ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Big Data and Its Impact on the Modern Data Mining Workflow

Yannis Kotidis

kotidis@aueb.gr

Professor, Department of Informatics

Athens University of Economics and Business

About me

- Professor, Dept of Informatics, Athens University of Economics and Business
  - URL: http://pages.cs.aueb.gr/~kotidis/
  - Email: 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)

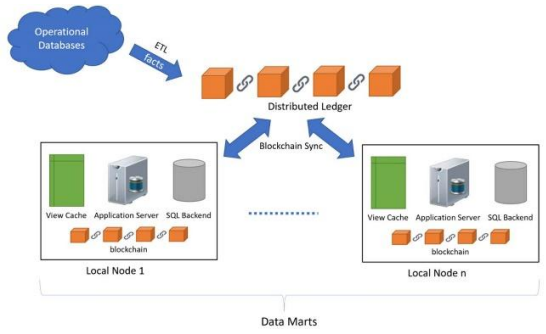
DeLorean: Compression, Indexing and Analysis Techniques for Time-Series Management

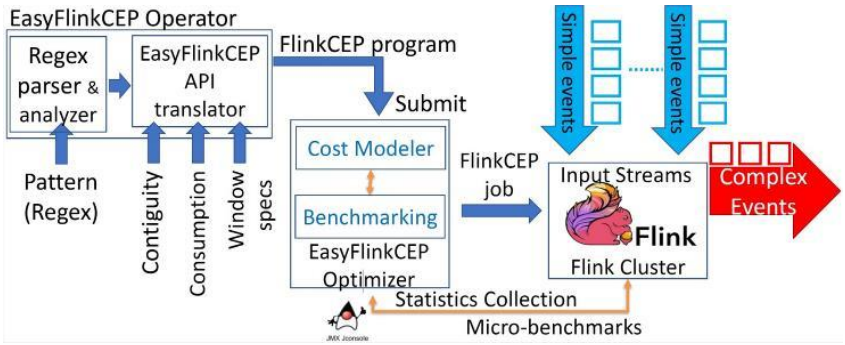
EasyFlinkCEP: Complex Event Processing using user-defined patterns

Smart-Views: Decentralized OLAP View Management using Blockchains

RECOST: 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 & models (categorical, ordinal, numerical, hierarchical, vectors, sets, strings, time series, distributions, graphs), data streams
  - 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

Course Materials: E-class Platform

- Announcements, Assignments, Grades
- Lecture and Tutorial Slides
  - Available in the "Documents" folder.
  - Suggested reading: Mining of Massive Datasets, Jure Leskovec, Anand Rajaraman, Jeff Ullman, *http://www.mmds.org/*
- Optional Slides & Code Examples
  - Found in the "Code&Data" subfolder

Course Evaluation Structure (TBD!)

- **Written exams**
  - Mid-term Examination (during the 4th lecture), Weight: 30%
  - Final Examination, Weight: 30%
- **The written exams will evaluate:**
  - Theoretical understanding
  - Practical problem-solving skills
  - Topics covered will be based on class sessions
- Assignment, Weight: 40%

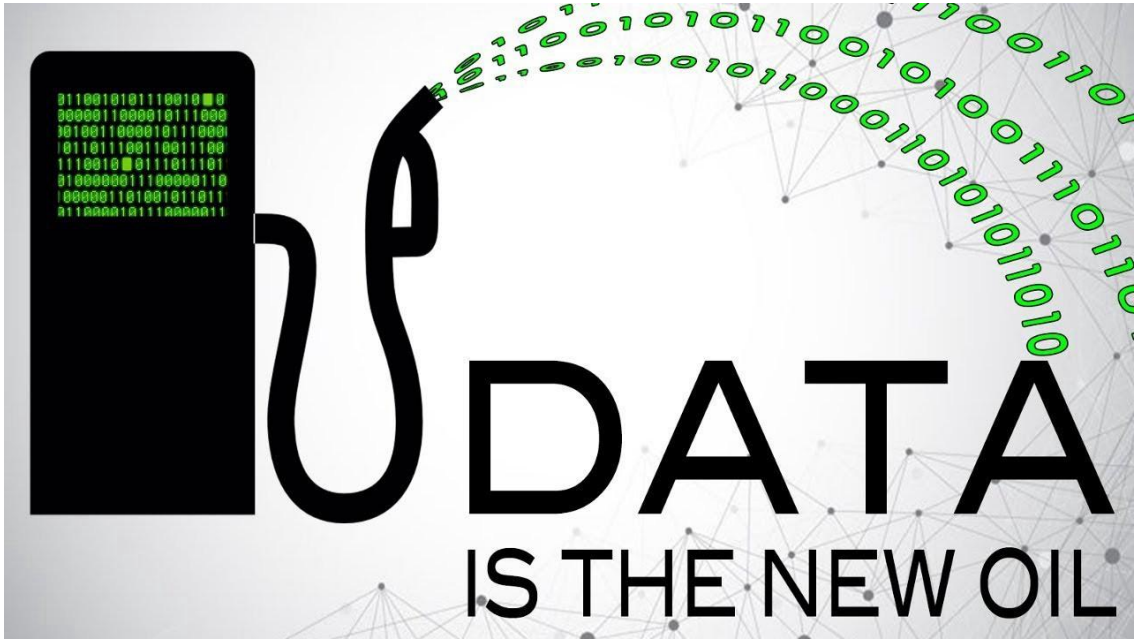
Big Data is everywhere and knows it all!

Data is the new oil in the 21st century

- 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 21st century

- 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!

- 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!

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 over1PB/year)
- Business Applications (e.g. Bank/Credit Card transactions)

Gather whatever data you can whenever possible

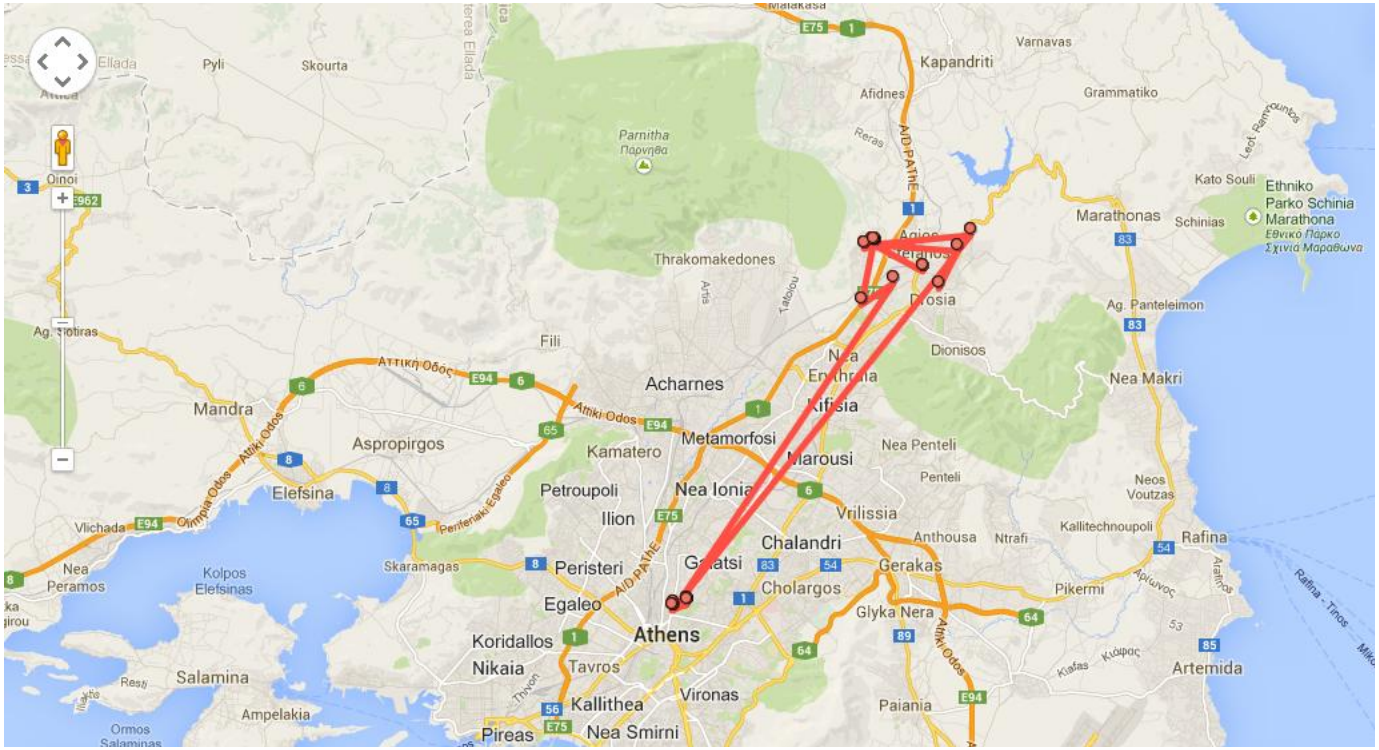
- Data will have value in the future
- …even for purposes not envisioned!

Digital Data Explosion

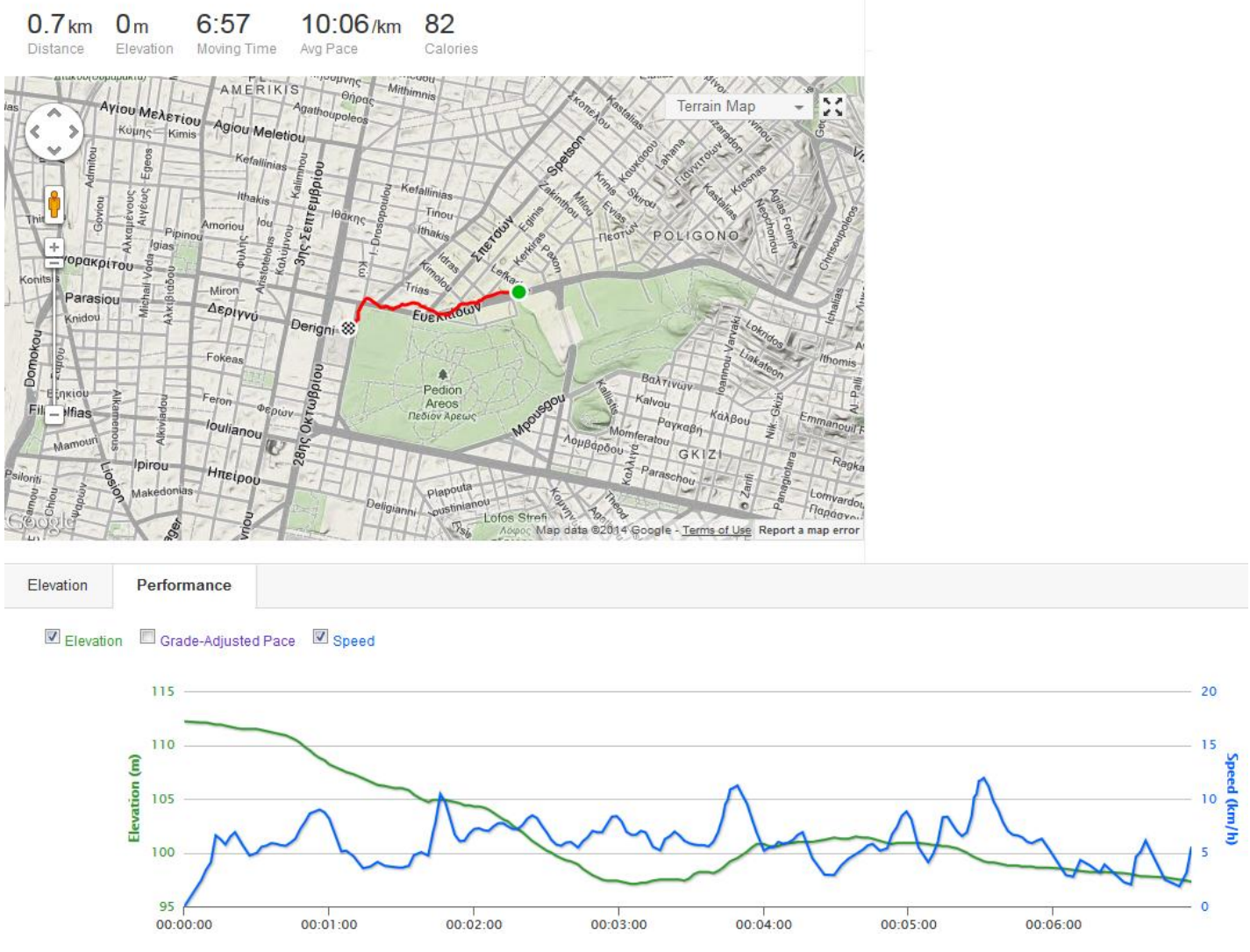
- Everything we see, read, hear, write, and measure can now be in a digital form!!
- Every few days we generate more data than we did from the dawn of civilization until 2000
  - Multimedia, scientific, sensor, etc. data is becoming prevalent

How to use this data to drive new growth?

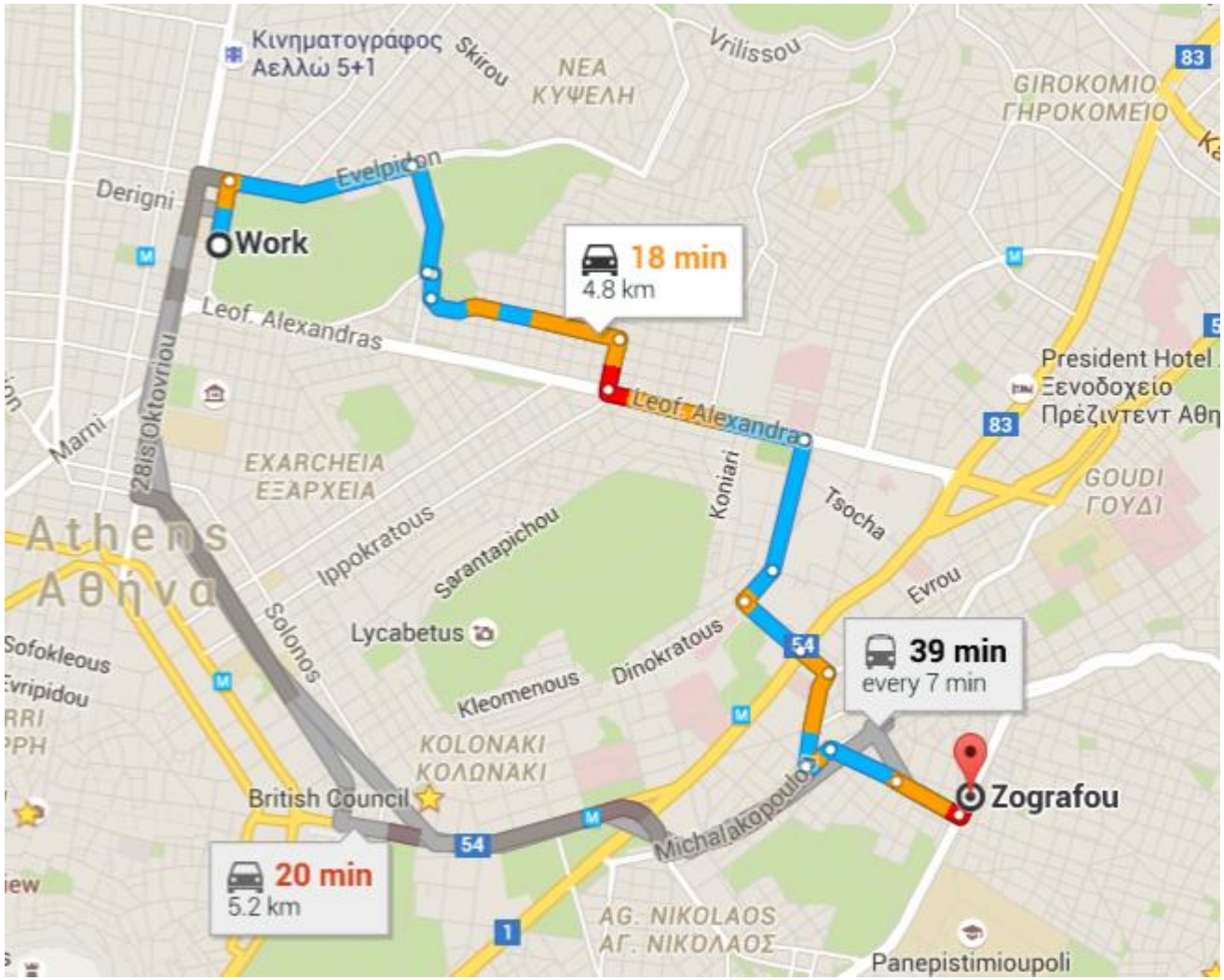
Location history based on cellular connections



Phone GPS tracking

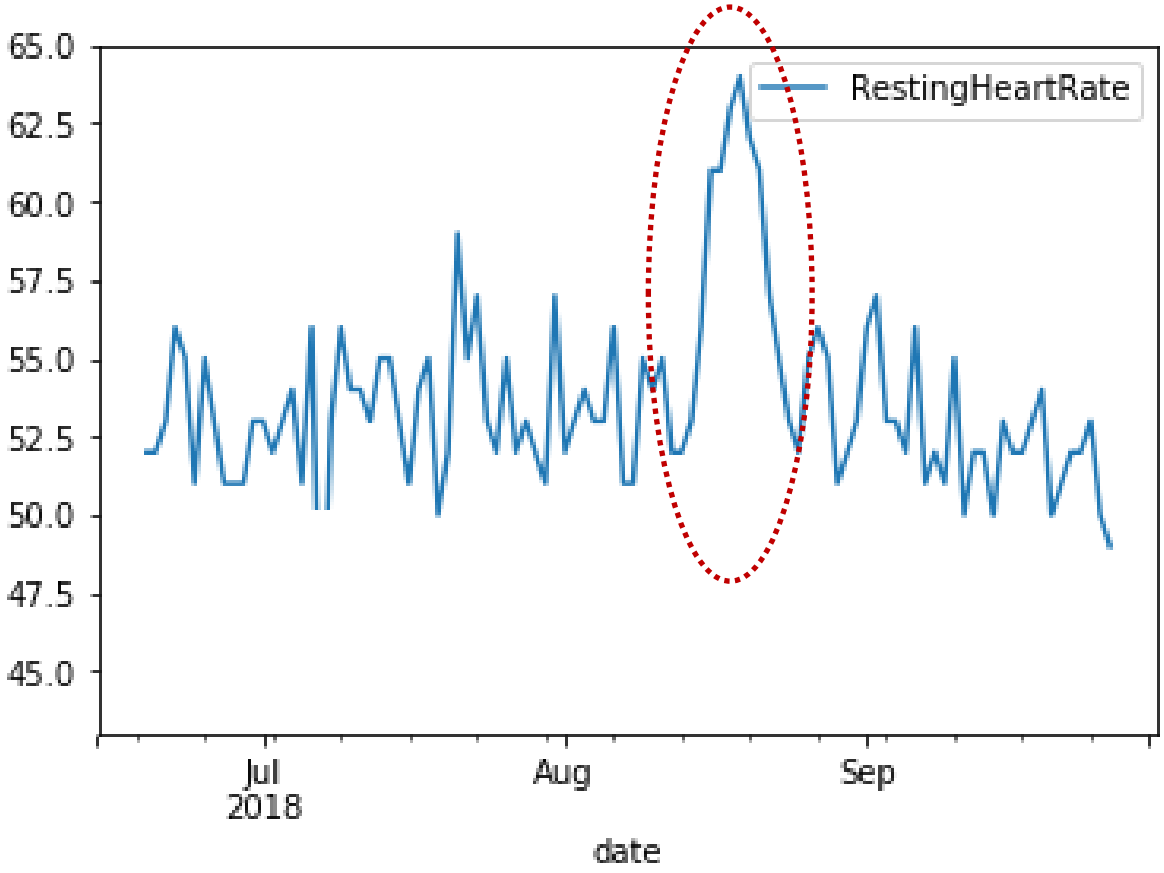


Provide useful services: Live traffic data



Another Application: Fitness Analytics

How do you explain this peak?

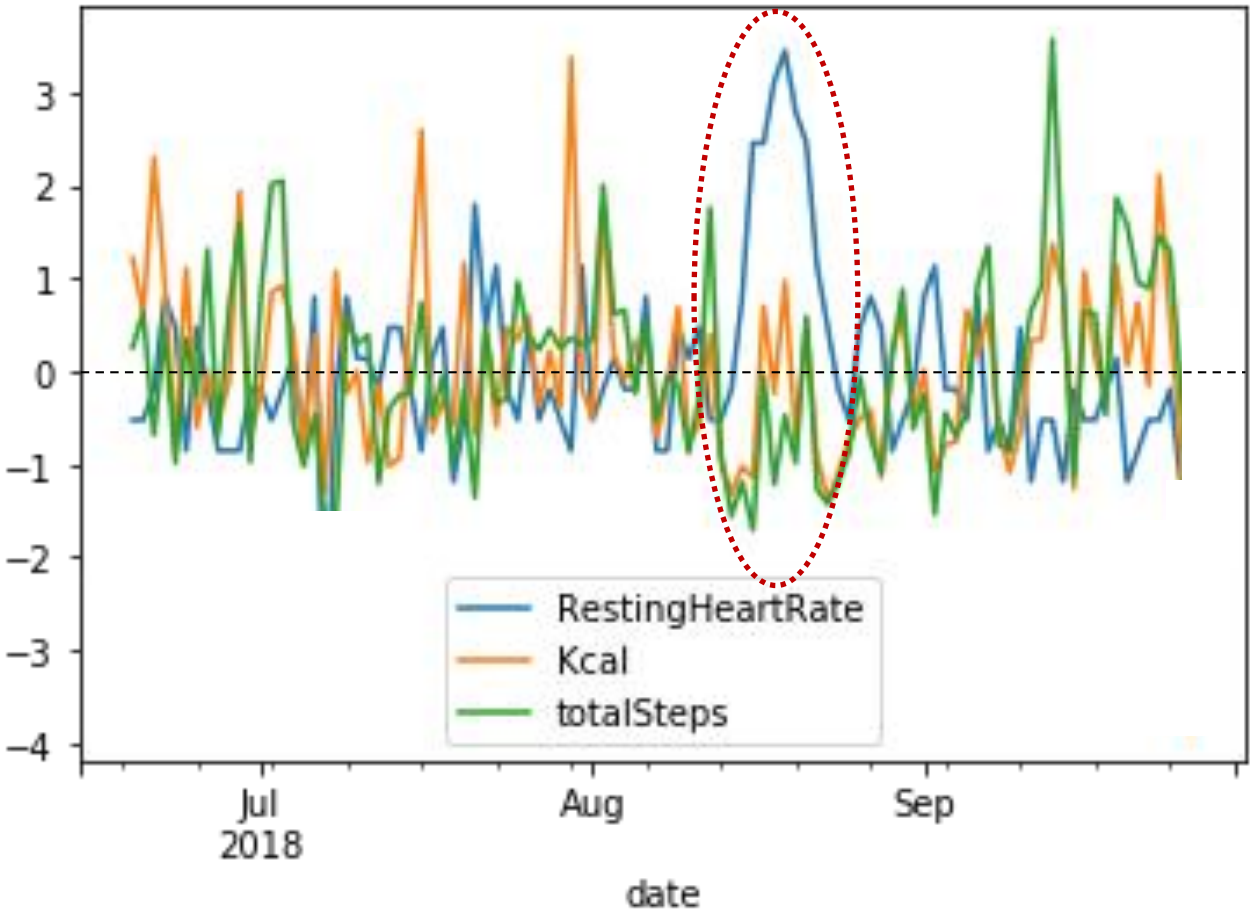


- **Descriptive analytics**: answer queries looking back in time [this example]
- **Predictive analytics**: forecasting [what is my marathon predicted time?]
- **Prescriptive analytics**: planning in order to achieve predicted outcomes [e.g. workout schedule]

More data → more insights

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

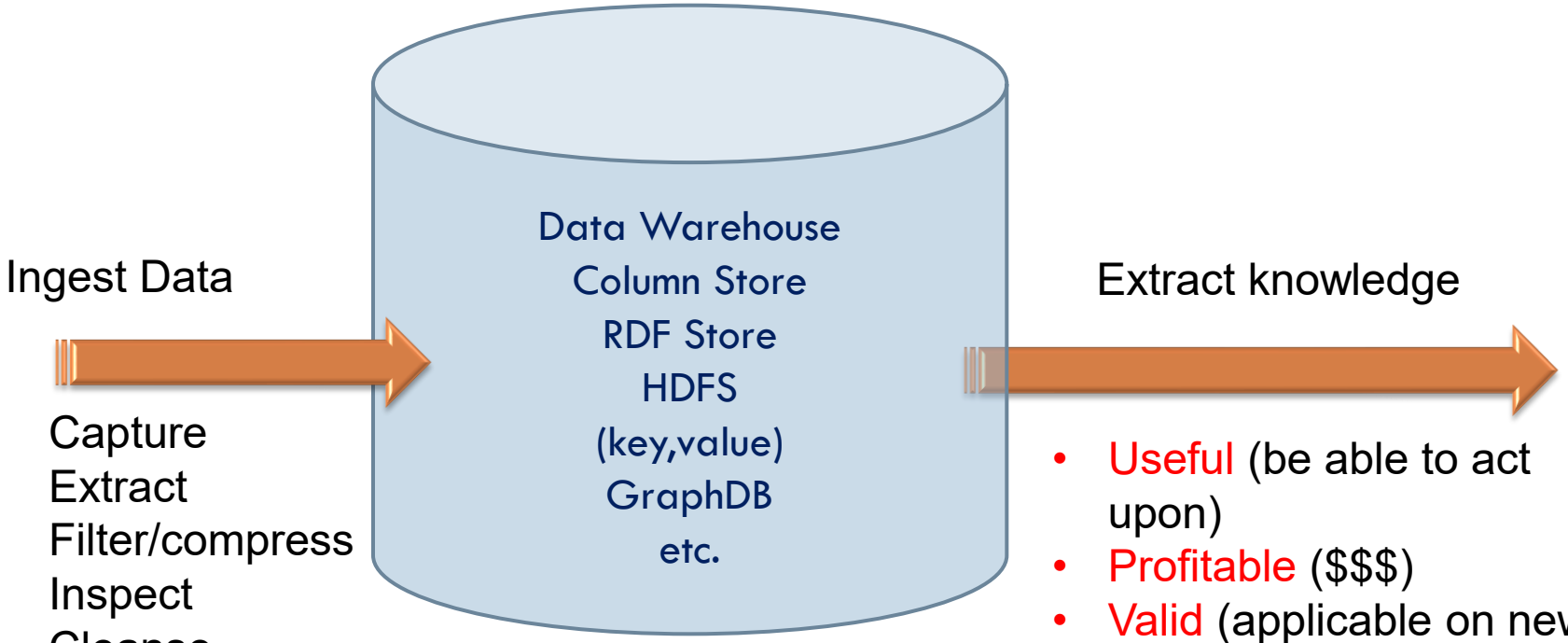
Exercise(down) → Fitness (down)



Data Mining

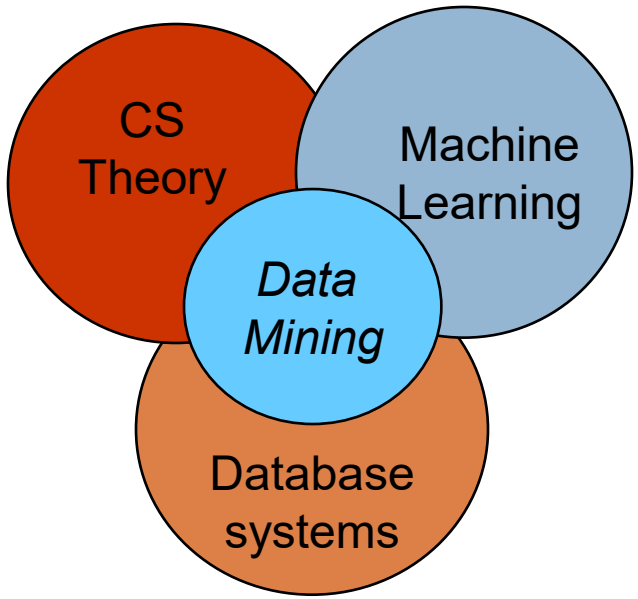
- The process of analyzing data to identify new information in the form of features, patterns or relationships
- Has become a well-established discipline related to Artificial Intelligence, Machine Learning and Statistical Analysis
  - Led by advances in computer hardware and database technology
   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

The big (&very abstract) picture



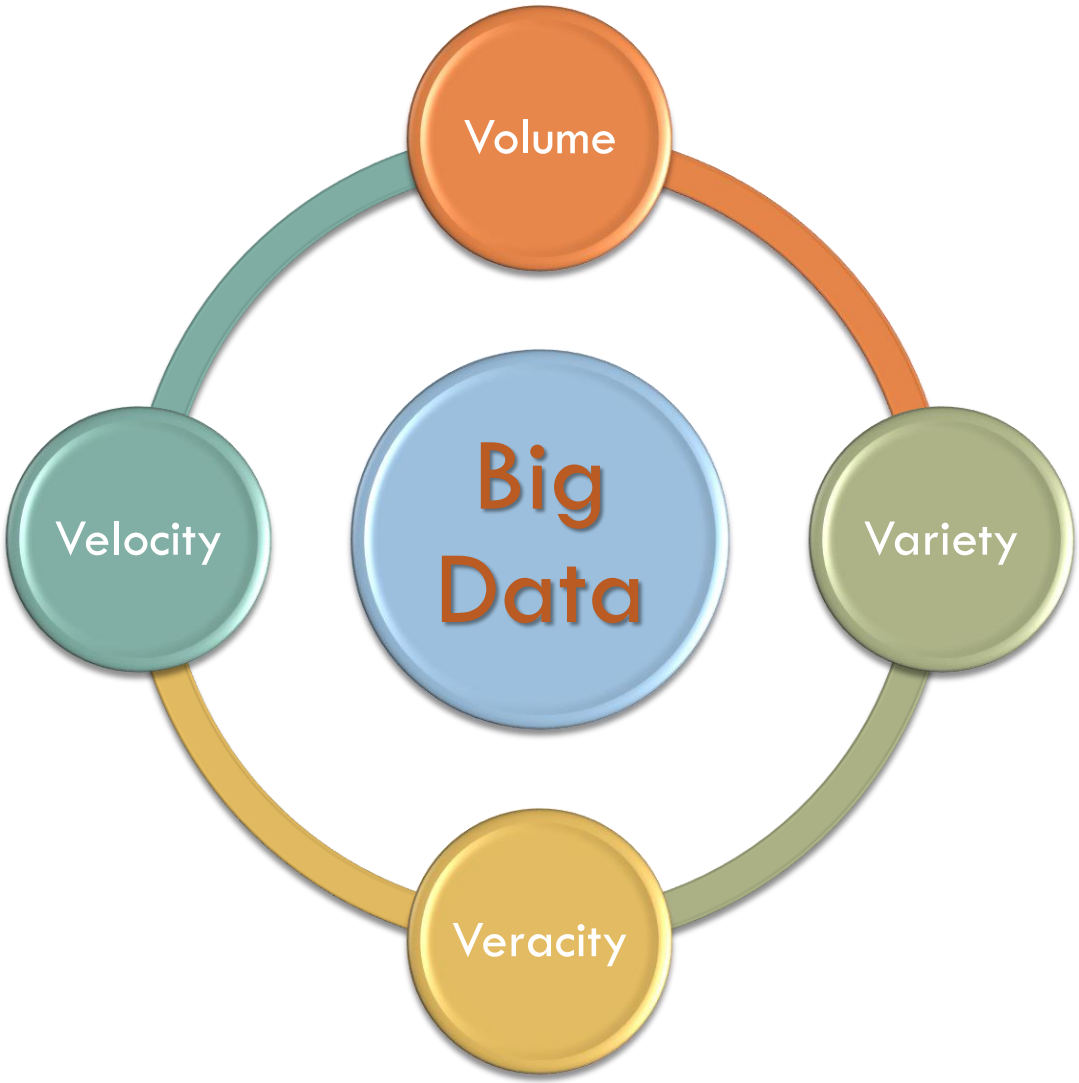
Data Mining: Cultures

- **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



Slide from J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Big Data – The 4 Vs



Big Data: Volume

- 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

- 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 to maximize its value.

Big Data: Variety

- 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

- 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?

Implications of Big Data

- Hardware struggles to keep up with data growth
  - Big Data platforms enable scalable computation
  - However, scaling hardware alone can be impractical or insufficient
- Can we scale our algorithms instead?
  - Yes, but it requires trade-offs in efficiency & accuracy

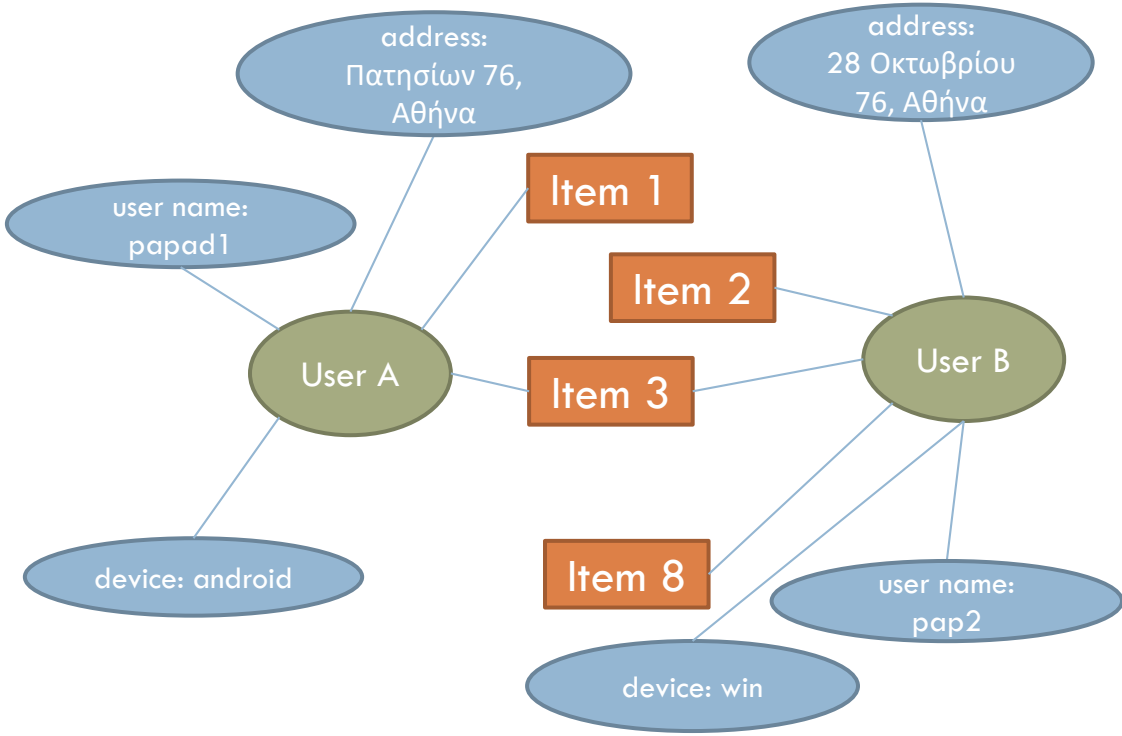
Running Example

- Assume we are a small start-up company with a few hundred users
- Management aims to increase profits by analyzing user profiles and purchase behaviours
- **A simple analytical objective:** identify all pairs of similar customers based on their data.
- What insights can be derived from this analysis?
  - Customer segmentation and clustering
  - Personalized recommendations

Representing Customer Profile & Activity Data

- Customer data can be modelled in multiple ways
  - **Graph models:** representing customers as nodes and their relationships or interactions as edges.
  - **Relational models:** storing customer attributes and transactions in structured tables.
  - **Vector representations:** encoding customer features as numerical vectors for machine learning applications.

Property Graph Data Model

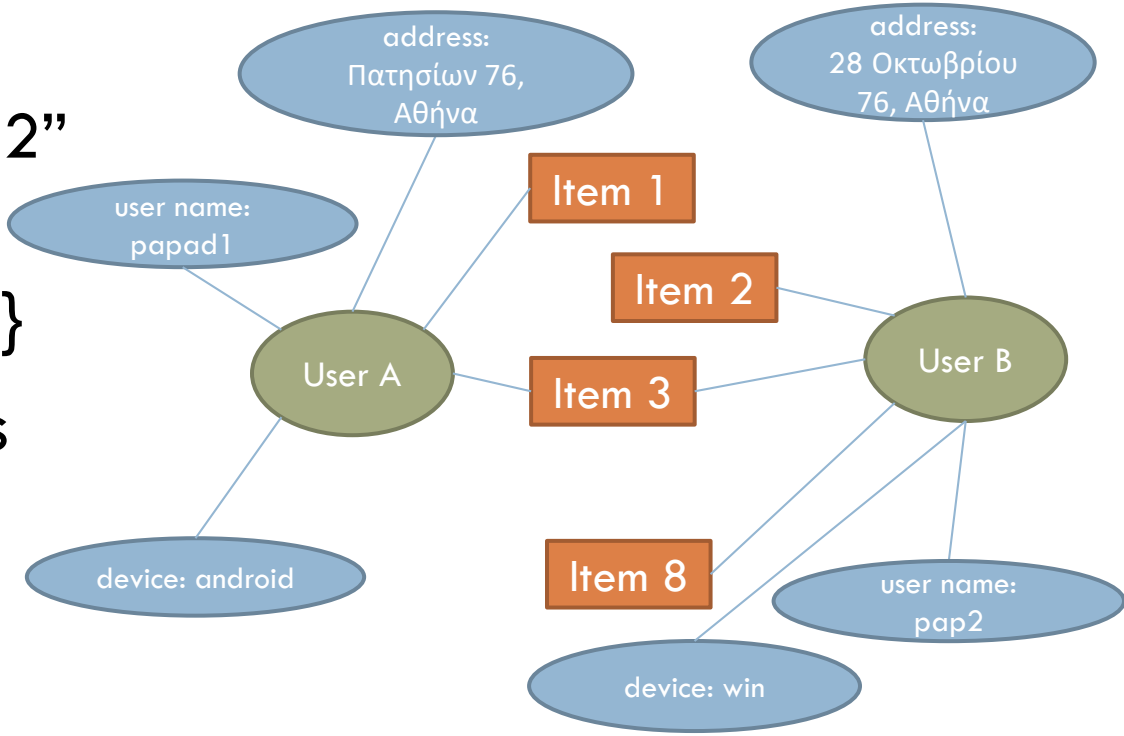


Relational/tabular Data Model

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

How to derive similar pairs?

- Compare attribute values
  - "papad1" vs "pap2"
  - {item1,item3} vs {item2,item3,item8}
- Vector comparisons
  - Cosine similarity?
  - Euclidian distance?



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

How to derive similar pairs?

- "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

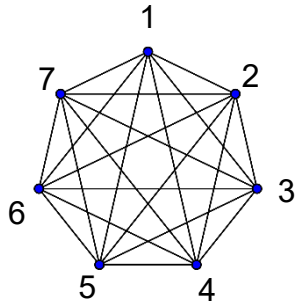
Οκτωβρίου 76, Αθήνα"

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**

- 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, our company customer base gets 100 times larger!
  - How long will in take for the same task to compute?



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

Scale-up versus Scale-out

- 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

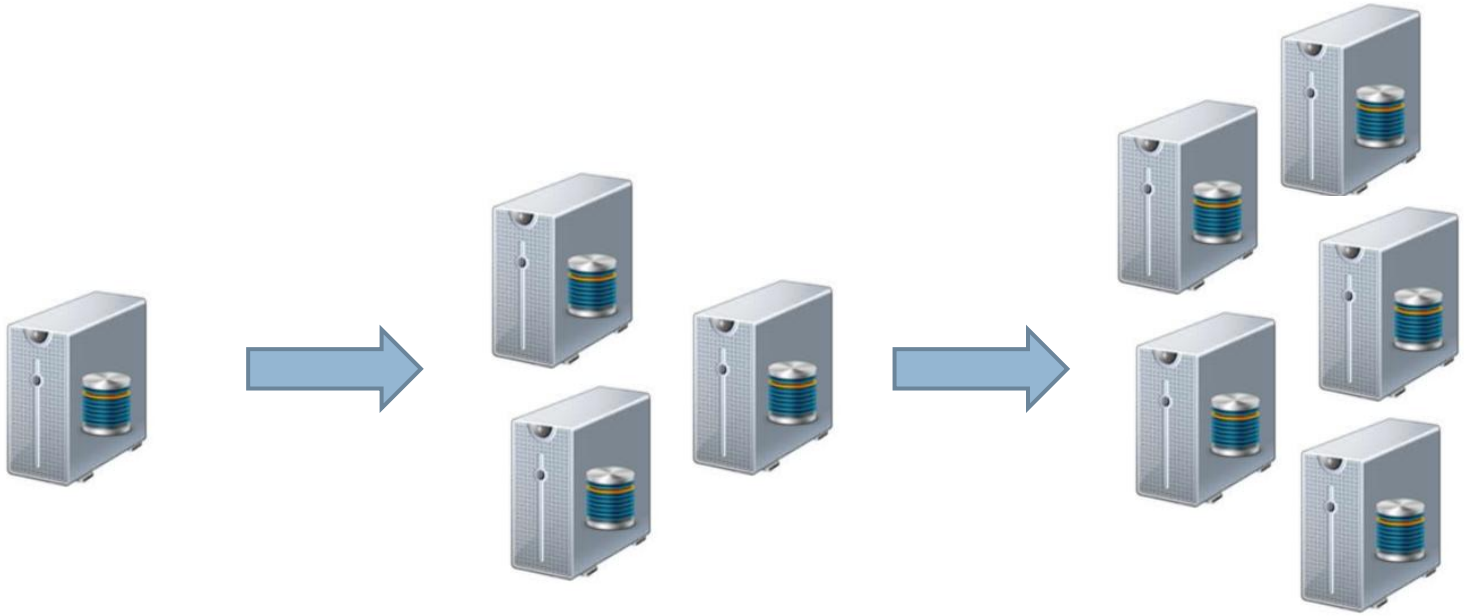
- In cloud computing environments this translates to moving up to larger, more powerful instances

AWS Compute Optimized instances

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

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

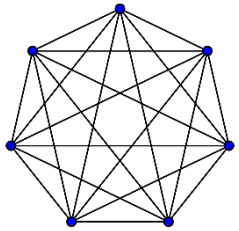


How to proceed?

- Scale-up / scale-out at best result in linear reduction of running time
- But scaling x 10000 times also increases cost by (roughly) x10000!

Recall complexity of pair-wise computations

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



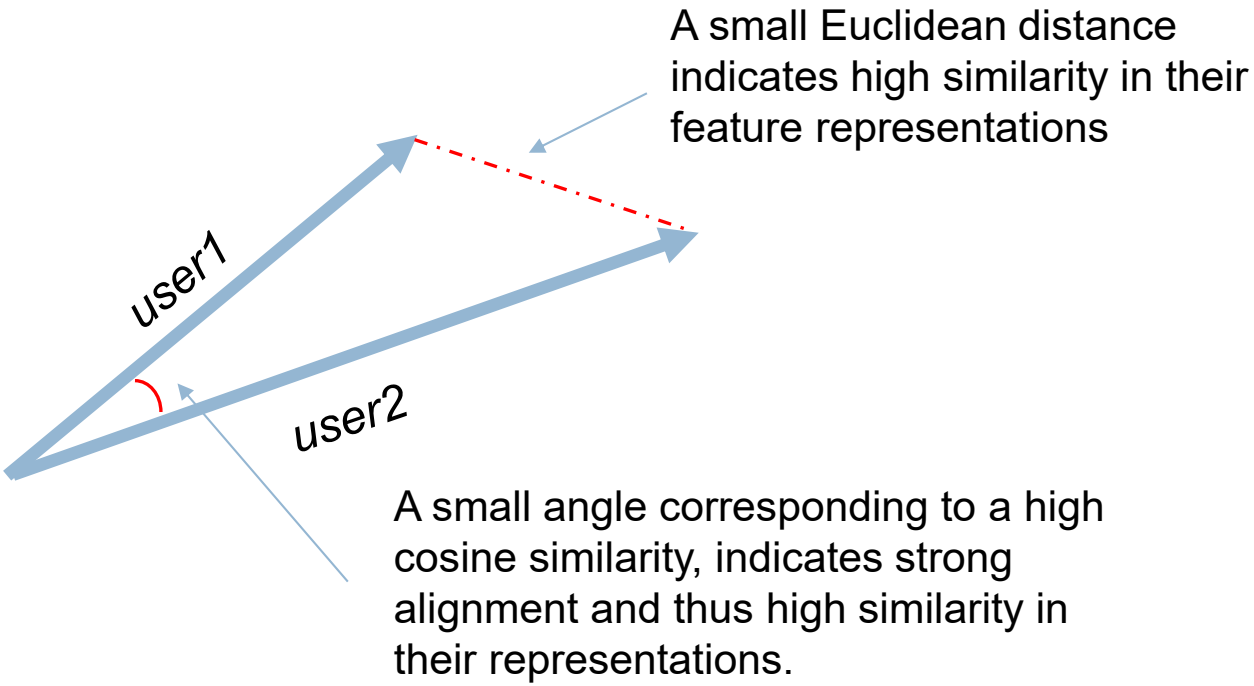
- For a Nearest-Neighbor query NN-$k(q)$ we need to compute $n-1$ distances (using q as a source)
- How do we scale these queries?
  - With the use of indexes!
    - Hierarchical Navigable Small Words (HNSW)
    - Locality Sensitive Hashing (LSH), discussed next

How to scale similarity calculations?

- A **similarity index** is required to support pairwise evaluation.
  - If *user1* is highly similar to *user2*, the index should suggest this pair. Failing to do so would result in a **false negative**.
  - Conversely, if *user1* is highly dissimilar to *user2*, the index should exclude this pair. Otherwise, it would lead to a **false positive**.
- In essence, we aim to avoid unnecessary calls to the sim() function, while still maintaining **high accuracy**.

How to compare vectors?

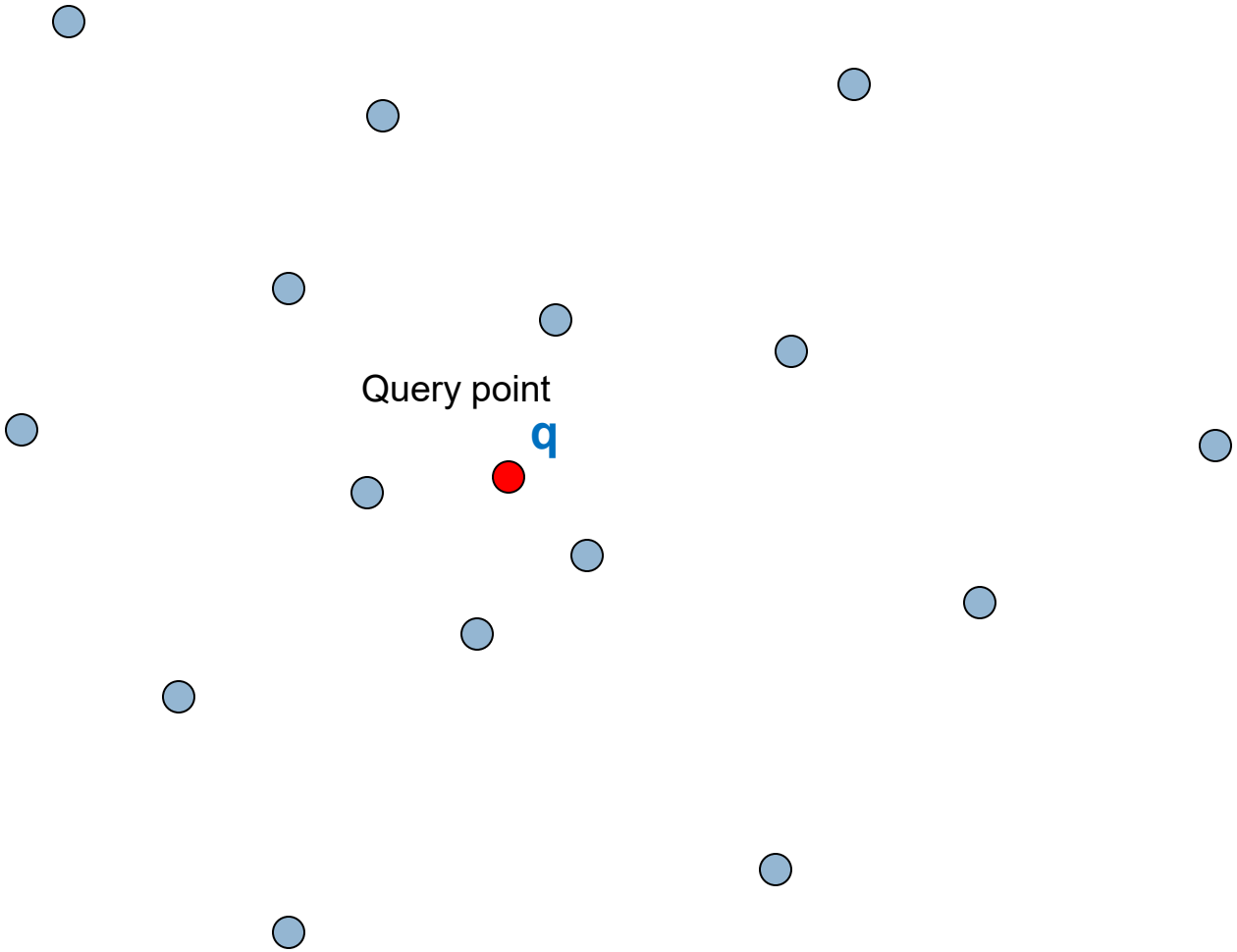
- Euclidean distance
- Cosine similarity



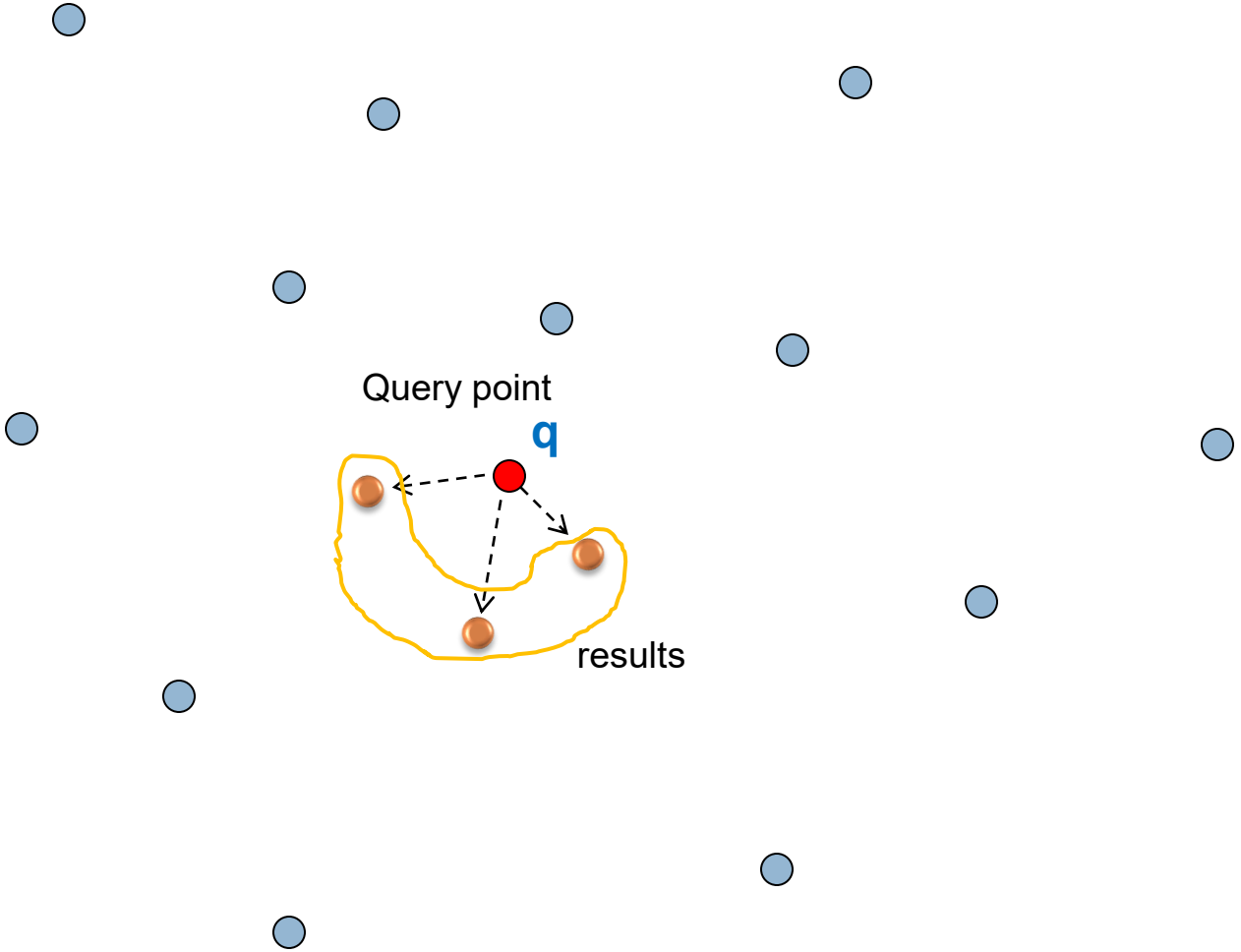
Nearest Neighbor Query Example

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

Nearest Neighbor Query

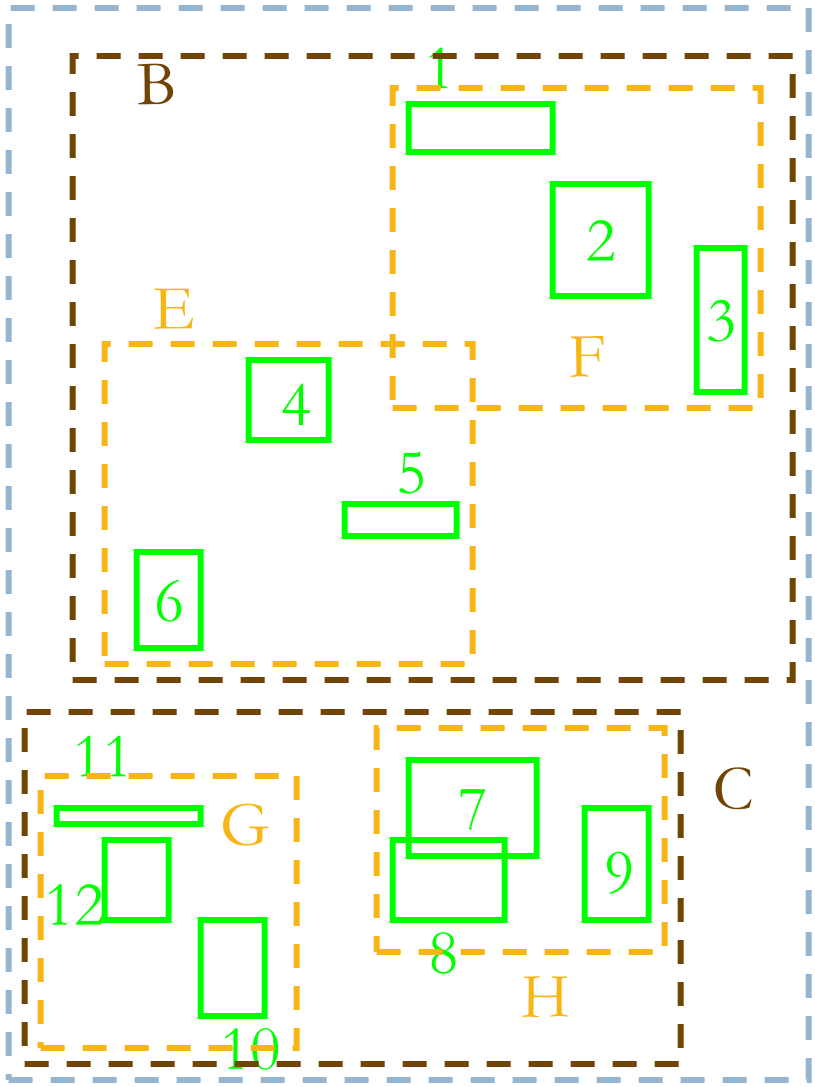


Nearest Neighbor Query



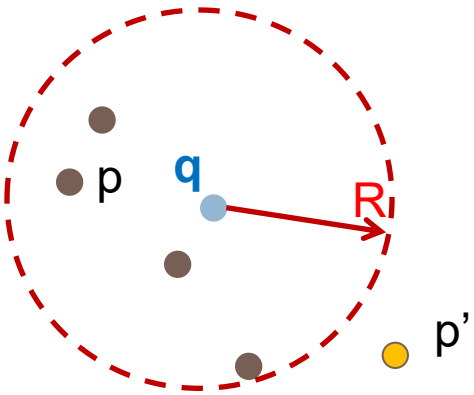
Dimensionality Curse

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



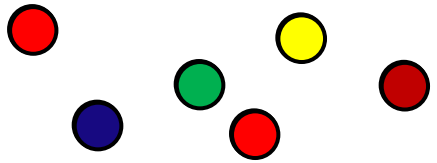
Approximation to the rescue

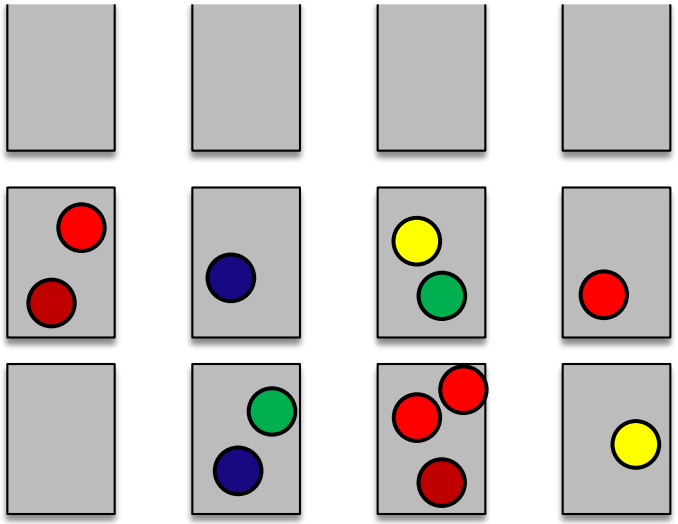
- Approximate range query: build index data structure which
  - If there is a point p: dist(p,q)≤R, p is returned with high probability
    - i.e. high probability of not missing a true NN = few false negatives
  - If dist(p',q)>R, the probability of returning p' is small
    - Most results are true NN = few false positives
      - False positives may be pruned in a second step



Locality Sensitive Hashing (LSH)

- Assign items to **buckets** using a hash function **h(x)**
  - E.g. h(●)=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

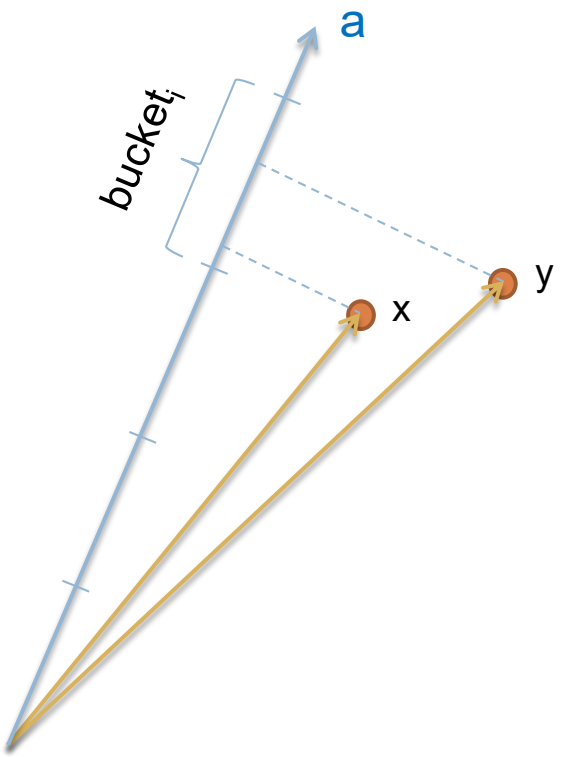




Buckets (0-3)

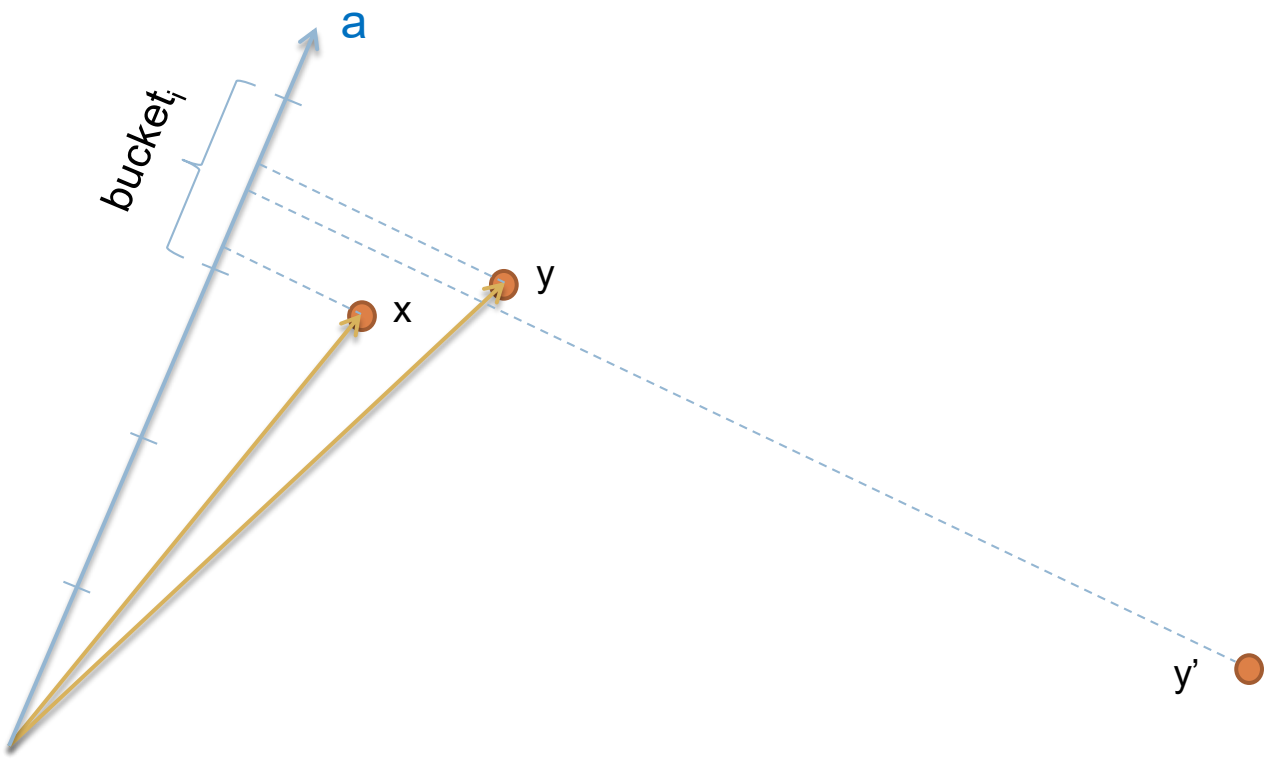
Intuition of a hash function that preserves the Euclidian Distance

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



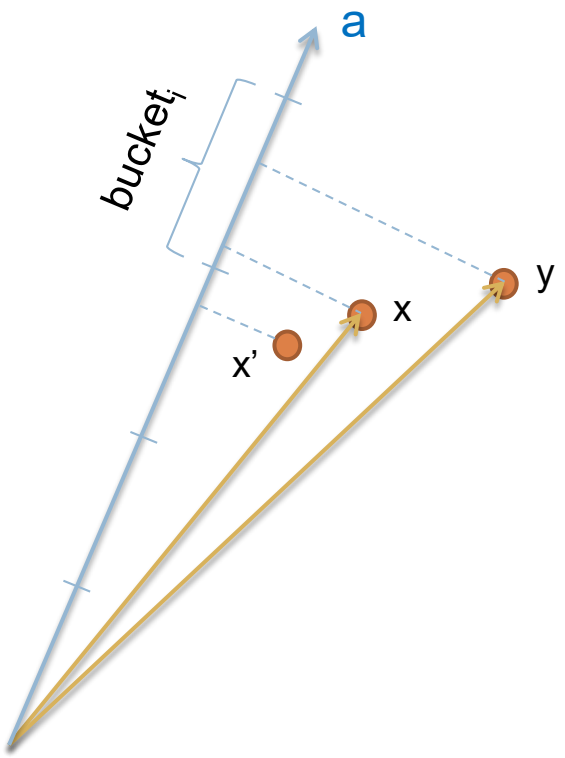
Recall : length of projection of vector x onto a is the inner product a.x

False positives



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

False negatives



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

More on LSH

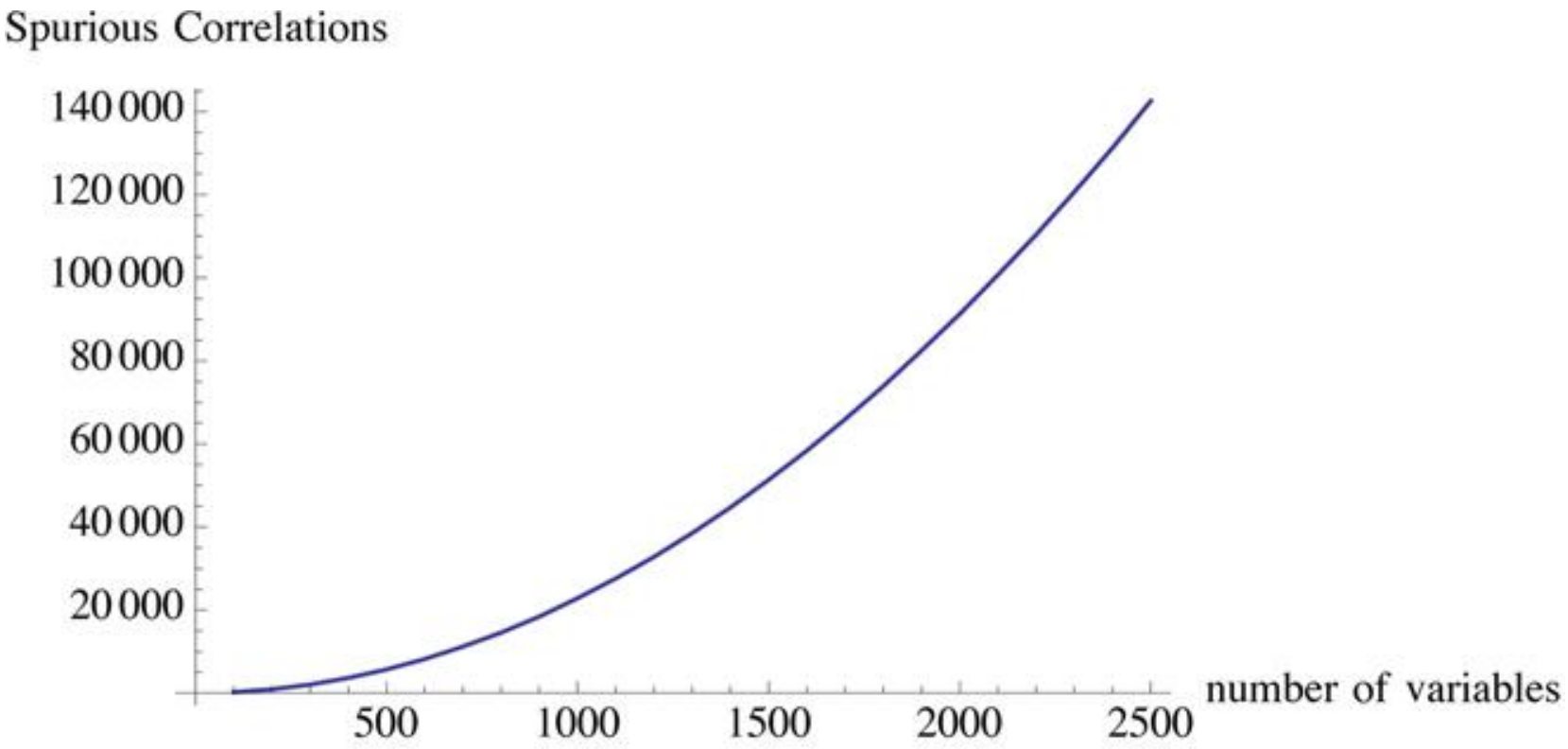
- Families of hash function exist for several popular distance (similarity) metrics
  - Euclidian, Manhattan, Jaccard
- Hashing on multiple hash tables is parallelizable

  …and map-reduce-able!

More implications of 4Vs in data analysis

- Big data → big noise → big errors?

Beware the Big Errors of 'Big Data' (Nassim N. Taleb)



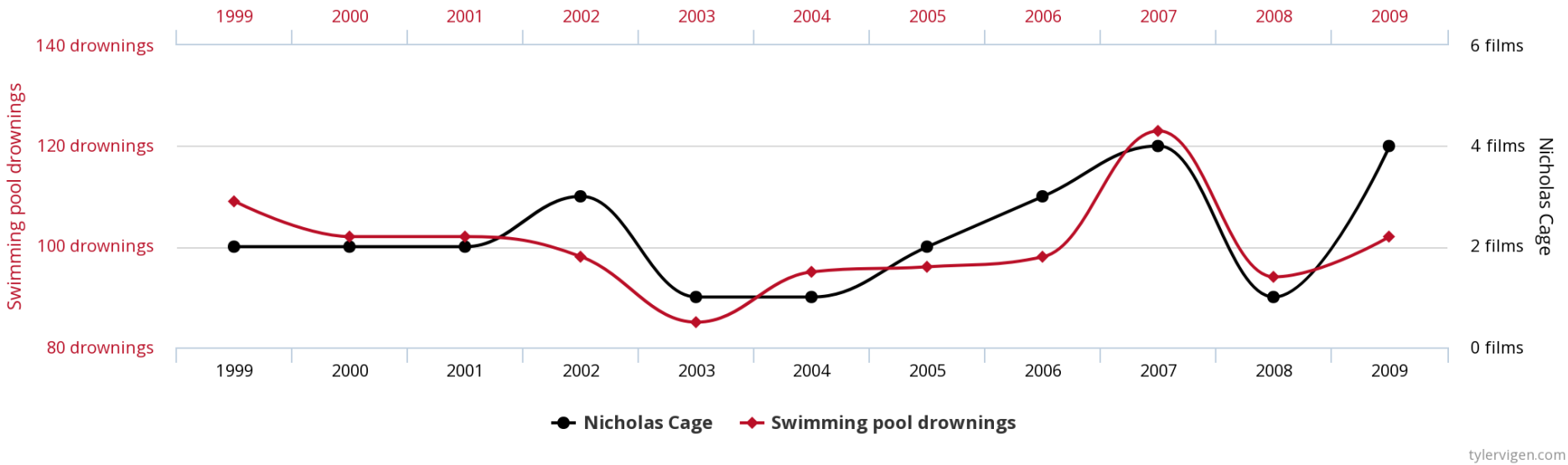
Spurious Correlations

(http://tylervigen.com/)

Number of people who drowned by falling into a pool

correlates with

Films Nicolas Cage appeared in

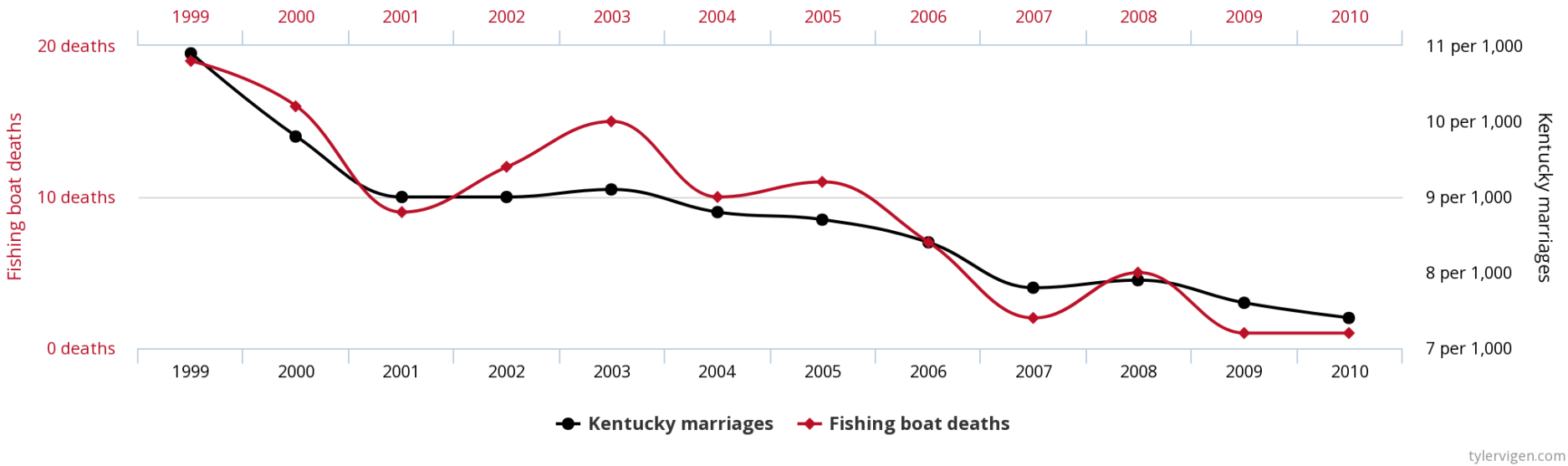


Correlation: 66.7%

Spurious Correlations

(http://tylervigen.com/)

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



Correlation: 95.2%

Rhine Paradox* – (1)

- 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!

Image generated with chatgpt

*example from http://infolab.stanford.edu/~ullman/mmds.html

Rhine Paradox – (2)

- 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)

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

Meaningfulness of Answers

- 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"

- There are1 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

*example from http://www.mmds.org/

Simplifying Assumptions

- 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
  - Loot at booking records of 100K hotels
  - Examine hotel records for 1000 days to find evil-doers

Assume Random Behavior

- 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

- Probability of observing an event on random data is very small: 0.00000000000000001
- Thus, if we see one such pair of people it should be investigated, right?

The effects of big data in calculations

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

$$\binom{n}{2} \approx \frac{n^2}{2} = 5*10^{17}$$ 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

- $(5*10^{17}) * (5*10^{5}) * 10^{-18} = 250000$ events

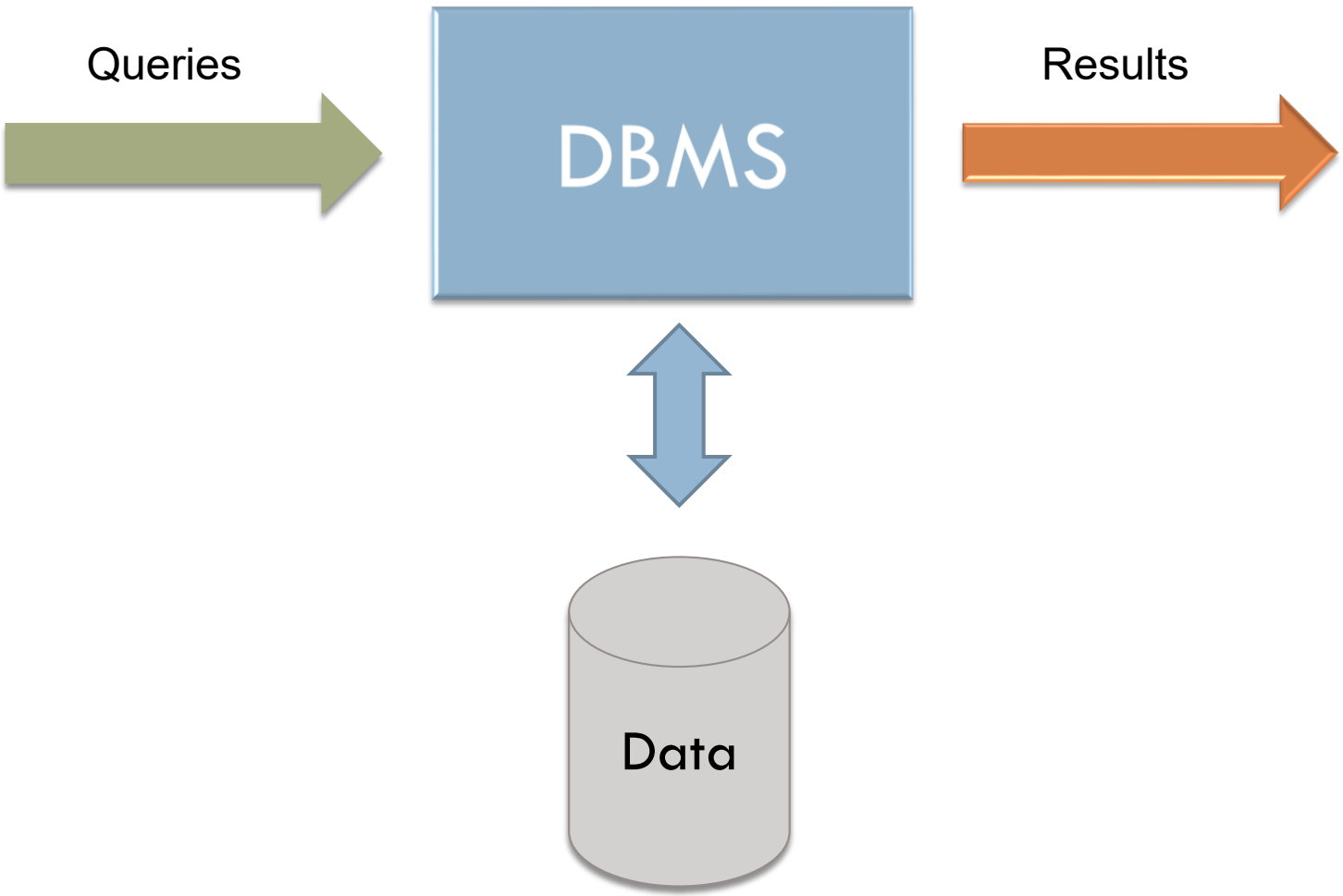
Pairs of people — Pairs of days — Probability of an event

- 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 only100 real evil-doers
  - Is it feasible? Does it justify the intrusion on people's lives?

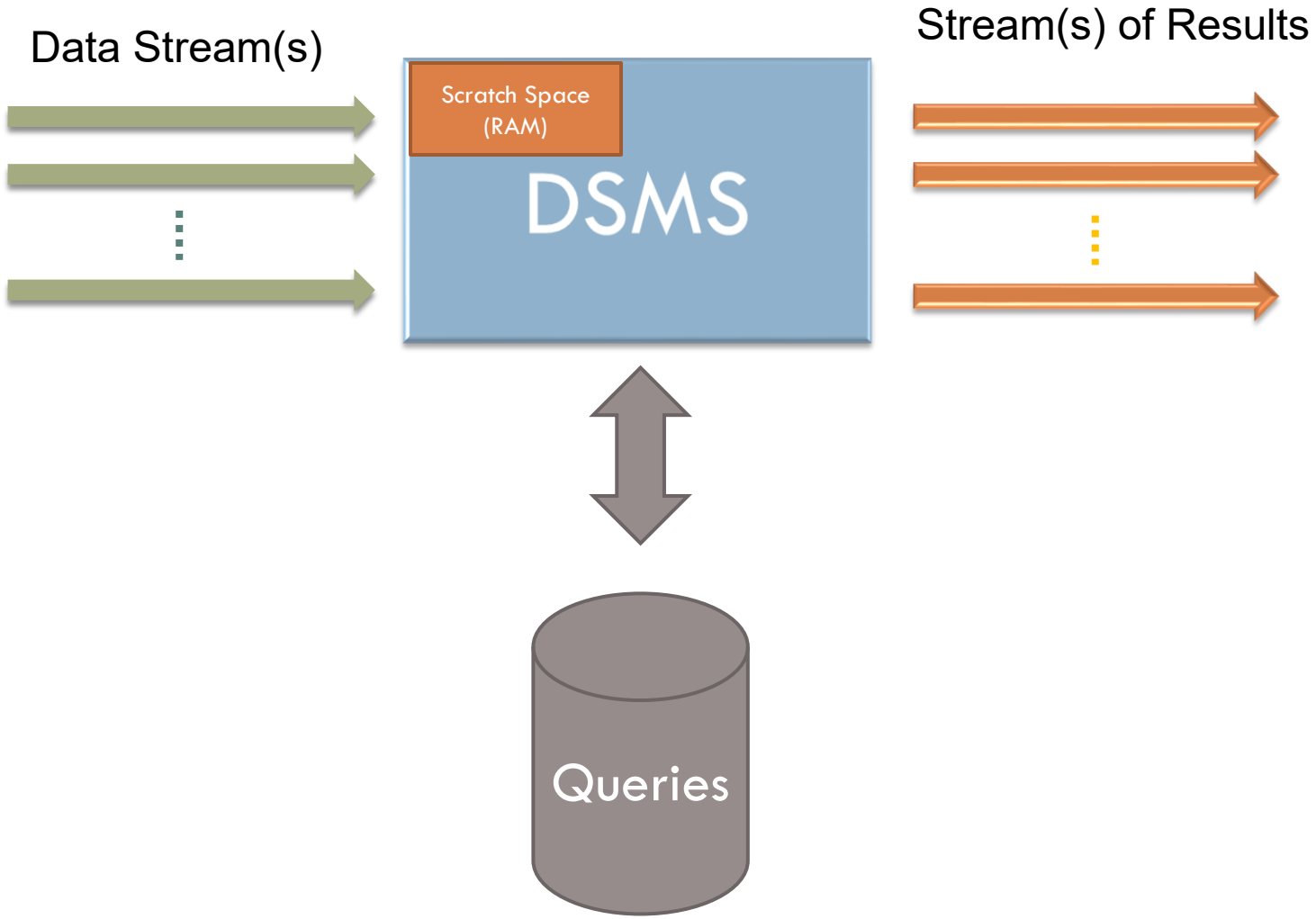
Big Data: Velocity

Transitioning from Batch to Real-Time Stream Processing

Traditional Data Processing

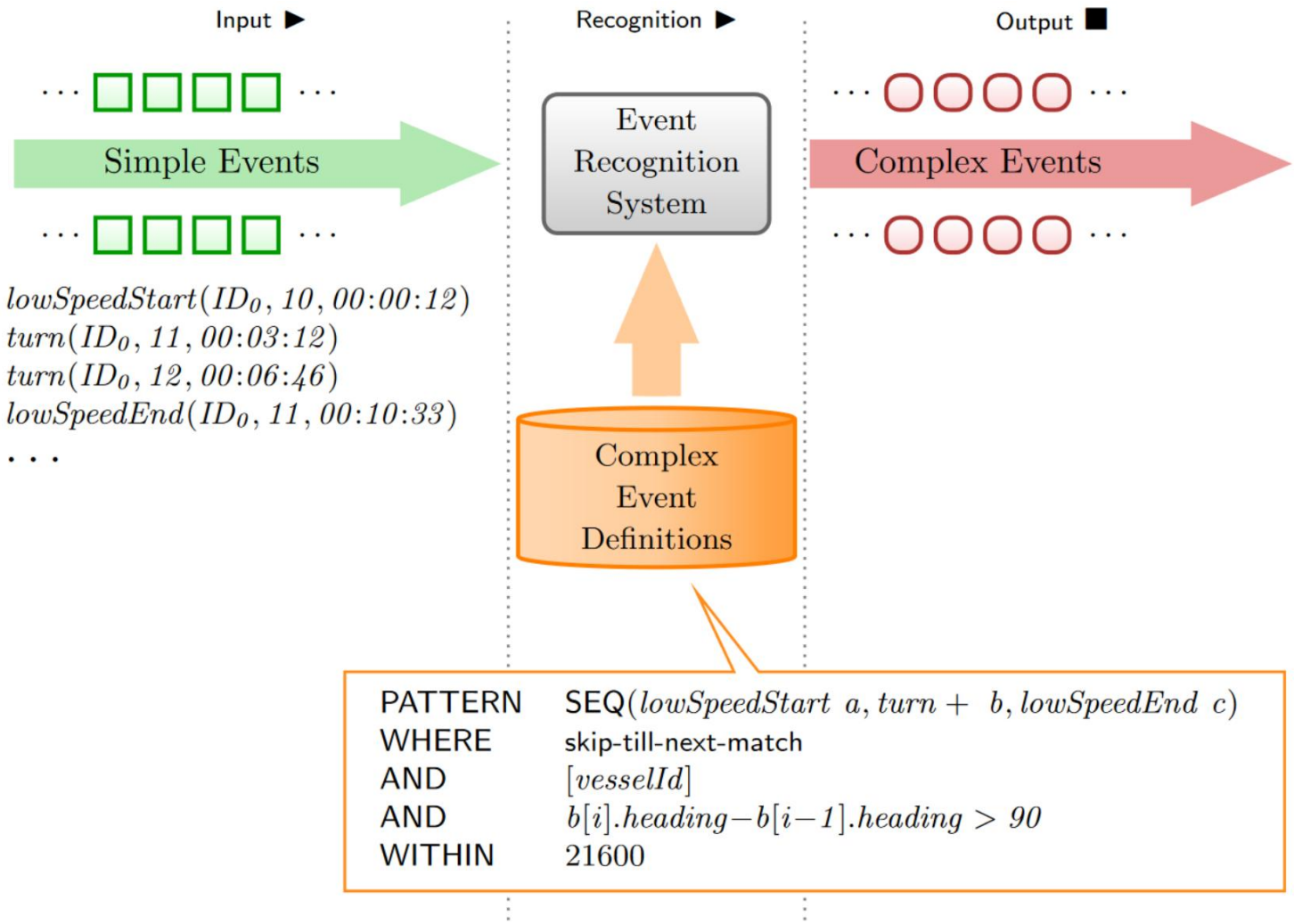


Stream (real-time) Processing



Related topic: Complex Event Processing

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



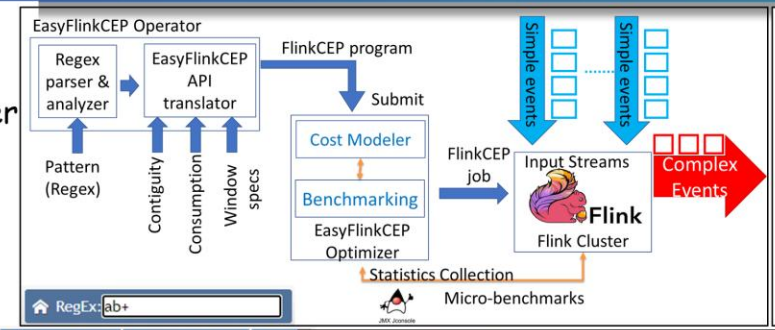
Nikos Giatrakos, Eleni Kougioumtzi, Antonios Kontaxakis,
Antonios Deligiannakis, Yannis Kotidis

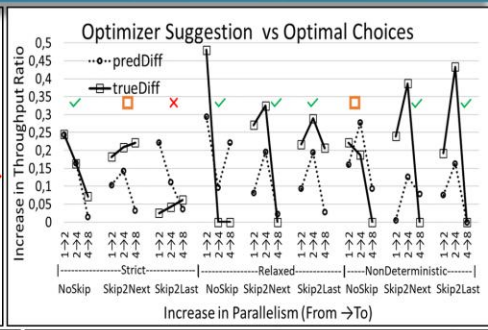
ATHENA

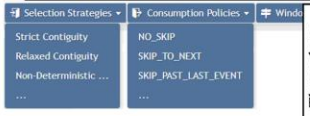
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

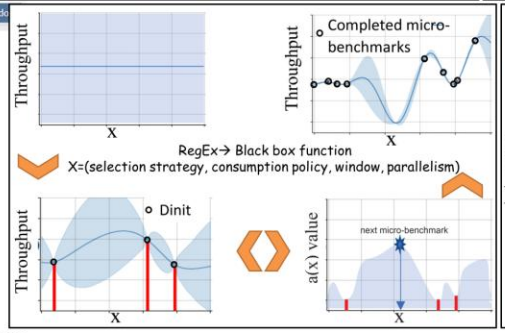
EasyFlinkCEP: Big Event Data Analytics for Everyone

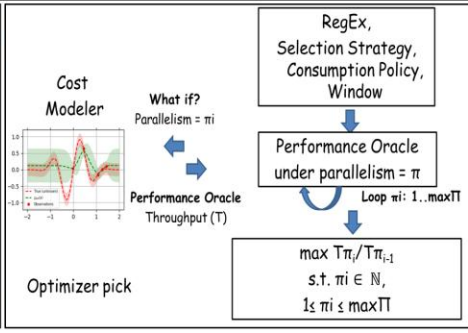
- Complex Event Processing (CEP) in FlinkCEP:
  - † Distributed processing over computer clusters
  - † Language of high expressive power
  - × Low level language (Scala/Java)
  - × Cluster admin decisions needed
- EasyFlinkCEP Prototype
  - EasyFlinkCEP Operator
    - No coding, uses Regular Expressions
    - graphical definition of CEP parameters
  - EasyFlinkCEP Optimizer
    - automates FlinkCEP job configurations (parallelism)





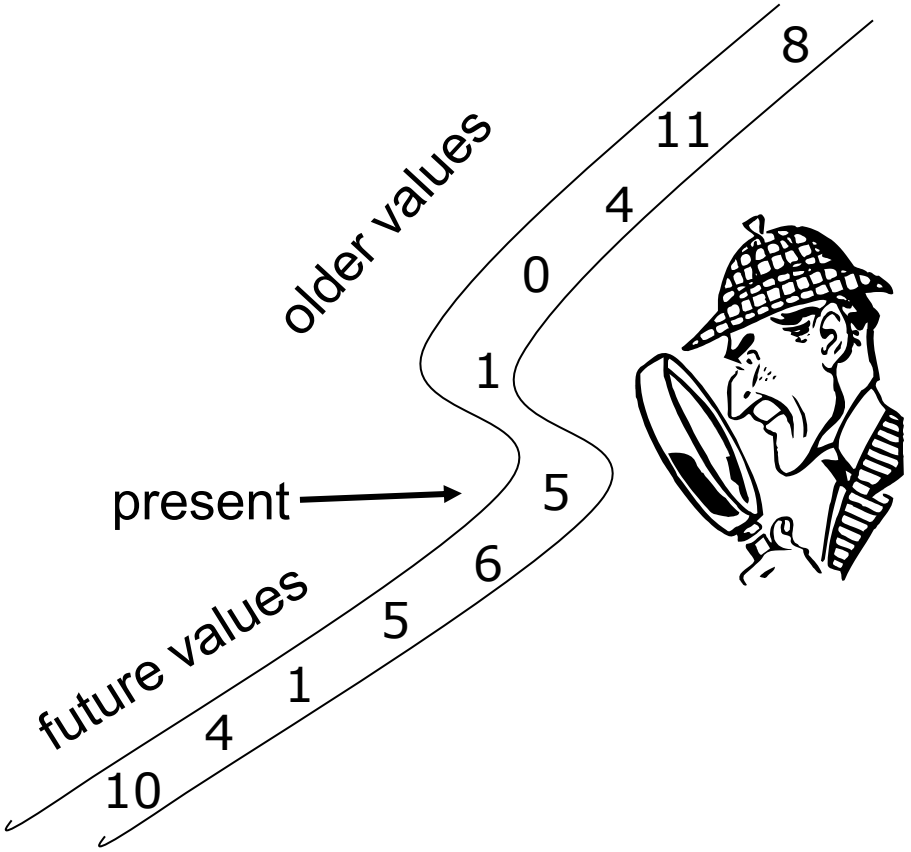






Achievements: (1) Non-programmer event analysts rapidly develop and deploy new CEP pipelines. (2) Exploits the processing capacity of modern hardware without requiring practitioners to make cluster administration decisions.

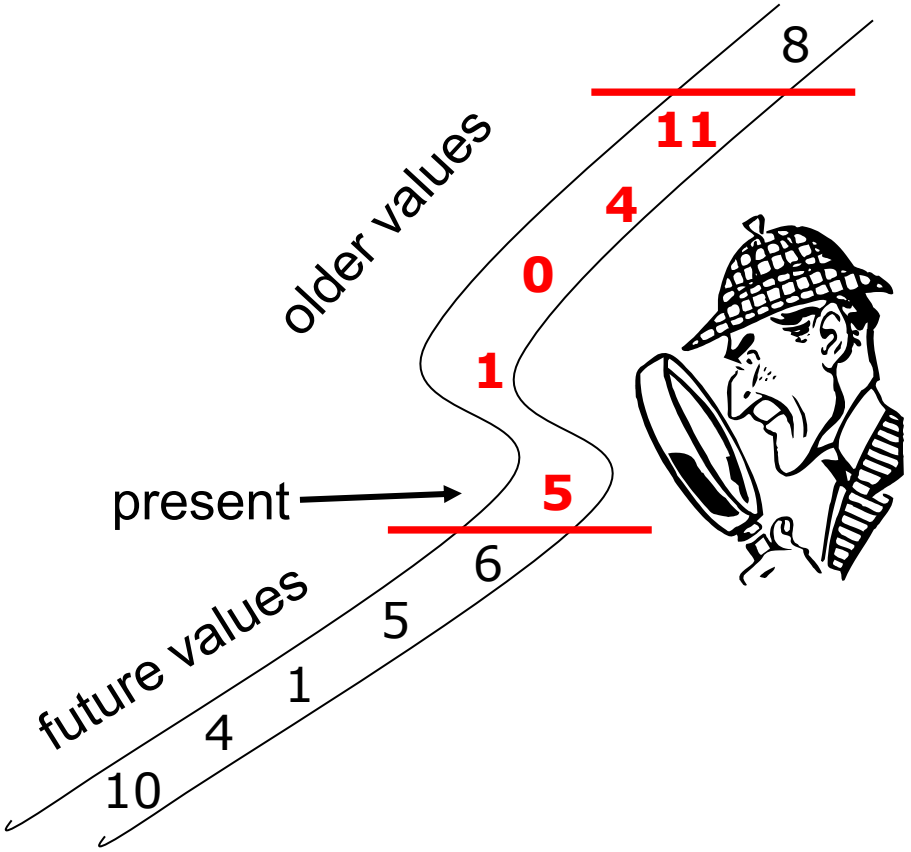
Stream processing: exact computations over infinite data streams



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 *continuous* queries
  - Number of items seen
  - Maximum/minimum value in a stream?
  - Average value?
  - Median value?
  - Most frequent value?
  - Number of distinct items?

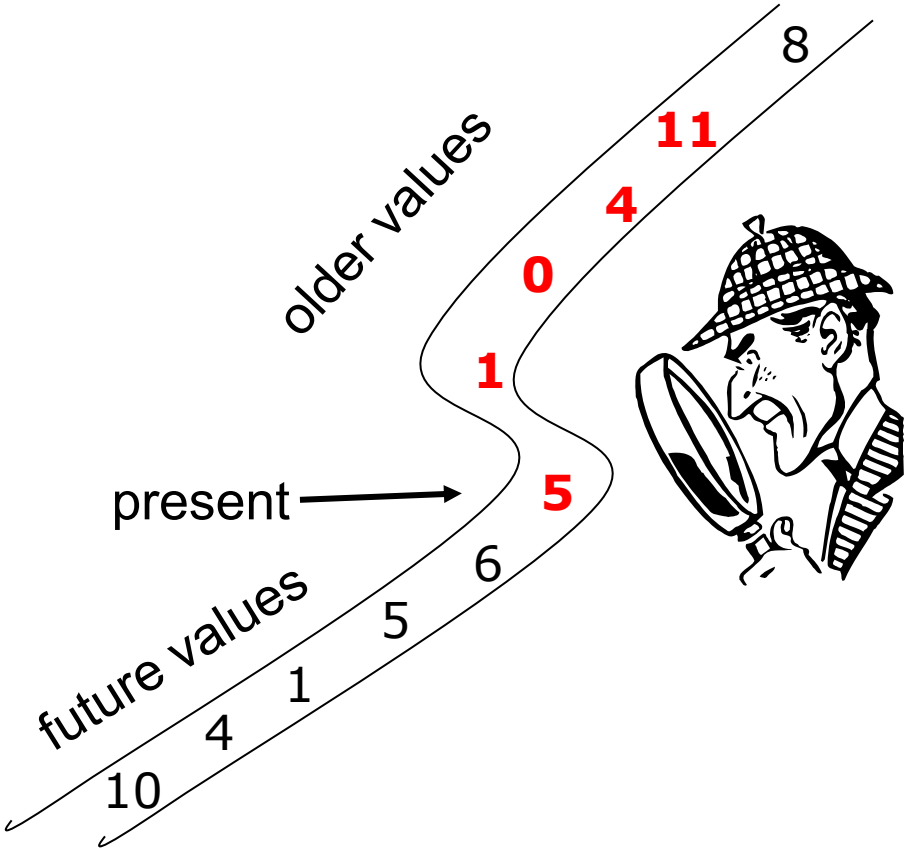
Windows: used to segment a continuous data stream into batches of tuples



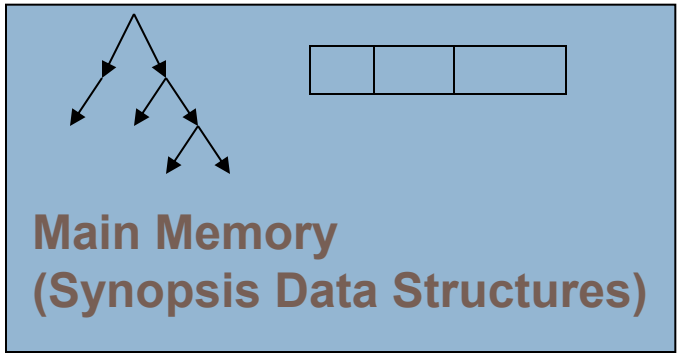
Data Stream

- 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

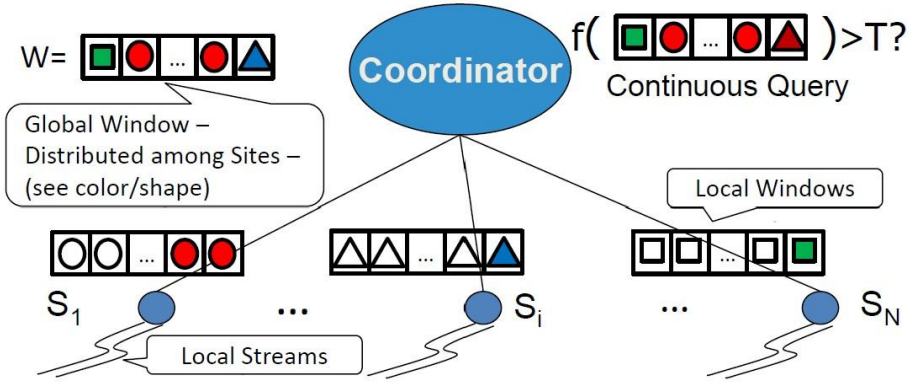


Data Stream

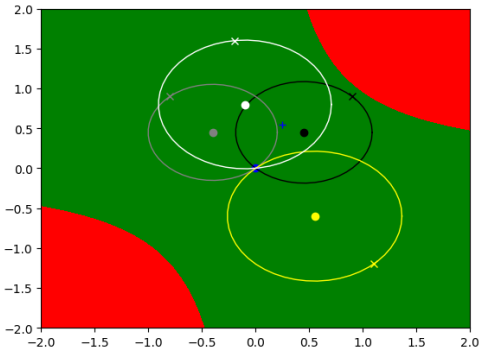


- Maintain synopses over the data and use them at query time to produce approximate answers
  - Example: data sketches, discussed next

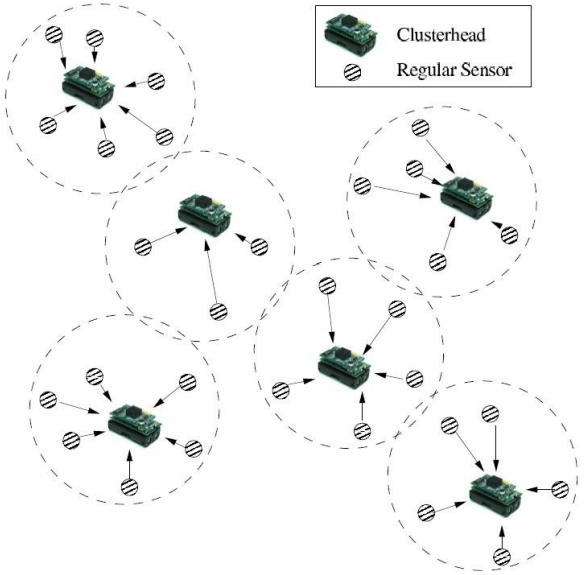
Distributed Stream Processing



Distributed Monitoring Example (by Nikos Giatrakos)



Geometric Monitoring

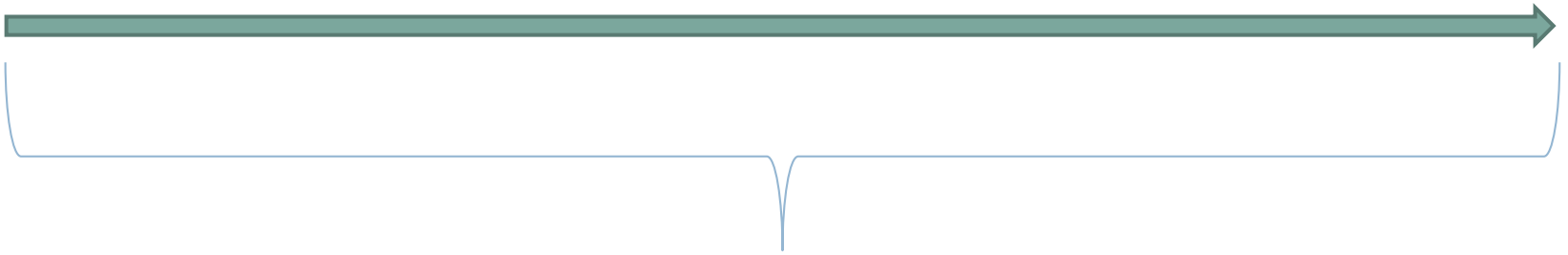


In-network outlier detection (Kotidis et. al)

Stream processing example (1): Most frequent items

- 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

stream 1

$m_a = 5$

$m_b = 4$

$m_c = 3$

$m_d = 3$

S=2nd moment = 25+16+9+9=59

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

stream 2

$m_a = 12$

$m_b = 1$

$m_c = 1$

$m_d = 1$

S=2nd moment = 144+1+1+1=147

S is a measure of how uneven the distribution is

2nd moment estimation caveat: memory

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

stream 1

$m_a = 5$

$m_b = 4$

$m_c = 3$

$m_d = 3$

S=2nd moment = 25+16+9+9=59

For a stream with M distinct items, we need O(M) memory!

Another Streaming Application

- 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

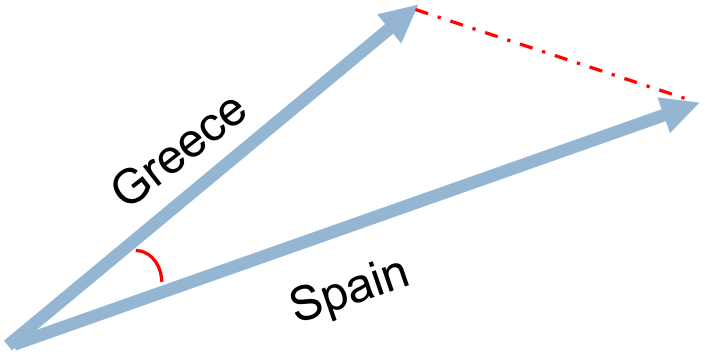
"Bag" of hashtags for Greece

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

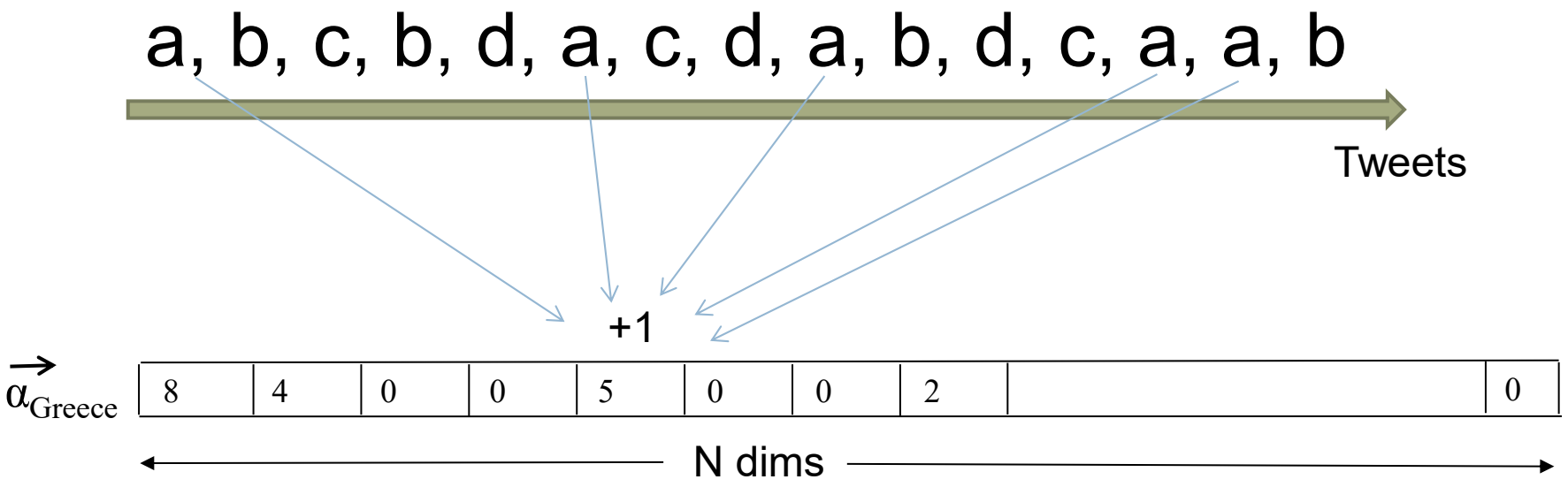
Compute frequencies

A simple idea (just to keep going)

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



Frequency vector for Greece



- Each hashtag (string) in 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

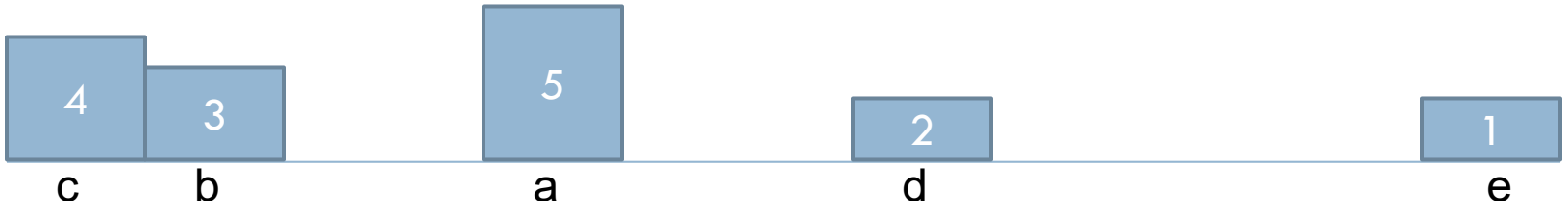
Time

How I chose to represent this information in my program:

$\vec{\alpha}_{\text{Greece}}$

4	3	0	0	5	0	0	2		1
c	b			a			d		e

What I am actually capturing with the above representation:



Note

- This representation is convenient 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("#paobc")=7

Slot 0 Slot 1 Slot 7

$\vec{\alpha}_{Greece}$

8	4	0	0	5	0	0	2		0

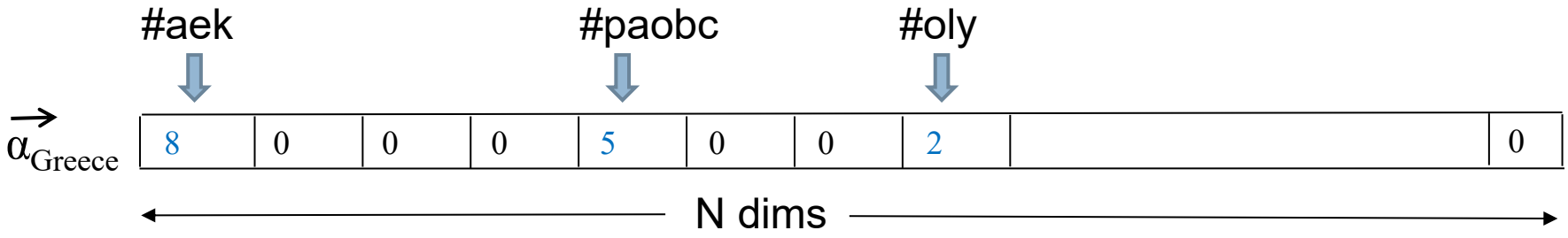
N dims

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

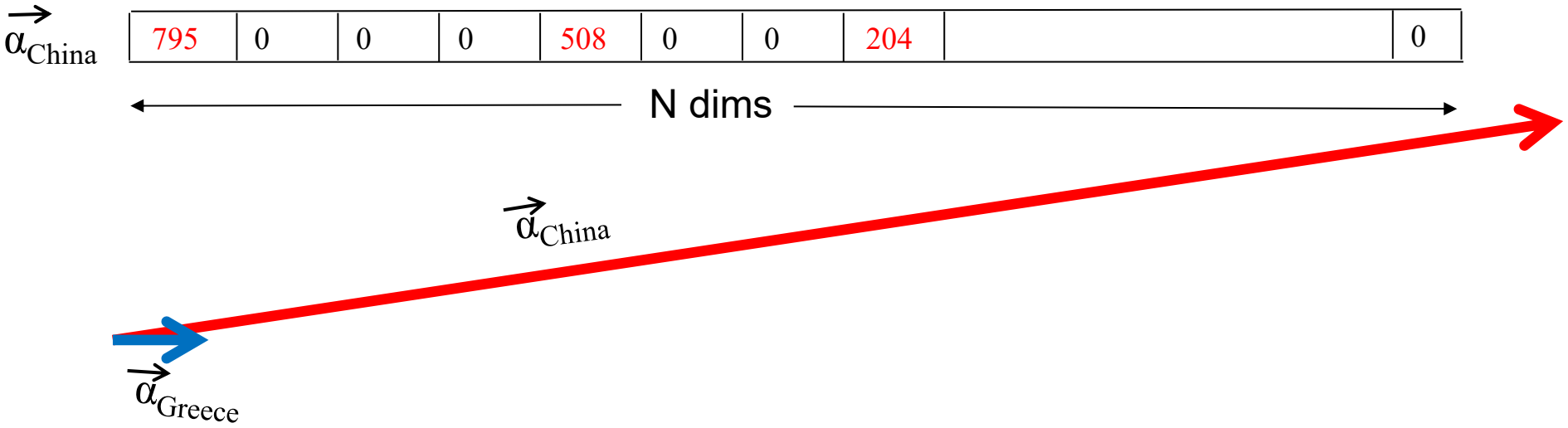
- In this example key values are the different hashtags (that we do not know in advance)
- We need a function h(s) that maps a string s to an integer (next lecture)

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

- Vector for Greece

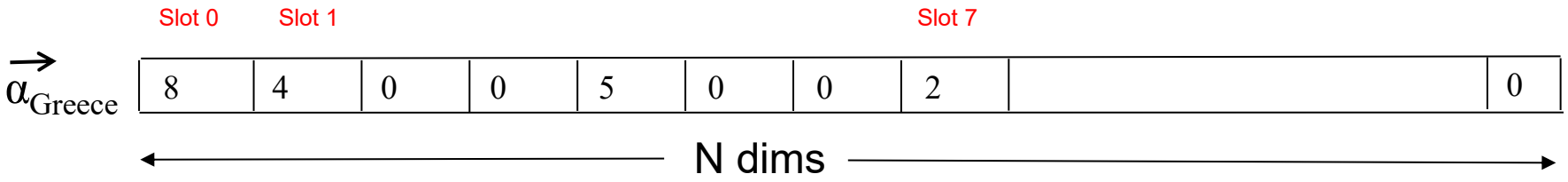


- Vector for China (not likely but for the shake of this example…)



Ooops!

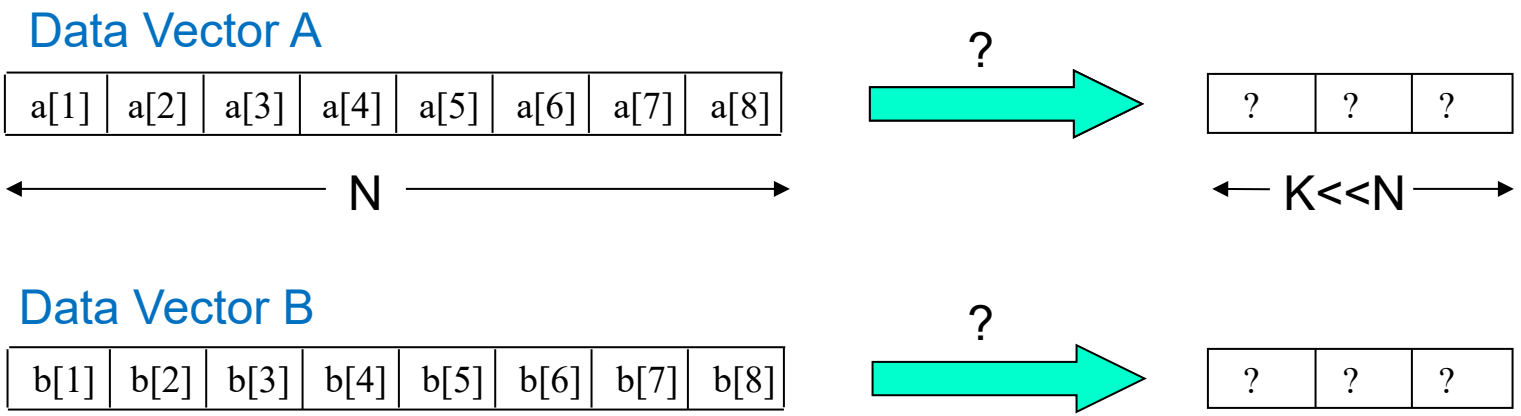
- In order to avoid *collisions*, I chose a large number for N (e.g. $N=2^{32} = 4294967296$)
  - Thus, hash values are 4-byte integers
- Each slot holds a 4-byte integer counter (freq)
- Thus, each vector occupies 16GB of RAM if stored "uncompressed"



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

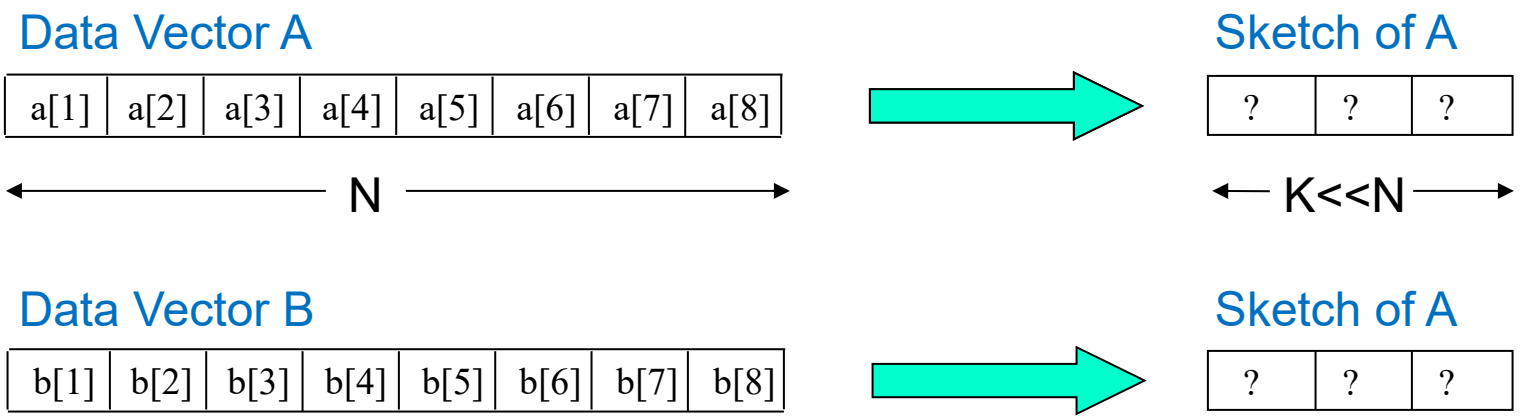
Data sketches

- 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



Challenge

- Is it possible to compute these low-dimensional sketches by relying solely on data updates, without needing the original high-dimensional data vectors?

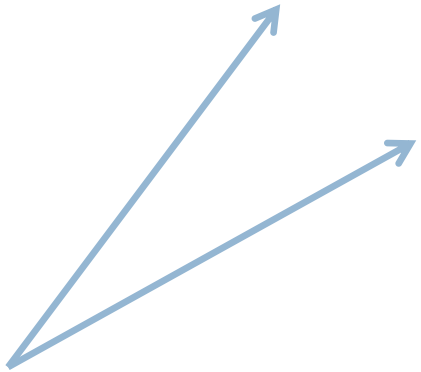


Dot (inner) product between two vectors

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

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

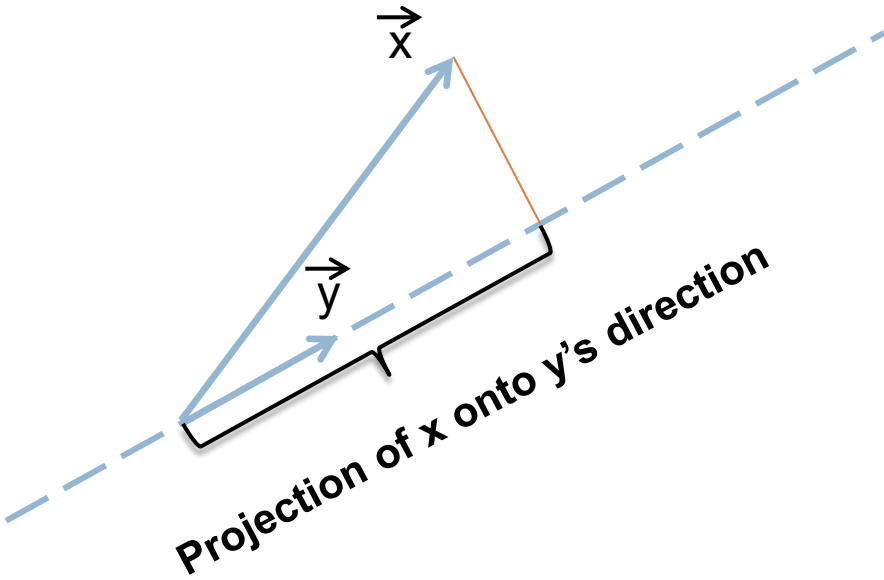
$\vec{x} = (1,3,0,5)$

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

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

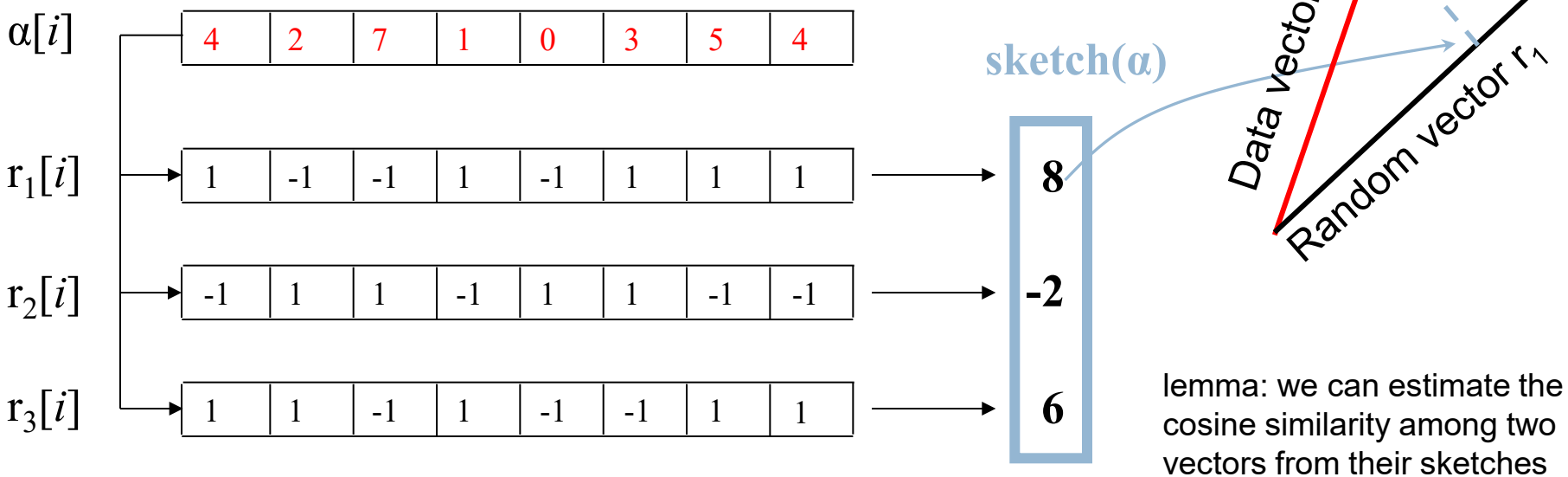
$\vec{x} \cdot \vec{y} = 1/2+3/2+5/2 = 9/2$

$= |\vec{x}| * 1 * \cos(\theta(\vec{x}, \vec{y}))$



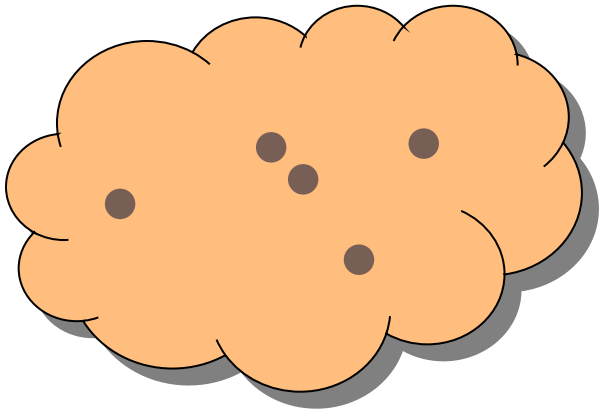
Linear Projections (AMS Sketches)

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

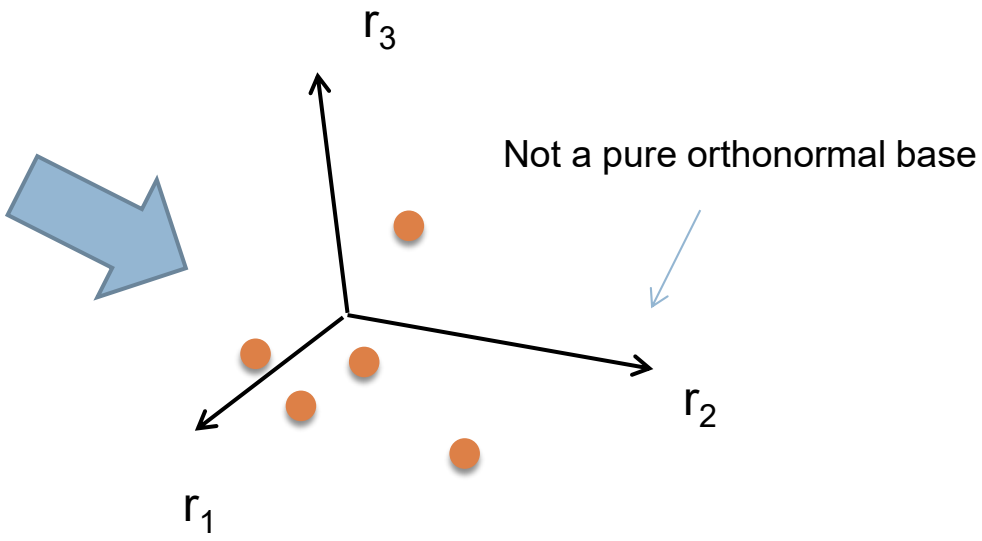


lemma: we can estimate the cosine similarity among two vectors from their sketches

Random Projections



High-dimensional space

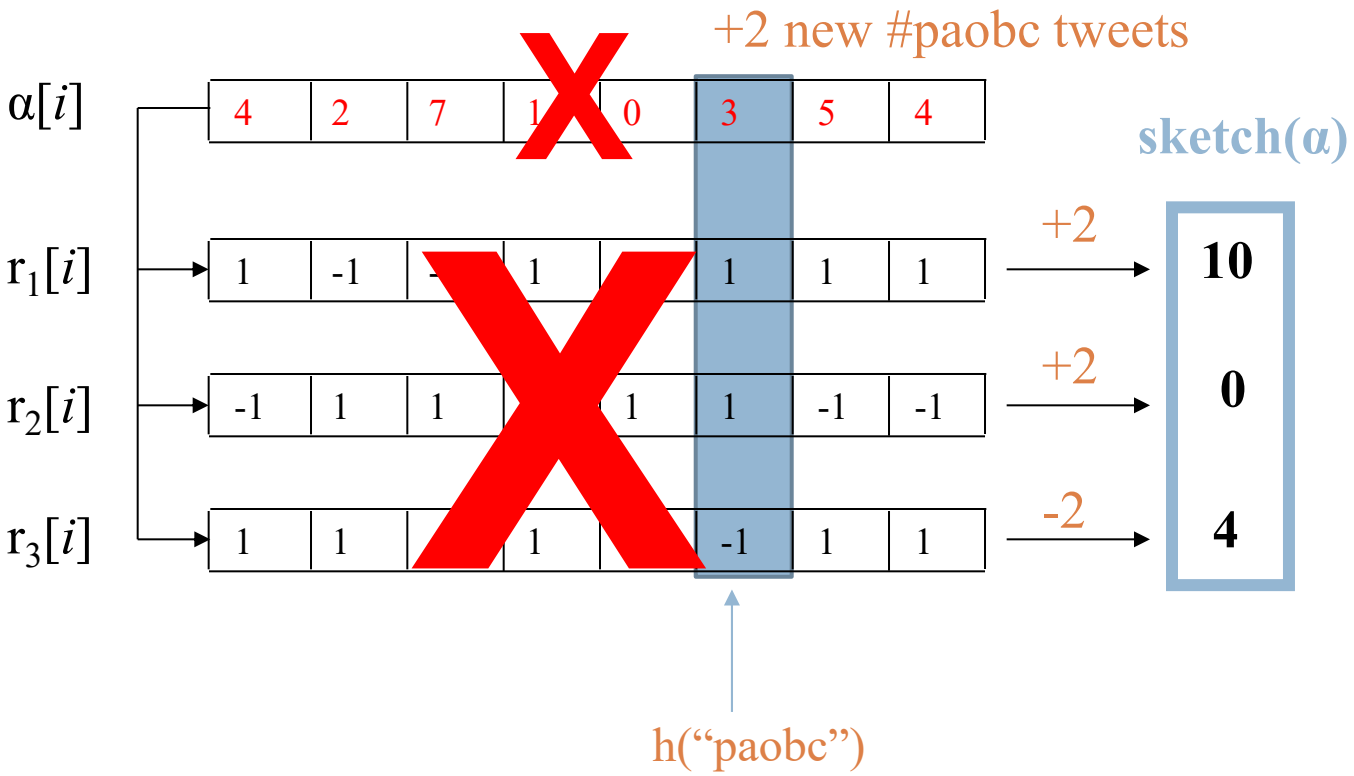


What can we do with them

- Estimate norms, distances, dot products with (ε,δ)-guarantees
  - ε: approximation error
  - δ: 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

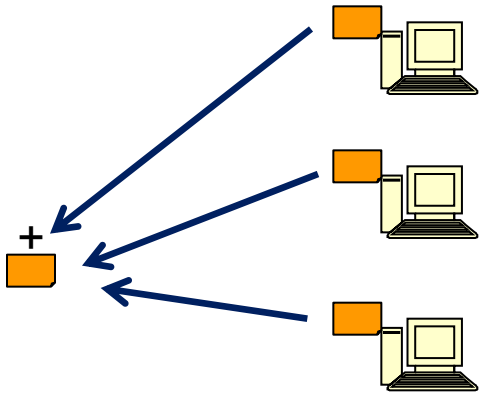
Updatability (addresses velocity)

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



Sketches: Recap

- 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

Personal View

- 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

- 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

Word Cloud from today's lecture

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

- Noga Alon, Yossi Matias, Mario Szegedy: The Space Complexity of Approximating the Frequency Moments. STOC 1996: 20-29.
- G Cormode, S Muthukrishnan: An improved data stream summary: the count-min sketch and its applications. Journal of Algorithms 55 (1), 58-75
- 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.
- 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).
- P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. In Journal of Computer and System Sciences, 31:182–209, 1985.

THANK YOU!