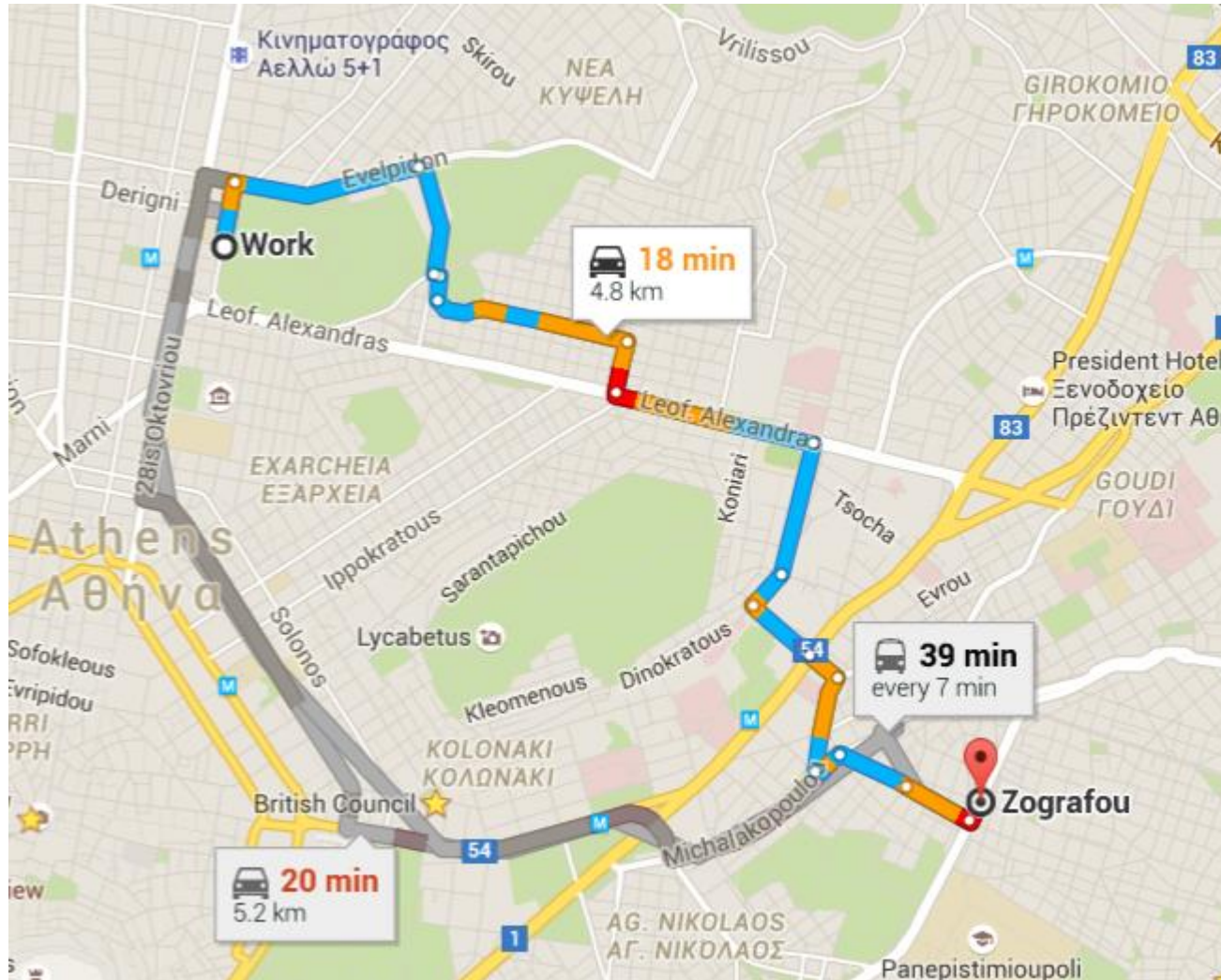
Elevation  Grade-Adjusted Pace  Speed





# Provide useful services: Live traffic data

29



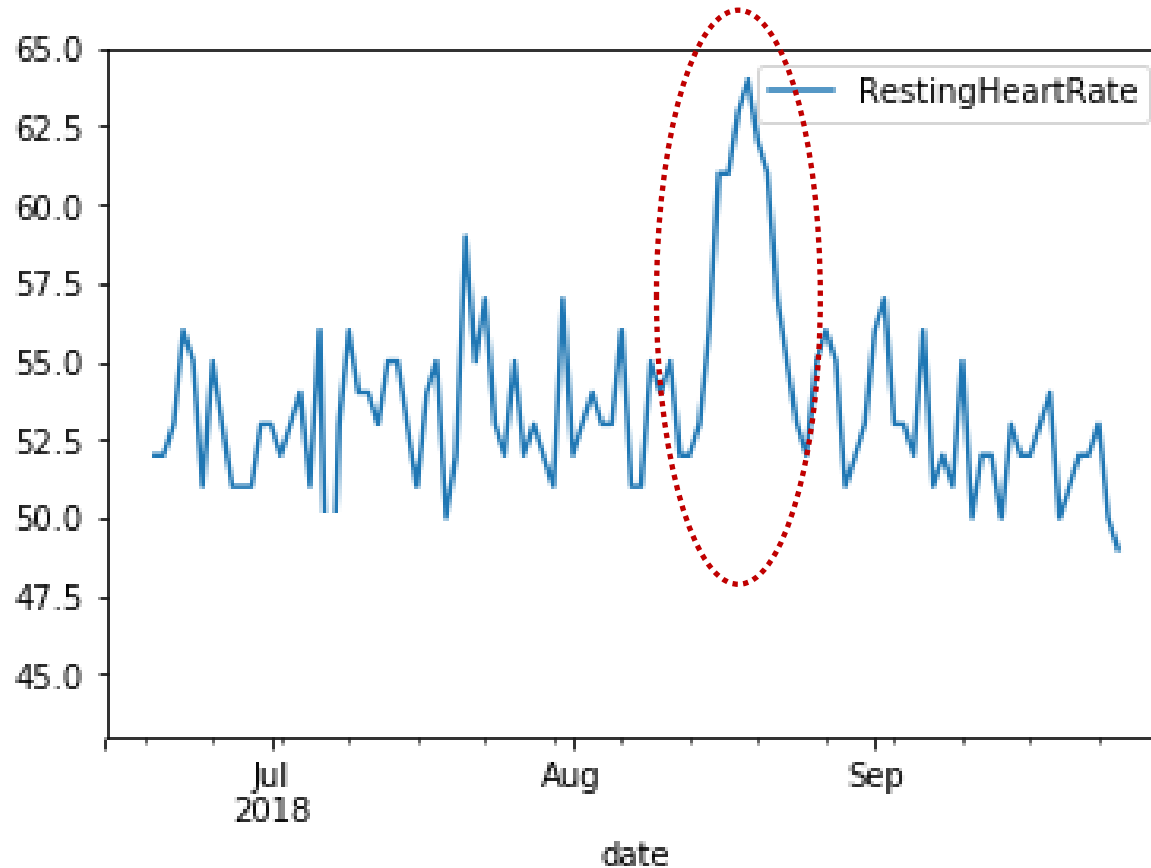


# Another Application: Fitness Analytics



31

How do you explain this peak?



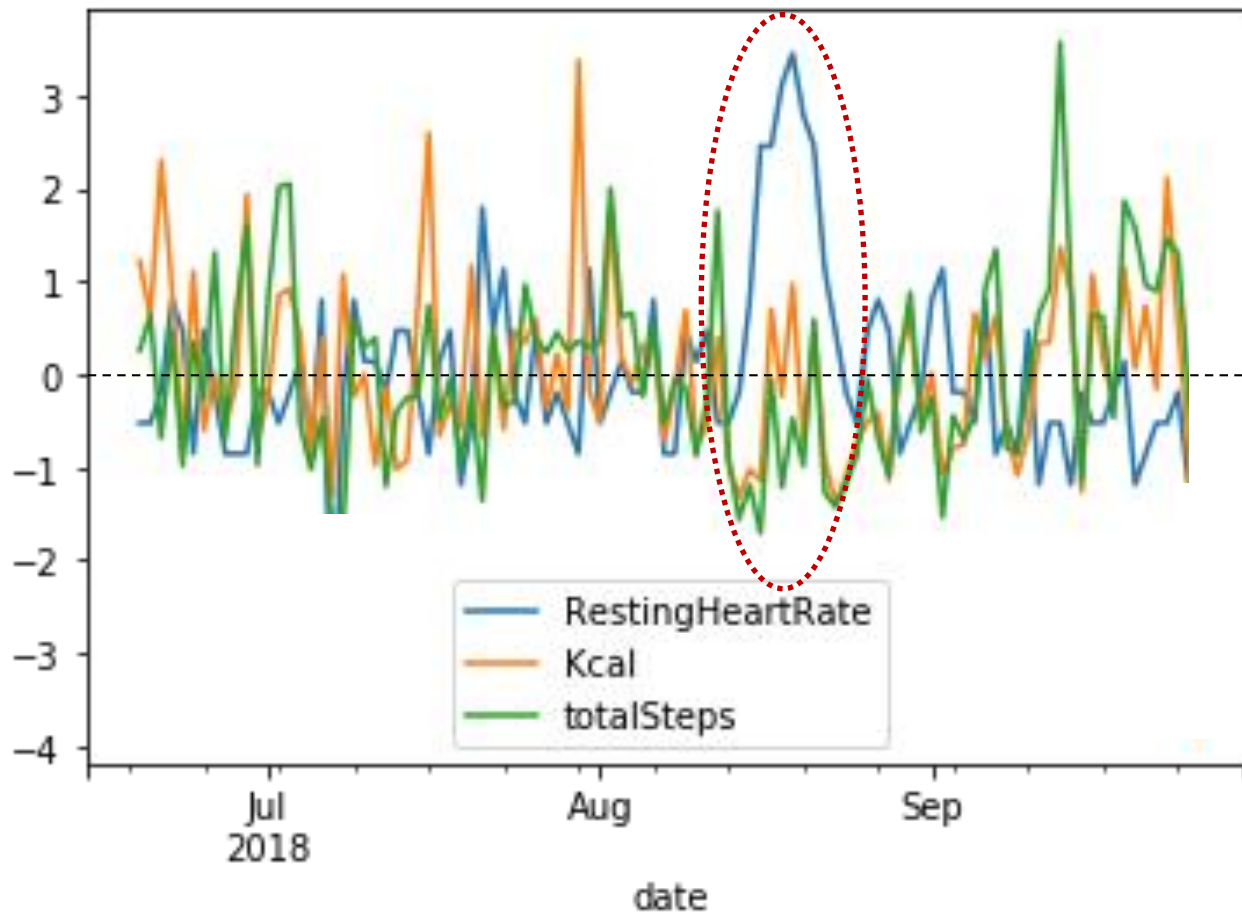
- **Descriptive analytics:** answer queries looking back in time [this example]
- **Predictive analytics:** forecasting [what is my marathon predicted time?]
- **Prescriptive analytics:** planning in order to achieve predicted outcomes [e.g. workout schedule]

# More data → more insights

32

Kcal(down) & totalSteps(down) → RestHR(up)

Exercise(down) → Fitness (down)



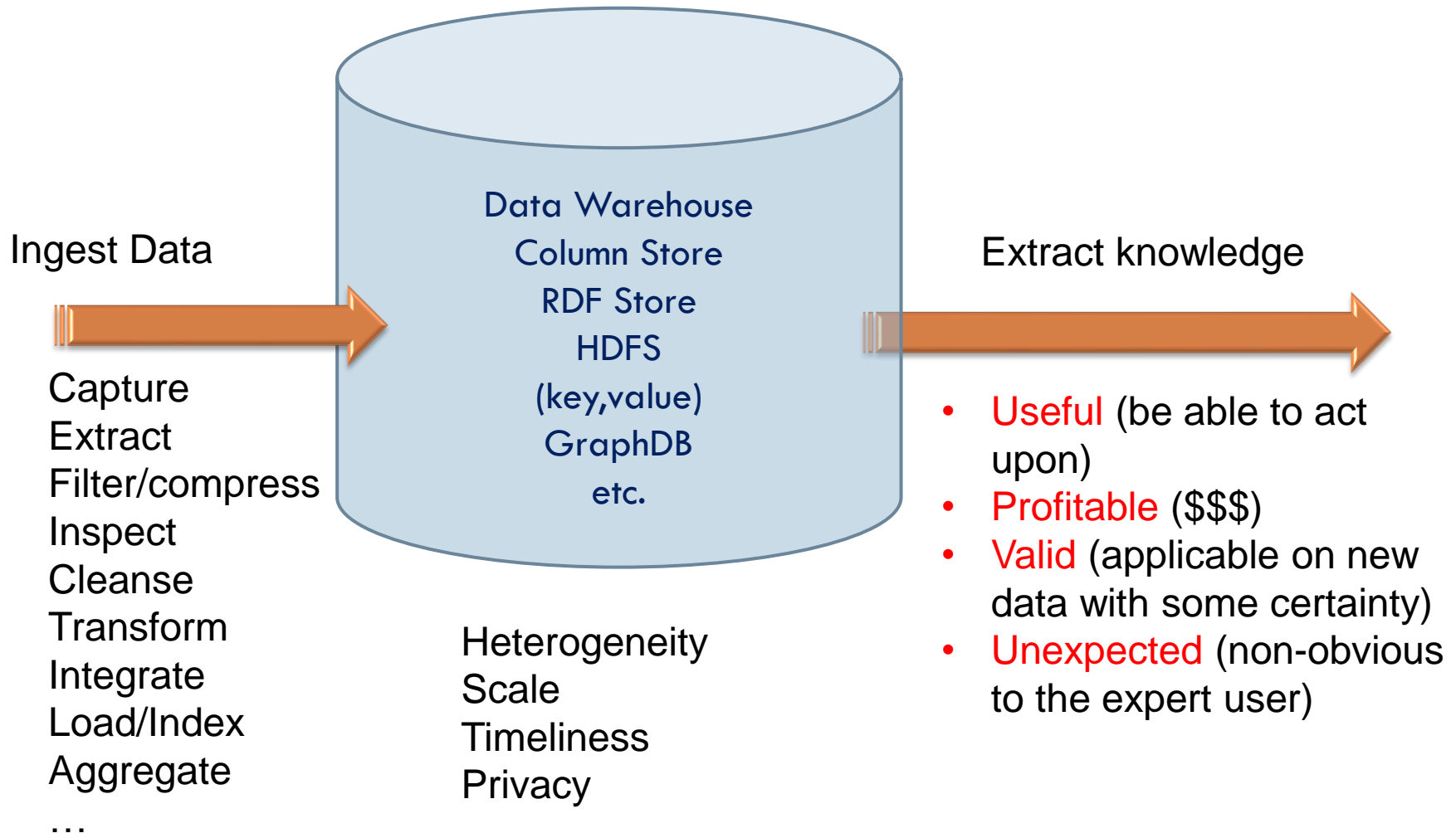
# Data Mining

33

- The process of analyzing data to identify **new information** in the form of **features, patterns** or **relationships**
- Has become a well-established discipline related to Artificial Intelligence, Machine Learning and Statistical Analysis
  - ▣ Led by advances in computer hardware and database technology
    - Data warehousing, Business Intelligence, Cloud Computing, Big Data frameworks

# The big (&very abstract) picture

34

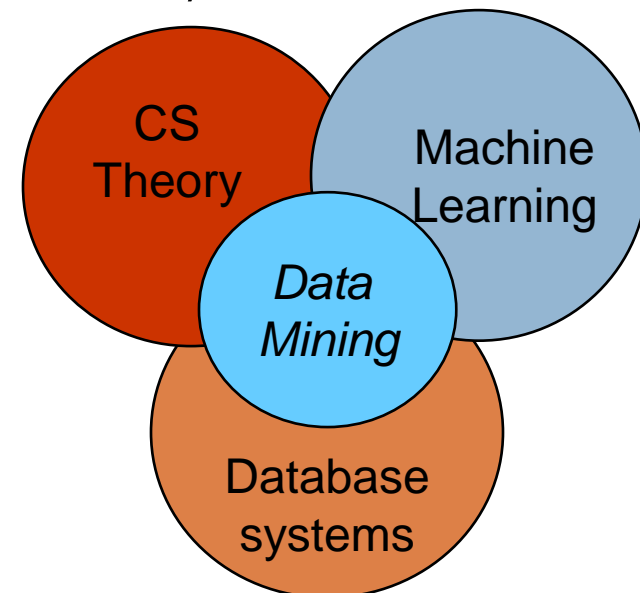


# Data Mining: Cultures

35

## □ Different cultures:

- To a DB person, data mining is an extreme form of **analytic processing** – queries that examine large amounts of data
  - Result is the query answer, focus on the process/algorithm
- To a ML person, data-mining is the **inference of models**
  - Result is the parameters of the model



# Limiting factors

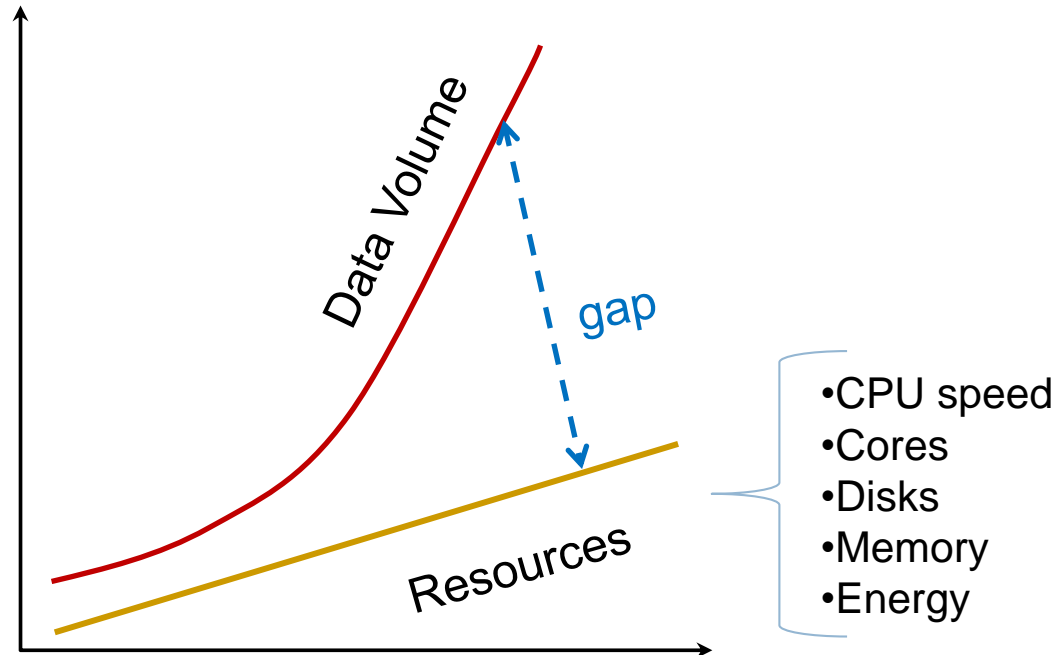
36

- Hardware
  - ▣ Technological/economical limitations
- Software
  - ▣ Complexity of deploying, monitoring and maintaining computing resources
    - Huge advances in this area in the past decades ✓
  - ▣ Algorithmic complexity of certain tasks
    - Theory and practice need to converge

# Mind the (technological) gap!

37

- Data is scaling faster than compute resources, and CPU speeds are static.
  - ▣ Moore's law is not coming to our rescue anymore...

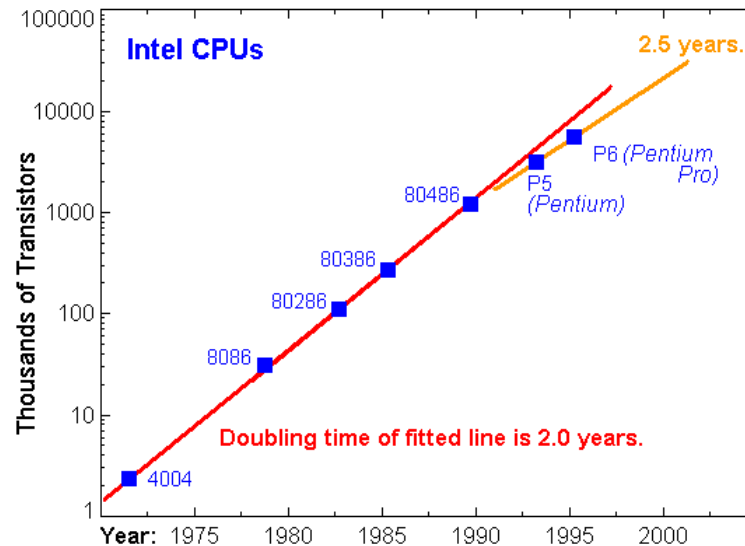


# Gordon Moore's Law

## co-founder of Intel, 1965

38

- **Prediction:** the number of transistors per square inch on integrated circuits had doubled every year since the integrated circuit was invented. Moore predicted that this trend would continue for the foreseeable future.



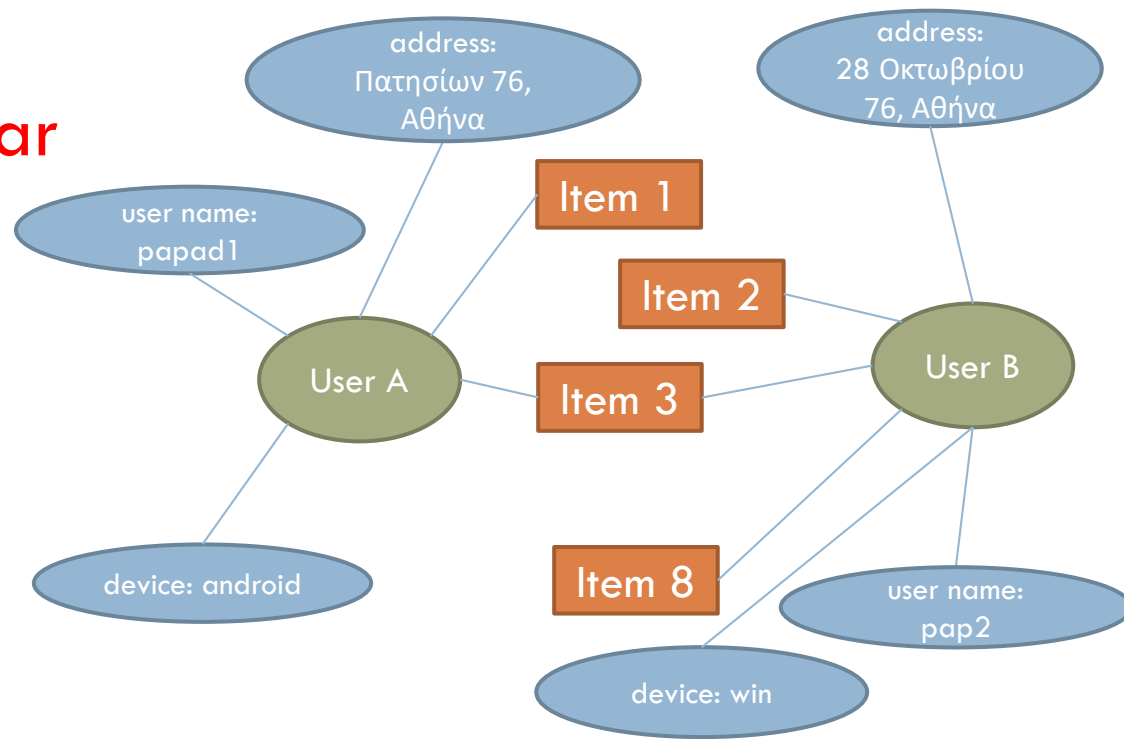
- x2 transistors per square inch / 18 months
- HDDs: < 0.5 euro/GB



# Complexity of a simple (yet very important) computation

39

- Given a collection of customer data, **compute most similar pairs**
- Applications
  - ▣ Identity Resolution
  - ▣ Fraud detection
  - ▣ Customer Segmentation
  - ▣ Collaborative filtering



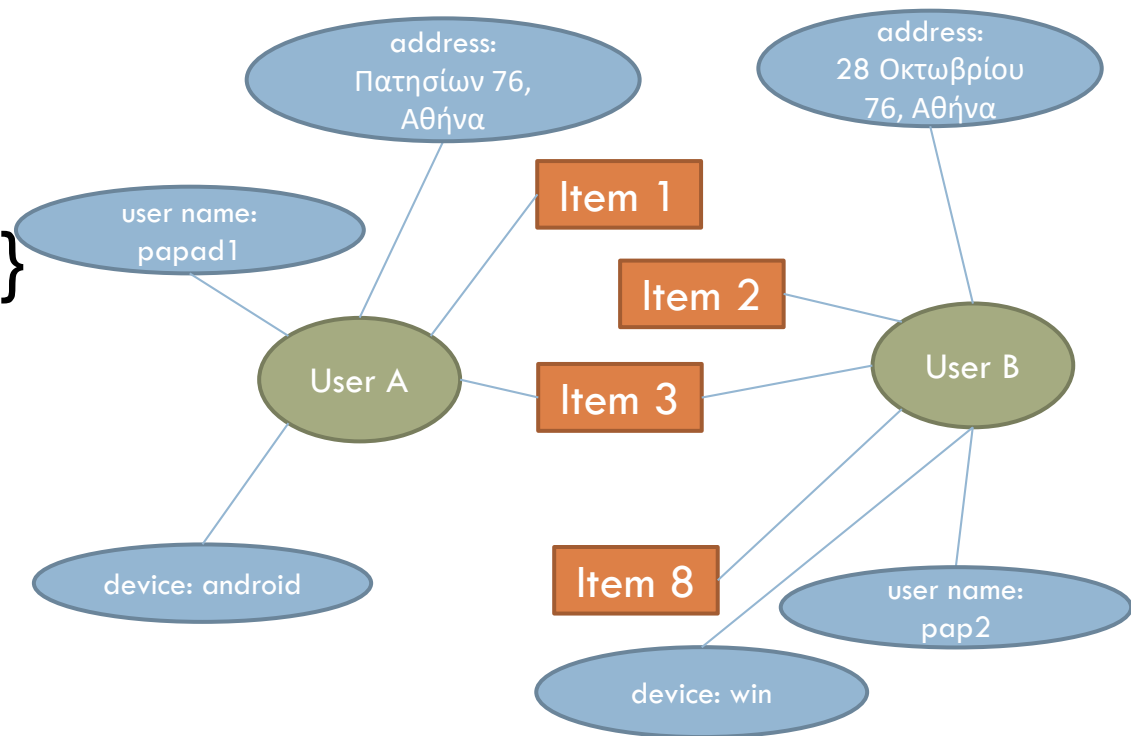
userA: papad1,android,{item1,item3}, "Πατησίων 76, Αθήνα"  
userB: pap2,win,{item2,item3,item8}, "28 Οκτωβρίου 76, Αθήνα"

...

# How to compare customers based on these data?

40

- “papad1” vs “pap2”
- android vs win
- {item1,item3} vs {item2,item3,item8}
- “Πατησίων 76, Αθήνα” vs “28 Οκτωβρίου 76, Αθήνα”



userA: papad1,android,{item1,item3}, “Πατησίων 76, Αθήνα”  
userB: pap2,win,{item2,item3,item8}, “28 Οκτωβρίου 76, Αθήνα”

...

# How to compare customers based on these data?

41

- “papad1” vs “pap22”
- android vs win

address 1

address 2

**We will address these problems in the next lecture!**

For now assume we have a function `sim(user1,user2)` that estimates the similarity between two users based on their data attribute values

Οκτωβρίου 76,  
Αθήνα”

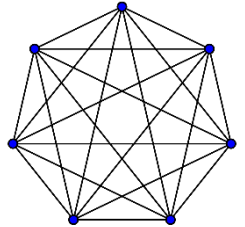
deviceid2

userA: papad1,android,{item1,item3}, “Πατησίων 76, Αθήνα”  
userB: pap2,win,{item2,item3,itemN}, “28 Οκτωβρίου 76, Αθήνα”  
...

# Let us focus on the **running time**

42

- Run brute-force **all-pairs** similarity computation in your favorite programming language
- For  $n$  customers we need  $\frac{n(n-1)}{2}$  comparisons
  - ▣ CS theory: task complexity is  $O(n^2)$
  - ▣ Assume task completes in 5 minutes (yeah!)
- In a year from now, dataset gets 100 times larger
- How long will it take for the same task to compute?



$$\text{Ans: } \sim 100^2 * 5\text{min} = \frac{50000}{60 * 24} \text{ days} = 34,7 \text{ days} > 1 \text{ month!}$$

# Scale-up versus Scale-out

43

- They refer to different strategies for expanding your computing resources to handle the growth of work
- **Scale-up** (vertical scaling): adding more/better resources to an existing system to reach a desired state of performance
  - ▣ More powerful CPU/GPU
  - ▣ More RAM
  - ▣ Larger/faster Disks
  - ▣ More network interfaces
- In cloud computing environments this translates to moving up to larger, more powerful **instances**



# AWS Compute Optimized instances

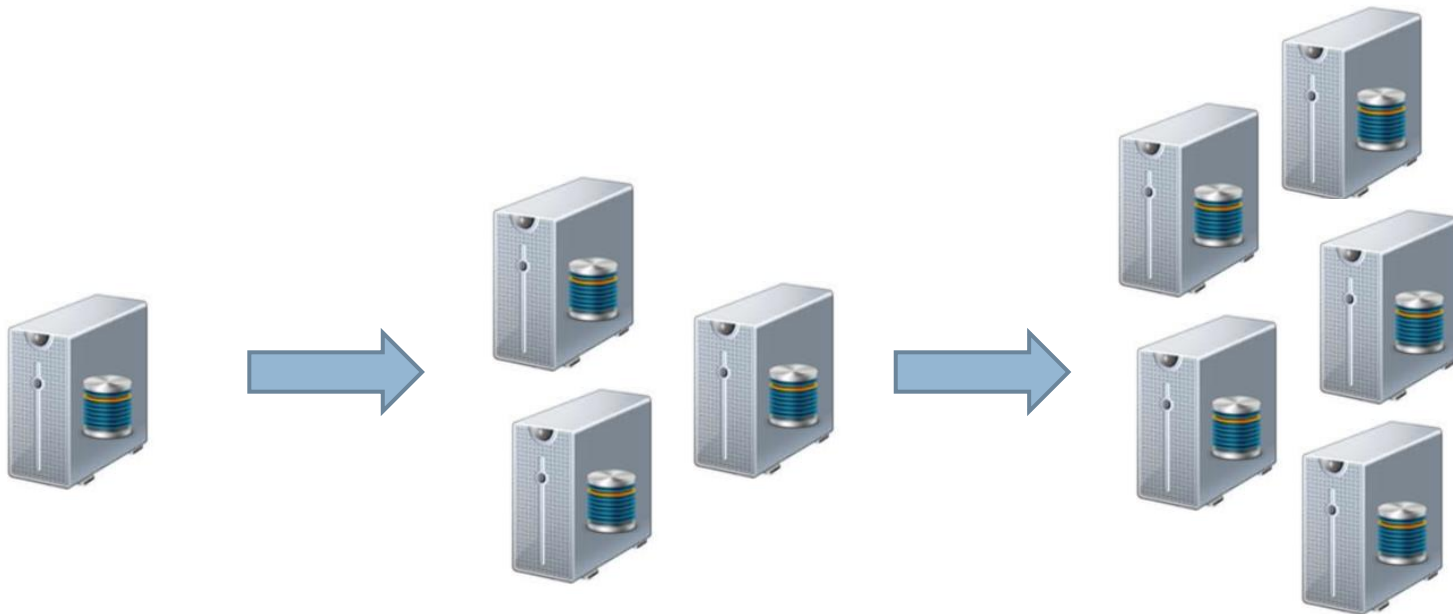
44

Model	vCPU	Memory (GiB)	Instance Storage (GiB)	Network Bandwidth (Gbps)
c5.large	2	4	EBS-Only	Up to 10
c5.xlarge	4	8	EBS-Only	Up to 10
c5.2xlarge	8	16	EBS-Only	Up to 10
c5.4xlarge	16	32	EBS-Only	Up to 10
c5.9xlarge	36	72	EBS-Only	10
c5.12xlarge	48	96	EBS-Only	12
c5.18xlarge	72	144	EBS-Only	25
c5.24xlarge	96	192	EBS-Only	25
c5.metal	96	192	EBS-Only	25
c5d.large	2	4	1 x 50 NVMe SSD	Up to 10
c5d.xlarge	4	8	1 x 100 NVMe SSD	Up to 10
c5d.2xlarge	8	16	1 x 200 NVMe SSD	Up to 10
c5d.4xlarge	16	32	1 x 400 NVMe SSD	Up to 10
c5d.9xlarge	36	72	1 x 900 NVMe SSD	10
c5d.12xlarge	48	96	2 x 900 NVMe SSD	12
c5d.18xlarge	72	144	2 x 900 NVMe SSD	25
c5d.24xlarge	96	192	4 x 900 NVMe SSD	25
c5d.metal	96	192	4 x 900 NVMe SSD	25

# Horizontal scaling

45

- **Scale-out:** increase capacity by adding more instances
  - May reduce costs by employing less sophisticated resources to accommodate variable workloads



# Big Data – The 4 Vs

46





# Big Data: Volume

47

- Organizations process terabytes or even petabytes of raw data.
  - Turn **12 terabytes** of Tweets created each day into improved product sentiment analysis.
  - Typical bottleneck: move data across the memory hierarchy
    - One HDD reads 12TB @ 145 MB/sec in ~24 hours
    - One NVMe SSD reads 12TB @ 2GB/sec in ~1,7 hours
    - Common tasks require multiple passes over the dataset
    - Need to also consider CPU processing time...

# Big Data: **Velocity**

50

- Ability to react quickly to **streaming** data
  - ▣ **Real-Time Enterprise**: reduce the gap between when data is recorded in an organization and when it is available for information processing and decision-making.
  - ▣ For **time-sensitive processes** such as catching fraud, big data must be used as it **streams** into your enterprise in order to maximize its value.
  - ▣ Scan 5 million trade events created each day to identify potential fraud.

# Big Data: Variety

51

- Big data is any type of data - **structured** and **unstructured** data (or anything in between) such as text, sensor data, time series, audio, video, click streams, log files, graph structures and more.
  - ▣ Does the relational model meet your data needs?
    - Key-value pairs?, column-based, documents-oriented? graphs?
- New insights are found when analyzing these data types together.
  - ▣ But they often live in different (data) eco-systems...

# Big Data: **Veracity**

52

- How to deal with **uncertain** or **imprecise** data
  - ▣ In traditional applications there was always the assumption that the data is **certain**, **clean**, and **precise**
- Need to act based on information collected from disparate sources or the social web
  - ▣ How much **faith** can we put in social media data like Tweets, Facebook posts?
  - ▣ Can you explain your ML models?
  - ▣ How can we act upon information/processes if we don't completely trust or understand them?
  - ▣ Can you modify your algorithms to provide strong statistical guarantees over imprecise data?

# Implications of 4Vs in data analysis

53

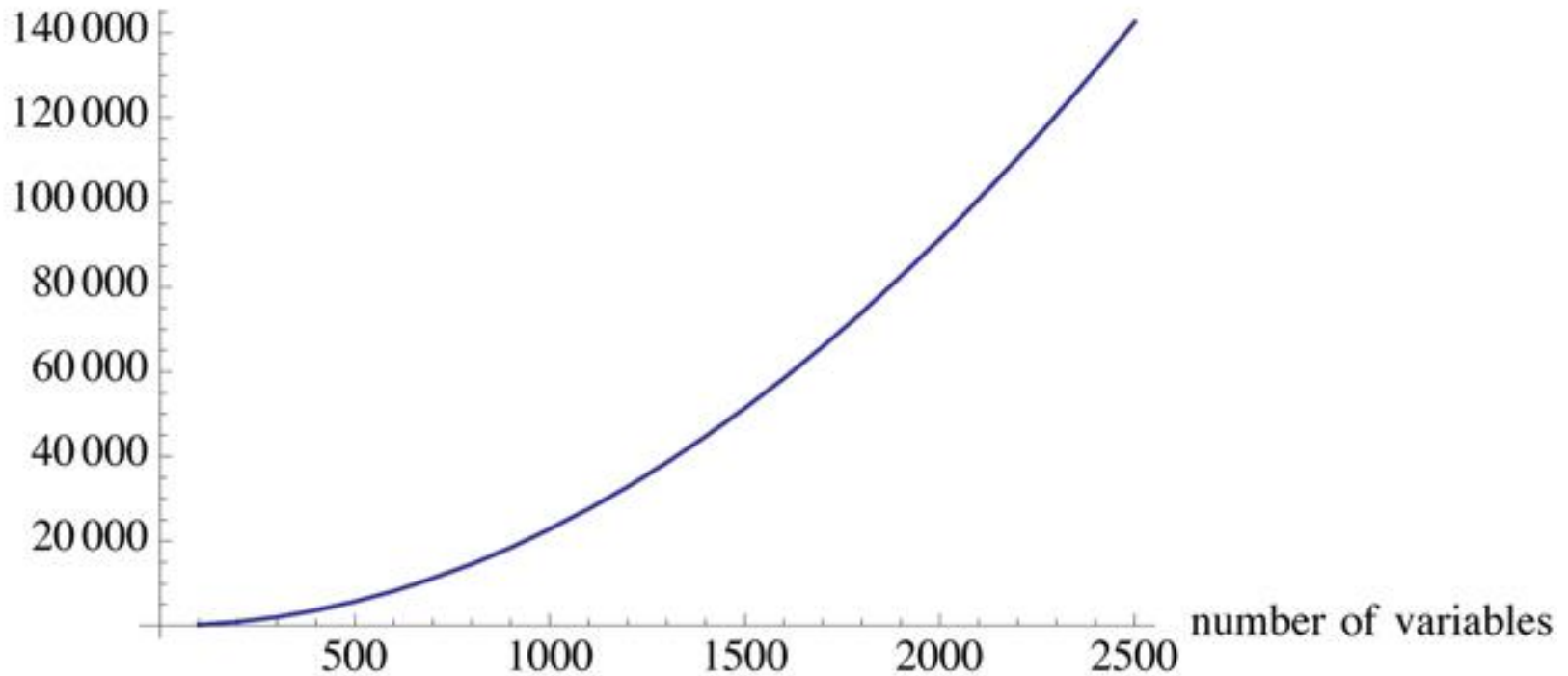
- Big data → big noise → big errors?
- Need faster/scalable algorithms
  - ▣ Partitioning/Parallelization helps sometimes but best you can hope for is linear speed up
    - Some algorithms are hard to parallelize
  - ▣ Theoretical bounds reached
    - Trade accuracy for efficiency
  - ▣ Provision for data that is always in flux

# Beware the Big Errors of 'Big Data'

(Nassim N. Taleb)

54

Spurious Correlations



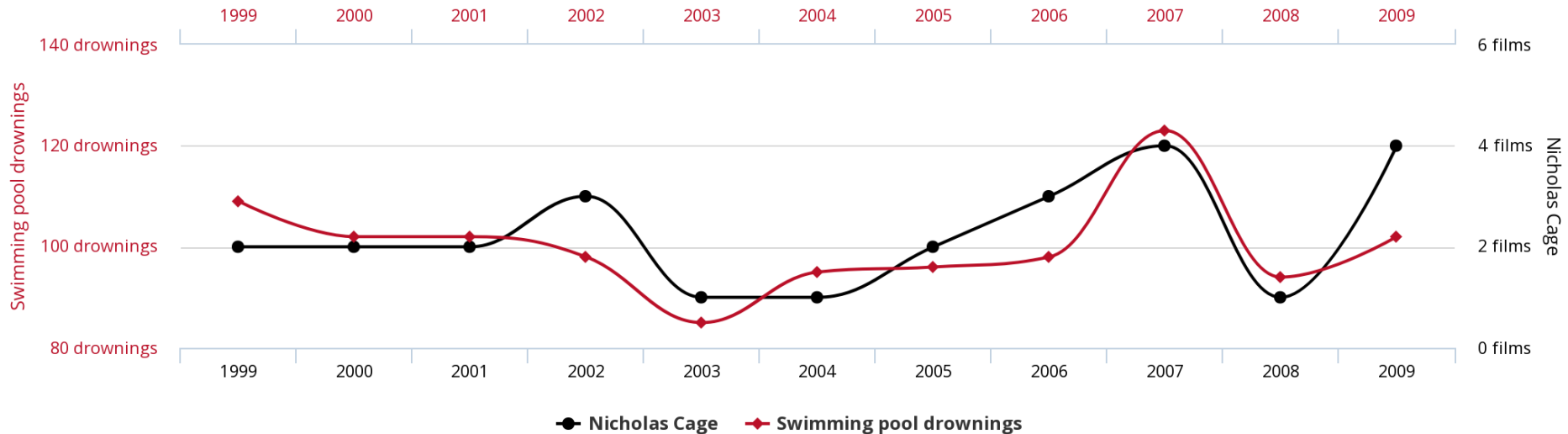
# Spurious Correlations

(<http://tylervigen.com/>)

55



**Number of people who drowned by falling into a pool**  
correlates with  
**Films Nicolas Cage appeared in**



tylervigen.com

Correlation: 66.7%

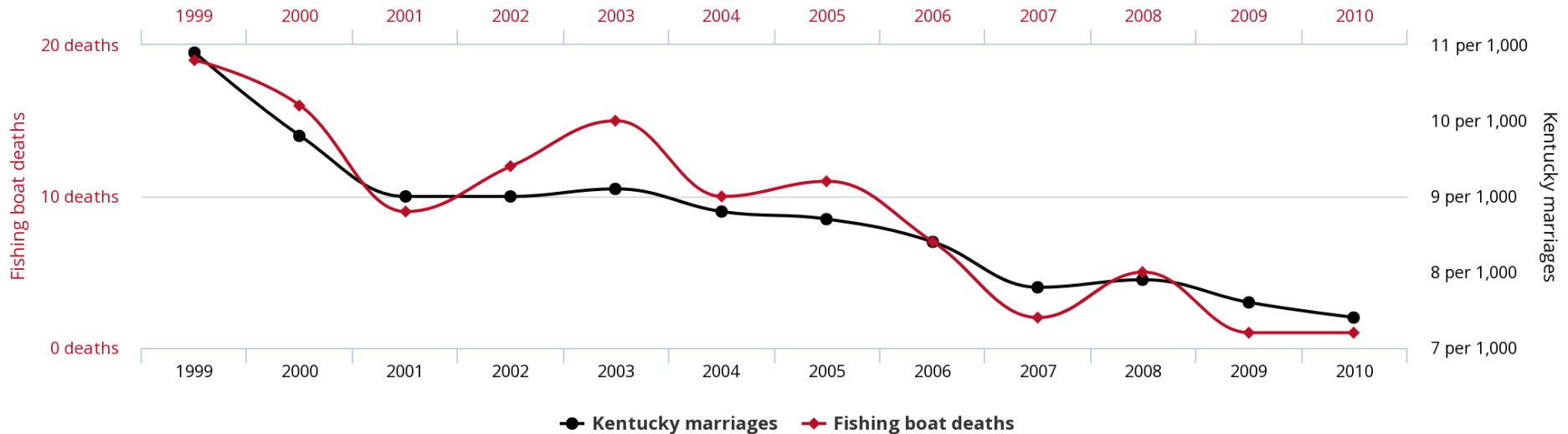
# Spurious Correlations

(<http://tylervigen.com/>)

56



**People who drowned after falling out of a fishing boat**  
correlates with  
**Marriage rate in Kentucky**



tylervigen.com

Correlation: 95.2%



# Rhine Paradox\* – (1)

57

- Joseph Rhine was a parapsychologist in the 1950's who hypothesized that some people had Extra-Sensory Perception (ESP).
- He devised an experiment where subjects were asked to guess 10 hidden cards – red or blue.
- He discovered that almost 1 in 1000 had ESP – they were able to get all 10 right!

# Rhine Paradox – (2)

58

- He told these people they had ESP and called them in for another test of the same type.
- Alas, he discovered that almost all of them had lost their ESP.
- What did he conclude?
  - ▣ Answer on next slide.

# Rhine Paradox – (3)

59

- He concluded that you shouldn't tell people they have ESP; it causes them to lose it.

# Meaningfulness of Answers

60

- A big data-mining risk is that you will “discover” patterns that are meaningless.
- When looking for a property make sure that the property does not allow so many possibilities that random data will surely produce facts “of interest”

# Example Case\*: detect “evil-doers”

62

- There **are 1 billion** people out there who might be evil-doers
- Out of those, about **100** are indeed evil-doers
  - ▣ This is the number of “events” (people) you expect to find in your investigation
- **Make a hypothesis**: a pair of people should be under investigation if they visit on two different days the same hotel
  - ▣ Maybe different hotel on each day

# Simplifying Assumptions

63

- Look for people who, on two different days, were both at the same hotels.
  - 1 billion people who might be evil-doers
  - A person goes to a hotel one day in 100
  - Look at booking records of 100K hotels
  - Examine hotel records for 1000 days to find evil-doers

# Assume Random Behavior

64

- Probability **any two** people both decide to visit a hotel on any given day is  $0.01 * 0.01 = 0.0001$ 
  - ▣ There are  $10^5$  hotels to choose from
    - Assume hotels are visited with same probability (not realistic)
  - ▣ Thus, probability that they visit the same hotel is  $p = 10^{-4} * 10^{-5} = 10^{-9}$
- Probability that they visit the same hotel on two different days is  $p^2 = 10^{-18}$  (hotels may differ on the two days)

# What we have found

65

- Probability of observing an **event** on **random data** is very **small**: **0.00000000000000000001**
- Thus, if we see one such pair of people it should be investigated, right?



# The effects of big data in calculations

66

- Event = two people were at the same hotel on two different days
- There are  $n=1$  billion people resulting in


$$\binom{n}{2} \approx \frac{n^2}{2} = 5 * 10^{17} \quad \text{pairs of people}$$

- Similarly, there are  $\binom{1000}{2} \approx \frac{1000^2}{2} = 5 * 10^5$  pairs of days to look for

# Expected number of events

67

$$\square (5 \cdot 10^{17}) * (5 \cdot 10^5) * 10^{-18} = 250000 \text{ events}$$

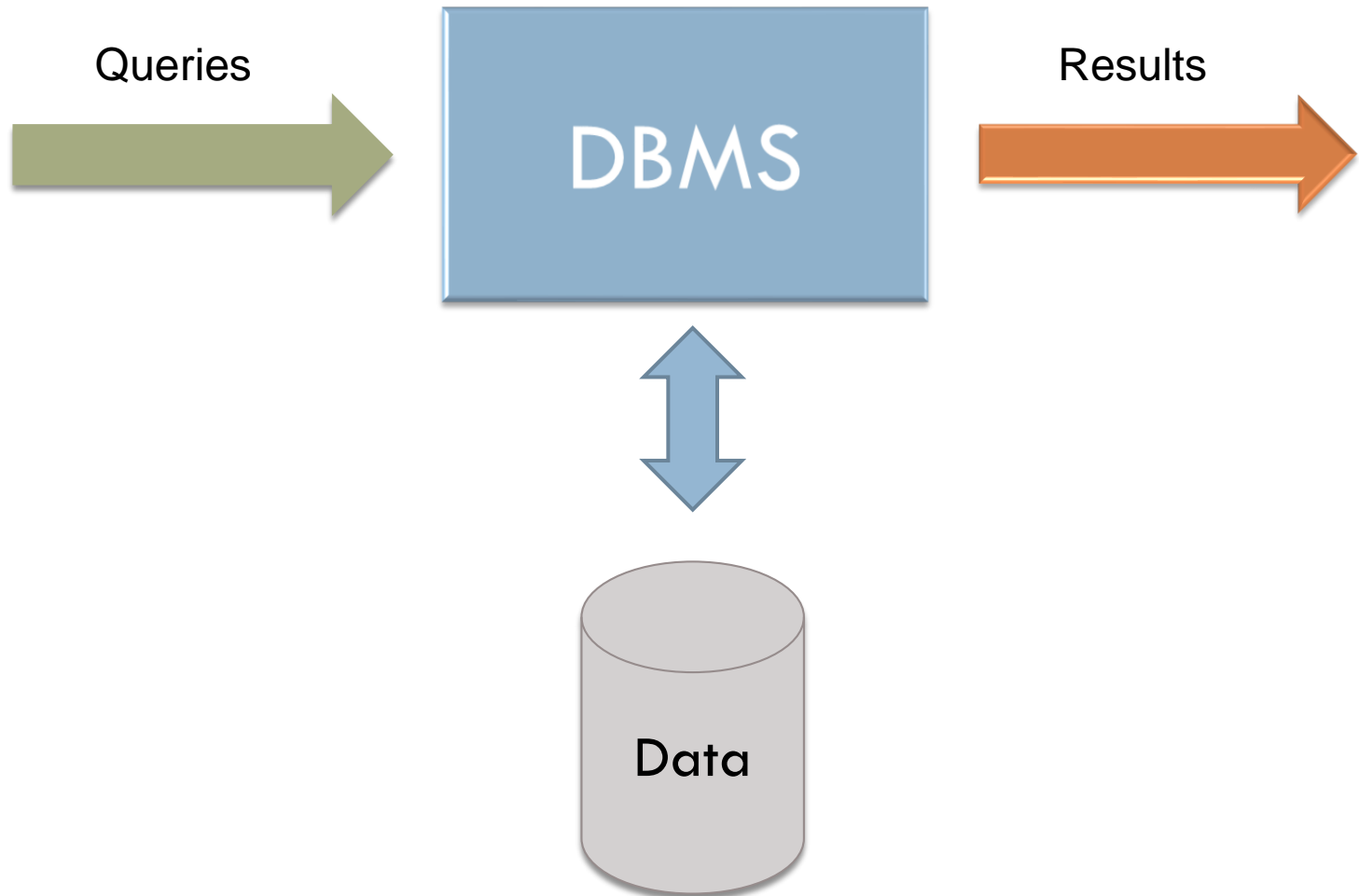


Pairs of people      Pairs of days      Probability of an event

- Each event is a pair of people (who have visited the same hotels on two occasions)
- Thus, there are 250K pairs of people to investigate
  - ▣ But there are only 100 real evil-doers
  - ▣ Is it feasible? Does it justify the intrusion on people's lives?

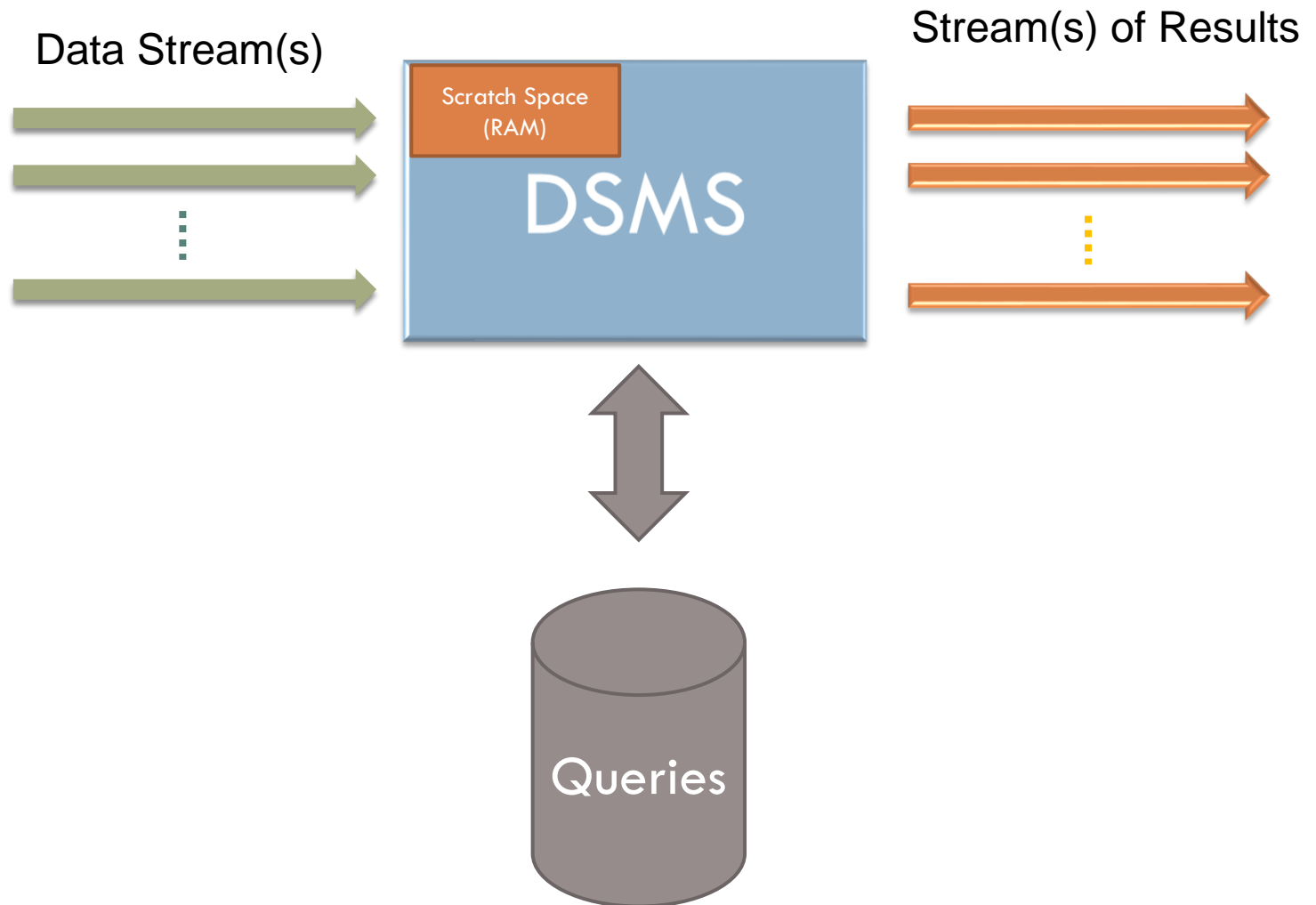
# Traditional Data Processing

69



# Stream (real-time) Processing

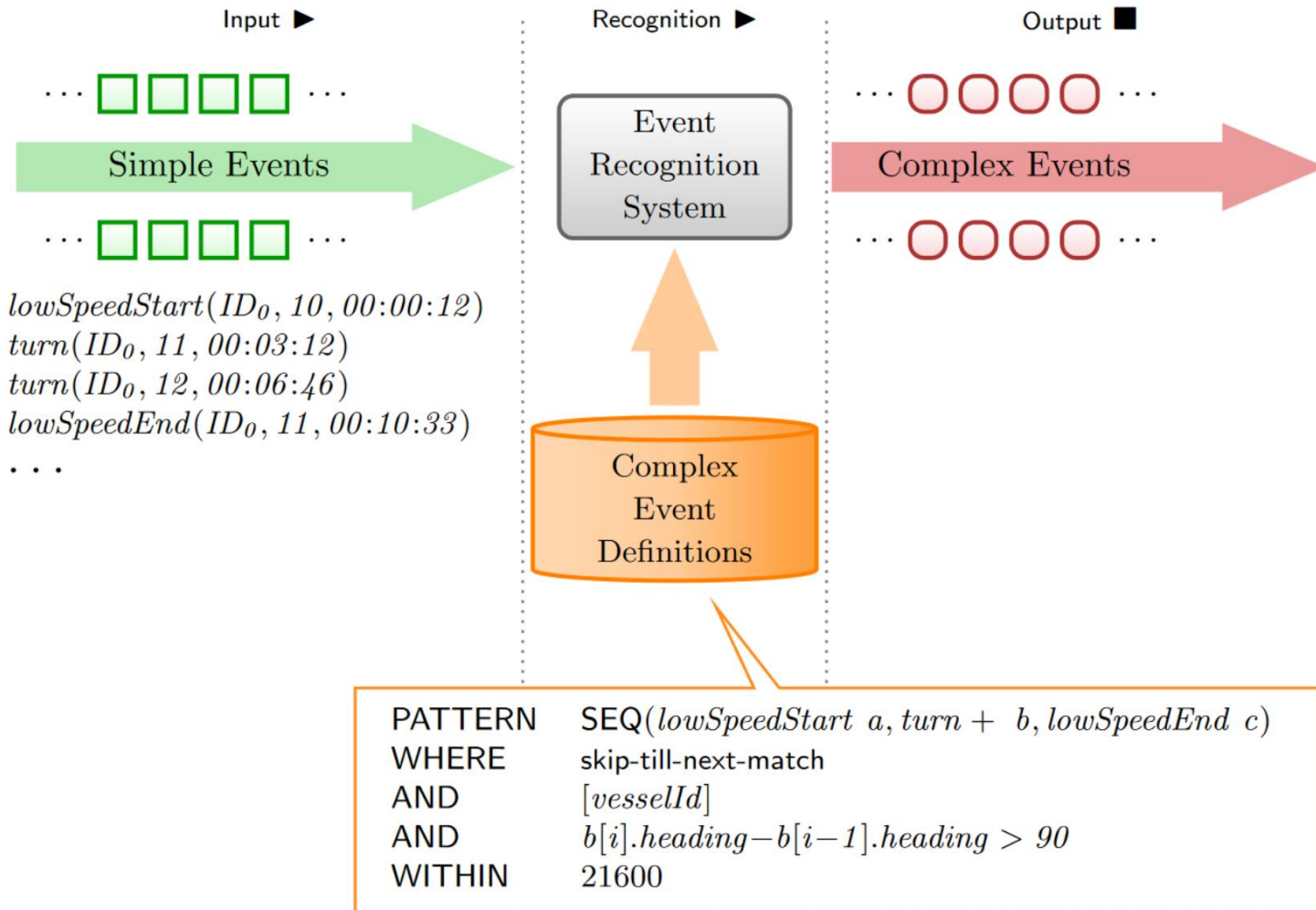
70



# Related topic: Complex Event Processing

(example from: [http://cer.iit.demokritos.gr/publications/papers/2020/VLDB-D-19-00003\\_CRVersion.pdf](http://cer.iit.demokritos.gr/publications/papers/2020/VLDB-D-19-00003_CRVersion.pdf))

71



# EasyFlinkCEP: Big Event Data Analytics for Everyone

- **Complex Event Processing (CEP) in FlinkCEP:**
  - † Distributed processing over computer clusters
  - † Language of high expressive power
  - × Low level language (Scala/Java)
  - × Cluster admin decisions needed

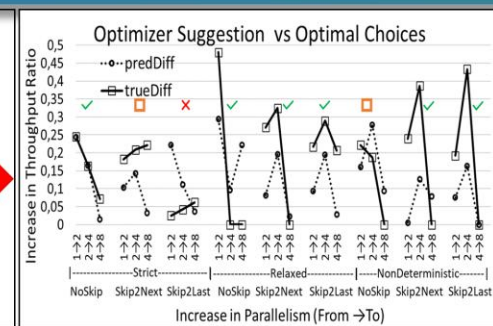
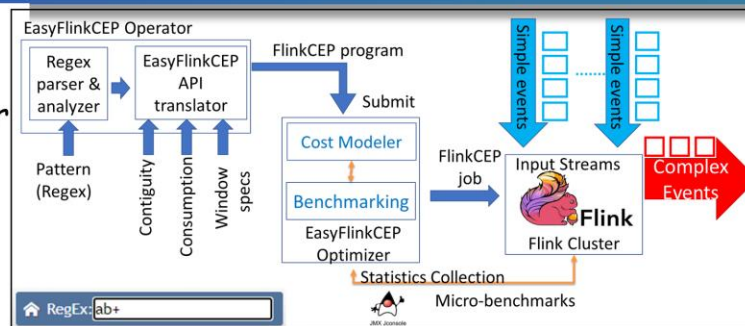
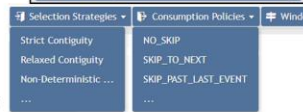
- **EasyFlinkCEP Prototype**

- **EasyFlinkCEP Operator**

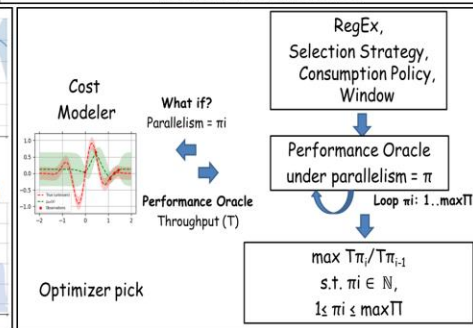
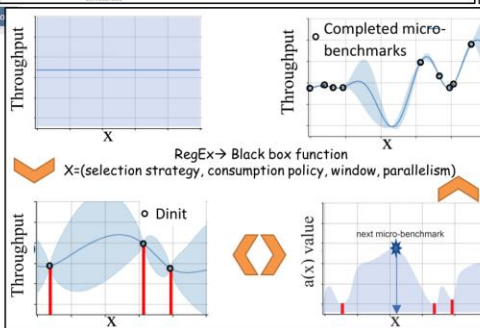
- No coding, uses Regular Expressions
- graphical definition of CEP parameters

- **EasyFlinkCEP Optimizer**

- automates FlinkCEP job configurations (parallelism)

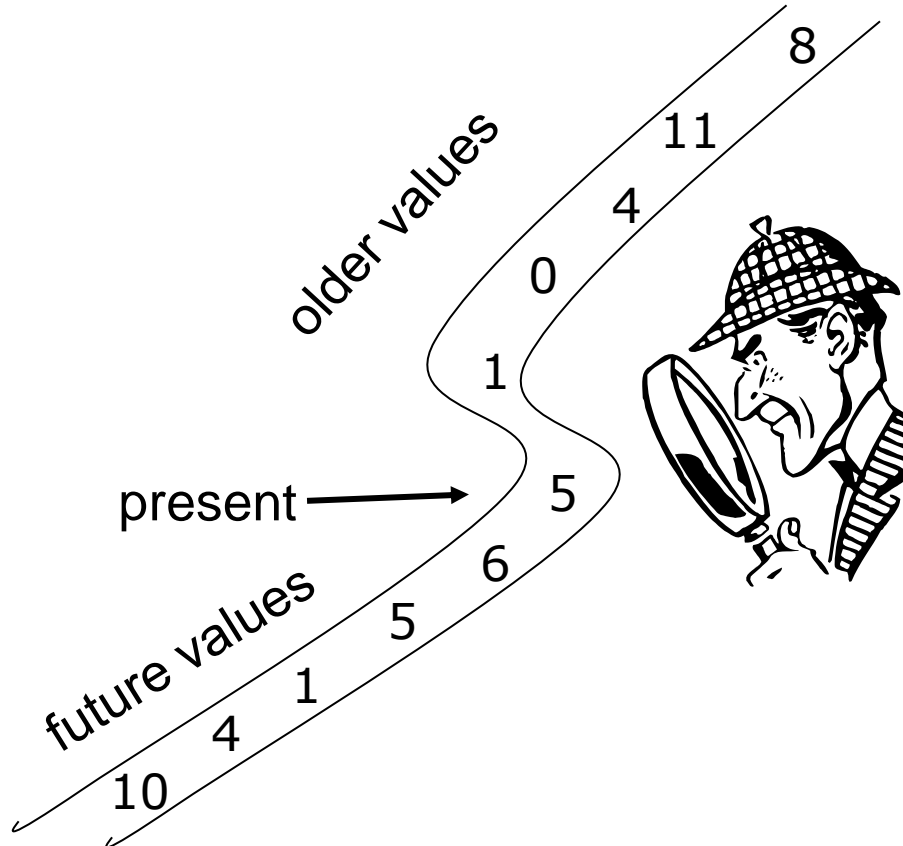
UI panels for 'Selection Strategies' and 'Consumption Policies'. Selection Strategies include 'Strict Contiguity', 'Relaxed Contiguity', and 'Non-Deterministic...'. Consumption Policies include 'NO\_SKIP', 'SKIP\_TO\_NEXT', and 'SKIP\_PAST\_LAST\_EVENT'.



**Achievements:** (1) Non-programmer event analysts rapidly develop and deploy new CEP pipelines. (2) Exploits the processing capacity of modern hardware without requiring practitioners to make cluster administration decisions.

# Stream processing: exact computations over **infinite** data streams

73

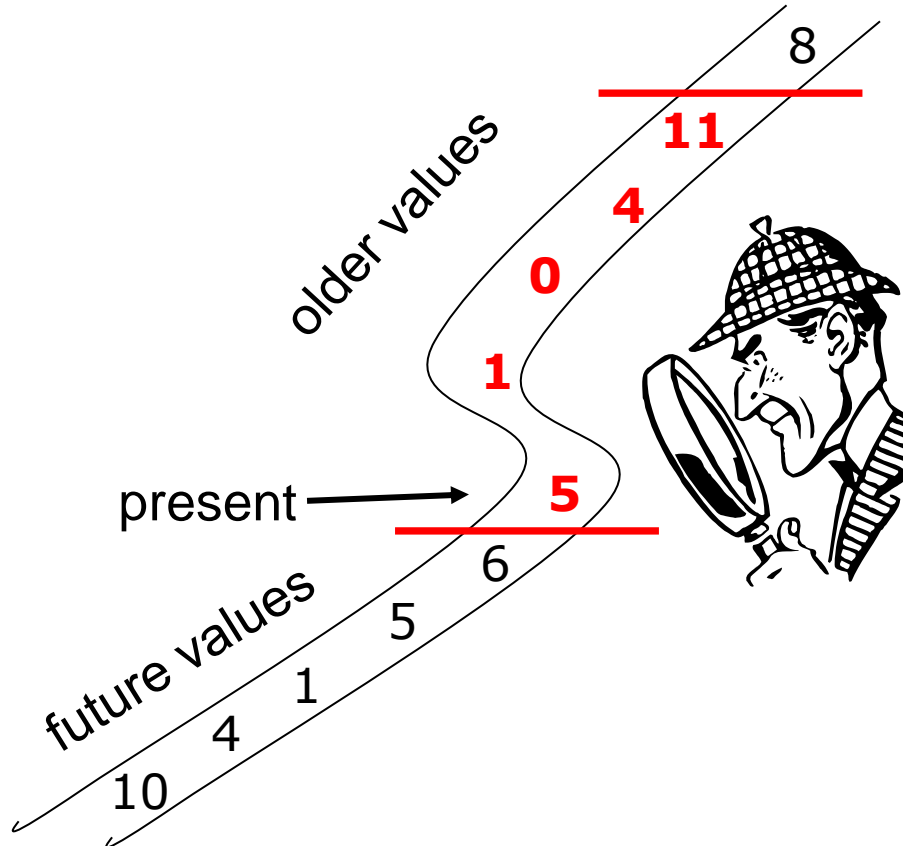


**Data Stream**

- For a stream of  $n$  values, some problems require  $O(n)$  memory in order to compute a definite answer, others don't
- **Consider the following computations**
  - ▣ Maximum/minimum value in a stream
  - ▣ Average value?
  - ▣ Median value?
  - ▣ Most frequent value?

# Windows: used to segment a continuous data stream into batches of tuples

74



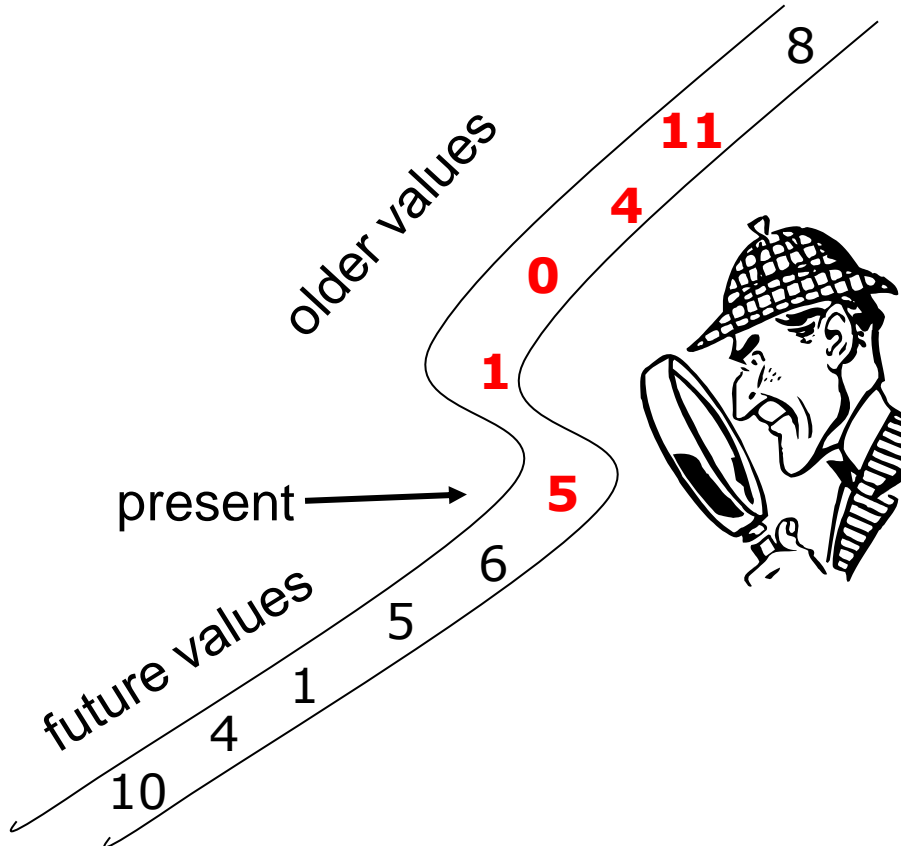
Data Stream

- Restrict the computation on the most recent tuples (**window**)
  - ▣ Tuple-based window: 5 most recent tuples (example on the left)
  - ▣ Time-based window: all data in the past 15 minutes
- The previous issue still holds (for  $n$  = window size)

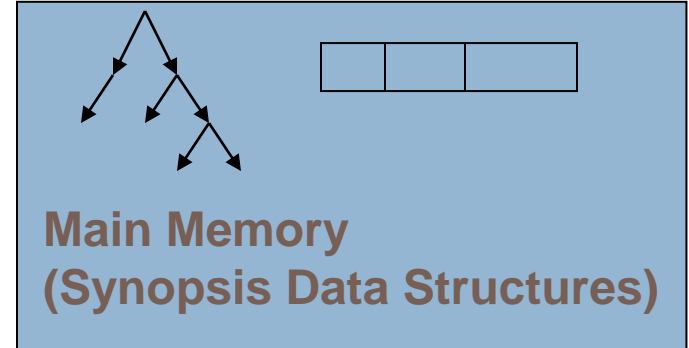


# Approximate query processing

75



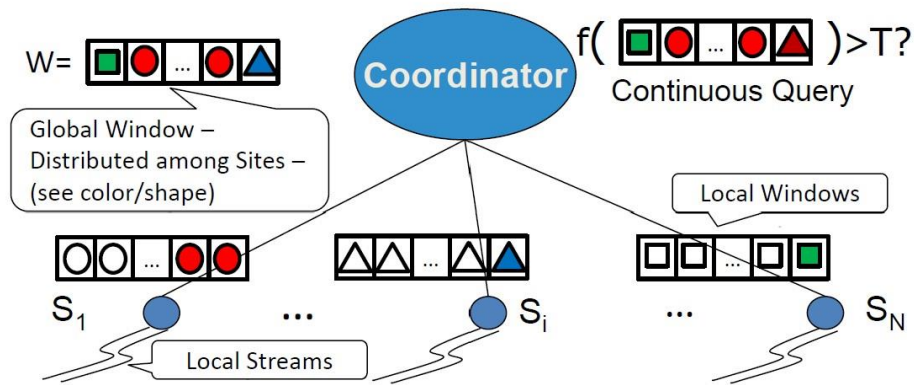
Data Stream



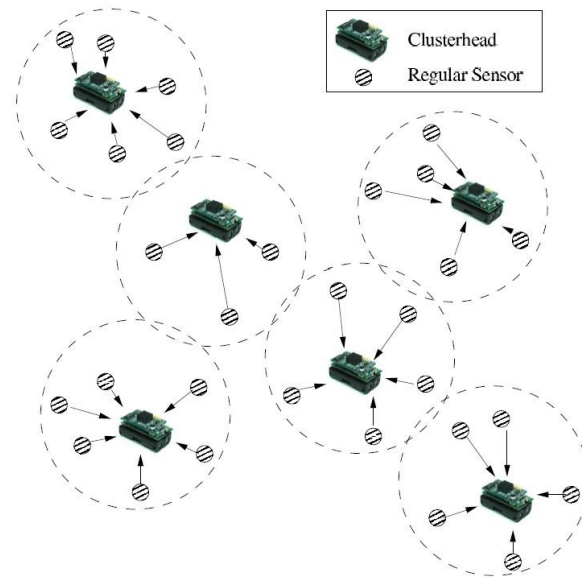
- Maintain **synopses** over the data and use them at query time to produce **approximate** answers
  - Example: data sketches, discussed next

# Distributed Stream Processing

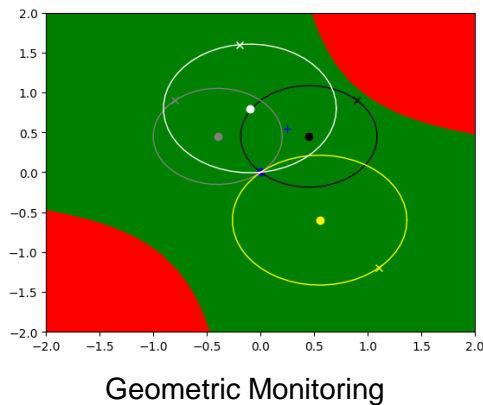
76



Distributed Monitoring Example (Nikos Giatrakos)



In-network outlier detection (Kotidis et. al)

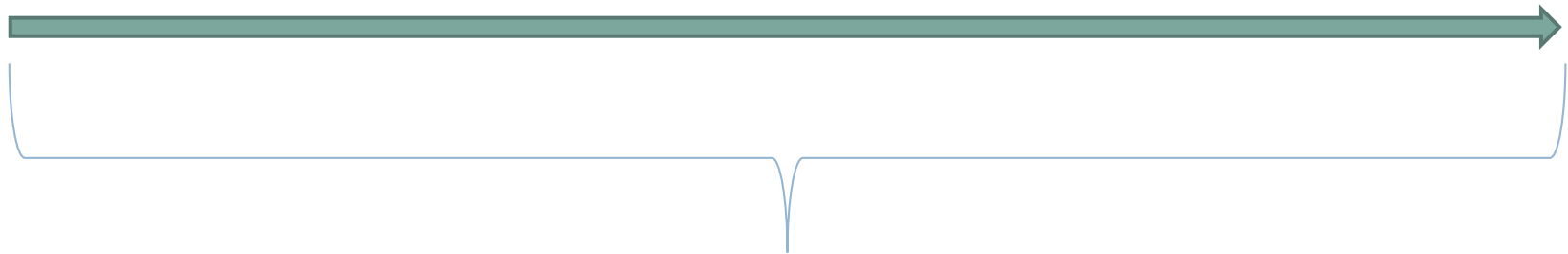


# Stream processing example (1): Most frequent items

77

- Compute most-frequent #hashtags each day
  - ▣ 500 million tweets each day
  - ▣ Multiple hashtags / tweet

#a, #b, #a, #a, #d, #b, #c, #a, #a, #b, #a, #a, #a, #a, #a



1 day, hundreds of millions stream records

# Stream processing example (2): moment estimation (Window=last 15 hashtags)

a, b, c, b, d, a, c, d, a, b, d, c, a, a, b

$$m_a = 5$$

$$m_b = 4$$

$$m_c = 3$$

$$m_d = 3$$

stream 1



$$S=2^{\text{nd}} \text{ moment} = 25+16+9+9=59$$

a, b, a, a, d, a, c, a, a, a, a, a, a, a

$$m_a = 12$$

$$m_b = 1$$

$$m_c = 1$$

$$m_d = 1$$

stream 2



$$S=2^{\text{nd}} \text{ moment} = 144+1+1+1=147$$

S is a measure of how uneven the distribution is

# Another Streaming Application

79

- Compare twitter usage across different countries **in real-time**
  - ▣ Connect to twitter API
  - ▣ Filter based on location (e.g. Greece)
  - ▣ Parse tweets, find hashtags
  - ▣ Compute frequencies
  - ▣ **Compare frequency distributions**



“Bag” of hashtags for Greece

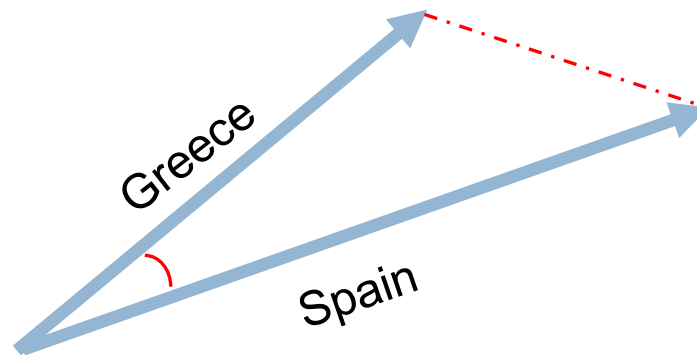
(#android,5)  
(#MeToo,1)  
(#iphone,1)  
(#NBAfinals,2)

Compute frequencies

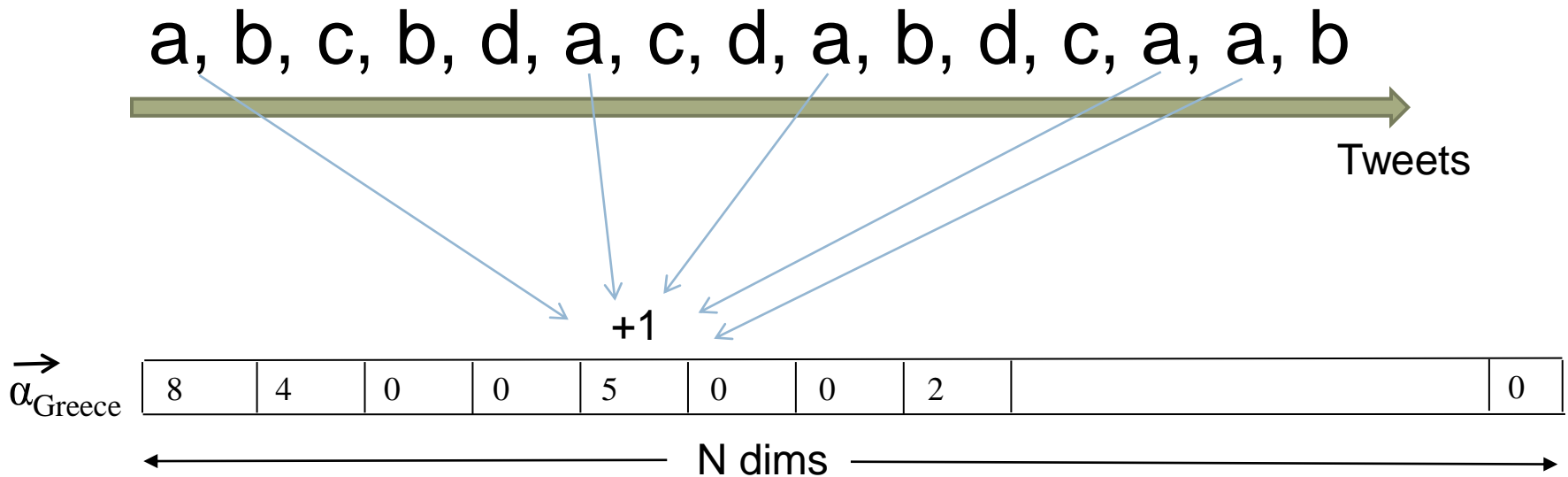
# A simple idea (just to keep going)

80

- For each country, **embed** all **frequencies** into a single **vector**
- Compare countries by comparing their respective vectors
  - ▣ We will discuss details of alternative metrics in the next lecture



# Frequency vector for Greece



- Each hashtag (string) is **mapped** to a coordinate in a N dimensional space
- The coordinate value is a counter that depicts the frequency of this particular hashtag

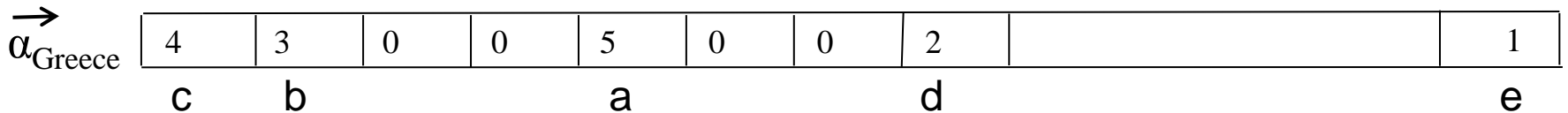
# What does it mean?

My raw data (hashtags = strings from Greek tweets):

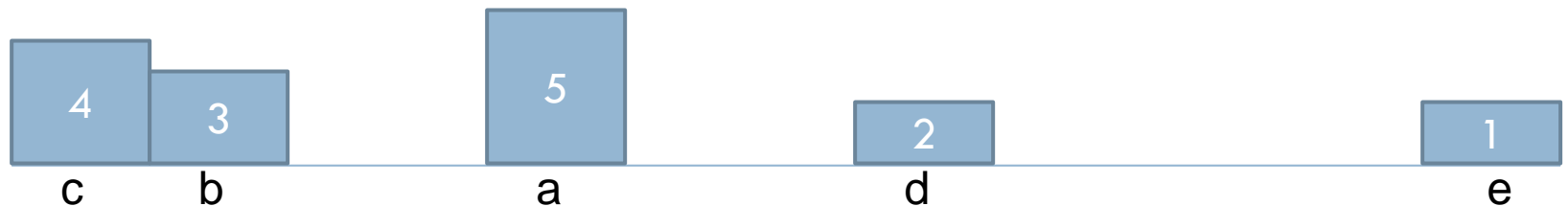
a, b, c, b, d, a, c, d, a, b, c, a, a, d, e, c

Time

How I chose to represent this information in my program:



What I am actually capturing with the above representation:





# Note

83

- This representation is “correct” if comparing frequency distributions using vectors is my means of analyzing this data
  - ▣ Dimensionality curse?
- Don’t jump into a “convenient” representation
- Consider the pros and cons of each approach
  - ▣ Goals, accuracy, performance tradeoffs, feasibility

# Hashing: map keys to an integer domain



.....#paobc.....#paobc

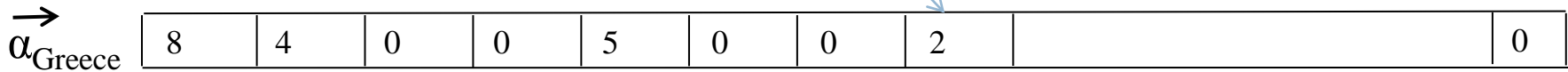


Tweets

$$h(\text{"#paobc"})=7$$

Slot 0      Slot 1

Slot 7



← N dims →

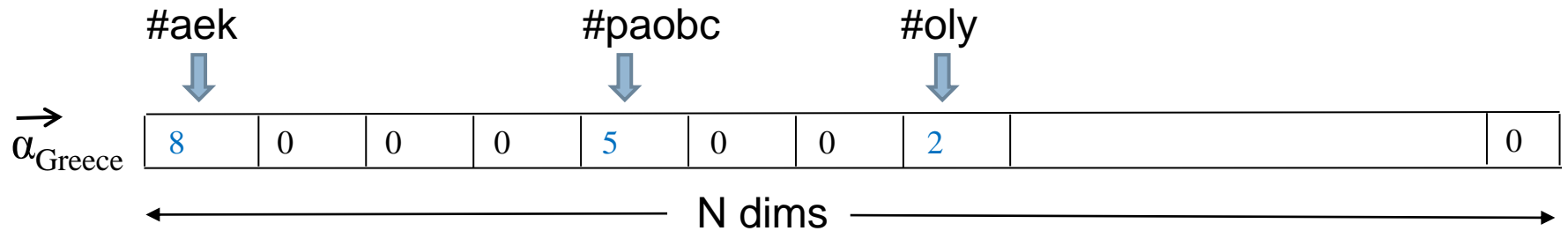
$N=2^m$  for  $h()$  returning  $m$ -bit integer values

- In this example key values are the different hashtags (that we do not know in advance)
- We need a function  $h(s)$  that maps a string  $s$  to an integer (next lecture)

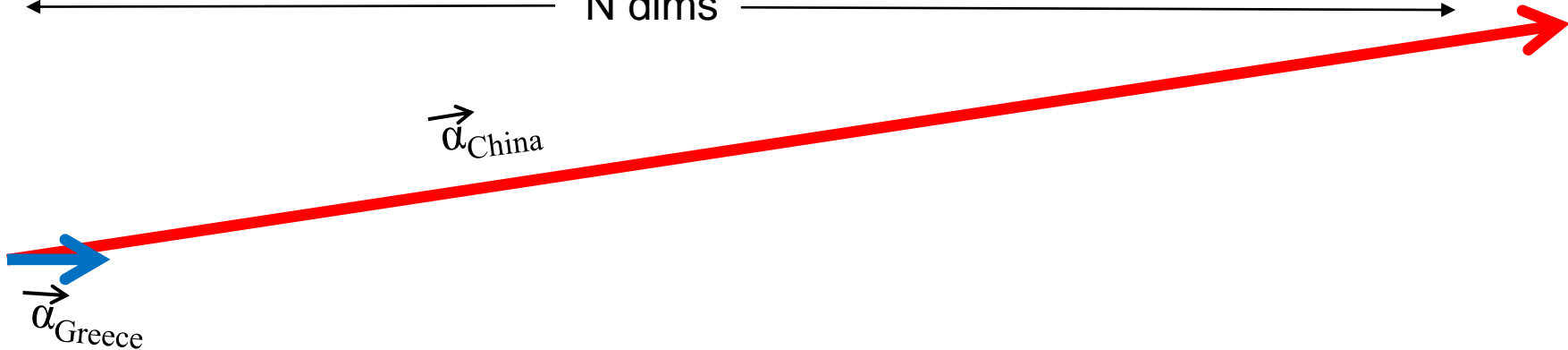
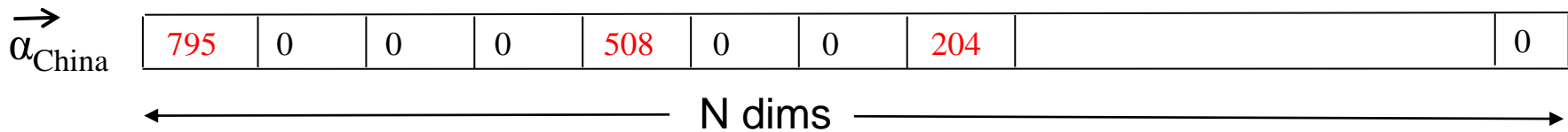
# Now that I have these vectors, I can compare them!

85

- Vector for Greece



- Vector for China (not likely but for the shake of this example...)

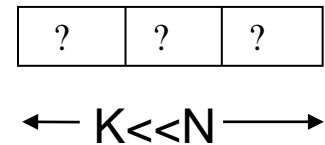
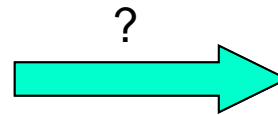
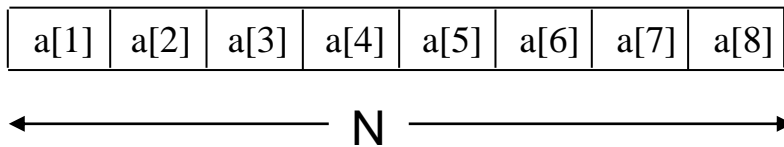




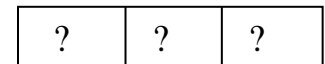
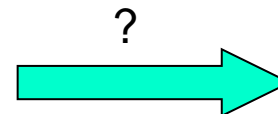
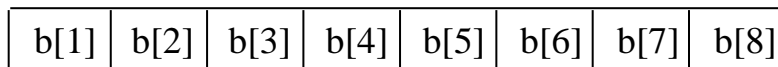
# Data sketches

- Mathematical representations of data that allow useful information to be extracted effectively
  - ▣ Preserve distance computations (approximately)
  - ▣ Have small time/space complexity
  - ▣ Allow for real-time changes in data

Data Item A



Data Item B



# Dot (inner) product between two vectors

- $\vec{x} \cdot \vec{y} = \sum(x_k * y_k)$

- Example:

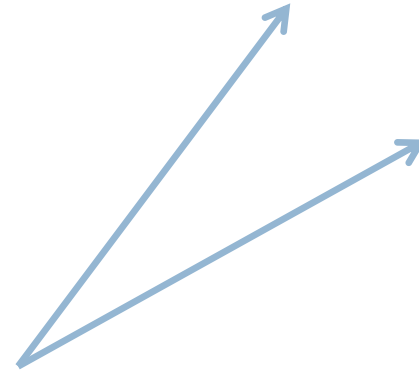
$$\vec{x} = (1, 3, 0, 5)$$

$$\vec{y} = (1, 0, 1, 6)$$

- Then:

$$\vec{x} \cdot \vec{y} = 1 * 1 + 3 * 0 + 0 * 1 + 5 * 6 = 31$$

$$= |\vec{x}| * |\vec{y}| * \cos(\theta(\vec{x}, \vec{y}))$$



# Dot product with unit vector

- $\vec{x} \cdot \vec{y} = \sum(x_k * y_k)$
- Example for unit vector  $\vec{y}$ :

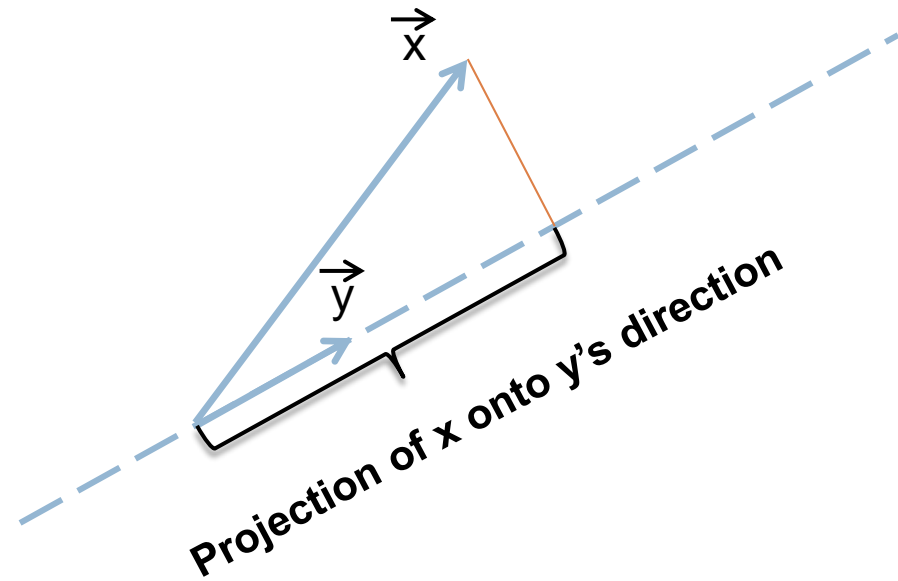
$$\vec{x} = (1, 3, 0, 5)$$

$$\vec{y} = (1/2, 1/2, 1/2, 1/2)$$

- Notice that  $|\vec{y}| = 1$

- Then:

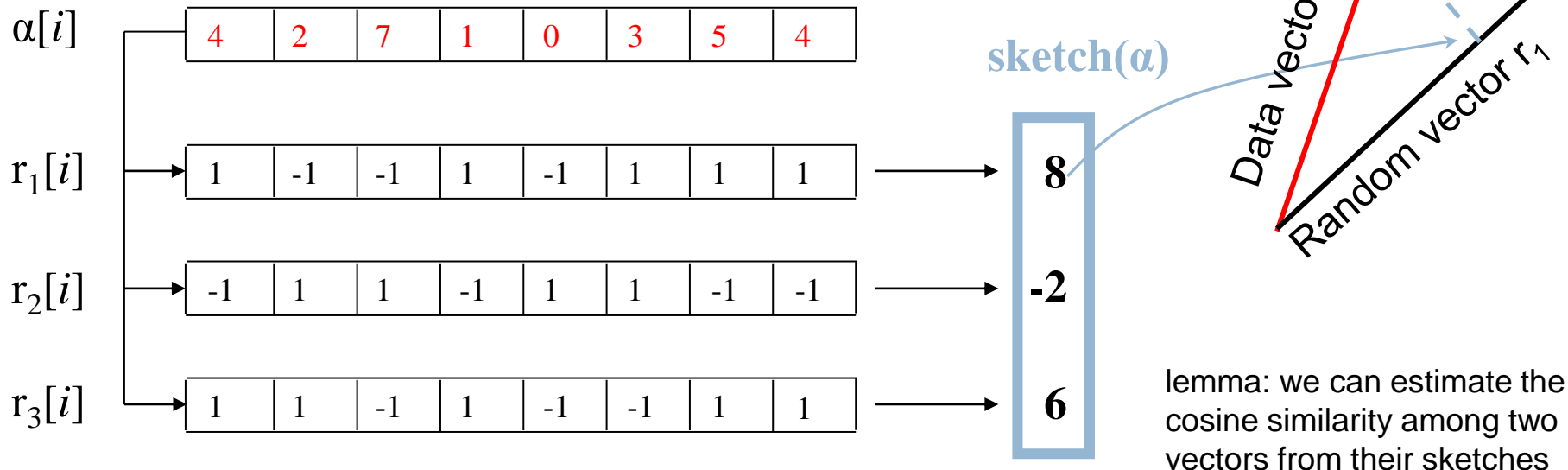
$$\begin{aligned}\vec{x} \cdot \vec{y} &= 1/2 + 3/2 + 5/2 = 9/2 \\ &= |\vec{x}| * 1 * \cos(\theta(\vec{x}, \vec{y}))\end{aligned}$$



# Linear Projections (AMS Sketches)

- [Alon96]: inner (dot) product of data vector with  $O(\log(N/\delta)/\epsilon^2)$  pseudo-random vectors  $\{-1,+1\}$  (linear projections)

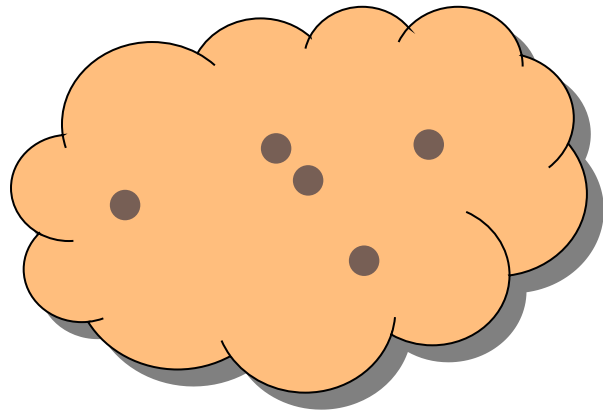
- $\epsilon$ : approximation error
- $\delta$ : failure probability



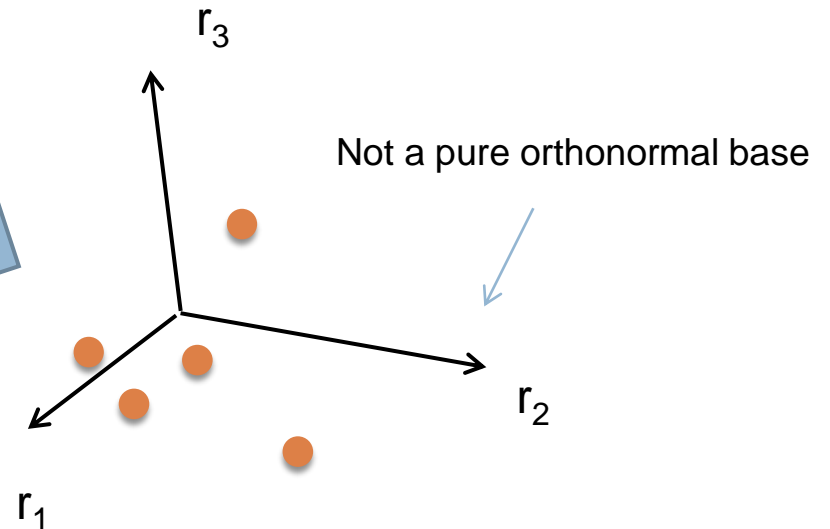
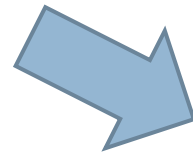


# Random Projections

93



High-dimensional space

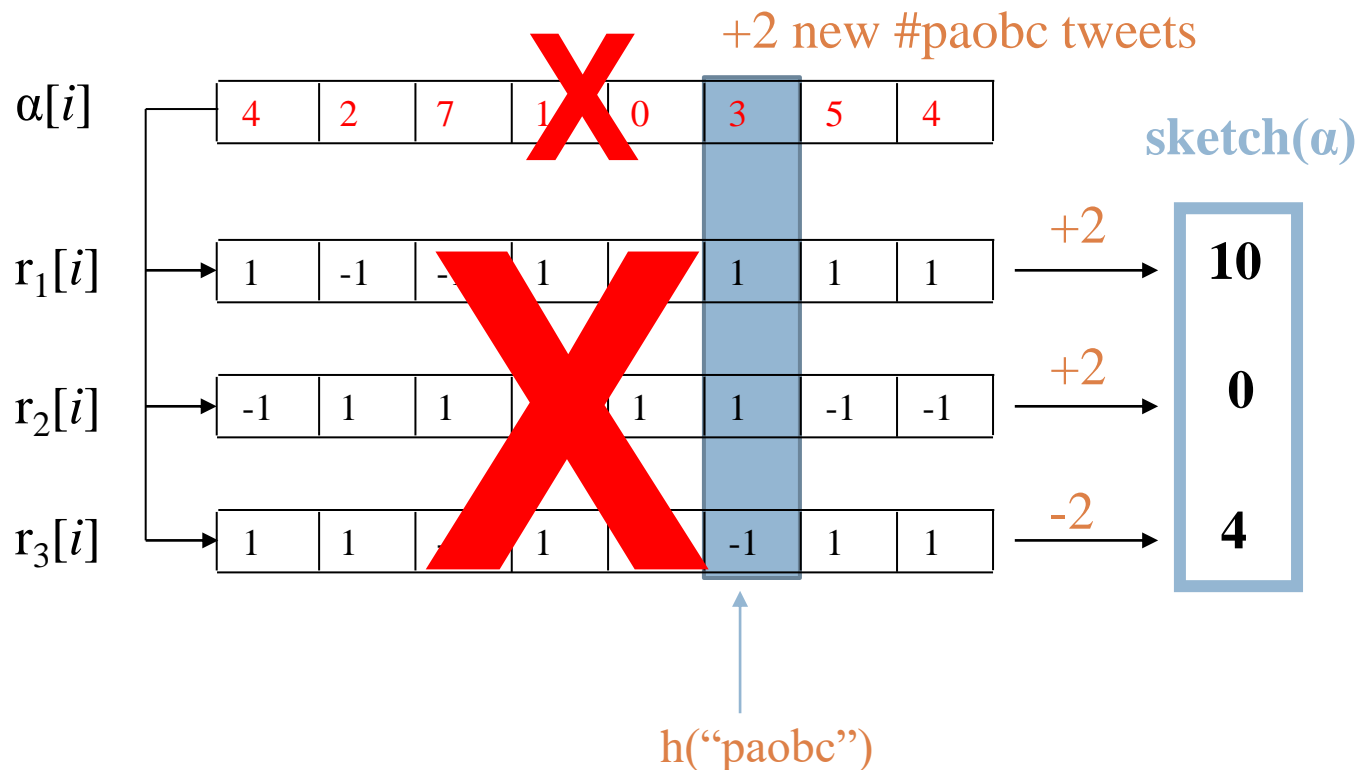


# What can we do with them

- Estimate norms, distances, dot products with  $(\epsilon, \delta)$ -guarantees
  - ▣  $\epsilon$ : approximation error
  - ▣  $\delta$ : failure probability
- Using much less space/time than using the real vectors
  - ▣ Need  $O(e^{-2}(\log 1/\delta))$  time and space (addressed volume, velocity)
    - Note: there are more accurate sketching techniques available in the literature

# Updatability (addresses velocity)

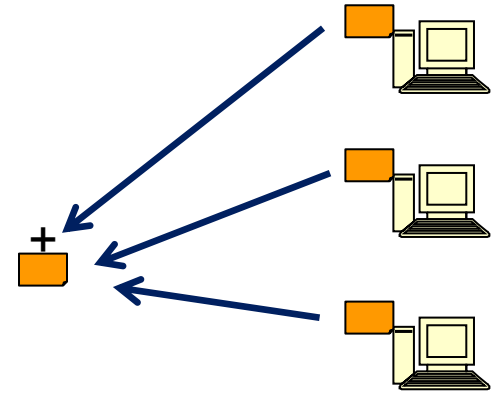
- Linearity permits **incremental updates**
  - Thus, suitable for streaming data!



# Sketches: Recap

96

- Simple Linear Projections  
(efficient computation)
- Permit *incremental* updates
- Sketches are *compassable*



# What we have achieved

- Reduced (significantly) memory footprint of the data analysis algorithm
  - Data now fits in memory → processing is orders of magnitude faster
- Have introduced (controlled) impression

# What is (still) missing?

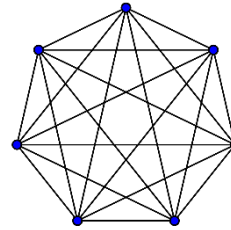
98

□ ?

# Recall complexity of pair-wise computations

99

- For an **all-pair** computation we need to perform  $n*(n-1)/2$  comparisons



- For a **Nearest-Neighbor** query  $NN-k(q)$  we need to compute  $n-1$  distances (using  $q$  as a source)
- How do we **scale** these computations?
  - **With the use of indexes!**

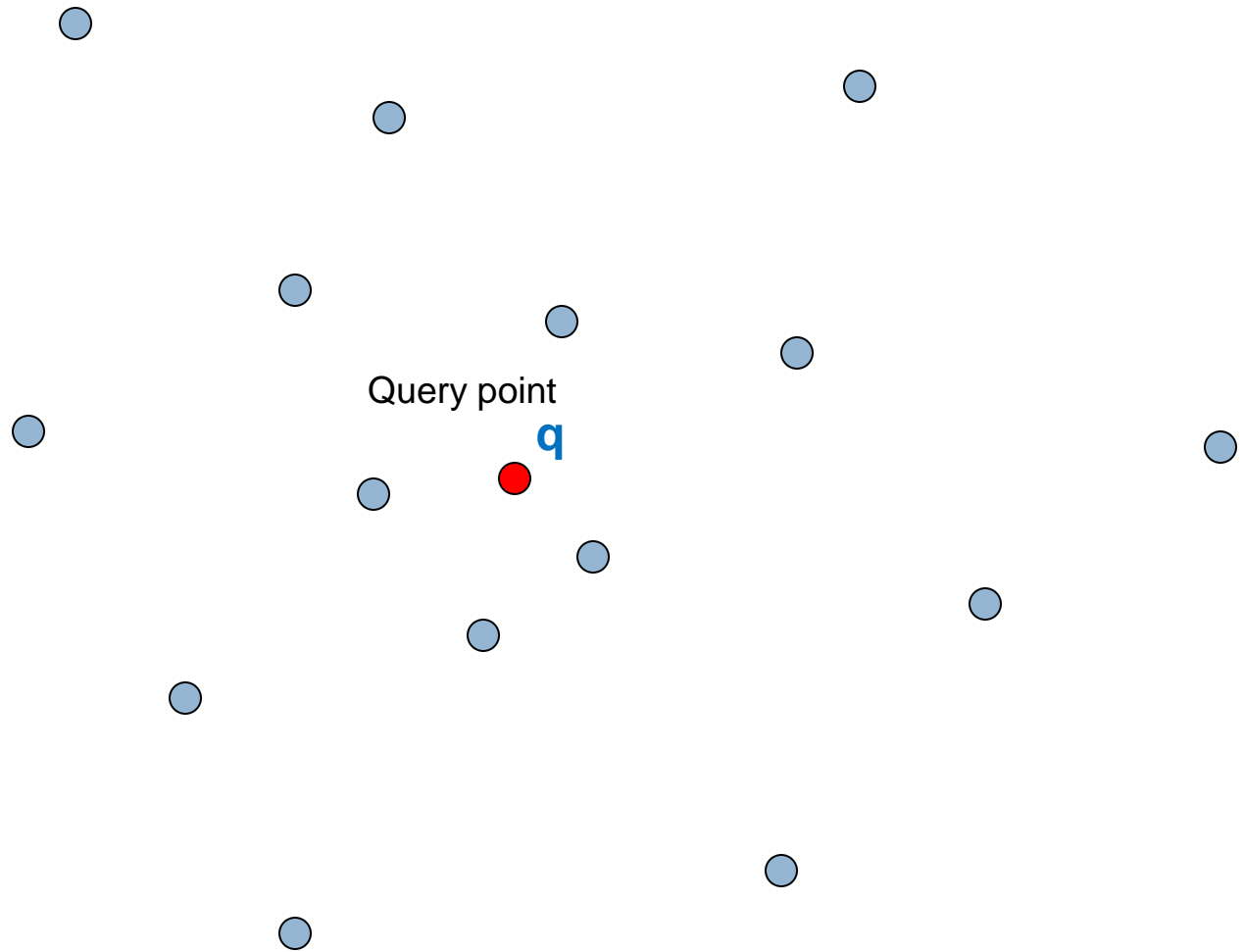
# Nearest Neighbor Queries Example

105

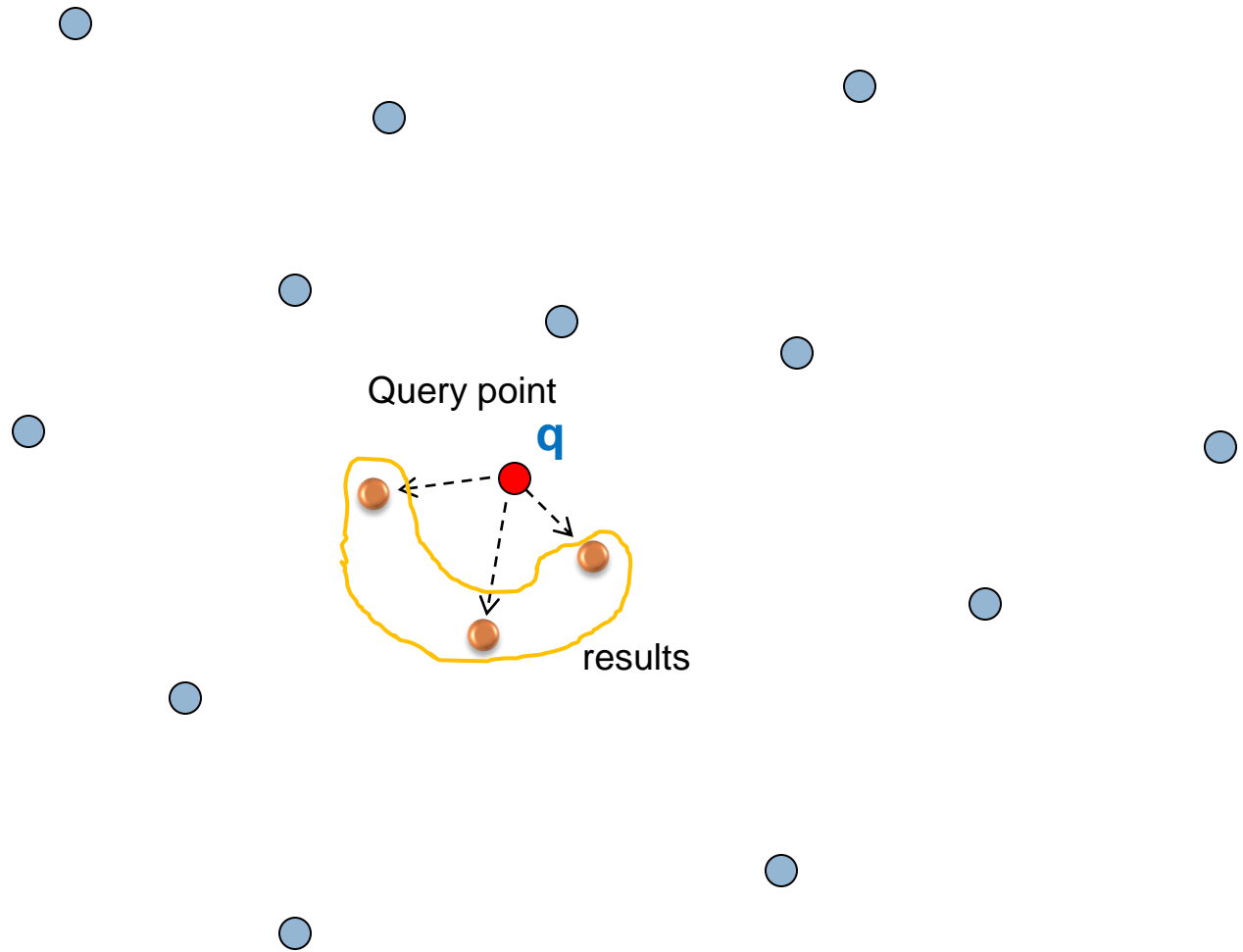
- Spatial (2-dim) domain: find me 3 restaurants nearest to **my location** (=query point **q**)
  - NN-k(q) (k=3) query



# Nearest Neighbor Query



# Nearest Neighbor Query

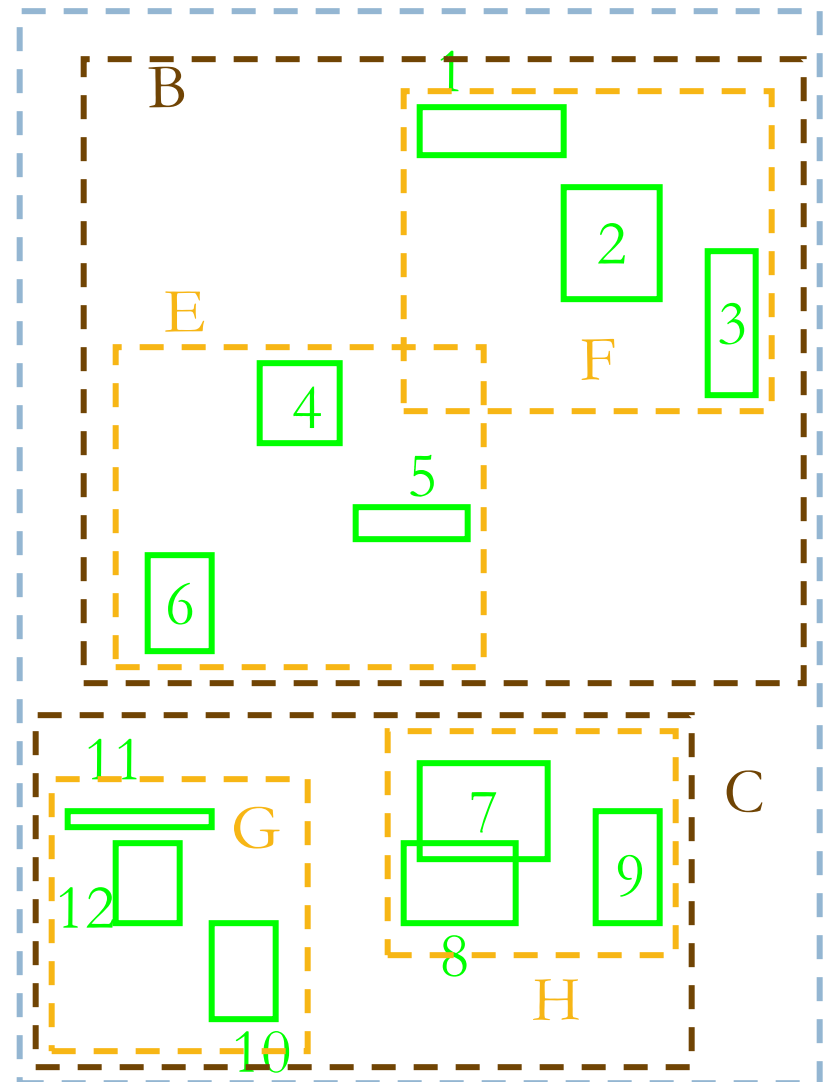


# Dimensionality Curse

## R-Tree

108

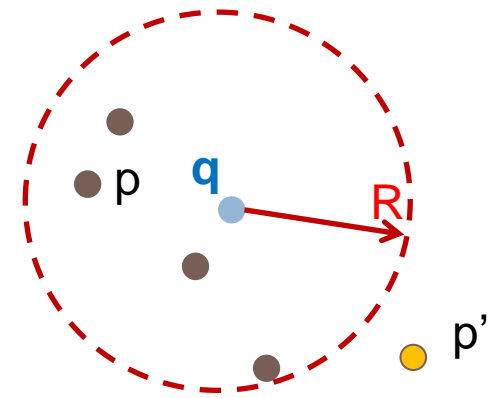
- **Curse of dimensionality** renders common indexes ineffectual
  - ▣ Often full scan of DB is preferable but slow
- Consider also: **velocity** & **volume** of big data



# “Approximation” to the rescue (again)

109

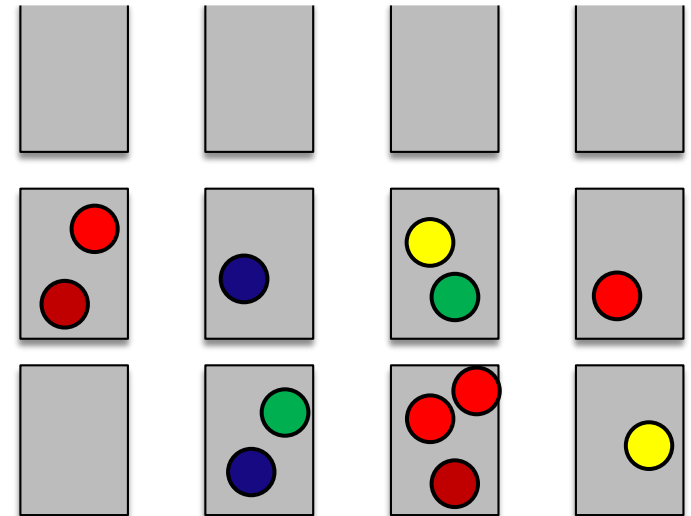
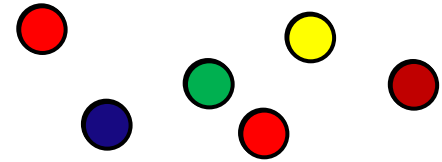
- Approximate k-NN problem: build data structure which
  - ▣ If there is a point  $p$ :  $\text{dist}(p,q) \leq R$ ,  $p$  is returned with high probability
    - i.e. high probability of not missing a true NN = few false negatives
  - ▣ If  $\text{dist}(p',q) > R$ , the probability of returning  $p'$  is small
    - Most results are true NN = **few false positives**
      - **False positives** may be **pruned** in a second step



# Locality Sensitive Hashing (LSH)

110

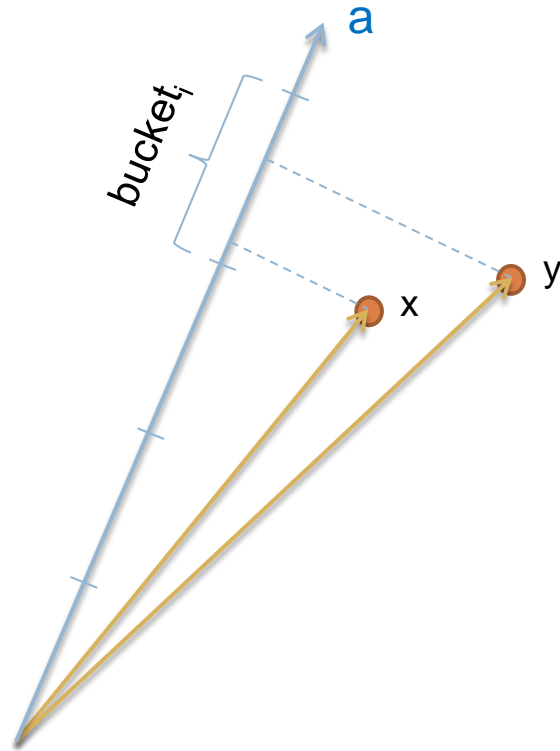
- Assign items to **buckets** using a hash function  $h(x)$ 
  - E.g.  $h(\bullet)=0$
  - Details of function  $h()$  depend on the preferred similarity metric:
    - Similar objects are hashed to the same bucket with high probability
    - Dissimilar objects are hashed to the same bucket with small probability
- Repeat several times



Buckets (0-3)

# Intuition of a hash function that preserves the Euclidian Distance

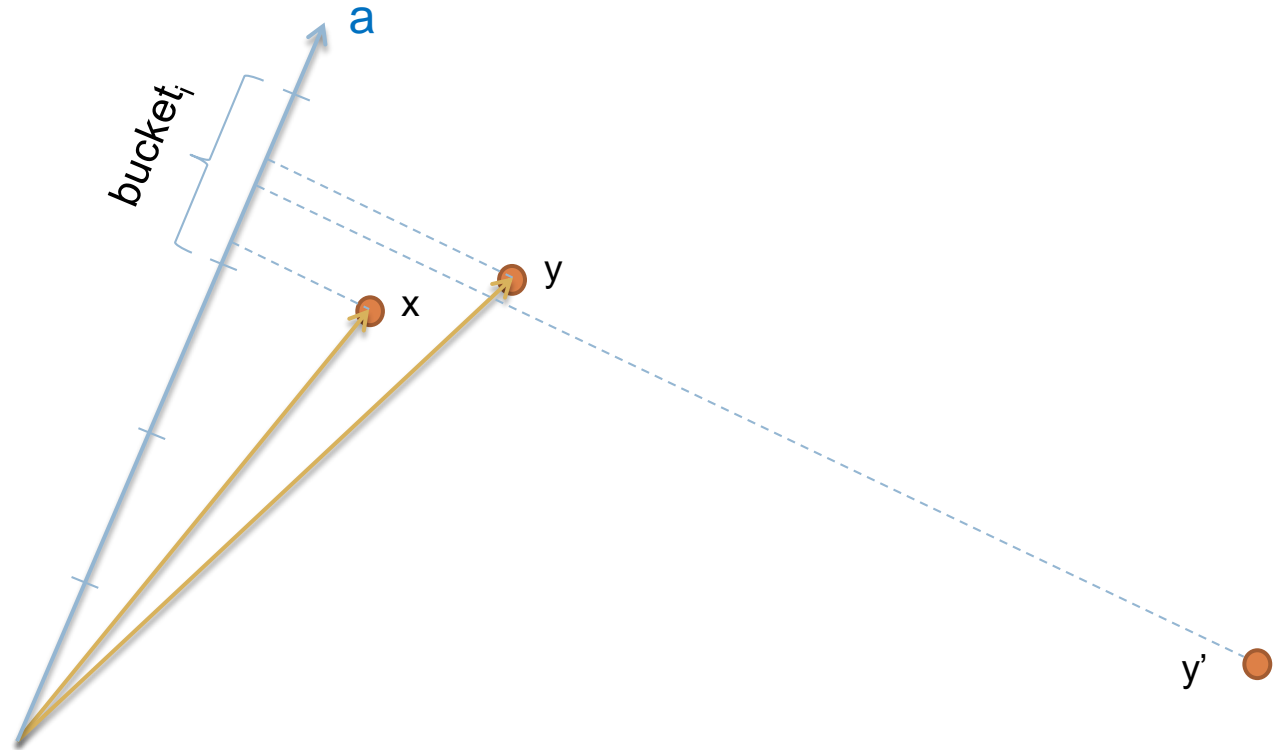
111



Recall : length of projection of vector  $x$  onto  $a$  is the **inner product  $a \cdot x$**

# False positives

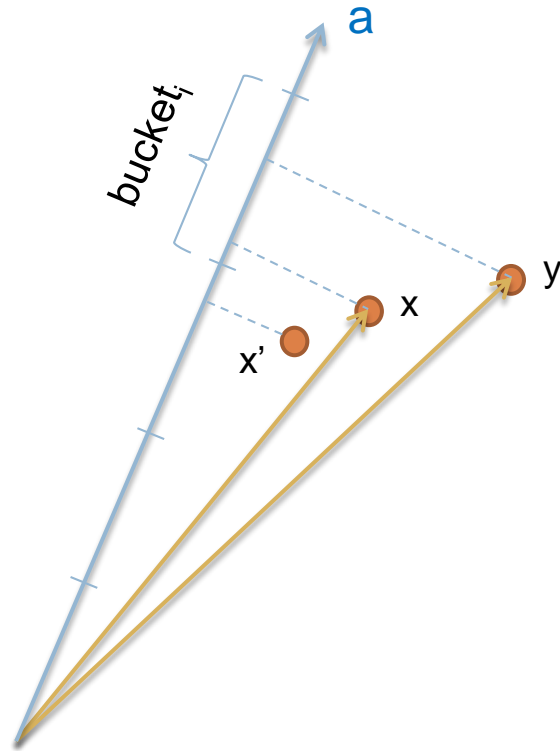
112



In this example  $y'$  is a **false positive** (will be pruned when computing true distance)

# False negatives

113



In this example  $x'$  is a **false negative** (unless we use additional hash functions)



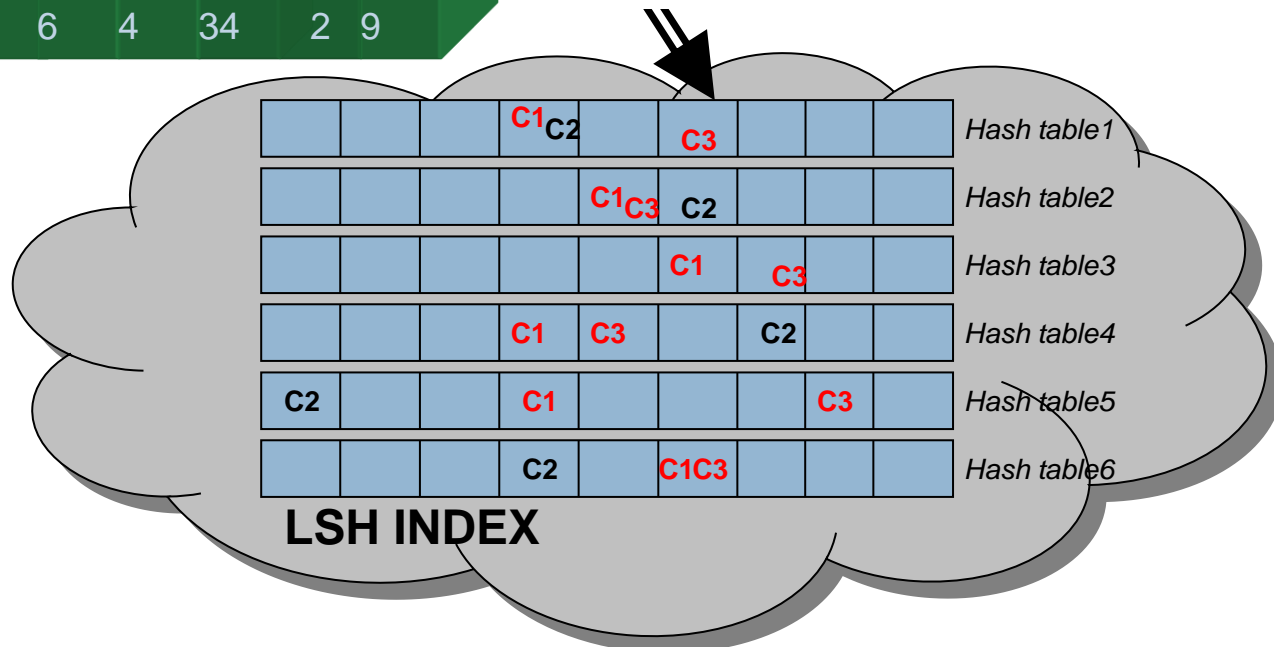
# LSH example for Basket Data

Customers' Purchases

<b>C1</b>	32	24	43	6	4	29
<b>C2</b>	4	7	3	56	48	9
<b>C3</b>	41	18	39	4	2	21
<b>C4</b>	9	13	17	4	11	14
<b>C5</b>	23	49	36	23	25	21
<b>C6</b>	7	6	4	34	2	9

$$h_{\alpha,b}(u) = \left\lfloor \frac{\alpha \cdot u + b}{r} \right\rfloor$$

ASSIGNING  
CUSTOMER VECTOR  
TO HASH BUCKETS



# More on LSH

115

- Families of hash function exist for several popular distance (similarity) metrics
  - ▣ Euclidian, Manhattan, Jaccard
- Hashing on multiple hash tables is parallelizable  
...and map-reduce-able!

# Personal View

125

- Past work on Data Streams from Databases and Algorithms community may help overcome some of the obstacles in dealing with Big Data
- We can revisit Data Analysis/Mining algorithms (Clustering, Eigenvector Analysis, Outlier detection) so as to benefit from methods and techniques developed in those areas

# Summary

126

- Big data raises several issues in Data Analysis
  - ▣ Scale, noise, dynamics, heterogeneity, inter-dependencies
- Data analysis itself can also be used to help improve the quality and trustworthiness of big data, understand its semantics, and provide intelligent querying functions
- Coordination & integration between different technological platforms is required
  - ▣ Data Warehouses/NoSQL platforms/DSMS/DM&ML libraries



# Bibliography on Sketches & LSH (beyond scope of this class)

129

- A. Gilbert, Y. Kotidis, S. Muthukrishnan, M. Strauss. Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries. In proceedings of the 27th Very Large Data Bases (VLDB) Conference, Rome, Italy, September 2001.
- Noga Alon, Yossi Matias, Mario Szegedy: The Space Complexity of Approximating the Frequency Moments. STOC 1996: 20-29.
- Distributed Similarity Estimation using Derived Dimensions. K. Georgoulas, Y. Kotidis. The VLDB Journal, Volume 21(1), pages 25-50, February 2012.
- Alexandr Andoni, Piotr Indyk: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Commun. ACM 51(1): 117-122 (2008).

THANK YOU!