Big Sequence Management

Karima Echihabi

Mohammed V University

Kostas Zoumpatianos

Harvard University &
Université de Paris

Themis Palpanas

Université de Paris &
French University Institute (IUF)

IEEE Symposium on Computers and Communications (ISCC), Rennes (France), July 2020

Acknowledgements

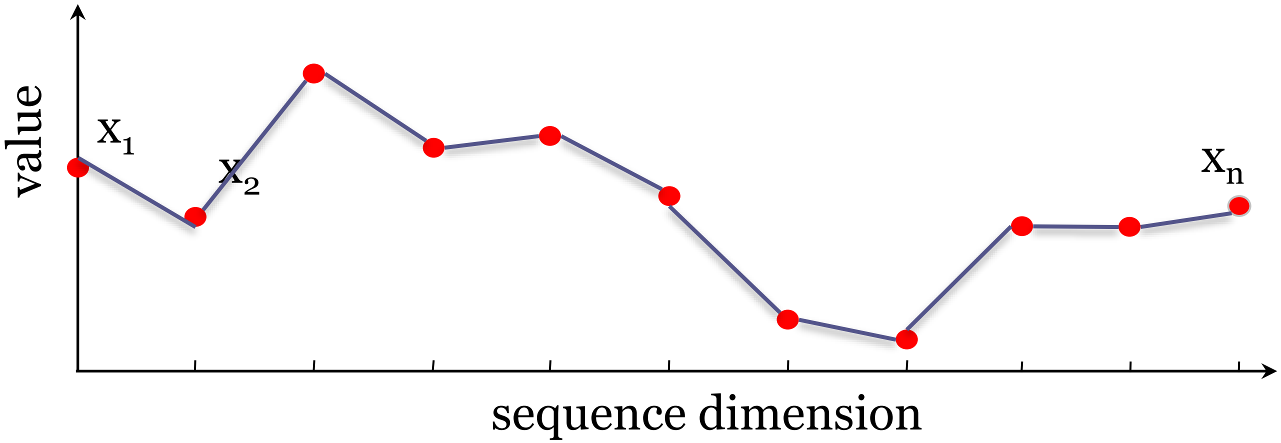
- thanks for slides to
  - Michail Vlachos
  - Eamonn Keogh
  - Panagiotis Papapetrou
  - George Kollios
  - Dimitrios Gunopulos
  - Christos Faloutsos
  - Panos Karras

Introduction, Motivation

Data series

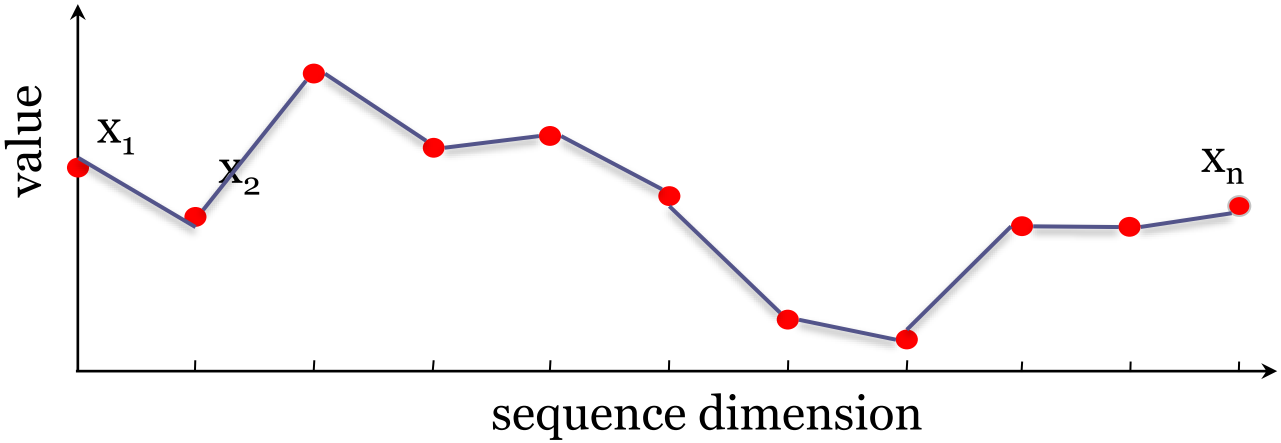
Data series

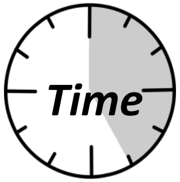
- Sequence of points ordered along some dimension



Data series

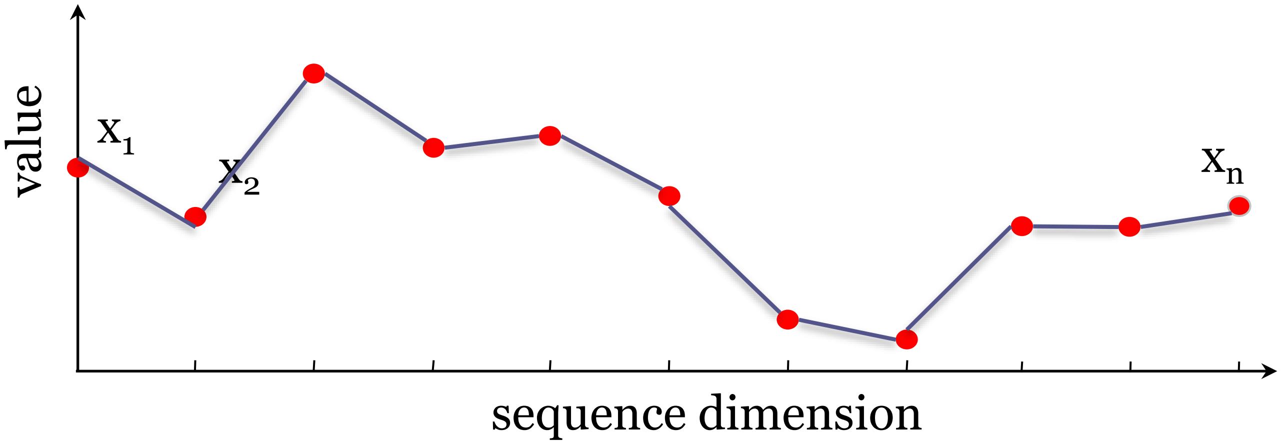
- Sequence of points ordered along some dimension

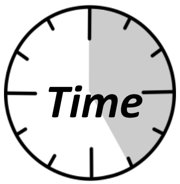


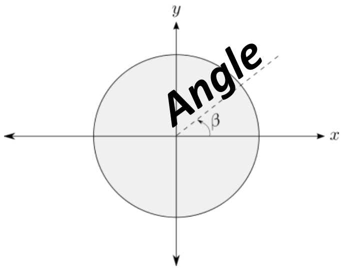


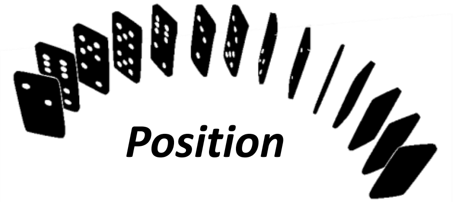
Data series

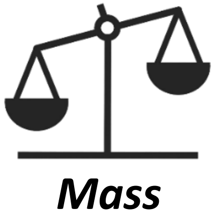
- Sequence of points ordered along some dimension

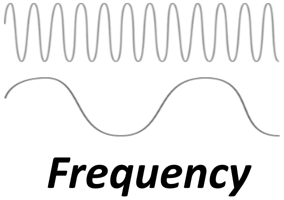






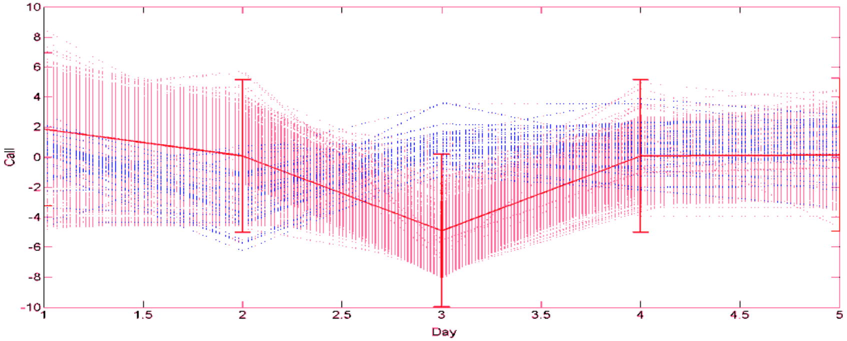




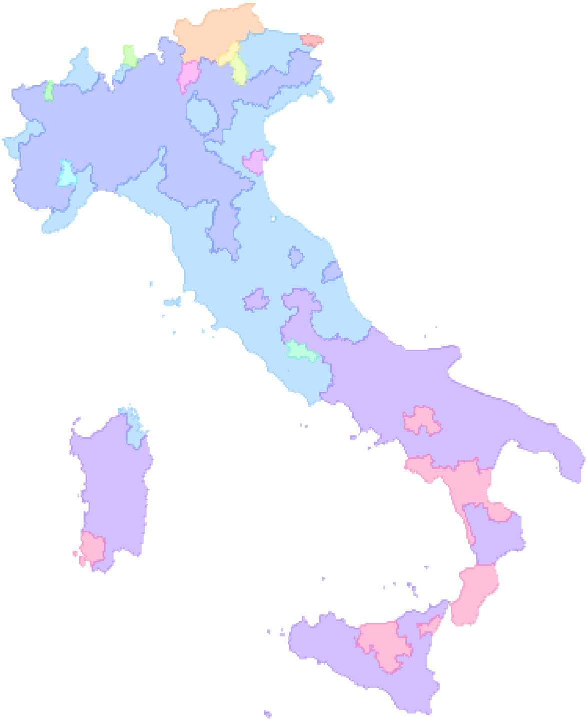


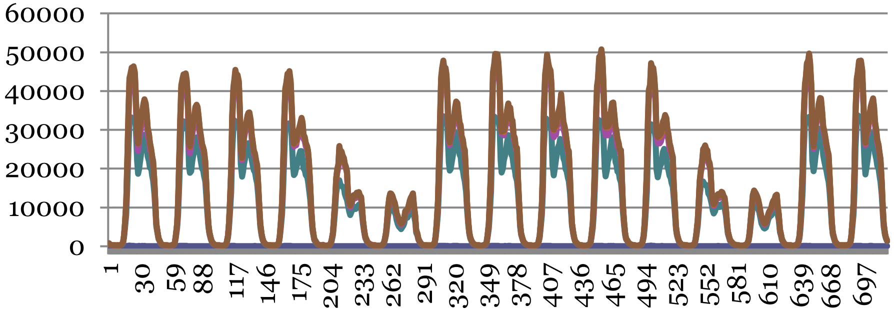
Telecommunications

- analysis of call activity patterns
  - Telecom Italia



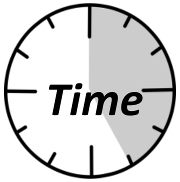
call activity for Easter Monday





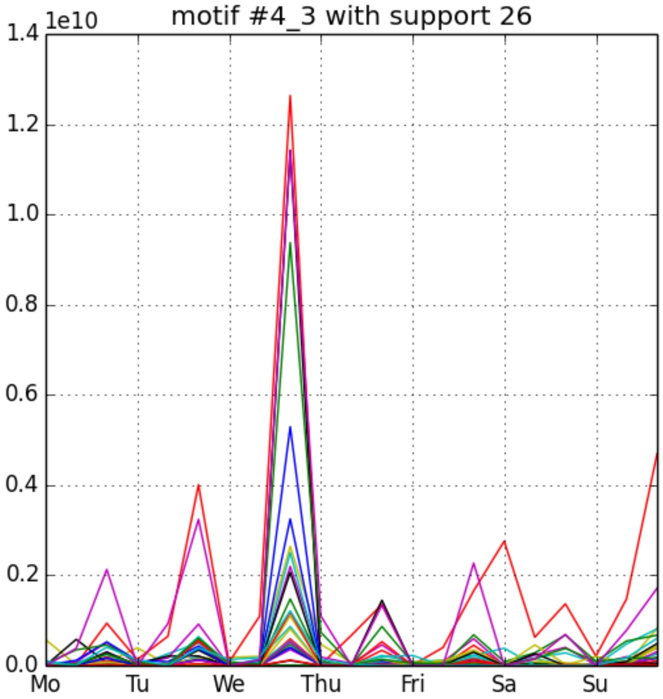
clustermap of incoming calls time series

average number of calls for 5 smallest clusters

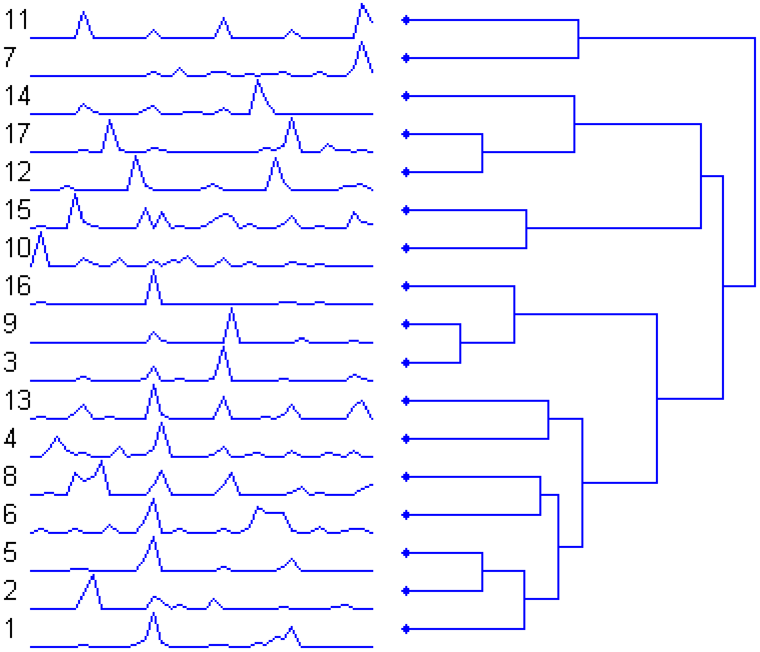


Home Networks

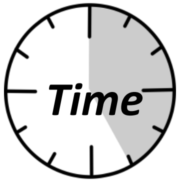
- temporal usage behavior analysis of home networks
  - Portugal Telecom



(previously unknown) frequent behavior pattern

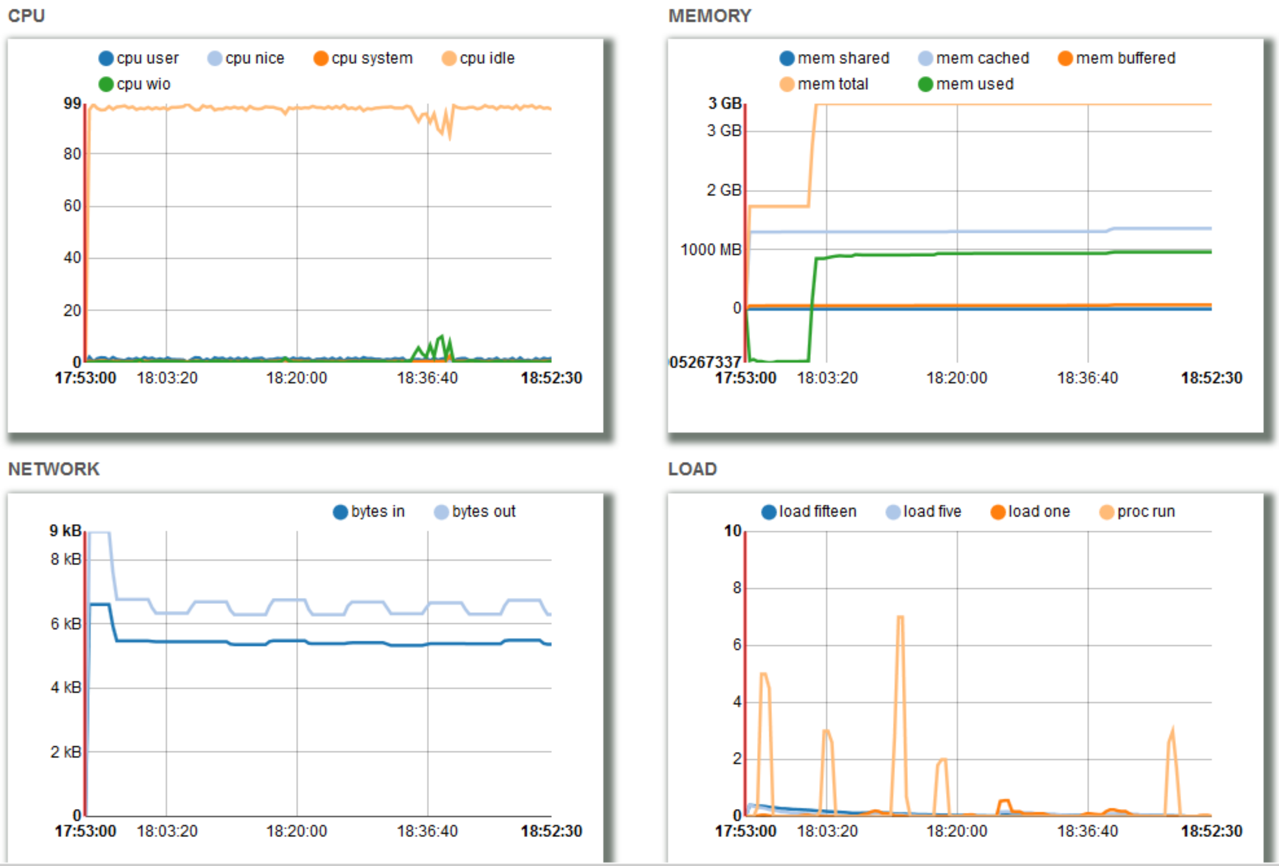


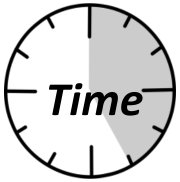
clustering based on user activity patterns



Data Centers

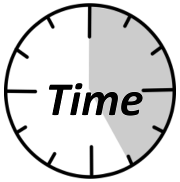
- cloud utilization/operation/health monitoring





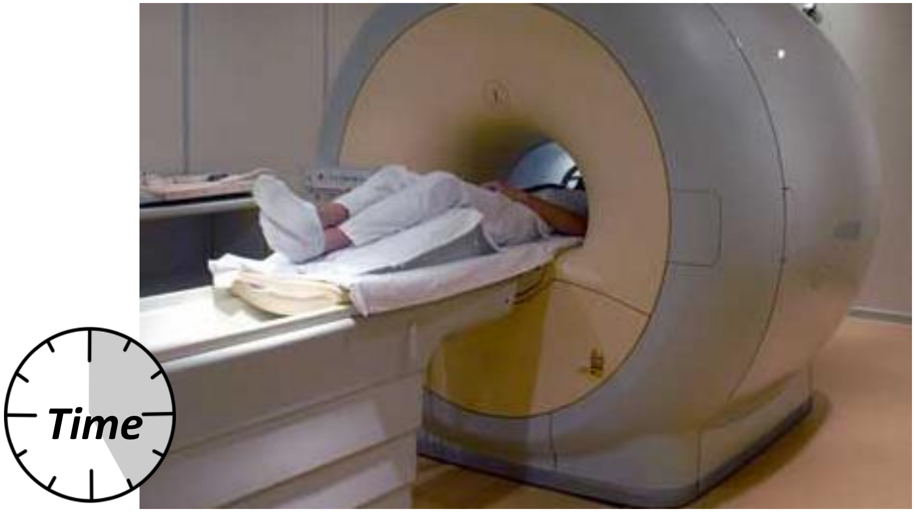
Neuroscience

- functional Resonance Magnetic Imaging (fMRI) data
  - primary experimental tool of neuroscientists
  - reveal how different parts of brain respond to stimuli



Neuroscience

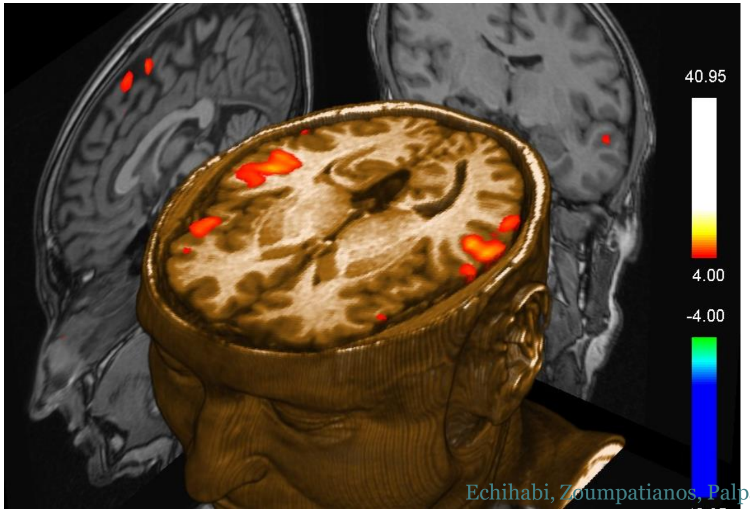
- functional Resonance Magnetic Imaging (fMRI) data
    - primary experimental tool of neuroscientists
    - reveal how different parts of brain respond to stimuli



Neuroscience

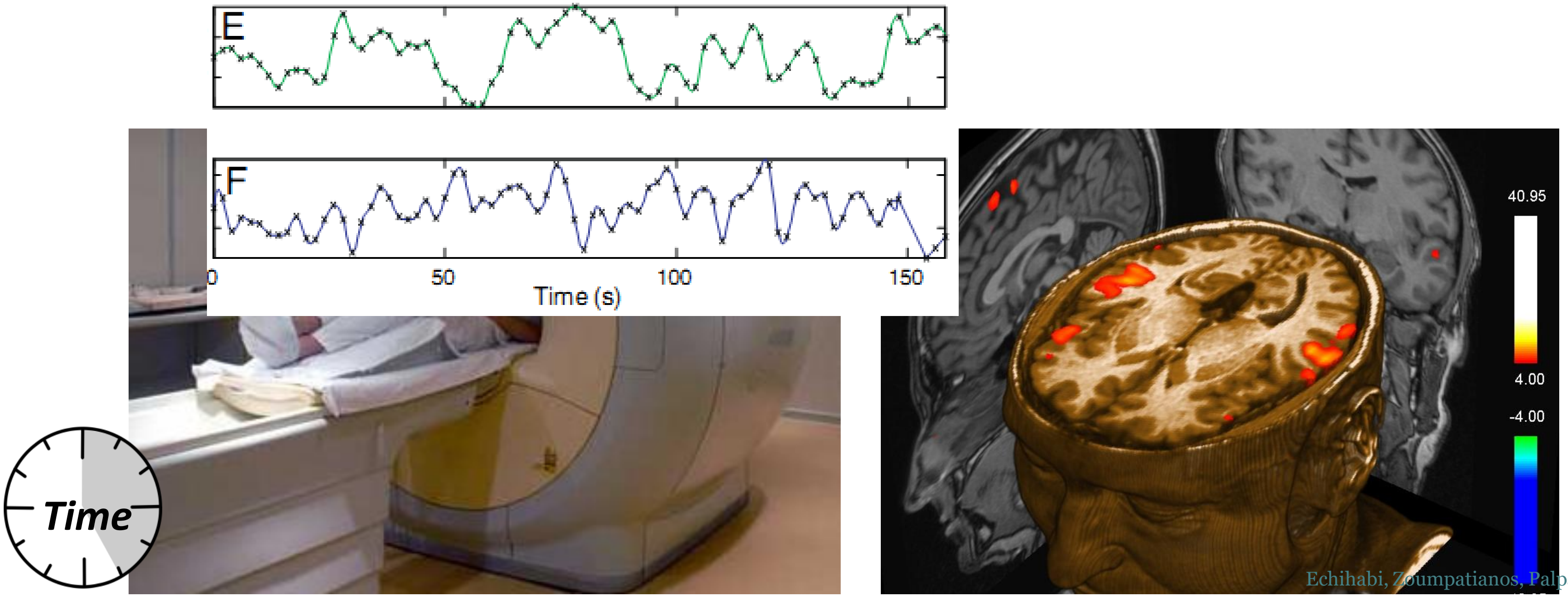
- functional Resonance Magnetic Imaging (fMRI) data
  - primary experimental tool of neuroscientists
  - reveal how different parts of brain respond to stimuli

Time

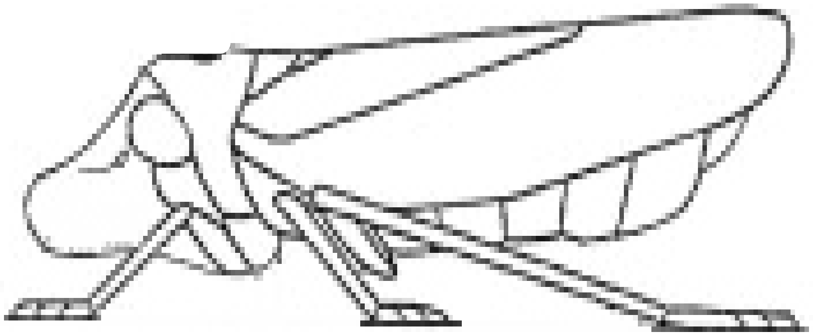


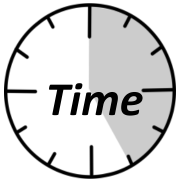
Neuroscience

- functional Resonance Magnetic Imaging (fMRI) data
  - primary experimental tool of neuroscientists
  - reveal how different parts of brain respond to stimuli

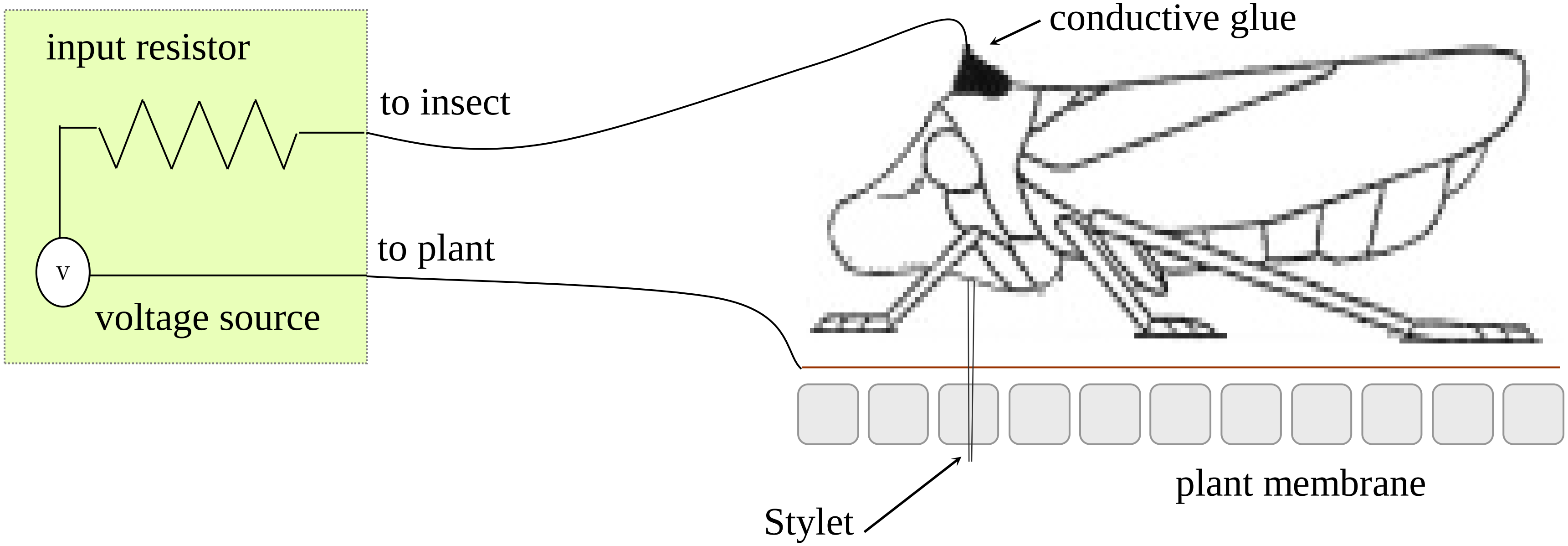


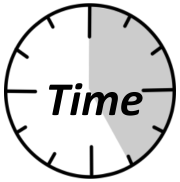
Entomology



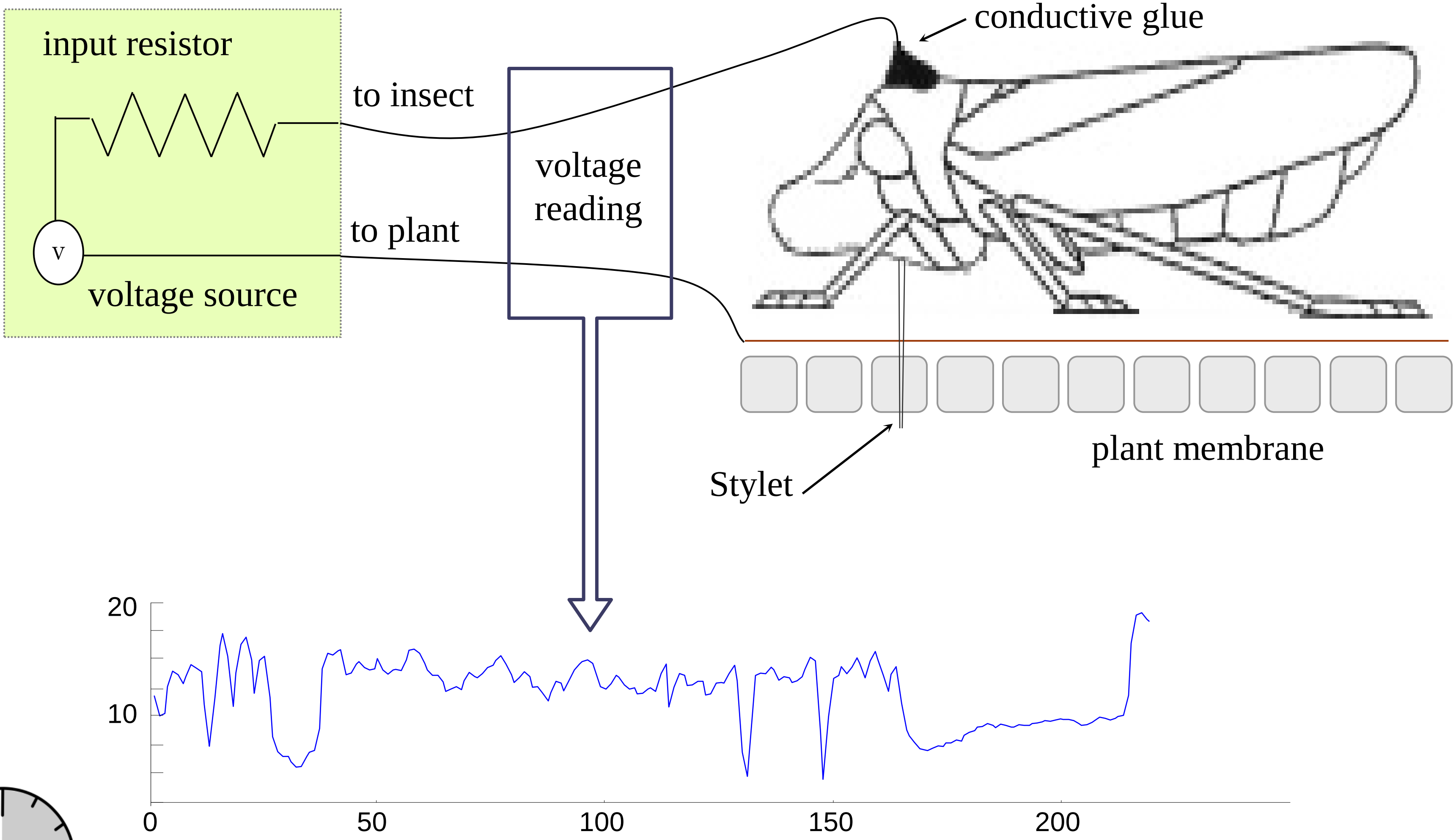


Entomology


input resistor
to insect
to plant
v
voltage source
conductive glue
plant membrane
Stylet

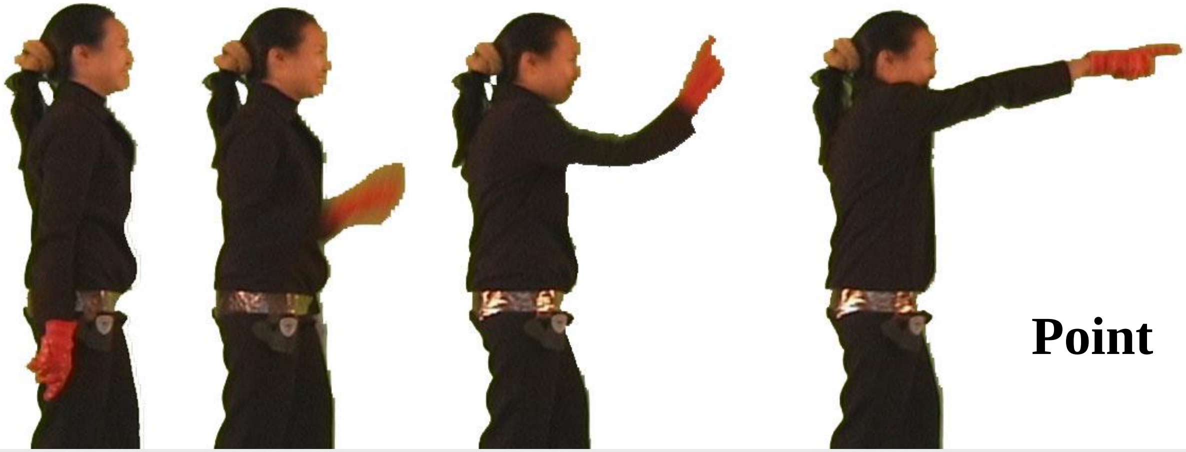

Time

Entomology

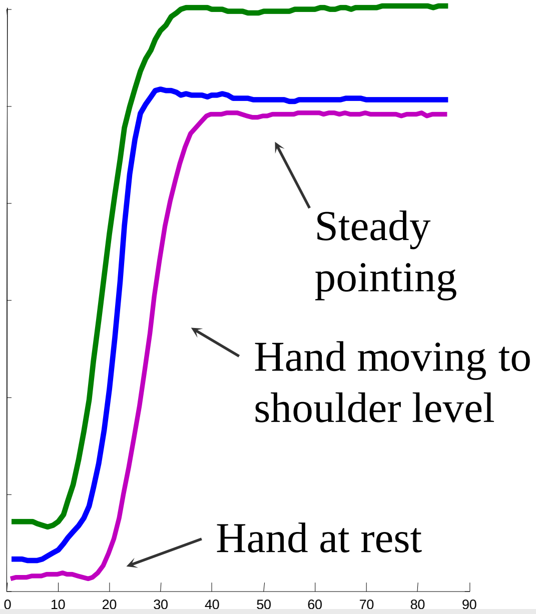


Time

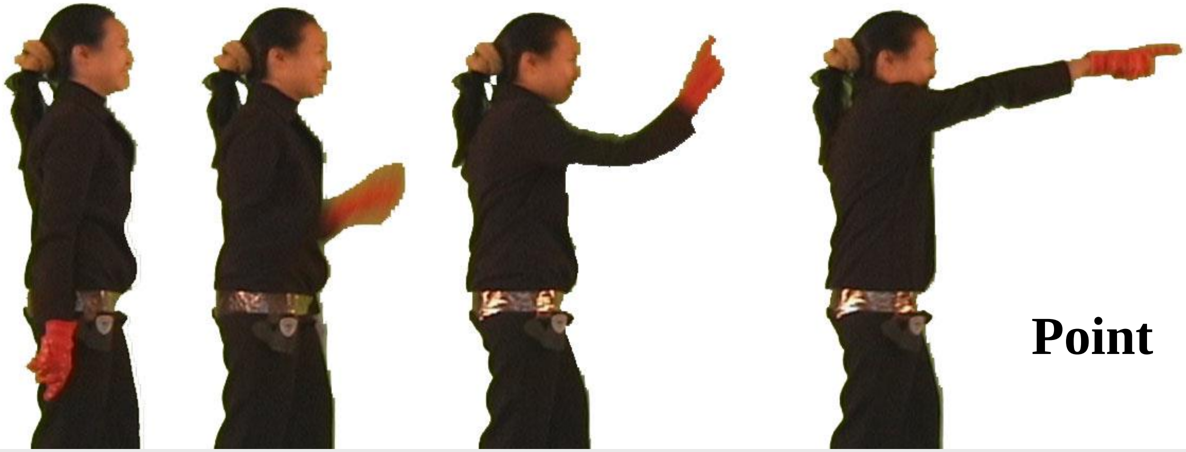
Time



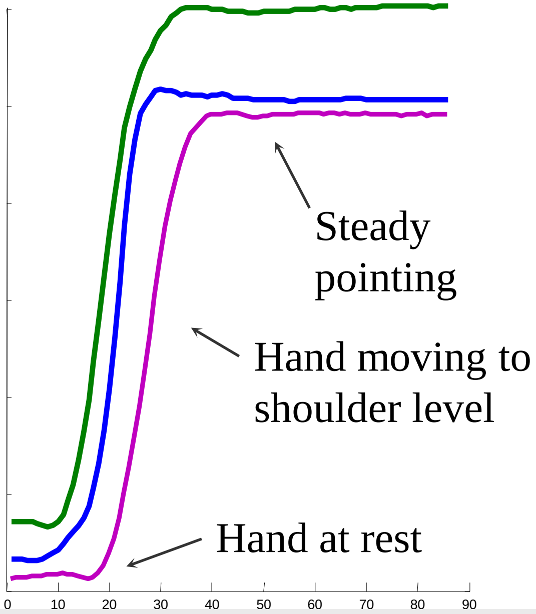
Point

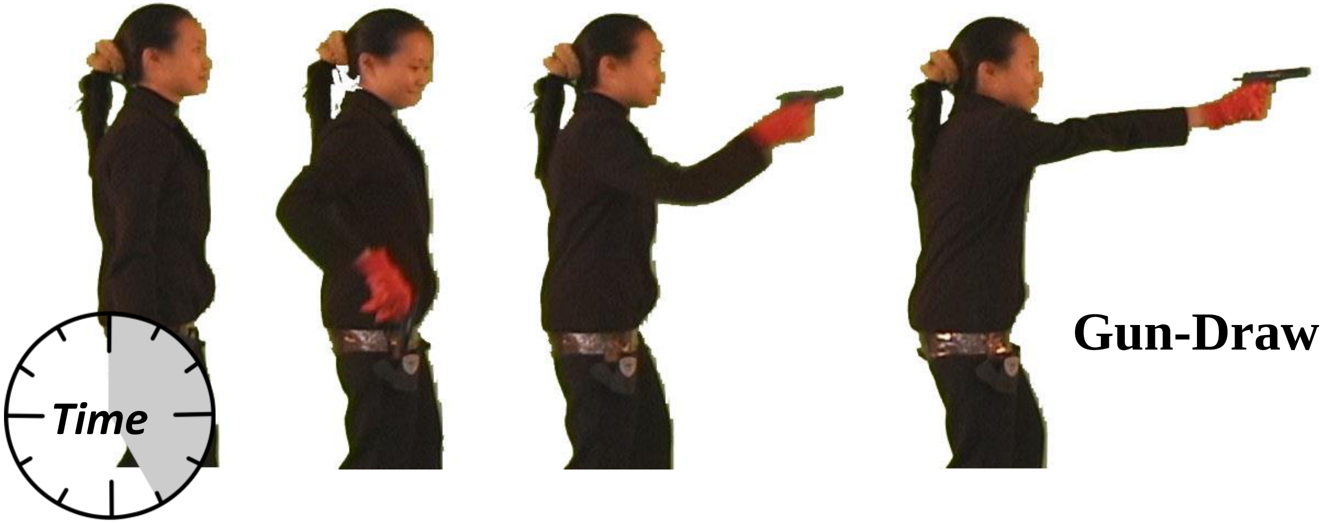

Steady pointing
Hand moving to shoulder level
Hand at rest
0
10
20
30
40
50
60
70
80
90

Time

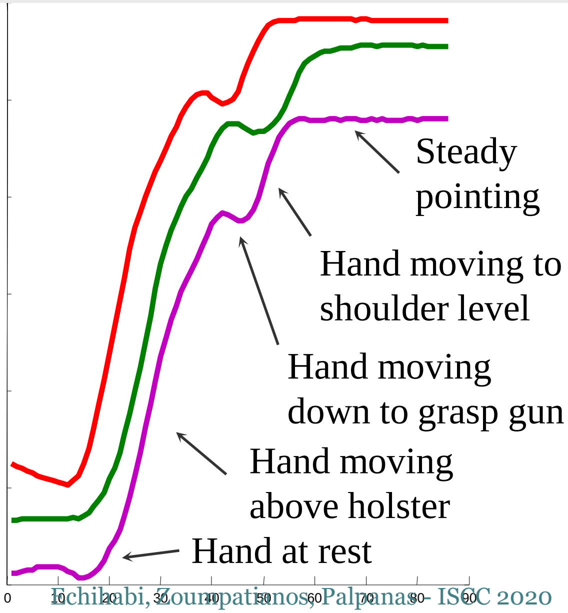


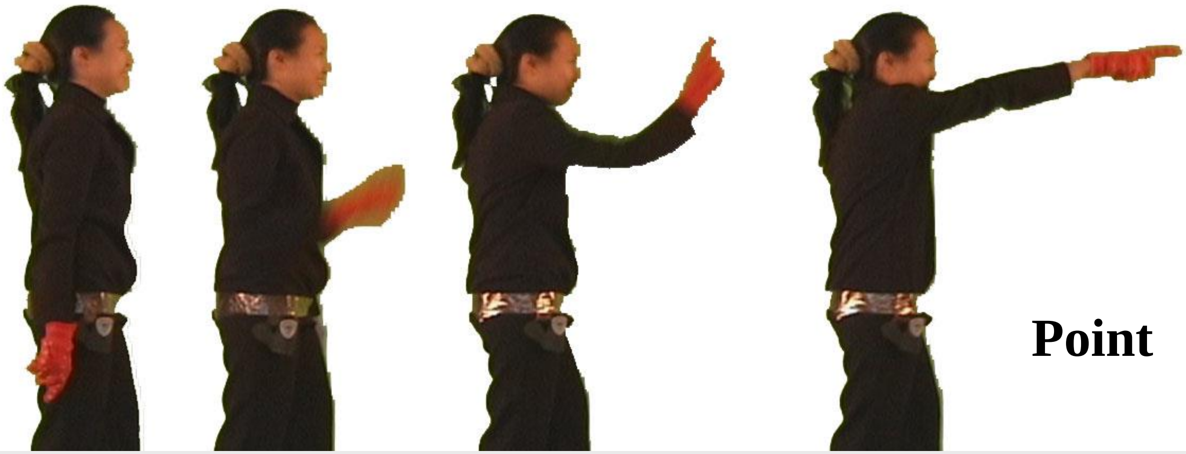
Point



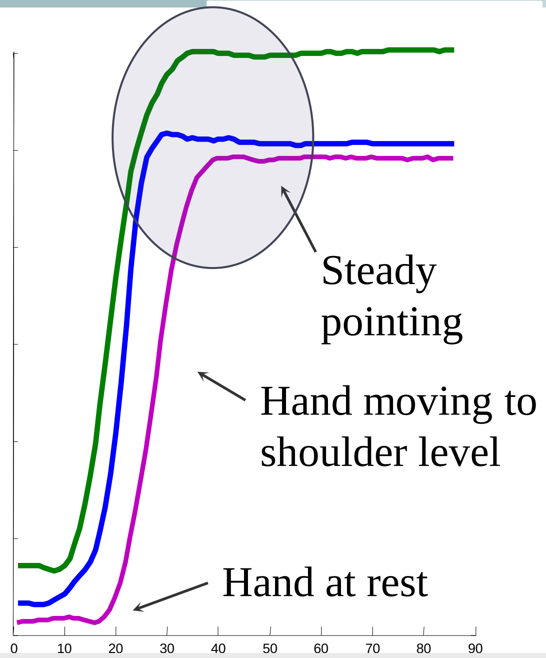


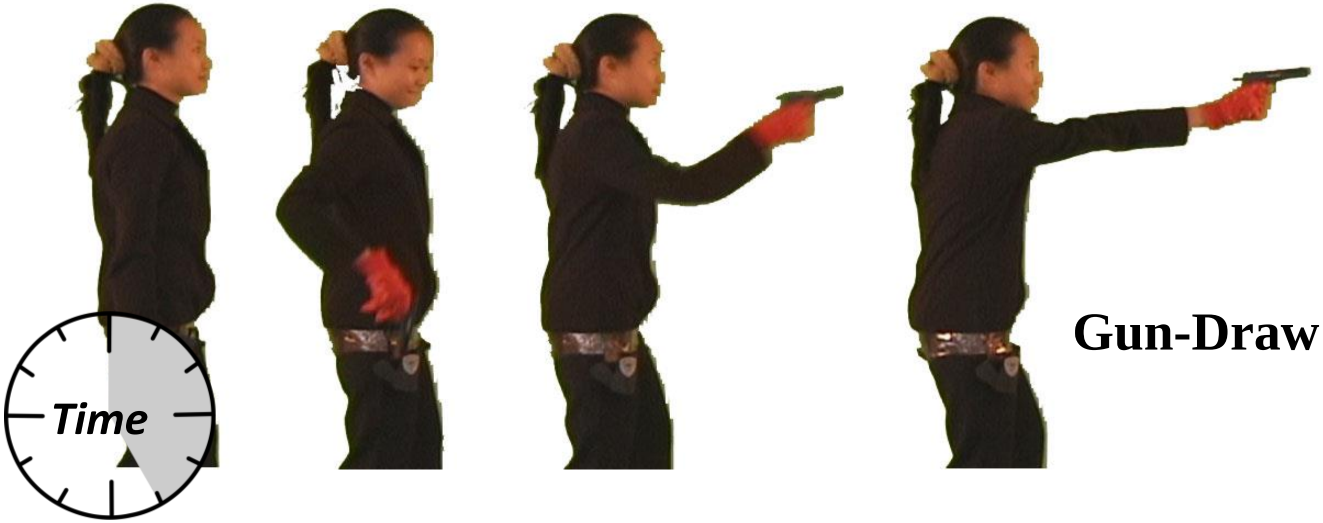
Gun-Draw



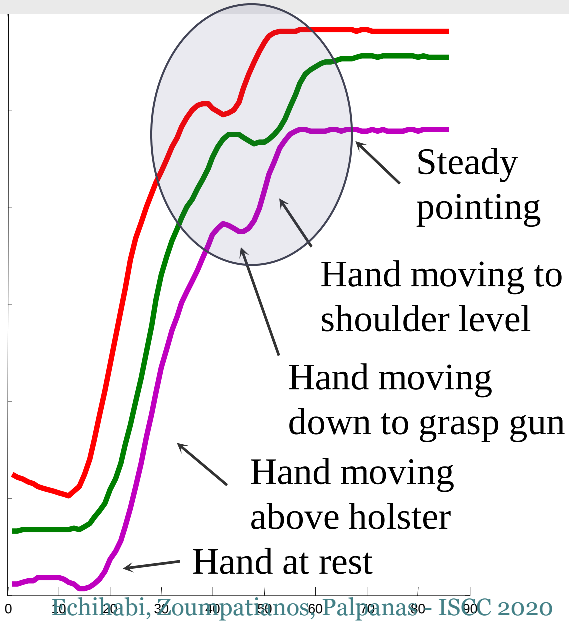


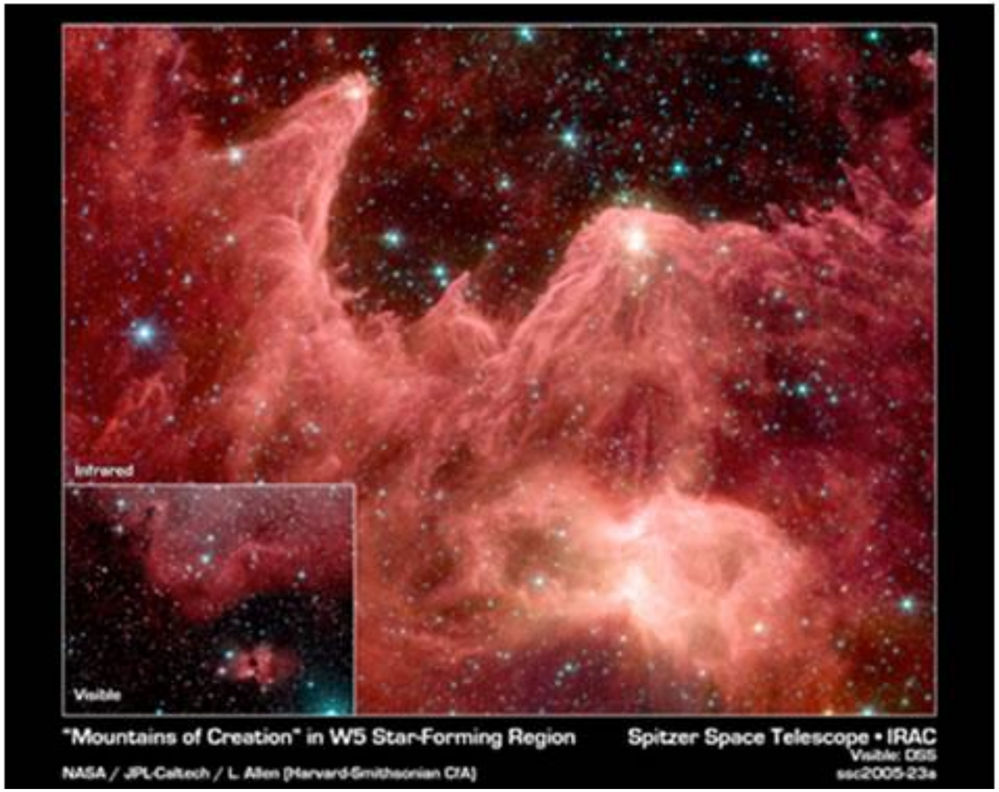
Point

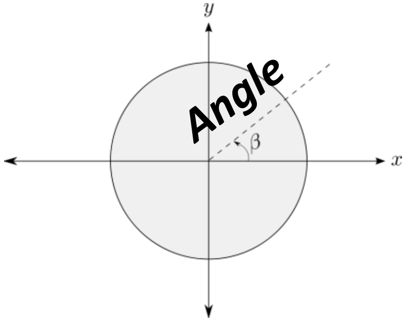




Gun-Draw

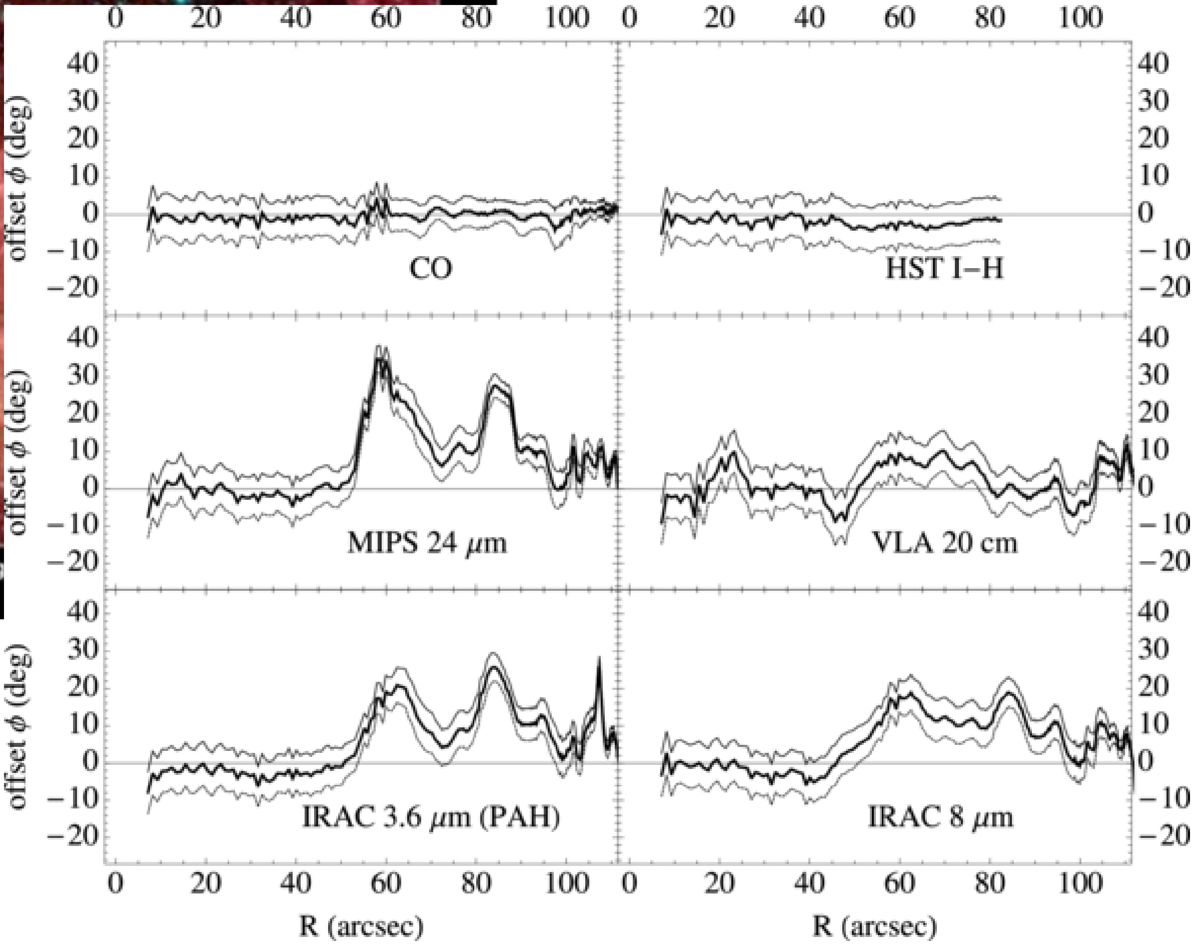


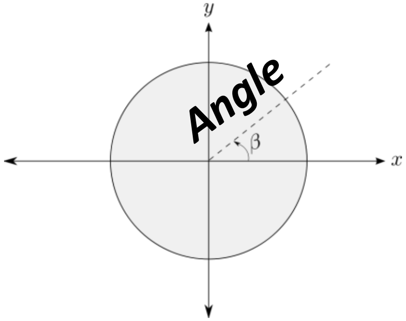

Infrared
Visible
"Mountains of Creation" in W5 Star-Forming Region
Spitzer Space Telescope • IRAC
Visible: DSS
NASA / JPL-Caltech / L. Allen (Harvard-Smithsonian CfA)
ssc2005-23a


y
Angle
β
x

"Mountains of Creation" in W5 Star-Forming

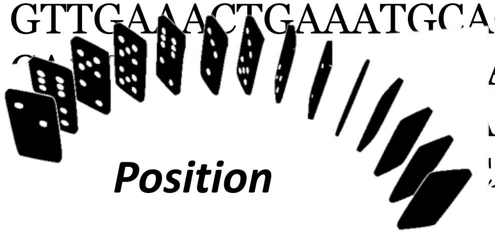
NASA / JPL-Caltech / L. Allen (Harvard-Smithsonian CfA)



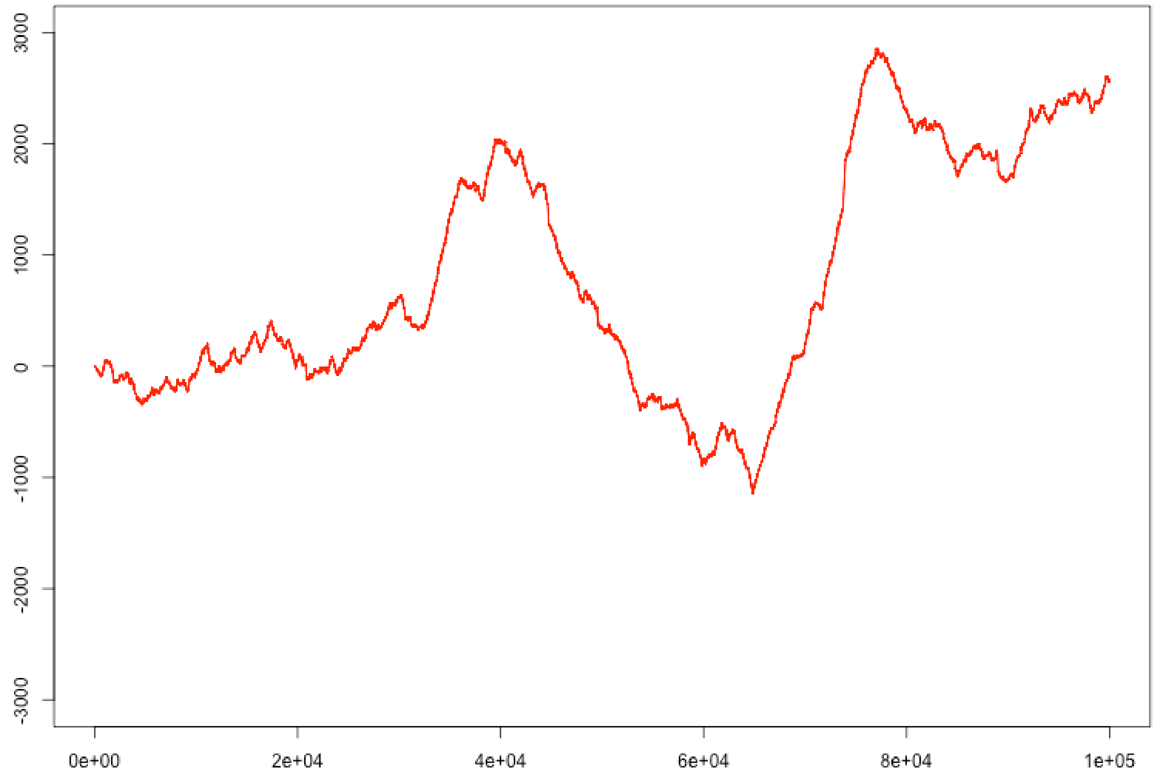


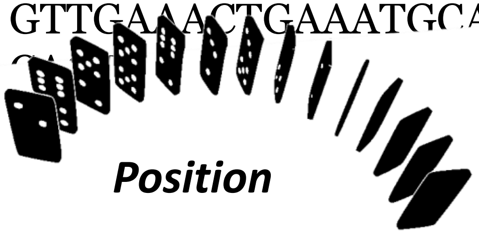
Schinnerer et al.

GTCAATGGCCAGGATATTAGAACAGTACTCTGTGAACCCTATTTATGGTGGCACCCCTTAGACTAA
GATAACACAGGGAGCAAGAGGTTGACAGGAAAGCCAGGGGAGCAGGGAAGCCTCCTGTAAAGAG
AGAAGTGCTAAGTCTCCTTTCTAAGGCACATGATGGATTCAAGGGAAAGCCACATTTGACTAAAGC
CCAAGGGATTGTTGCTTCTAATCCGATTTCTTGGCAGAAGATATTACAAACTAAGAGTCAGATTAA
TATGTGGGTGCCAAAATAAATAAACAAATAATTGAATAATCCCTGGAGGTTTAAGTGAGGAGAAA
CTCCTCCACAGCTTGCTACCGAGGCAGAACCGGTTGAAACTGAAATGCATCCGCCGCCAGAGGATC
TGTAAAAGAGAGGTTGTTACGAAACTGGCAACTGCCAACCAAAGTCCACCAATGGACAAGCAAAA
AAGAGCACTCATCTCATGCTCCCAAGGATCAACCTTCCCAGAGTTTTCACTTAAGTGGCCACCAAG
CCAGTTGTCAATCCAGGGCTTTGGACTGAAATCTAGGGCTTCATCCGCTACCTCAGAGTGTCTTCT
ATTTCTTCCAGCCAGTGACAAATACAACAAACATCTGAGATGTTTTAGCTATAAATCCTTTACAATT
GTTATTTATGTCTTAACTTTTGTTATACCTGGAAAAGTAGGGGAAACAATAAGAACATACTGTCTT
GGCCAAGCATCCAAGGTTAAATGAGTTATGGAAATTCATTTGGGAGCCAAGACATTGCACGTGGT
TATTTATTAGTCACCCAAGCATGTATTTTGCATGTCCATCAGTTGTTCTTGGCCAAAAGAGCAGAAT
CAATGAGCCGCTGCAGATGCAGACATAGCAGCCCCTTGCAGGGACAAGTCTGCAAGATGAGCATT
GAAGAGGATGCACAAGCCCGGTAGCCCGGGAAATGGCAGGCACTTACAAGAGCCCAGGTTGTTGC
CATGTTTGTTTTTGCAACTTGTCTATTTAAAGAGATTTGGGCAATGGCCAGGATATTAGAACAGTA
CTCTGTGAACCCTATTTATGGTAGCACCCCTTAGACTAAGATAACACAGGGAGCAAGAGGTTGACA
GGAAAGCCAGGGGAGCAGGGAAGCCTCCTGTAAAGAGAGAAGTGCTAAGTCTCCTTTCTAAGGCA
CATGATGGATCAAGGGAAAGTCACATTTGACTAAAGCCCAAGGGATTGTTGCTTCTAATCCGATTC
TTGGCAGAAGATATTGCAAACTAAGAGTCAGATTAATATGTGGGTGCCAAAATAAATAAACAAATA
ATTGAATAATCCCTGGAGGTTTAAGTGAGGAGAAACTCCTCCACACTTGCTACCGAGGCAGAACCG
GTTGAAACTGAAATGCACCCGCTGCCAGATTTATTAGTCACCCAAGCATGTATTTTGCATGTCCAT
AAGAACAGAATCAATGAGCCGCTGCAGATGCAGACATAGCAGCCCCTTG
GATGAGCATTGAAGAGGATGCACAAGCCCGGTAGCCCGGGAAATGGCA
AGGTTGTTGCCATGTTTGTTTTTGCAACTTGTCTTTTAAACAGATTTGA



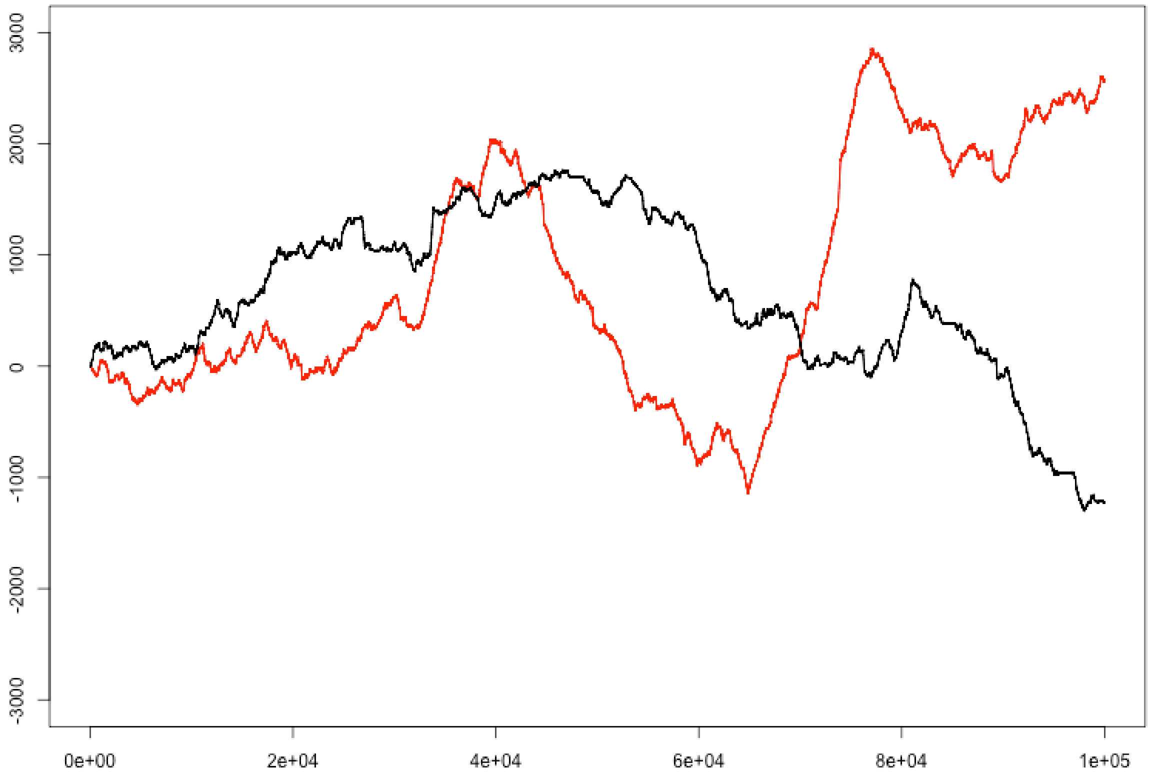
GTCAATGGCCAGGATATTAGAACAGTACTCTGTGAACCCTATTTATGGTGGCACCCCTTAGACTAA
GATAACACAGGGAGCAAGAGGTTGACAGGAAAGCCAGGGGAGCAGGGAAGCCTCCTGTAAAGAG
AGAAGTGCTAACTCTCCTTTCTAAGGCAGATGATGGATTCAAGGCAAAGGCAGATTTGACTAAAGC
CCAAGGATTG AGTCAGATTAA
TATGTGGGTGC TGAGGAGAAA
CTCCTCCACAGC GCCAGAGGATC
TGTAAAAGAGA ACAAGCAAAA
AAGAGCACTCA GGCCACCAAG
CCAGTTGTCAAT GAGTGTCTTCT
ATTTCTTCCAGC CCTTTACAATT
GTTATTTATGTC CATACTGTCTT
GGCCAAGCATCC TGCACGTGGT
TATTTATTAGTC AAGAGCAGAAT
CAATGAGCCGCT AGATGAGCATT
GAAGAGGATGC CAGGTTGTTGC
CATGTTTGTTTT TTAGAACAGTA
CTCTGTGAACCC AGAGGTTGACA
GGAAAGCCAGG TTTCTAAGGCA
CATGATGGATCA TAATCCGATTC
TTGGCAGAAGA ATAAACAAATA
ATTGAATAATCCCTGGAGGTTTAAGTGAGGAGAAACTCCTCCACACTTGCTACCGAGGCAGAACCG
GTTGAAACTGAAATGCACCCGCTGCCAGATTTATTAGTCACCCAAGCATGTATTTTGCATGTCCAT



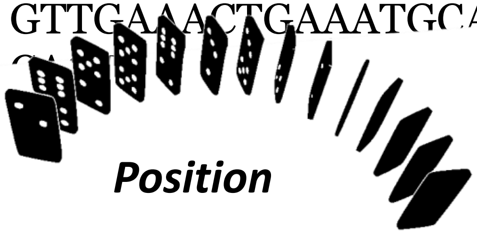


AAGAACAGAATCAATGAGCCGCTGCAGATGCAGACATAGCAGCCCCTTG
GATGAGCATTGAAGAGGATGCACAAGCCCGGTAGCCCGGGAATGGCA
AGGTTGTTGCCATGTTTGTTTTGCAACTTGTCTTTTAAACAGATTTGA

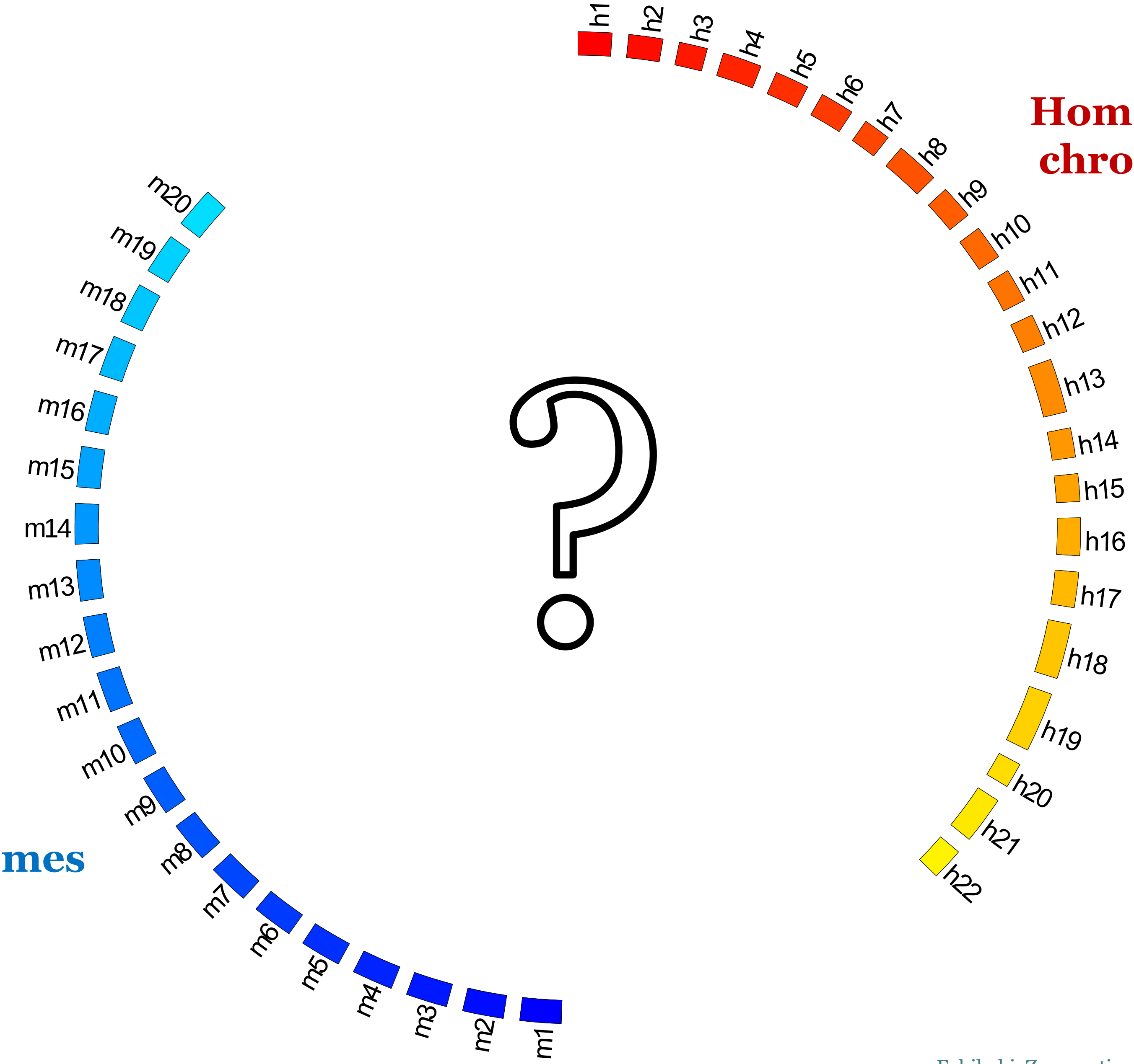
GTCAATGGCCAGGATATTAGAACAGTACTCTGTGAACCCTATTTATGGTGGCACCCCTTAGACTAA
GATAACACAGGGAGCAAGAGGTTGACAGGAAAGCCAGGGGAGCAGGGAAGCCTCCTGTAAAGAG
AGAAGTGCTAACTCTCCTTTCTAAGGCAGATGATGGATTCAAGGCAAAGGCAGATTTGACTAAAGC
CCAAGGATTG AGTCAGATTAA
TATGTGGGTGC TGAGGAGAAA
CTCCTCCACAG GCCAGAGGATC
TGTAAAAGAGA ACAAGCAAAA
AAGAGCACTCA GGCCACCAAG
CCAGTTGTCAA AGTGTCTTCT
ATTTCTTCCAG CCTTTACAATT
GTTATTTATGT CATACTGTCTT
GGCCAAGCATC TGCACGTGGT
TATTTATTAGT AAGAGCAGAAT
CAATGAGCCGC AGATGAGCATT
GAAGAGGATGC CAGGTTGTTGC
CATGTTTGTTT TTAGAACAGTA
CTCTGTGAACC AGAGGTTGACA
GGAAAGCCAGG TTTCTAAGGCA
CATGATGGATC TAATCCGATTC
TTGGCAGAAGA ATAAACAAATA



ATTGAATAATCCCTGGAGGTTTAAGTGAGGAGAAACTCCTCCACACTTGCTACCGAGGCAGAACCG
GTTGAAACTGAAATGCACCCGCTGCCAGATTTATTAGTCACCCAAGCATGTATTTTGCATGTCCAT
AAGAACAGAATCAATGAGCCGCTGCAGATGCAGACATAGCAGCCCCTTG
GATGAGCATTGAAGAGGATGCACAAGCCCGGTAGCCCGGGAATGGCA
AGGTTGTTGCCATGTTTGTTTTTGCAACTTGTCTTTTAAACAGATTTGA



Position

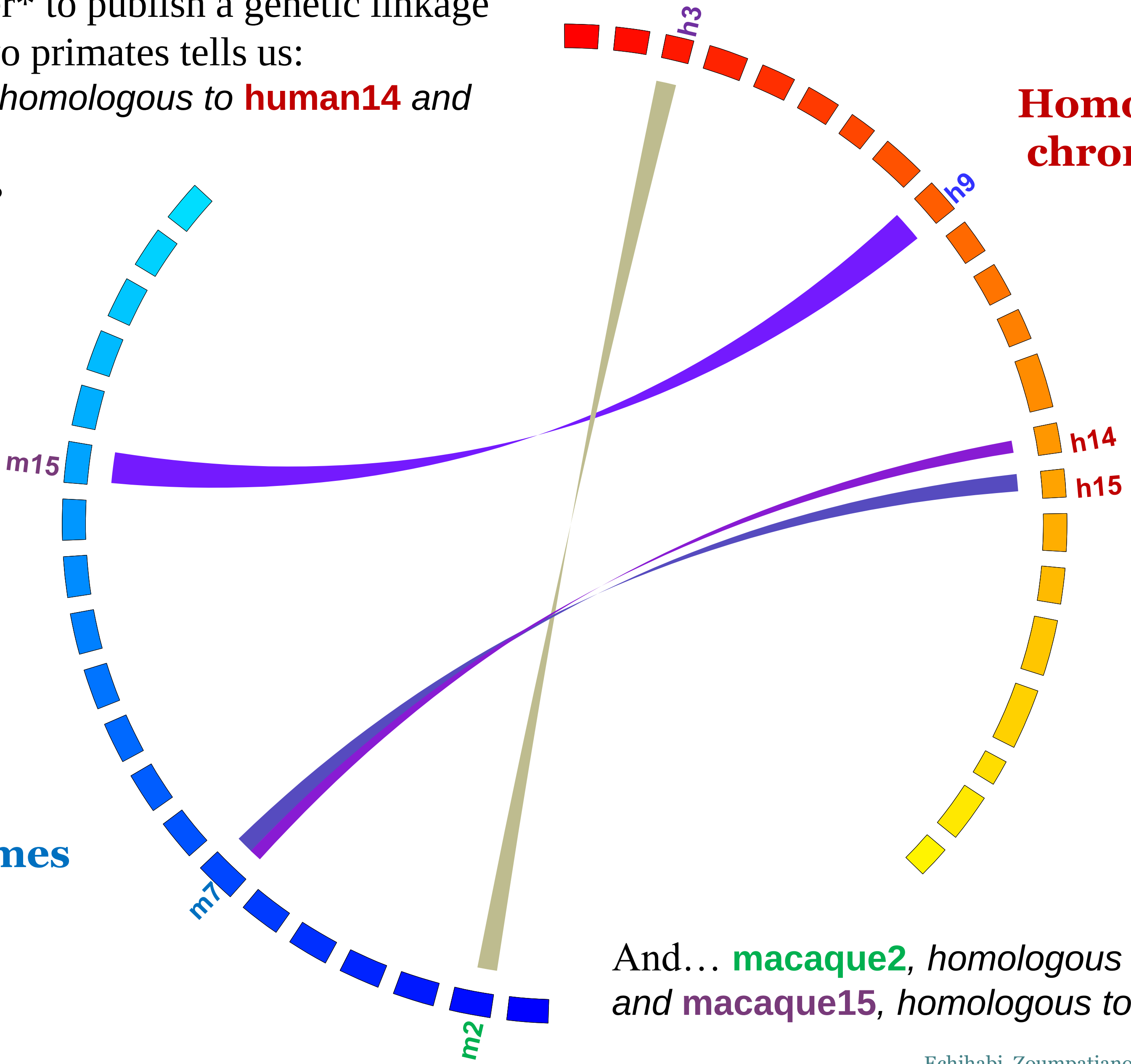


Homo Sapiens chromosomes

Rhesus Macaque chromosomes

The first paper* to publish a genetic linkage map of the two primates tells us:
macaque7 *is homologous to* **human14** *and* **human15**

*Rogers, J. et al. 2006



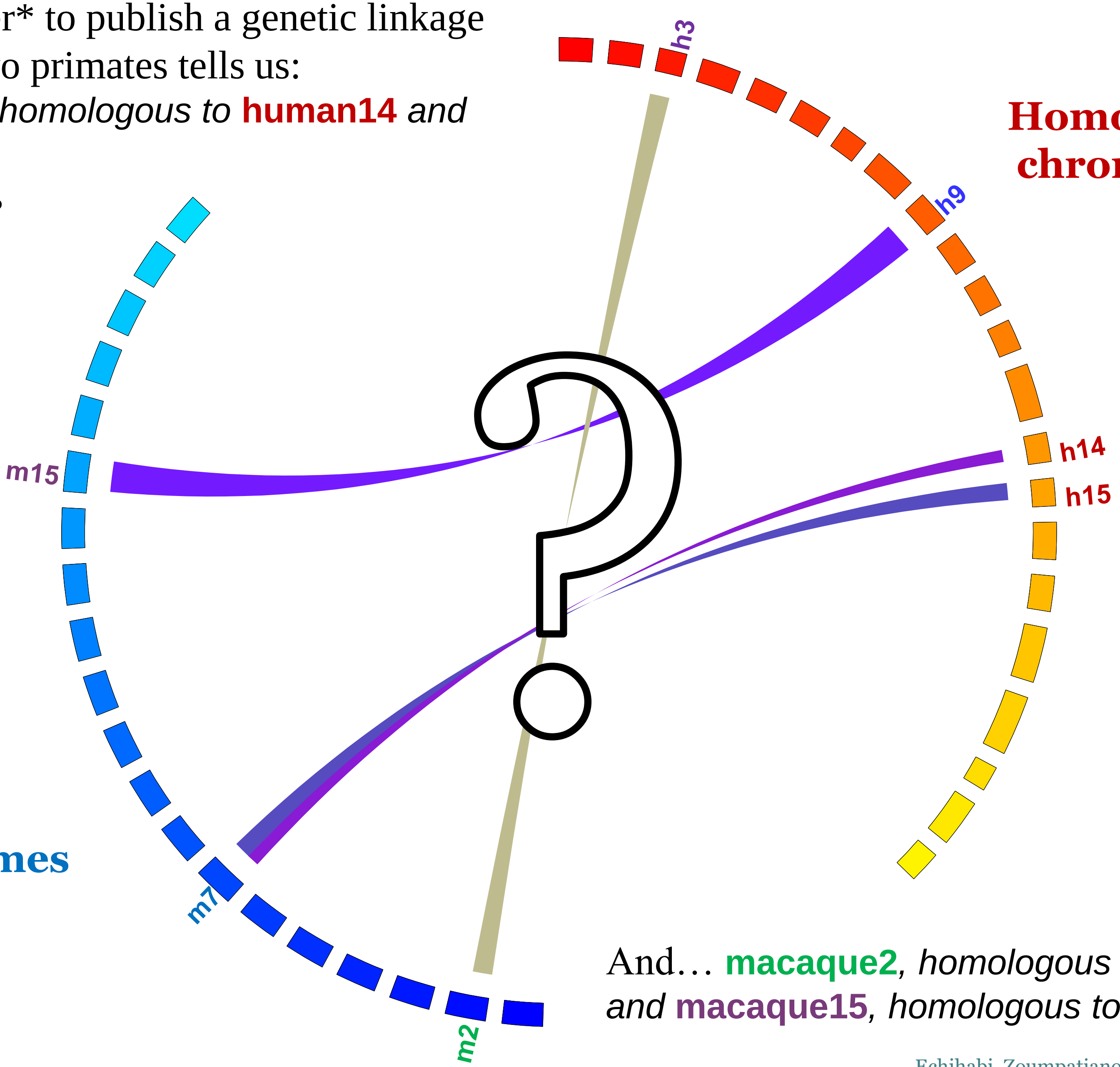
Homo Sapiens chromosomes

Rhesus Macaque chromosomes

And… **macaque2**, *homologous to* **human3**, *and* **macaque15**, *homologous to* **human9**...

The first paper* to publish a genetic linkage map of the two primates tells us:
macaque7 *is homologous to* **human14** *and* **human15**

*Rogers, J. et al. 2006

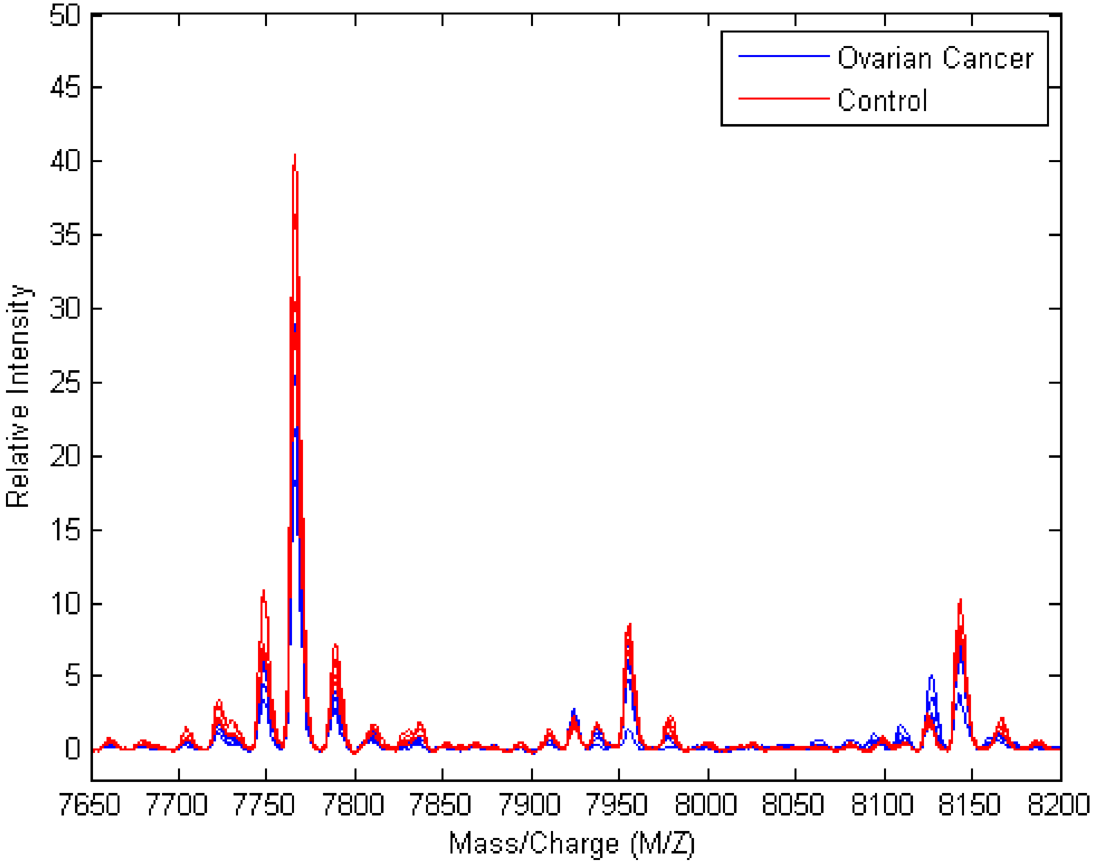


Homo Sapiens chromosomes

Rhesus Macaque chromosomes

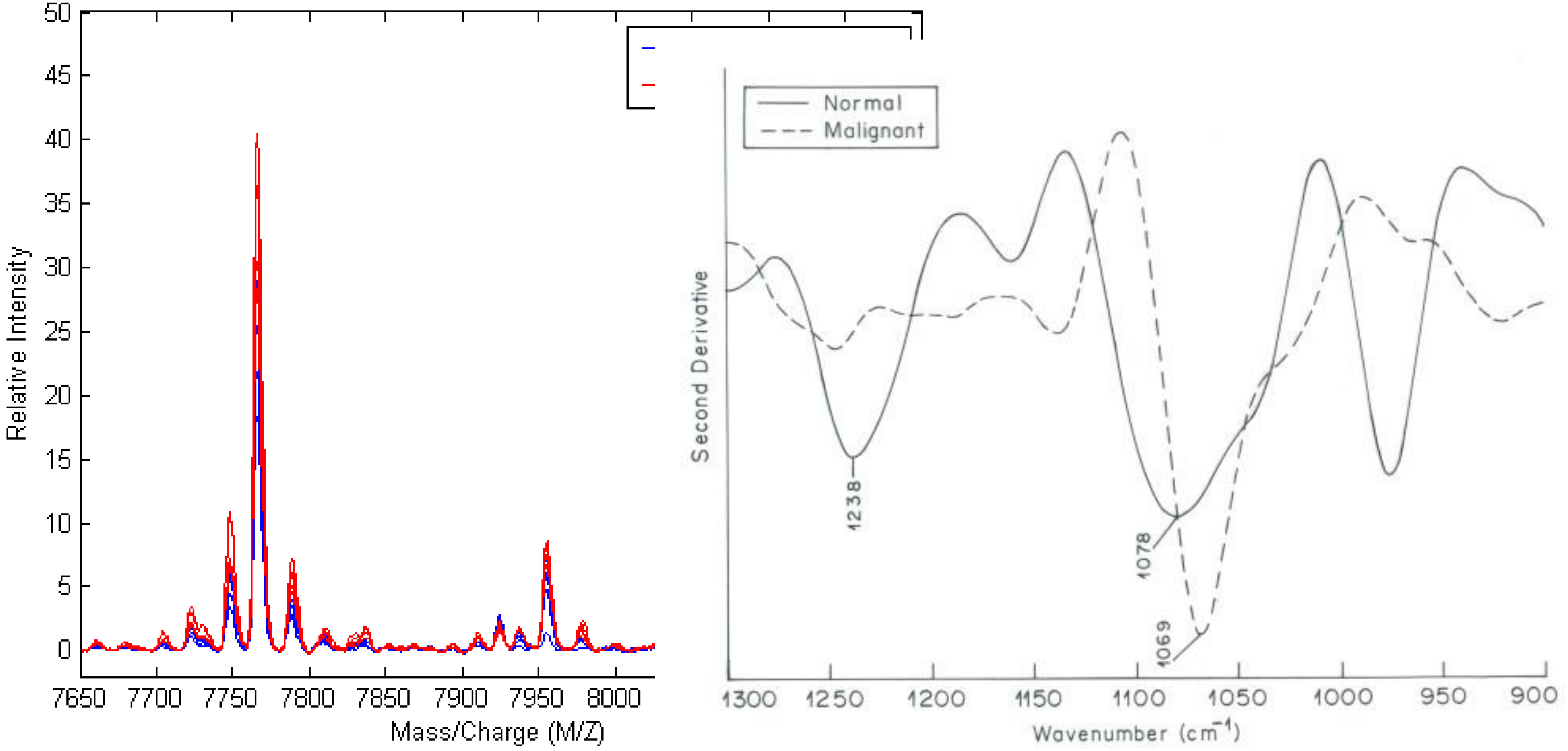
And… **macaque2**, *homologous to* **human3**, *and* **macaque15**, *homologous to* **human9**...

Medicine

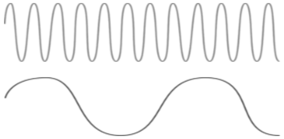


Mass

Medicine

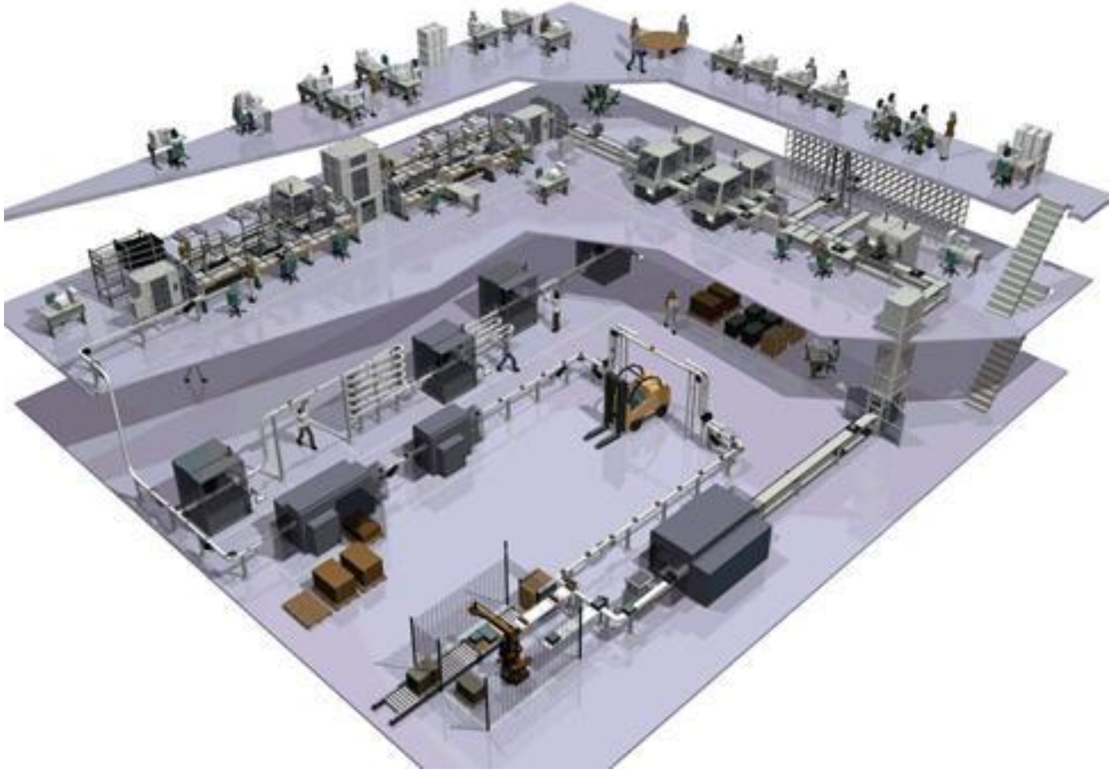


Mass



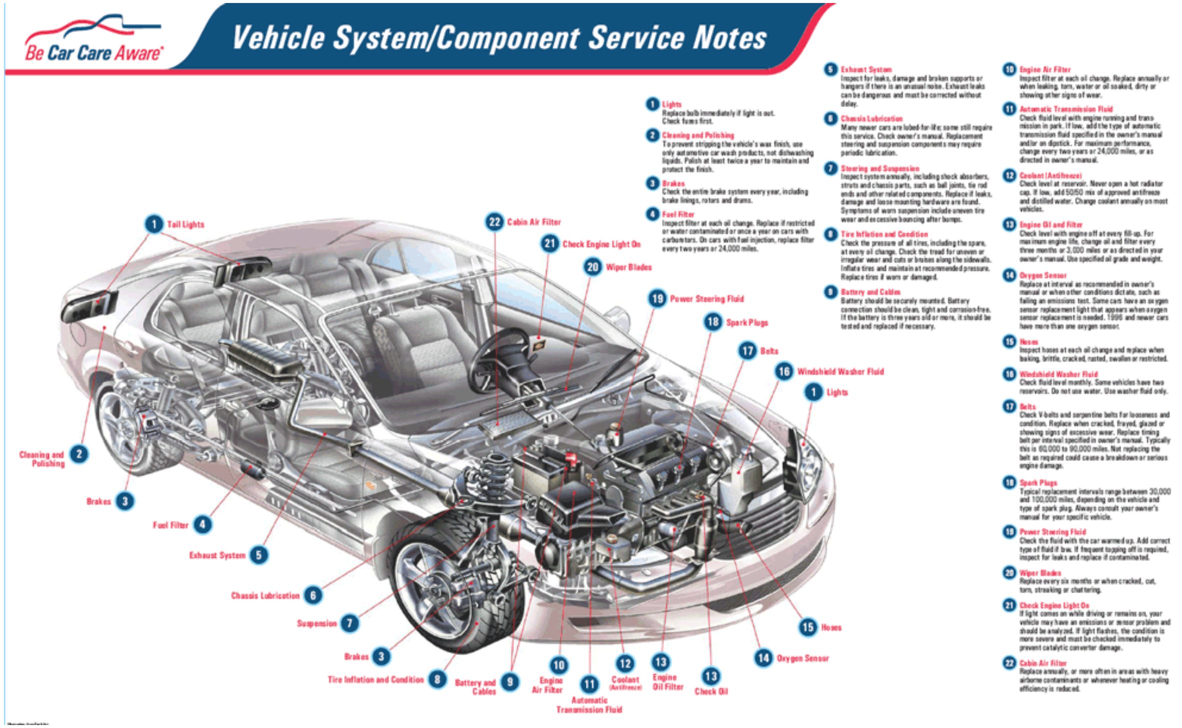
Frequency

Motivating Examples: Production Control System



Motivating Examples: Production Control System

Motivating Examples: Monitoring Vehicle Operation



Vehicle System/Component Service Notes

1 **Lights**
Replace bulb immediately if light is out. Check fuses first.

2 **Cleaning and Polishing**
To prevent stripping the vehicle's wax finish, use only automotive car wash products, not dishwashing liquids. Polish at least twice a year to maintain and protect the finish.

3 **Brakes**
Check the entire brake system every year, including brake linings, rotors and drums.

4 **Fuel Filter**
Inspect filter at each oil change. Replace if restricted or water contaminated or once a year on cars with carburetors. On cars with fuel injection, replace filter every two years or 24,000 miles.

5 **Exhaust System**
Inspect for leaks, damage and broken supports or hangers if there is an unusual noise. Exhaust leaks can be dangerous and must be corrected without delay.

6 **Chassis Lubrication**
Many newer cars are lubed-for-life; some still require this service. Check owner's manual. Replacement steering and suspension components may require periodic lubrication.

7 **Steering and Suspension**
Inspect system annually, including shock absorbers, struts and chassis parts, such as ball joints, tie rod ends and other related components. Replace if leaks, damage and loose mounting hardware are found. Symptoms of worn suspension include uneven tire wear and excessive bouncing after bumps.

8 **Tire Inflation and Condition**
Check the pressure of all tires, including the spare, at every oil change. Check the tread for uneven or irregular wear and cuts or bruises along the sidewalls. Inflate tires and maintain at recommended pressure. Replace tires if worn or damaged.

9 **Battery and Cables**
Battery should be securely mounted. Battery connection should be clean, tight and corrosion-free. If the battery is three years old or more, it should be tested and replaced if necessary.

10 **Engine Air Filter**
Inspect filter at each oil change. Replace annually or when leaking, torn, water or oil soaked, dirty or showing other signs of wear.

11 **Automatic Transmission Fluid**
Check fluid level with engine running and transmission in park. If low, add the type of automatic transmission fluid specified in the owner's manual and/or on dipstick. For maximum performance, change every two years or 24,000 miles, or as directed in owner's manual.

12 **Coolant (Antifreeze)**
Check level at reservoir. Never open a hot radiator cap. If low, add 50/50 mix of approved antifreeze and distilled water. Change coolant annually on most vehicles.

13 **Engine Oil and Filter**
Check level with engine off at every fill-up. For maximum engine life, change oil and filter every three months or 3,000 miles or as directed in your owner's manual. Use specified oil grade and weight.

14 **Oxygen Sensor**
Replace at interval as recommended in owner's manual or when other conditions dictate, such as failing an emissions test. Some cars have an oxygen sensor replacement light that appears when oxygen sensor replacement is needed. 1996 and newer cars have more than one oxygen sensor.

15 **Hoses**
Inspect hoses at each oil change and replace when baking, brittle, cracked, rusted, swollen or restricted.

16 **Windshield Washer Fluid**
Check fluid level monthly. Some vehicles have two reservoirs. Do not use water. Use washer fluid only.

17 **Belts**
Check V-belts and serpentine belts for looseness and condition. Replace when cracked, frayed, glazed or showing signs of excessive wear. Replace timing belt per interval specified in owner's manual. Typically this is 60,000 to 90,000 miles. Not replacing the belt as required could cause a breakdown or serious engine damage.

18 **Spark Plugs**
Typical replacement intervals range between 30,000 and 100,000 miles, depending on the vehicle and type of spark plug. Always consult your owner's manual for your specific vehicle.

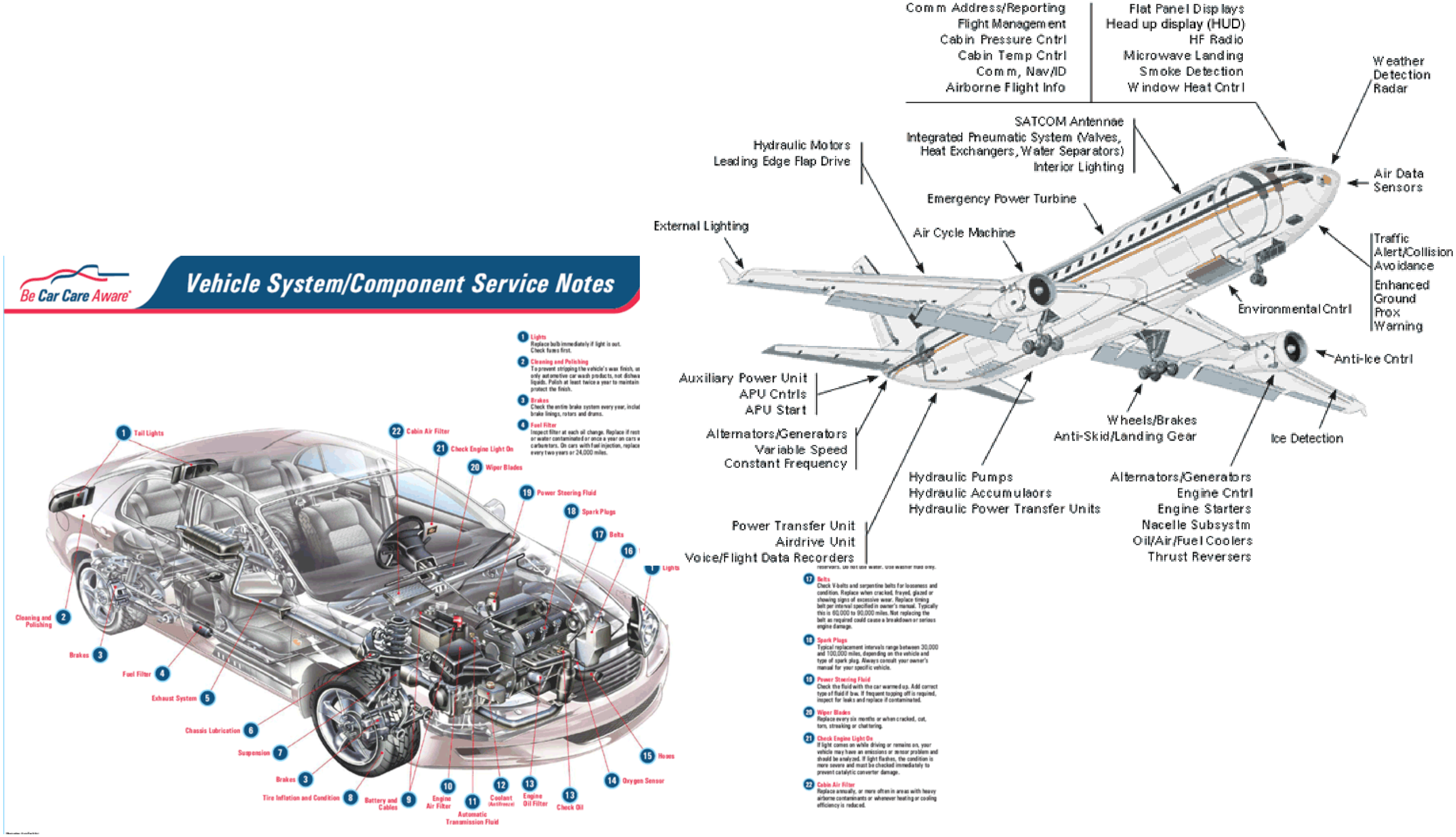
19 **Power Steering Fluid**
Check the fluid with the car warmed up. Add correct type of fluid if low. If frequent topping off is required, inspect for leaks and replace if contaminated.

20 **Wiper Blades**
Replace every six months or when cracked, cut, torn, streaking or chattering.

21 **Check Engine Light On**
If light comes on while driving or remains on, your vehicle may have an emissions or sensor problem and should be analyzed. If light flashes, the condition is more severe and must be checked immediately to prevent catalytic converter damage.

22 **Cabin Air Filter**
Replace annually, or more often in areas with heavy airborne contaminants or whenever heating or cooling efficiency is reduced.

Motivating Examples: Monitoring Vehicle Operation



1 **Lights**
Replace bulb immediately if light is out. Check fuses first.

2 **Cleaning and Polishing**
To prevent stripping the vehicle's wax finish, us only automotive car wash products, not dishwa liquids. Polish at least twice a year to maintain protect the finish.

3 **Brakes**
Check the entire brake system every year, includ brake linings, rotors and drums.

4 **Fuel Filter**
Inspect filter at each oil change. Replace if rest or water contaminated or once a year on cars w carburetors. On cars with fuel injection, replace every two years or 24,000 miles.

reservoirs. Do not use water. Use washer fluid only.

17 **Belts**
Check V-belts and serpentine belts for looseness and condition. Replace when cracked, frayed, glazed or showing signs of excessive wear. Replace timing belt per interval specified in owner's manual. Typically this is 60,000 to 90,000 miles. Not replacing the belt as required could cause a breakdown or serious engine damage.

18 **Spark Plugs**
Typical replacement intervals range between 30,000 and 100,000 miles, depending on the vehicle and type of spark plug. Always consult your owner's manual for your specific vehicle.

19 **Power Steering Fluid**
Check the fluid with the car warmed up. Add correct type of fluid if low. If frequent topping off is required, inspect for leaks and replace if contaminated.

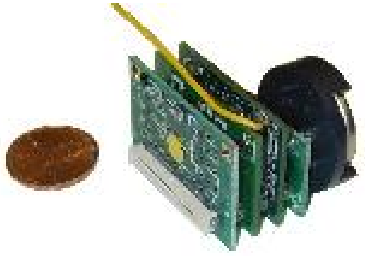
20 **Wiper Blades**
Replace every six months or when cracked, cut, torn, streaking or chattering.

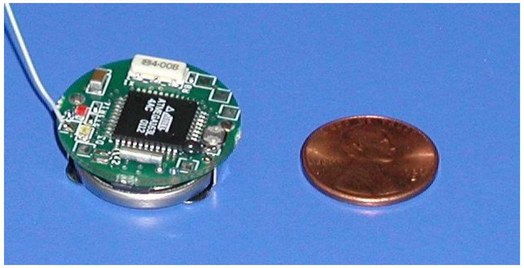
21 **Check Engine Light On**
If light comes on while driving or remains on, your vehicle may have an emissions or sensor problem and should be analyzed. If light flashes, the condition is more severe and must be checked immediately to prevent catalytic converter damage.

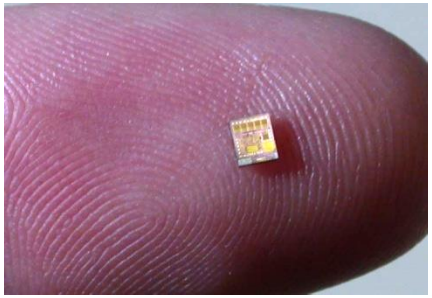
22 **Cabin Air Filter**
Replace annually, or more often in areas with heavy airborne contaminants or whenever heating or cooling efficiency is reduced.

Motivating Examples: Sensor Networks

- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology





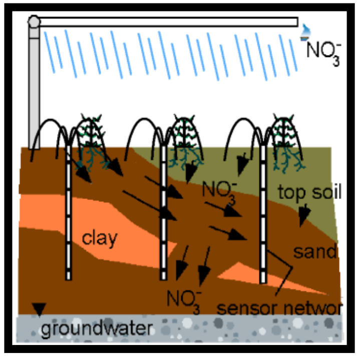


Motivating Examples: Sensor Networks

- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology

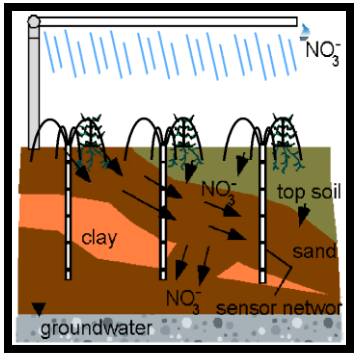
Motivating Examples: Sensor Networks

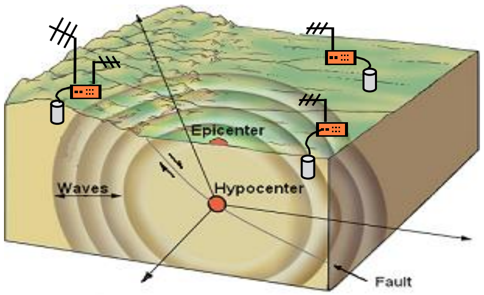
- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology



Motivating Examples: Sensor Networks

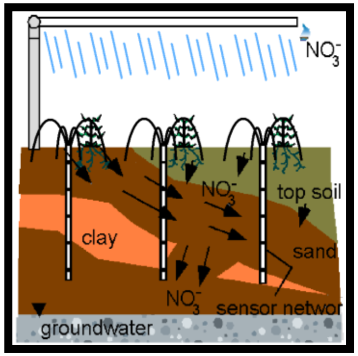
- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology

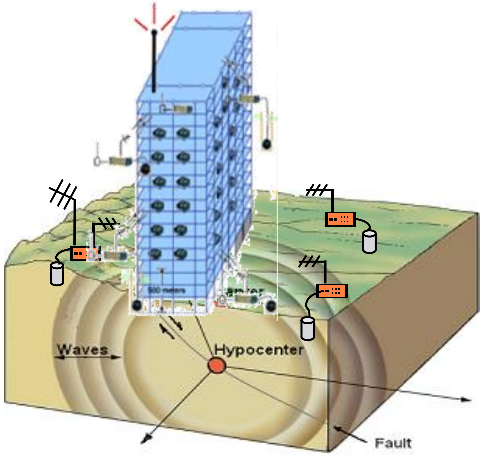




Motivating Examples: Sensor Networks

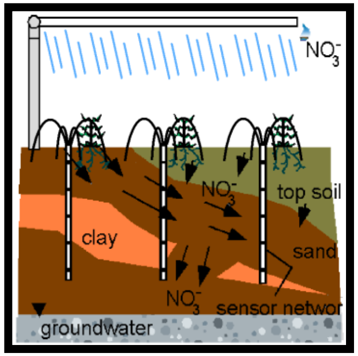
- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology

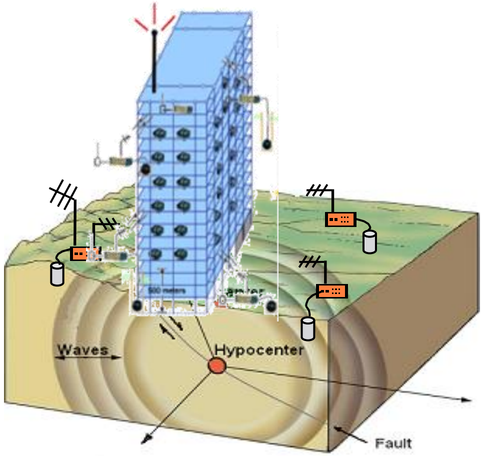




Motivating Examples: Sensor Networks

- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology

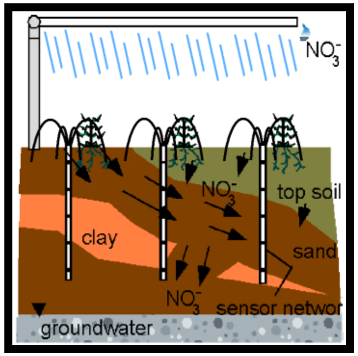


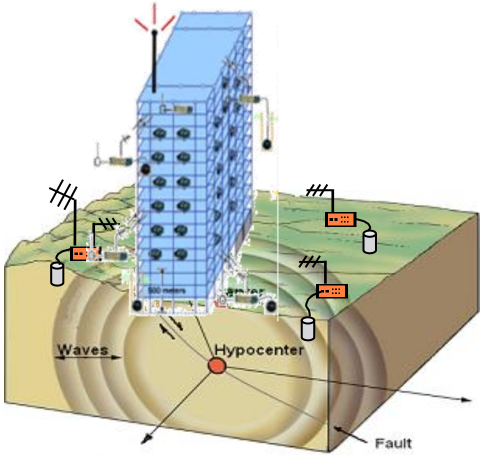


Motivating Examples:
Sensor Networks

- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology

- sensors unveil previously unobservable phenomena





Data as a Set
Data as a Sequence

- streaming data
  - window of interest
    - landmark window
    - sliding window (shifting window)

- may treat streaming data as a set, or as a sequence
  - depends on whether sequence is important

Data Series Anomalies Problem

- develop anomaly detection techniques based on sequences (data series), not on individual values
    - individual values can be normal, but their sequence can be abnormal!

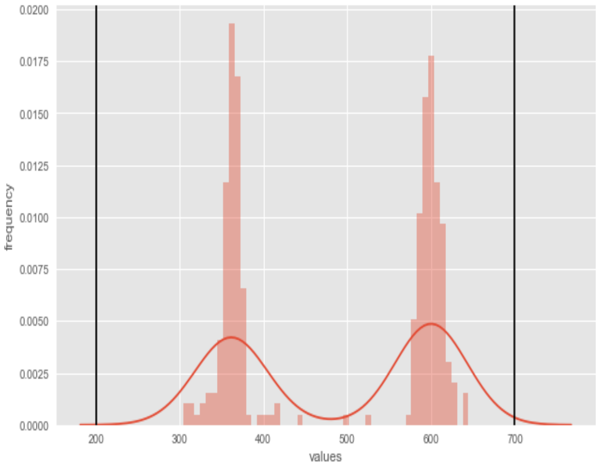
Data Series Anomalies Problem

- develop anomaly detection techniques based on sequences (data series), not on individual values
  - individual values can be normal, but their sequence can be abnormal!

150 points in a sequence S

Minimal critical value

Maximal critical value



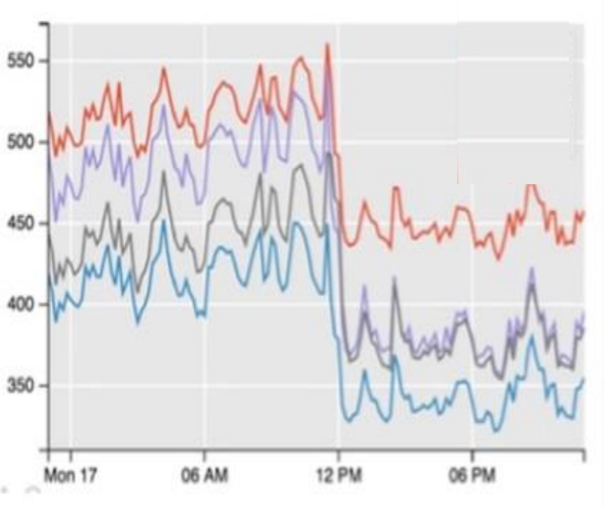
values are not outside critical thresholds

values are normal

Data Series Anomalies Problem

- develop anomaly detection techniques based on sequences (data series), not on individual values
  - individual values can be normal, but their sequence can be abnormal!

Sequence S



values are not outside critical thresholds

values are normal

sequences are abnormal

Data Series (Signal) Processing
Data Series Management

- lots of literature on data series processing
  - periodicity detection
  - data series modeling and forecasting
    - ARMA, ARIMA
  - outlier detection
    - focuses on next value

- we will focus on
  - sequences as first class citizens
  - very large sequence collections

Objectives

- get introduced to the data series data type
  - characteristics, properties, peculiarities
- learn about
  - data series representations
  - data series similarity matching
  - data series indexing
  - systems for data series management
  - challenges and open problems

Data Series Representations

Introduction

- lots of work on data series representations

Introduction

- lots of work on data series representations
  - techniques for representing/storing data series

Introduction

- lots of work on data series representations
  - techniques for representing/storing data series

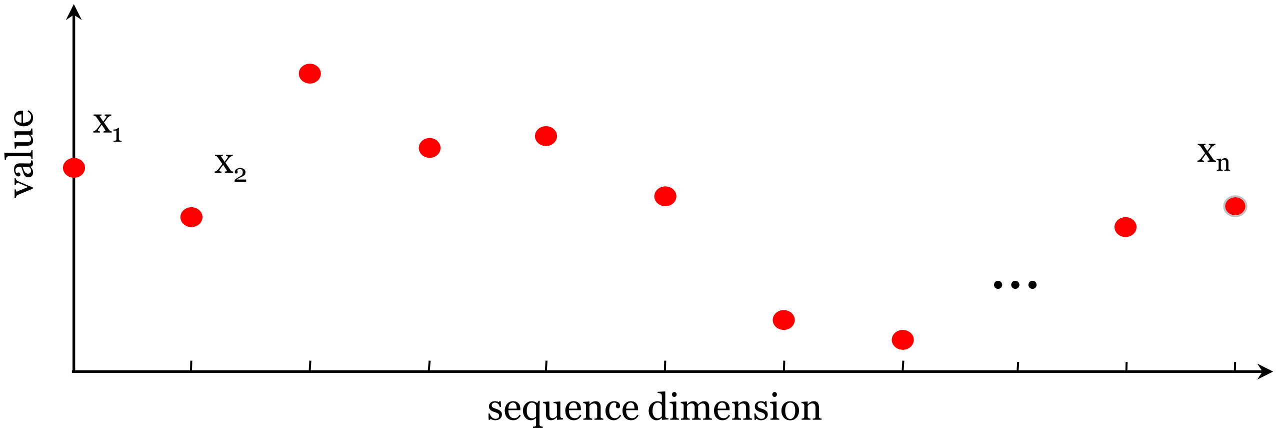
- main goal
  - summarize data series
  - render subsequent processing more efficient

Outline

- terminology and definitions
- motivation
- pre-processing tasks
- data series representation techniques

Data series

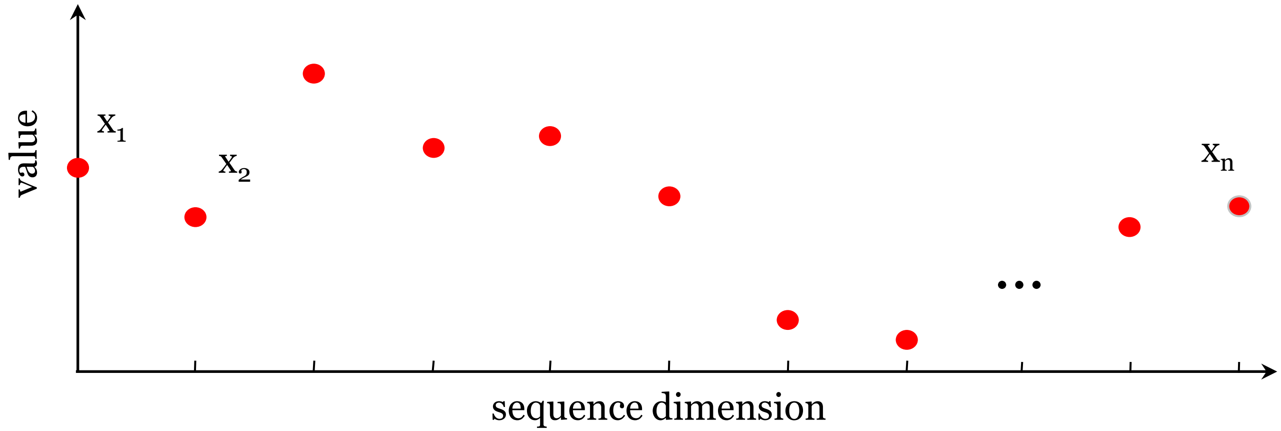
- Sequence of points ordered along some dimension



- terminology: we will use interchangeably
  - data series, time series, data sequence, sequence

Data series

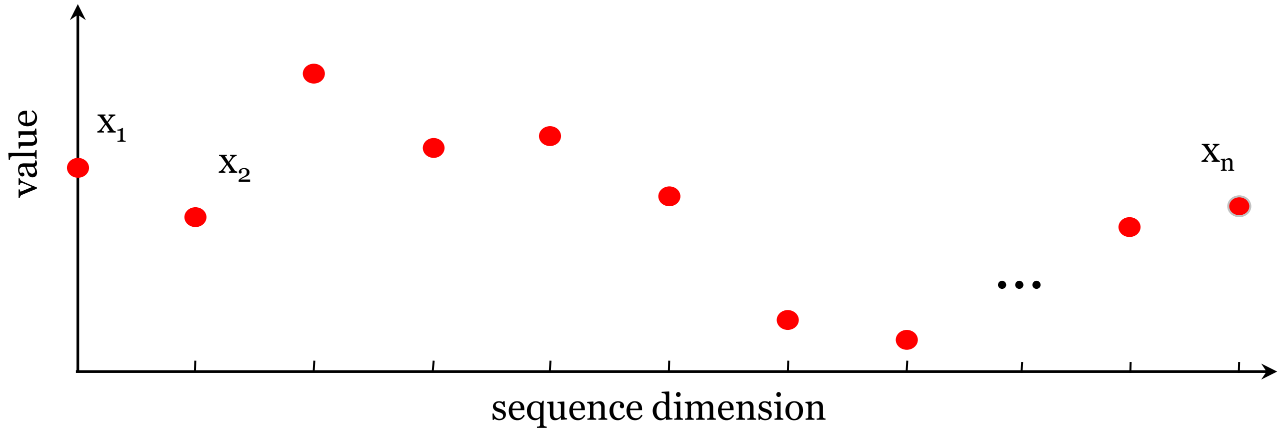
- Sequence of points ordered along some dimension



- number of data series values, n
  - length, or dimensionality

Data series

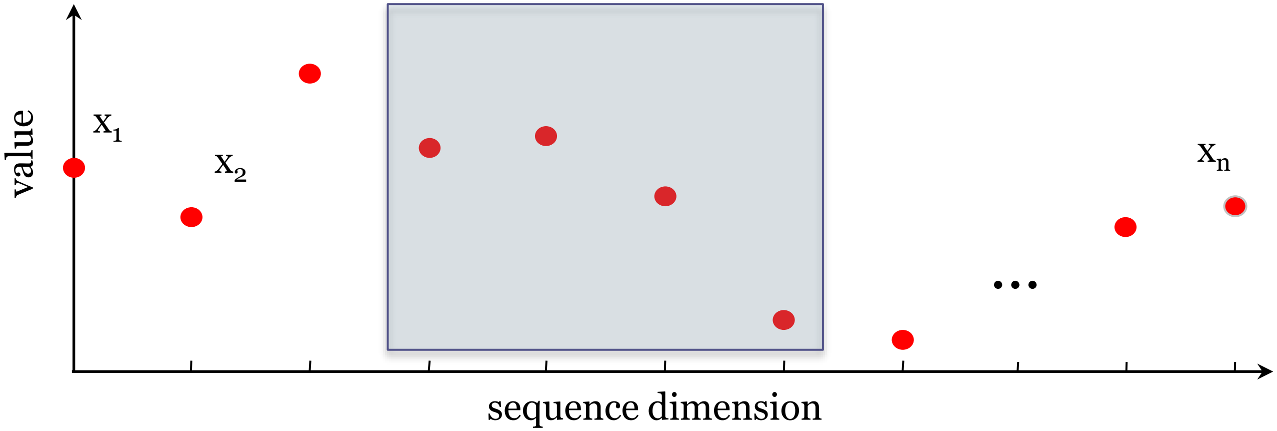
- Sequence of points ordered along some dimension



- subsequence
  - subset of contiguous values

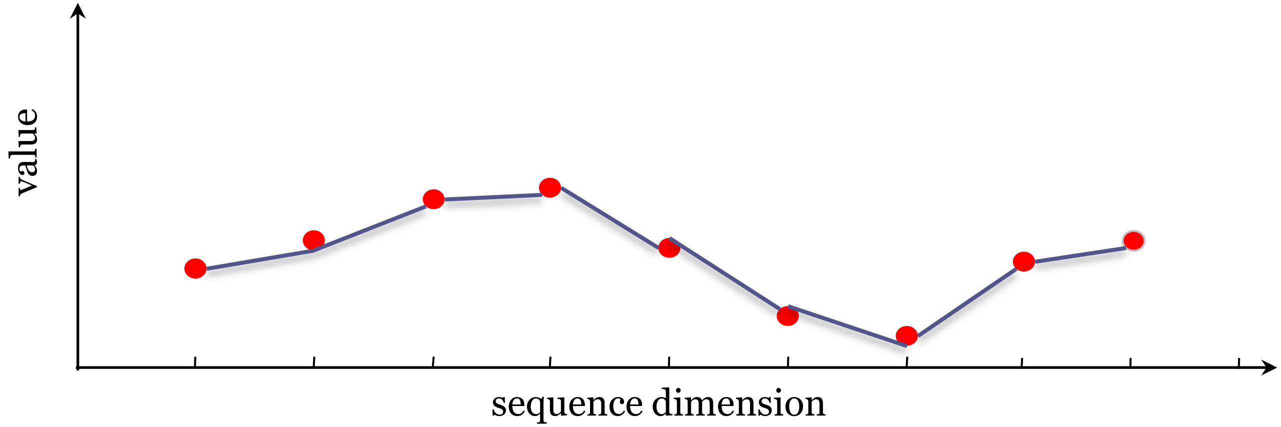
Data series

- Sequence of points ordered along some dimension

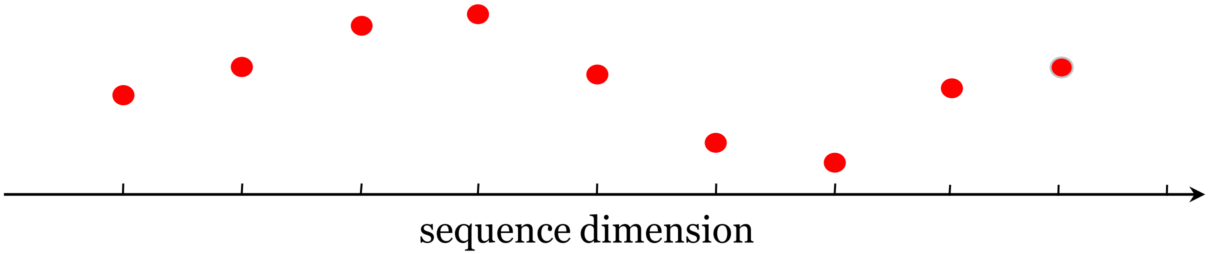


- subsequence
  - subset of contiguous values
  - eg, subsequence of length (dimensionality) 4

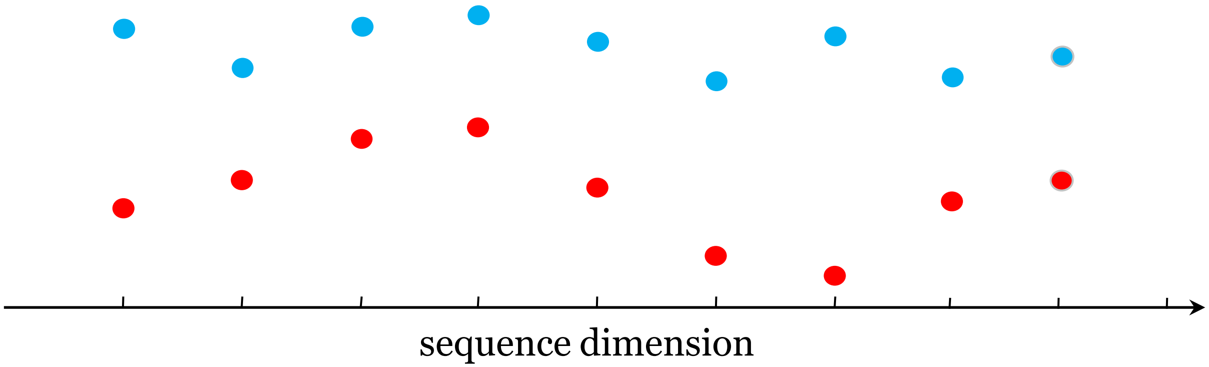
Data series
Distance



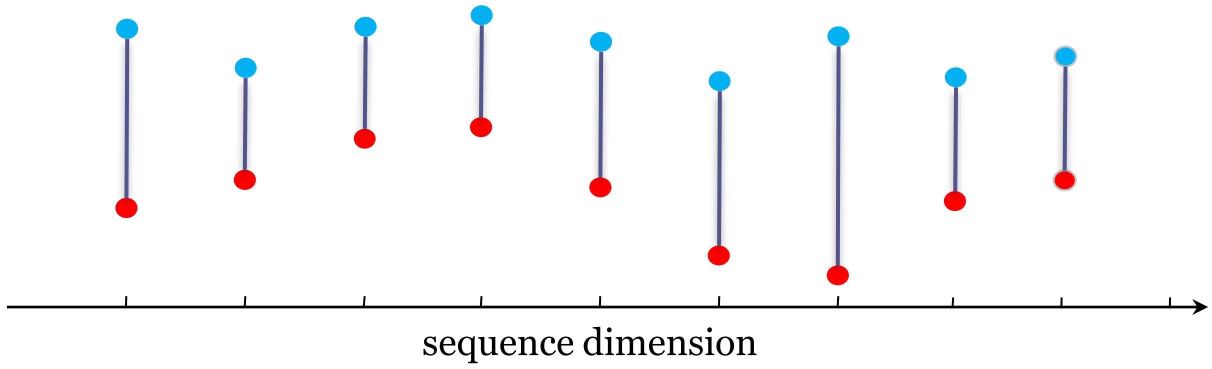
Data series
Distance



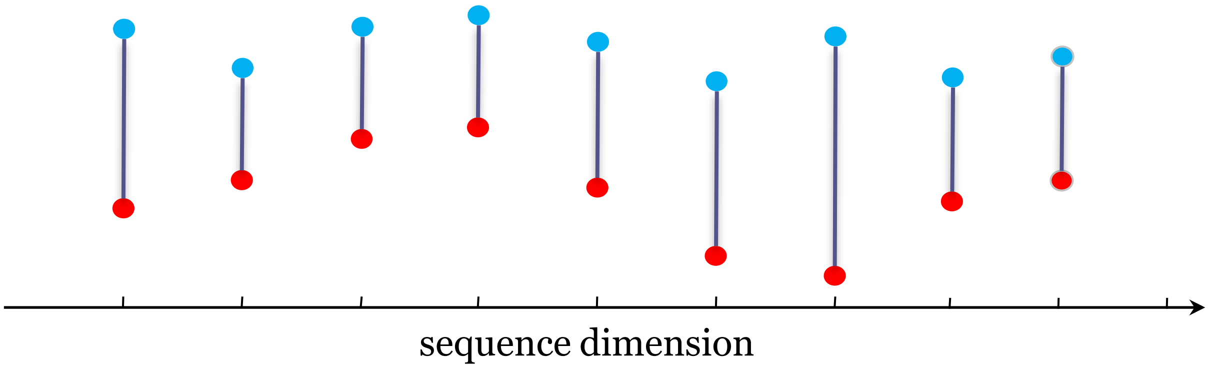
Data series Distance



Data series
Distance

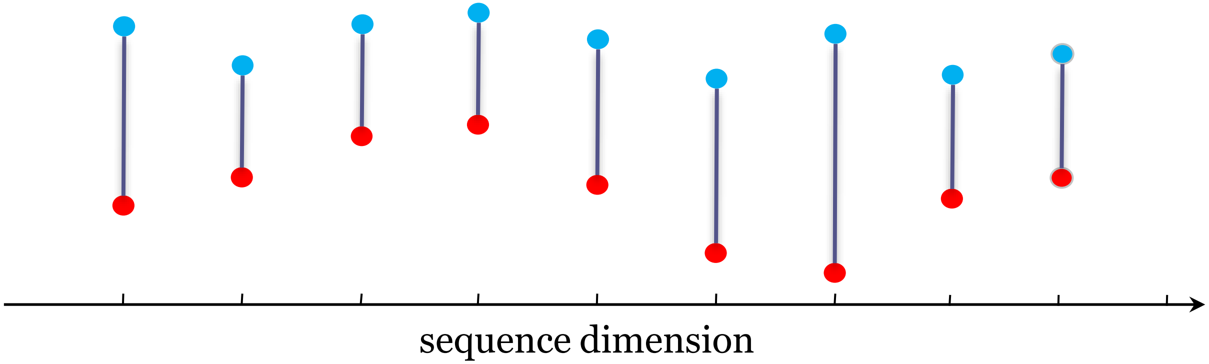


Data series Distance



- Euclidean distance
  - pair-wise point distance
  - $D(red, blue) = \sqrt{\sum_{i=1}^{n}(red_i \; - blue_i \;)^2}$

Data series
Reconstruction Error

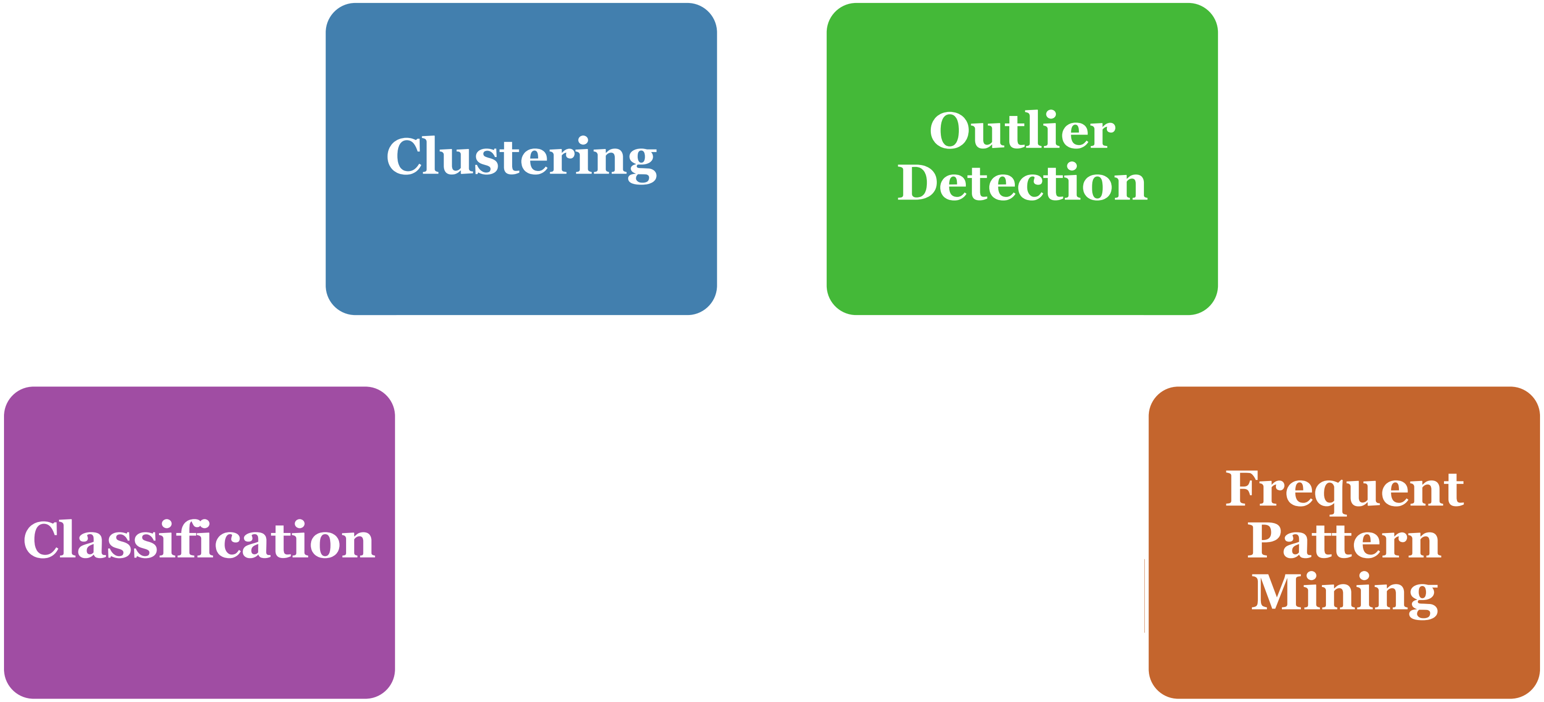


- Euclidean distance
  - pair-wise point distance
  - $D(red, blue) = \sqrt{\sum_{i=1}^{n}(red_i \ - blue_i \ )^2}$

Outline

- terminology and definitions
- motivation
- pre-processing tasks
- data series representation techniques

Analysis Tasks



Clustering

Outlier Detection

Classification

Frequent Pattern Mining

Analysis Tasks

- analyze evolution of values across x-dimension
- identify trends

Analysis Tasks

- analyze evolution of values across x-dimension
- identify trends

- treat data series as a first class citizen
  - analyze each data series as a single object
  - process all n-dimensions at once

Analysis Tasks
Subsequences

- often times the data series are very long
  - n >> 1
  - streaming data series

Analysis Tasks
Subsequences

- often times the data series are very long
  - n >> 1
  - streaming data series

- we then chop the long sequence in subsequences
  - e.g., using sliding window, or shifting window
  - pick carefully length of subsequence
    - should contain patterns of interest
- and process each subsequence separately

Analysis Tasks

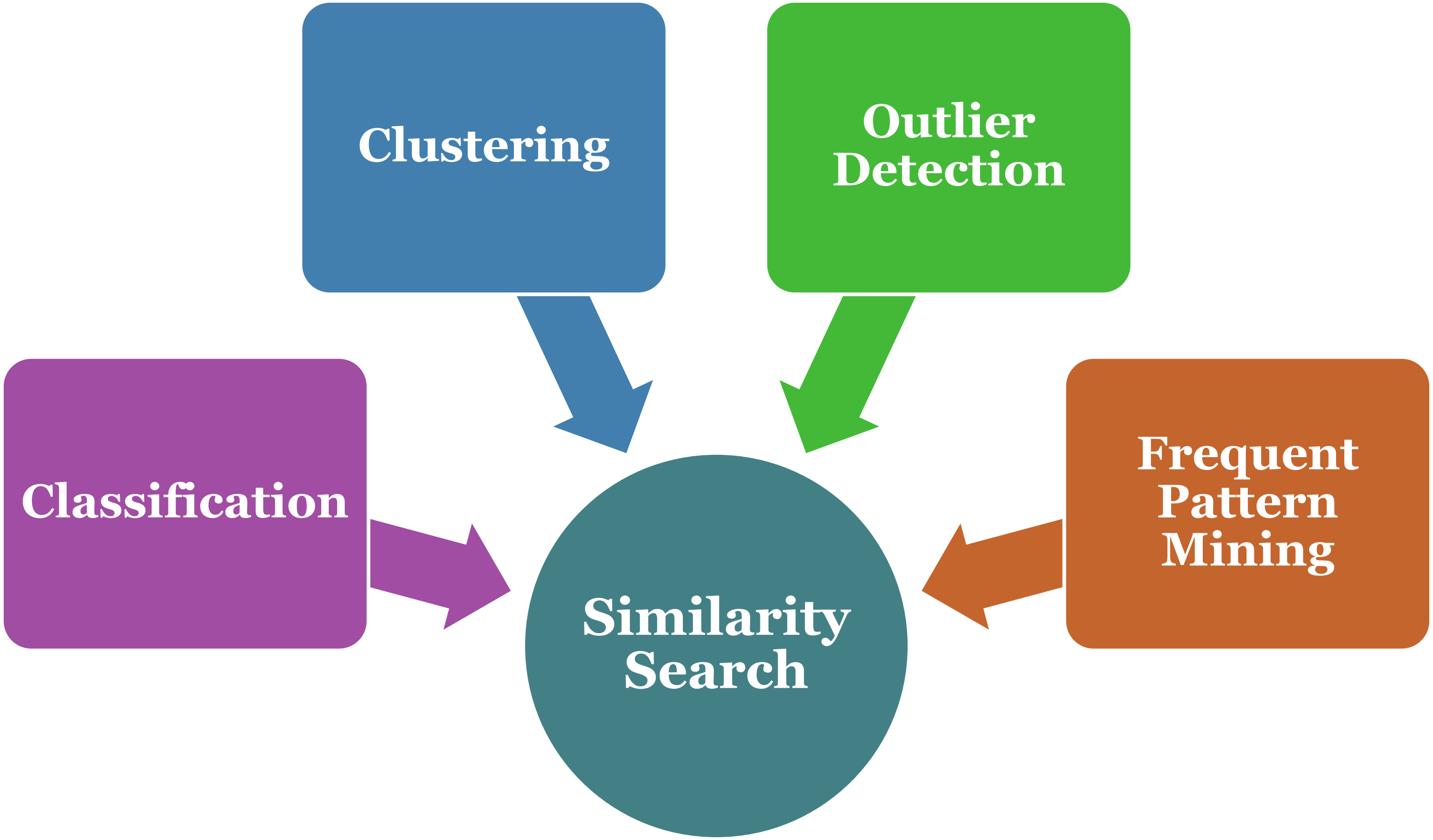
select values in time interval

select values in some range

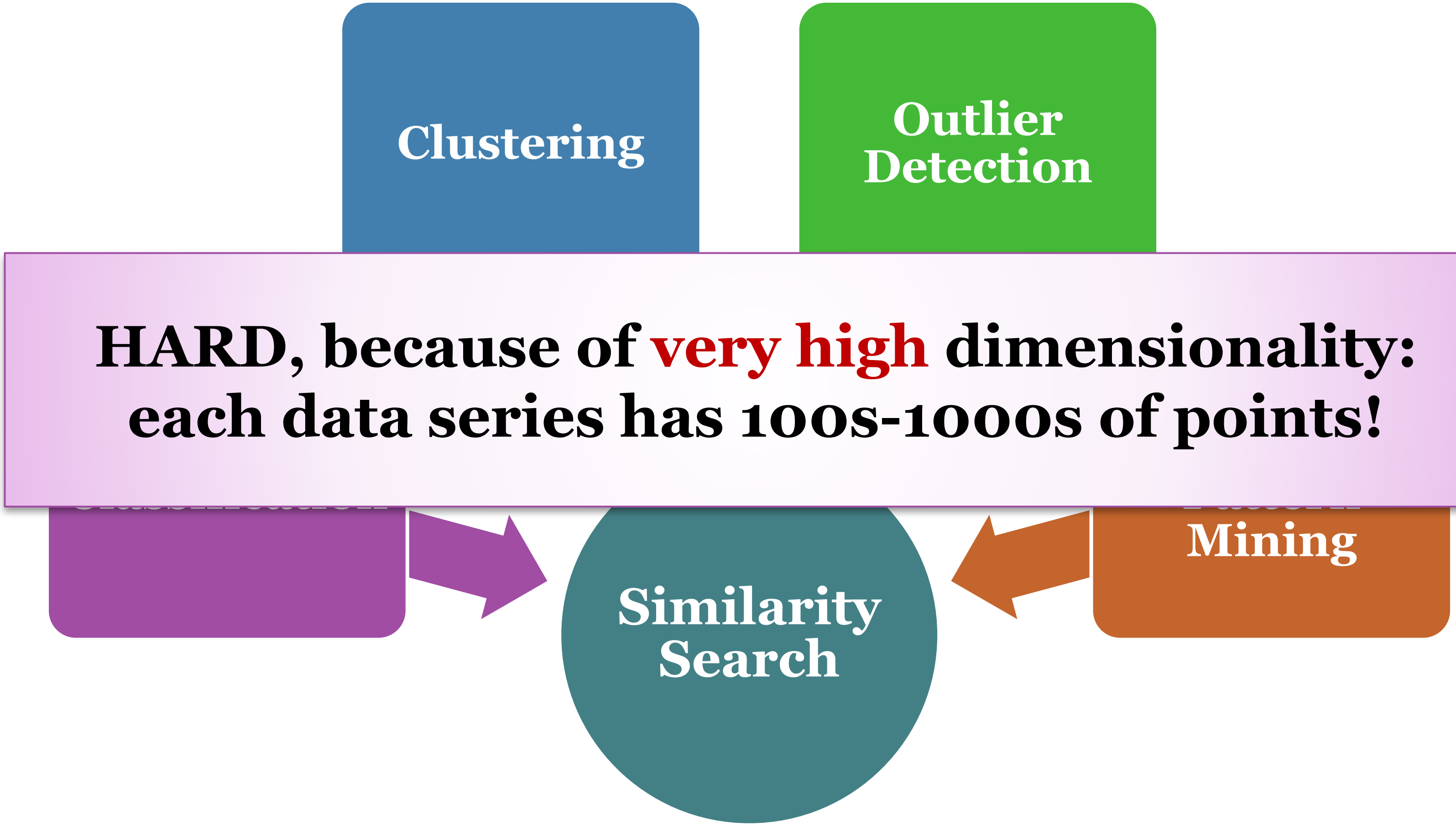
select some data series

combinations of those

Analysis Tasks

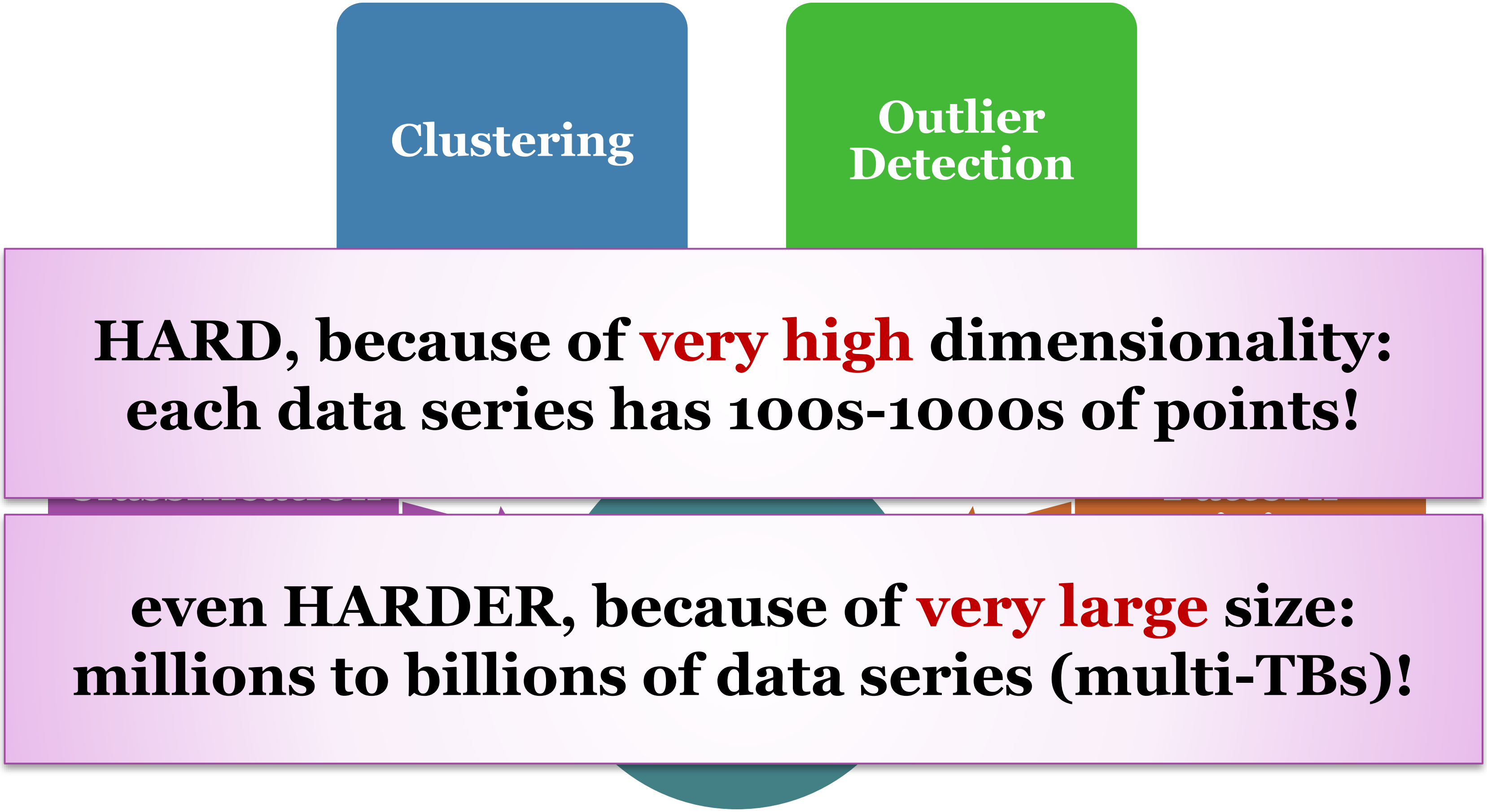


Analysis Tasks



HARD, because of very high dimensionality: each data series has 100s-1000s of points!

Analysis Tasks



HARD, because of very high dimensionality: each data series has 100s-1000s of points!

even HARDER, because of very large size: millions to billions of data series (multi-TBs)!

Motivation

- effective representation techniques to the rescue!
    - can significantly reduce the processing time
        - typically much smaller than original/raw data series

Motivation

- effective representation techniques to the rescue!
    - can significantly reduce the processing time
        - typically much smaller than original/raw data series

- will learn how to compute and use these representations

Motivation

- effective representation techniques to the rescue!
    - can significantly reduce the processing time
        - typically much smaller than original/raw data series
- will learn how to compute and use these representations
- these representations can further be used for indexing

Motivation

- effective representation techniques to the rescue!
  - can significantly reduce the processing time
    - typically much smaller than original/raw data series
- will learn how to compute and use these representations
- these representations can further be used for indexing
- all guarantee correct, exact results!

Outline

- terminology and definitions
- motivation
- pre-processing tasks
- data series representation techniques

Pre-Processing
z-Normalization

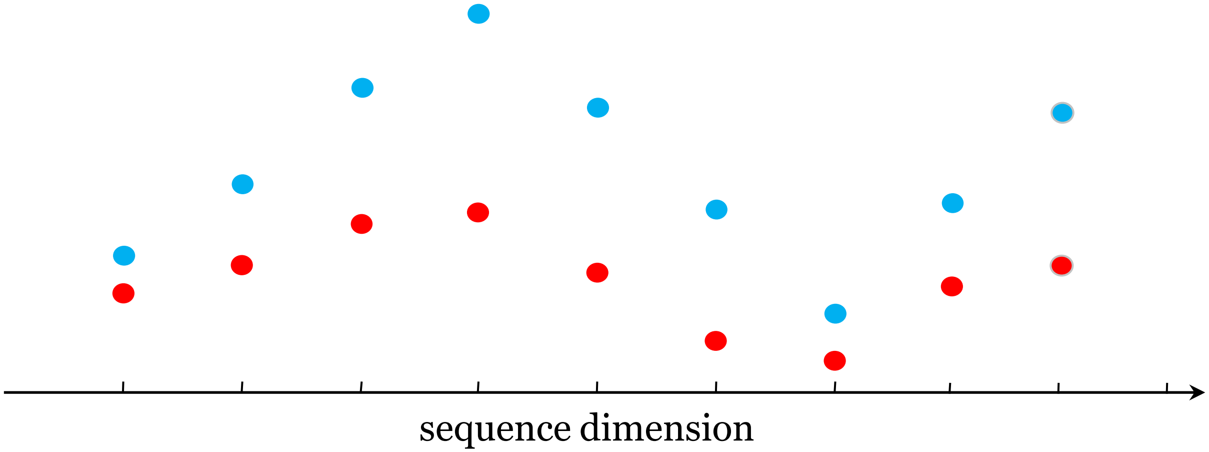
- data series encode trends
- usually interested in identifying similar trends

Pre-Processing
z-Normalization

- data series encode trends
- usually interested in identifying similar trends

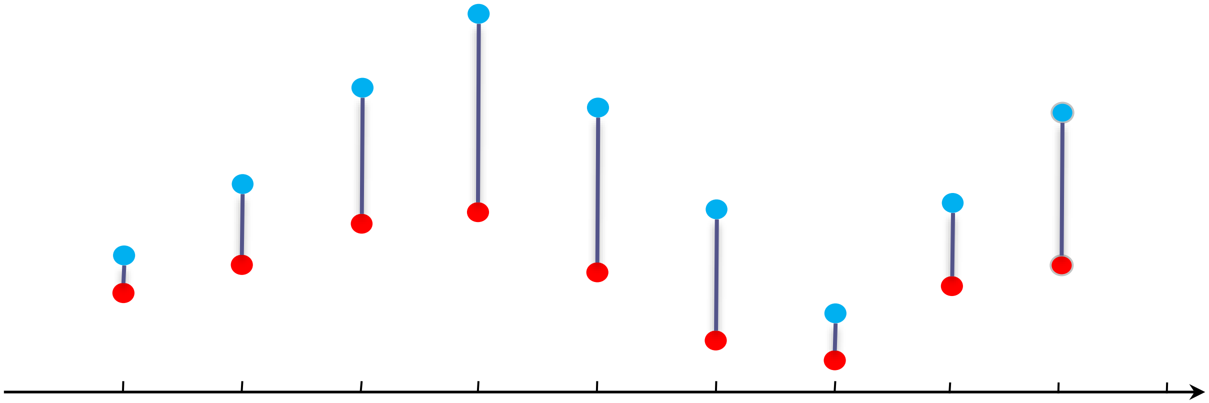
- but absolute values may mask this similarity

Pre-Processing
z-Normalization



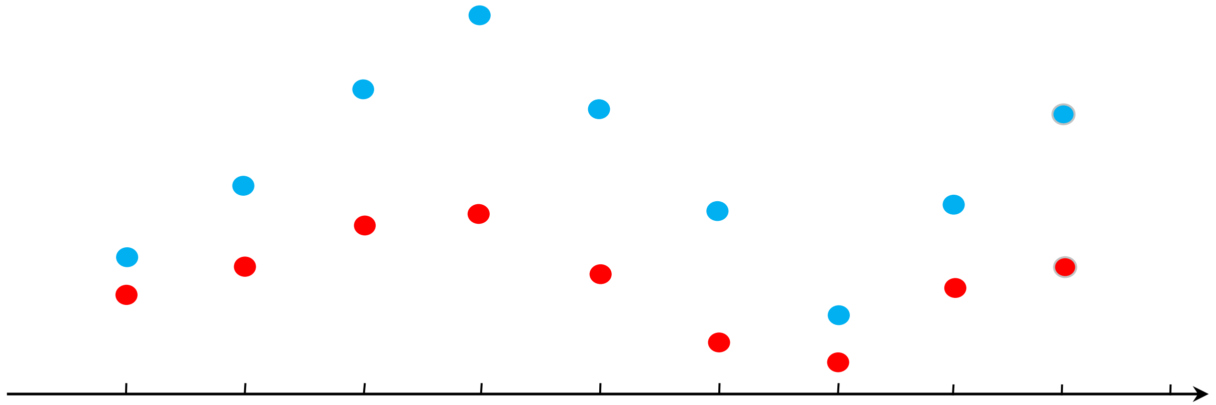
- two data series with similar trends

Pre-Processing
z-Normalization



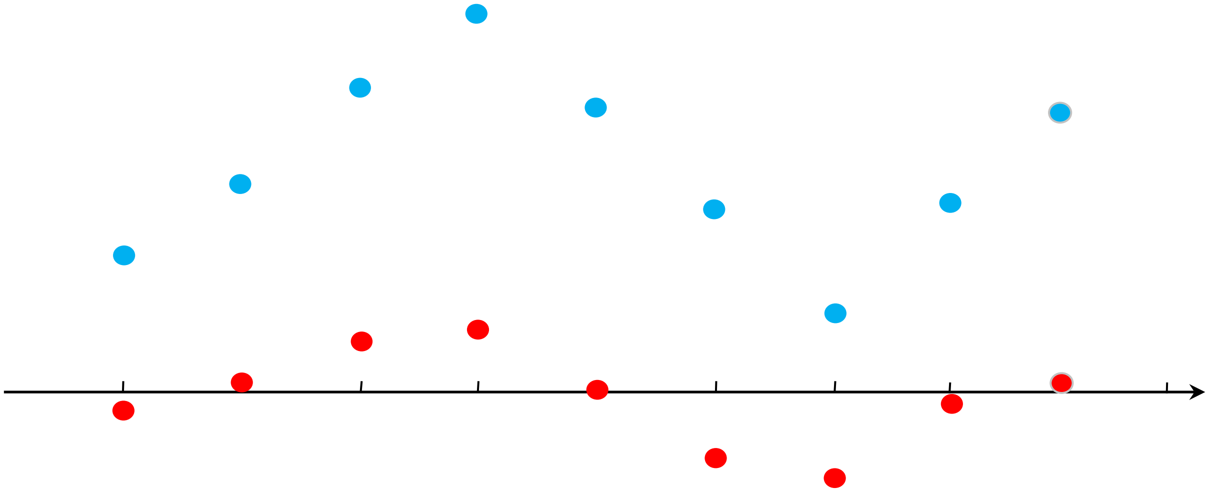
- two data series with similar trends
- but large distance…

Pre-Processing
z-Normalization



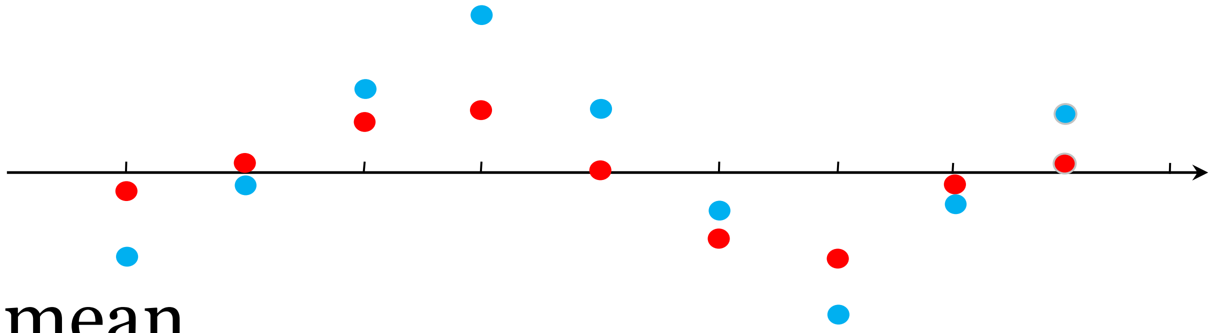
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing
z-Normalization



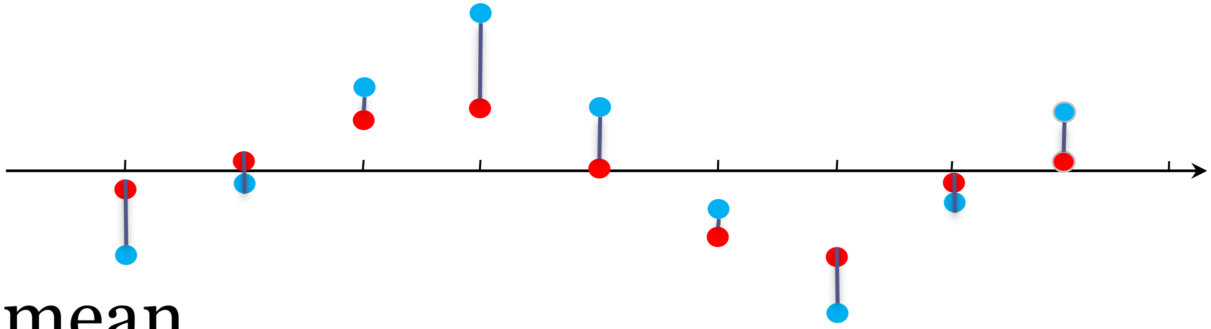
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing z-Normalization



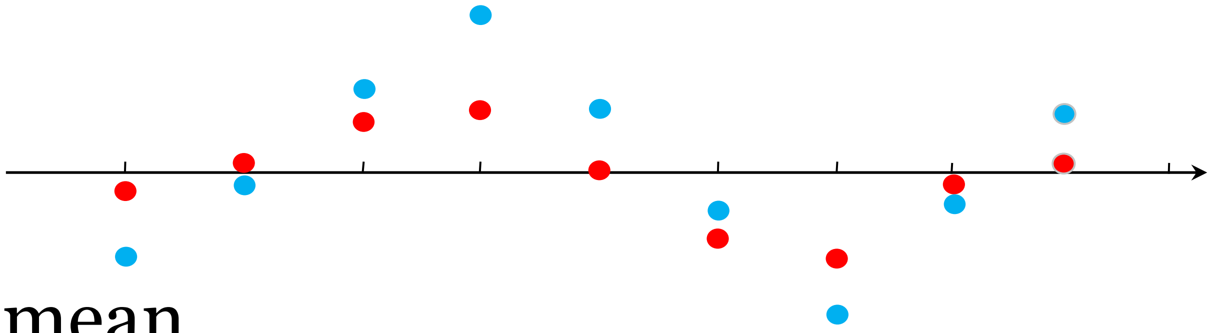
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing
z-Normalization



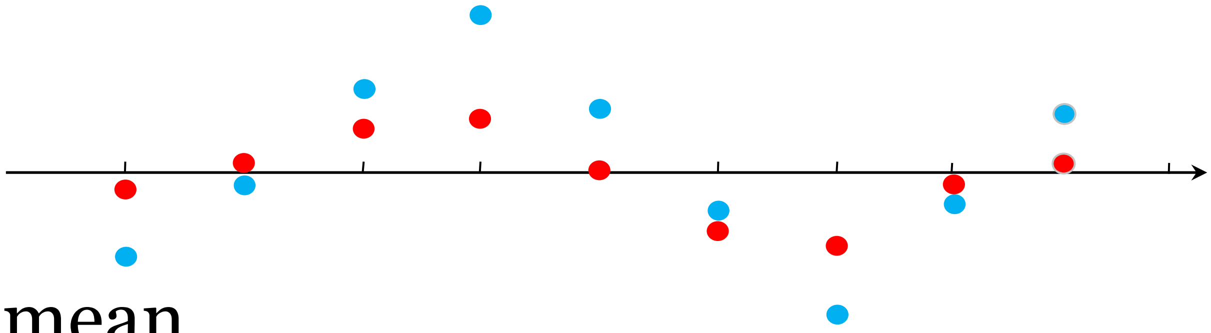
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing
z-Normalization



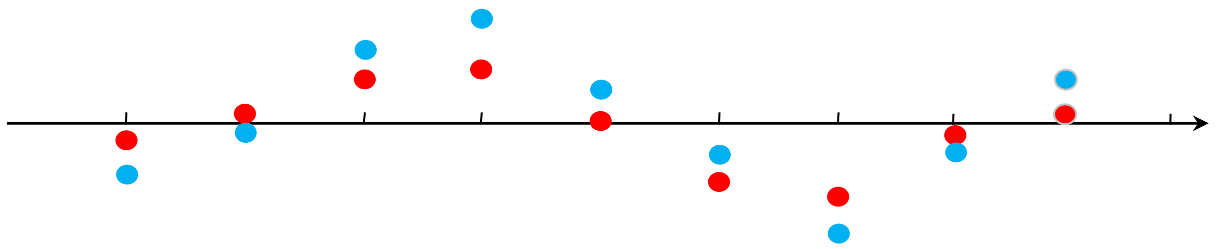
- zero mean
- standard deviation one
  - compute the standard deviation of the sequence
  - divide every value of the sequence by the stddev

Pre-Processing
z-Normalization



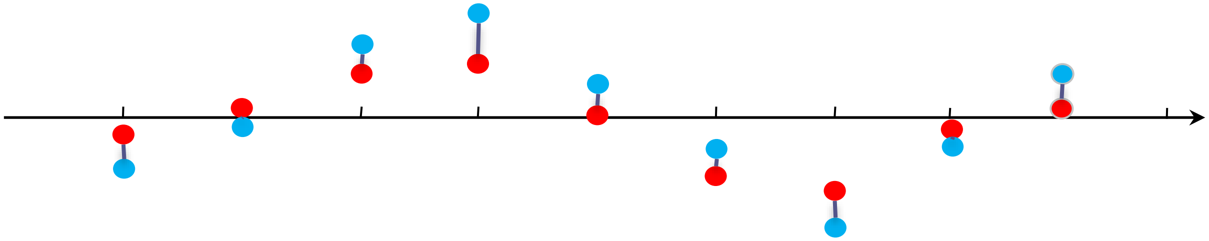
- zero mean
- standard deviation one
  - compute the standard deviation of the sequence
  - divide every value of the sequence by the stddev

Pre-Processing
z-Normalization



- zero mean
- standard deviation one
  - ▫ compute the standard deviation of the sequence
  - ▫ divide every value of the sequence by the stddev

Pre-Processing
z-Normalization



- zero mean
- standard deviation one

Pre-Processing z-Normalization

- when to z-normalize
  - interested in trends

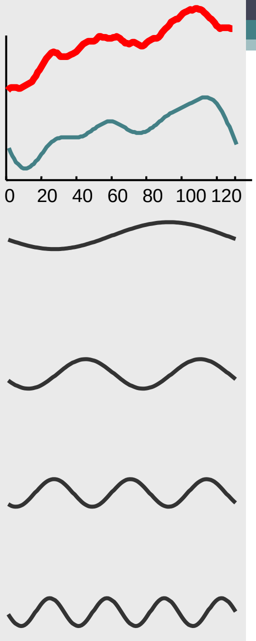
Pre-Processing z-Normalization

- when to z-normalize
  - interested in trends

- when not to z-normalize
  - interested in absolute values

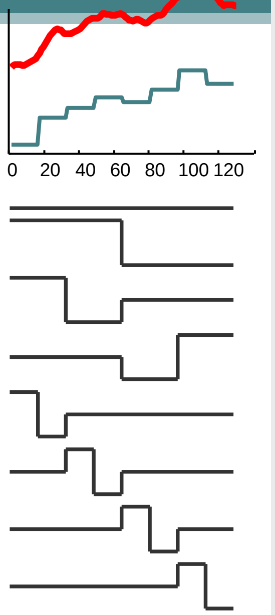
Outline

- terminology and definitions
- motivation
- pre-processing tasks
- data series representation techniques



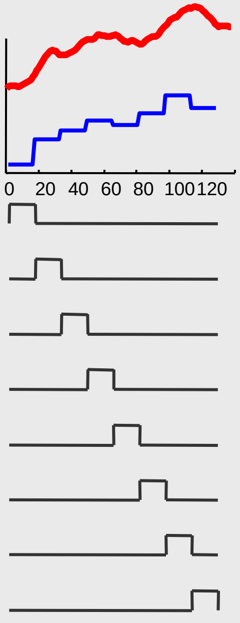
DFT

Agrawal, Faloutsos, &. Swami.
FODO 1993
Faloutsos, Ranganathan, &
Manolopoulos. SIGMOD 1994



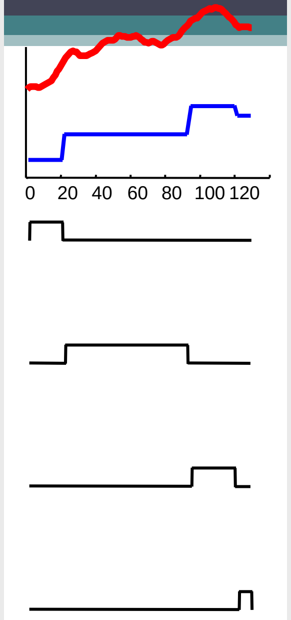
DWT

Chan & Fu. ICDE 1999



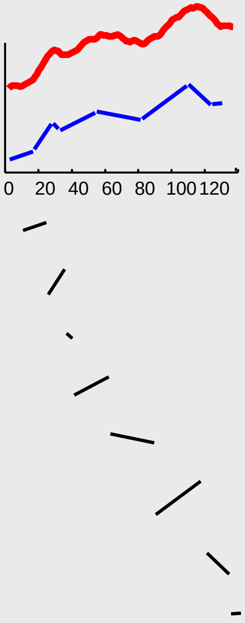
PAA

Keogh, Chakrabarti, Pazzani &
Mehrotra KAIS 2000
Yi & Faloutsos VLDB 2000



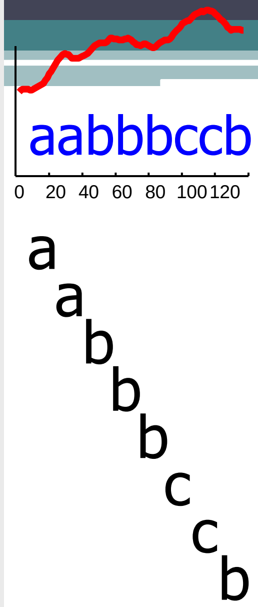
APCA

Keogh, Chakrabarti, Pazzani &
Mehrotra SIGMOD 2001



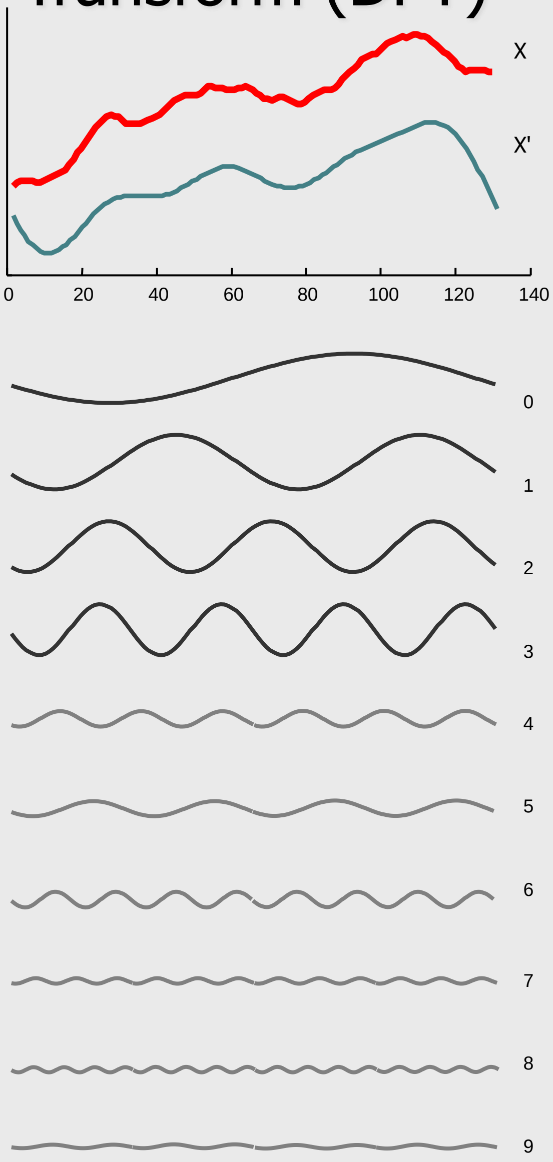
PLA

Morinaka,
Yoshikawa, Amagasa, &
Uemura, PAKDD 2001



SAX

Discrete Fourier Transform (DFT)



Basic Idea: Represent the time series as a linear combination of sines and cosines

Transform the data from the time domain to the frequency domain

Highlight the periodicities but keep only the first *n/2* coefficients

Jean Fourier

1768-1830

Why *n*/2 coefficients?

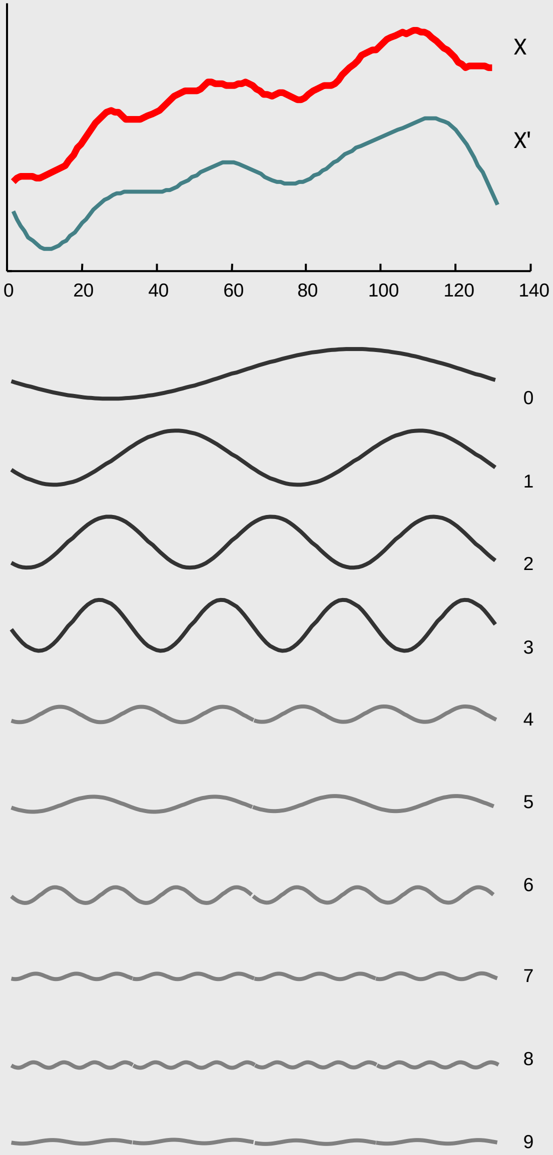
- ✓ Because they are symmetric

Excellent free Fourier Primer

Hagit Shatkay, The Fourier Transform - a Primer", Technical Report CS-95-37, Department of Computer Science, Brown University, 1995.

http://www.ncbi.nlm.nih.gov/CBBresearch/Postdocs/Shatkay/

Discrete Fourier Transform…recap



Pros and Cons of DFT as a time series representation

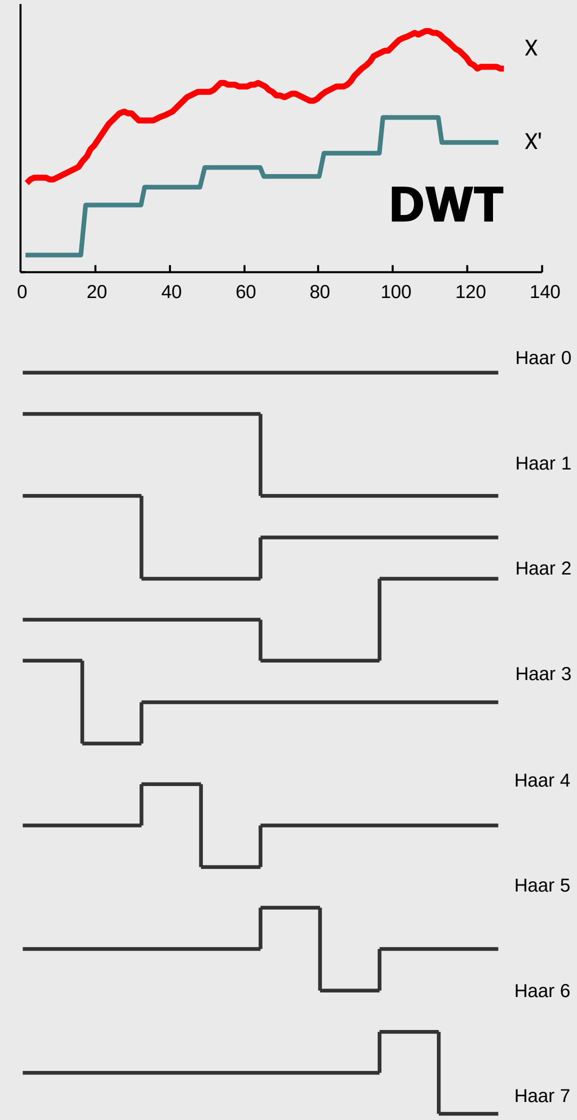
Pros:

- Good ability to compress most natural signals
- Fast, off the shelf DFT algorithms exist O(*n*log(*n*))

Cons:

- Difficult to deal with sequences of different lengths

Discrete Wavelet Transform (DWT)



Basic Idea: Represent the time series as a linear combination of Wavelet basis functions, but keep only the first *N* coefficients

Obtained from a single prototype wavelet *ψ(t)* called *mother wavelet* by *dilations* and *shifting:*

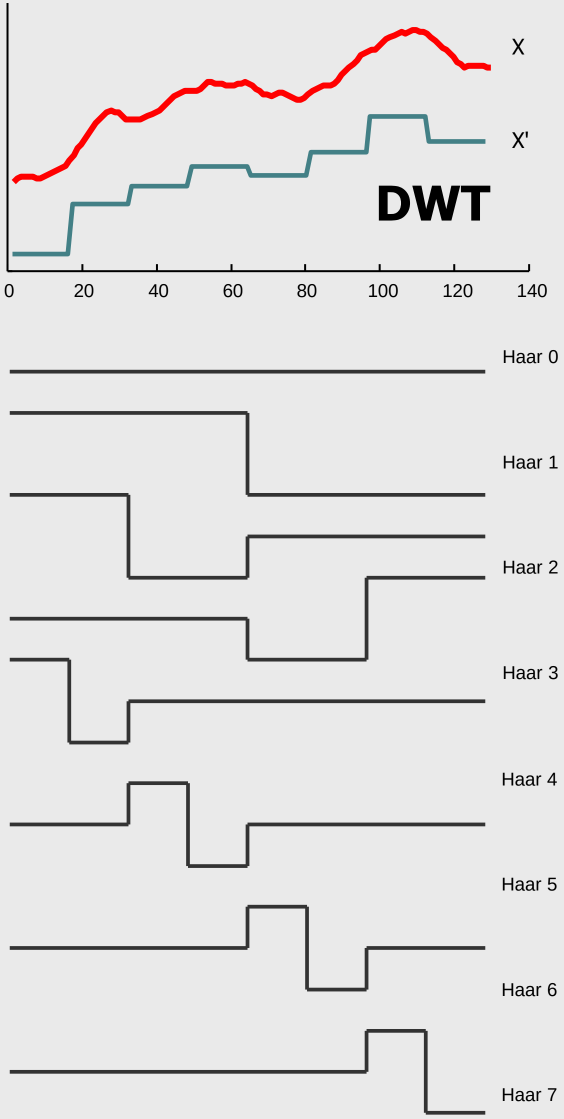
$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a})$$

where *a* is the scaling parameter and *b* is the shifting parameter

Excellent free Wavelets Primer

Stollnitz, E., DeRose, T., & Salesin, D. (1995). *Wavelets for computer graphics A primer: IEEE Computer Graphics and Applications.*

Discrete Wavelet Transform (DWT)



Pros and Cons of DWT as a time series representation

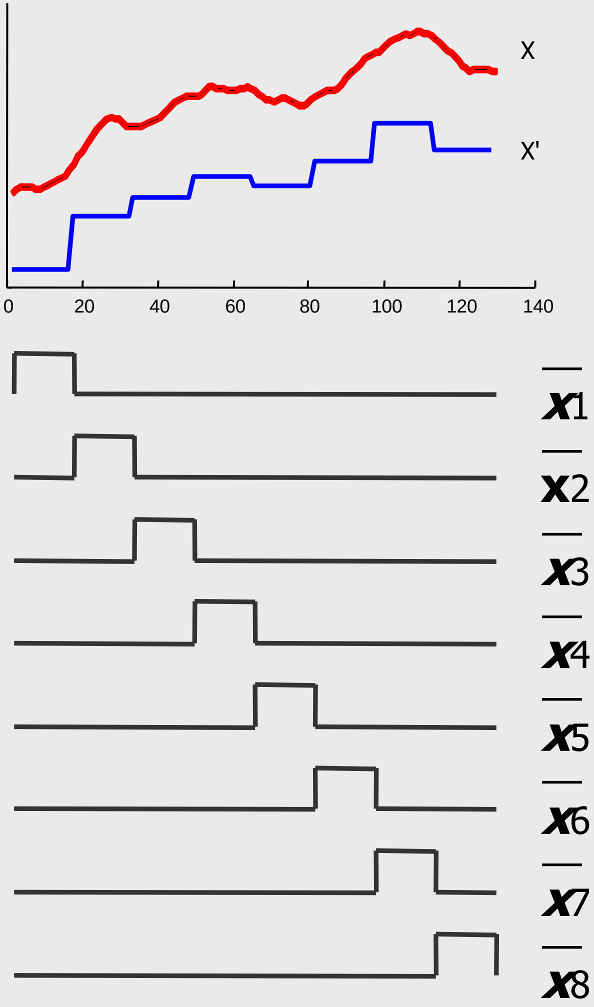
Pros:

- Good ability to compress stationary signals
- Can be computed in linear time

Cons:

- Signals must have a length $n = 2^{\text{some\_integer}}$
- Works best if N is $= 2^{\text{some\_integer}}$; Otherwise wavelets approximate the left side of signal at the expense of the right side

Piecewise Aggregate Approximation (PAA)



Basic Idea: Represent the time series as a sequence of box basis functions, each box being of the same length

Computation:

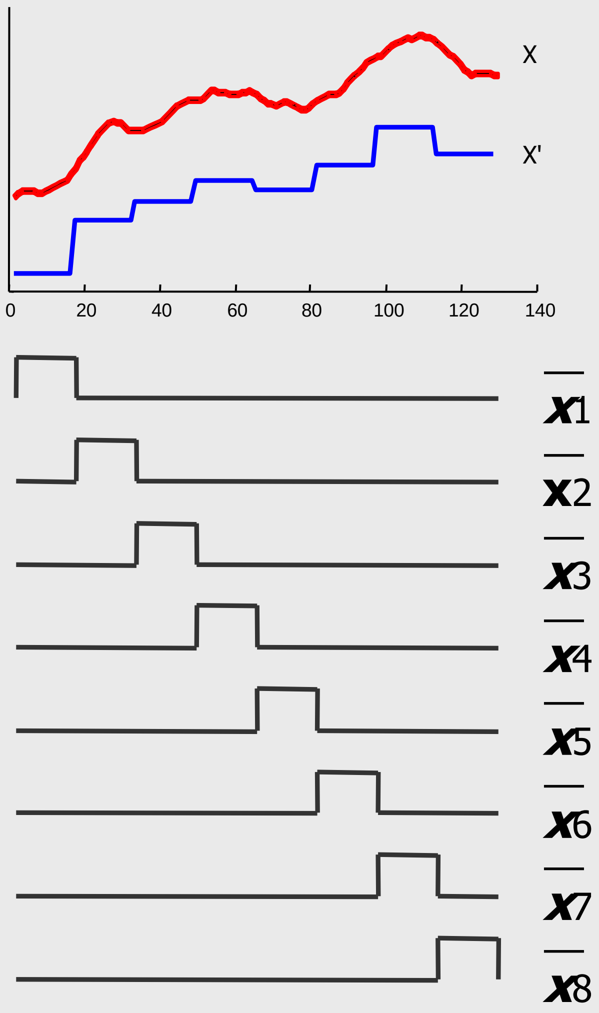
- X: time series of length n
- Can be represented in the N-dimensional space as:

$$\bar{x}_i = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} x_j$$

Keogh, Chakrabarti, Pazzani & Mehrotra, KAIS (2000)

Byoung-Kee Yi, Christos Faloutsos, VLDB (2000)

Piecewise Aggregate Approximation (PAA)



Pros and Cons of PAA as a time series representation.

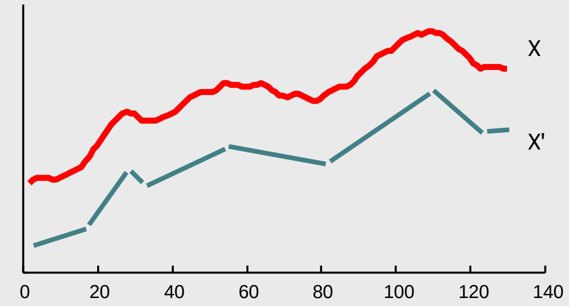
Pros:

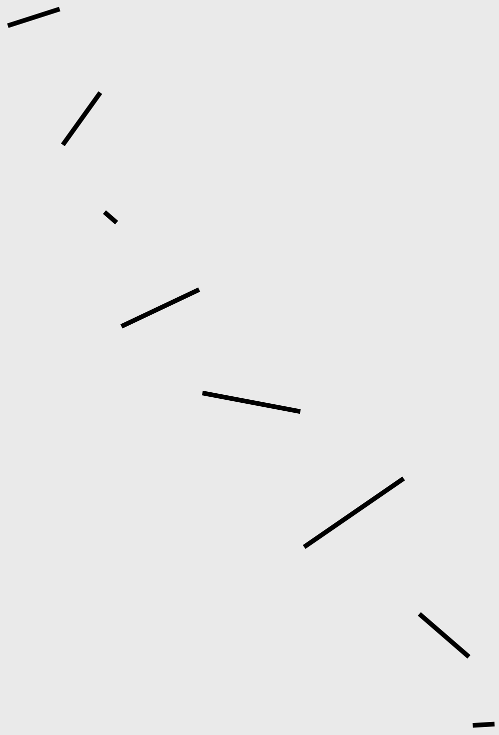
- *Extremely* fast to calculate
- As efficient as other approaches (empirically)
- Support queries of arbitrary lengths
- Can support any Minkowski metric
- Supports non Euclidean measures
- Supports weighted Euclidean distance
- *Simple!* Intuitive!

Cons:

- If visualized directly, looks aesthetically unpleasing

Piecewise Linear Approximation (PLA)





Basic Idea: Represent the time series (size n) as a sequence of straight lines (size N)

Lines could be **connected** **=> N/2** lines allowed

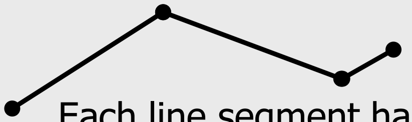
Lines could be **disconnected** **=> N/3** lines allowed

Empirical evidence on dozens of datasets suggests that **disconnected** is better

Also only **disconnected** allows a lower bounding Euclidean approximation

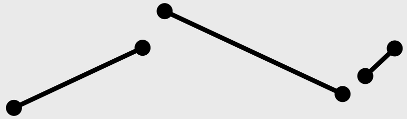
Karl Friedrich Gauss

1777 - 1855



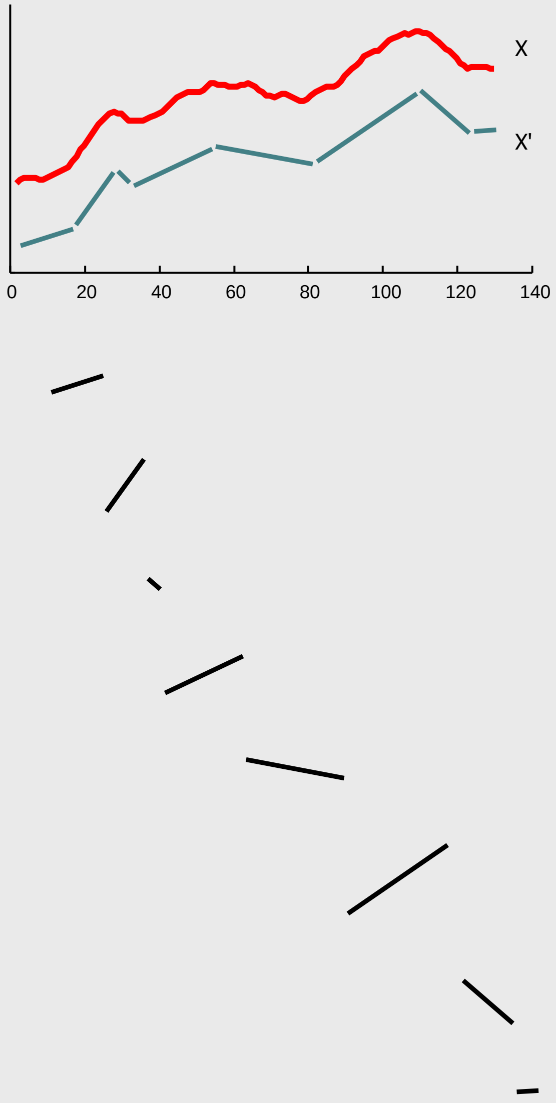
Each line segment has
- length
- left_height

(right_height can be inferred by looking at the next segment)



Each line segment has
- length
- left_height
- right_height

Piecewise Linear Approximation (PLA)



Pros and Cons of PLA as a time series representation

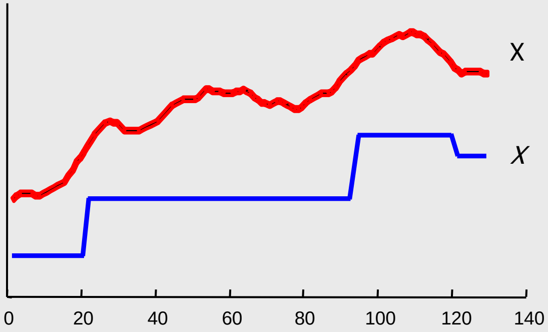
Pros:

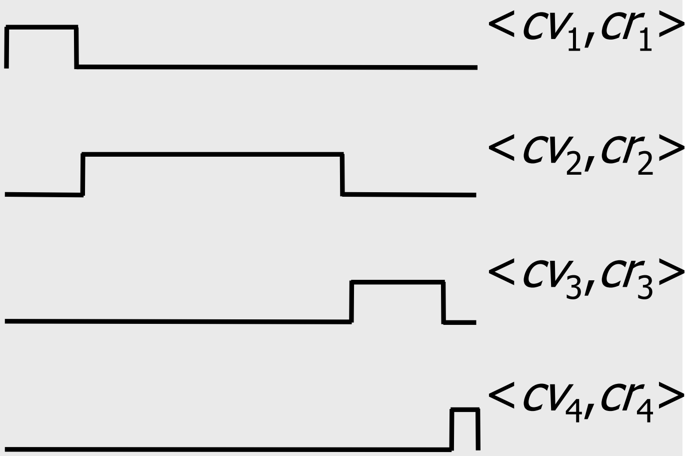
- Good ability to compress natural signals
- Fast linear time algorithms for PLA exist
- Able to support some interesting non-Euclidean similarity measures
- Already widely accepted in some communities (i.e., biomedical)

Cons:

- Not (currently) "indexable" by any data structure (but does allow fast sequential scan)

Adaptive Piecewise Constant Approximation (APCA)





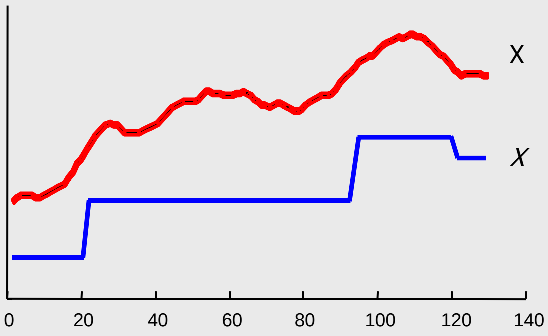
Basic Idea: Represent the time series as a sequence of box basis functions, each box being of the *different* length

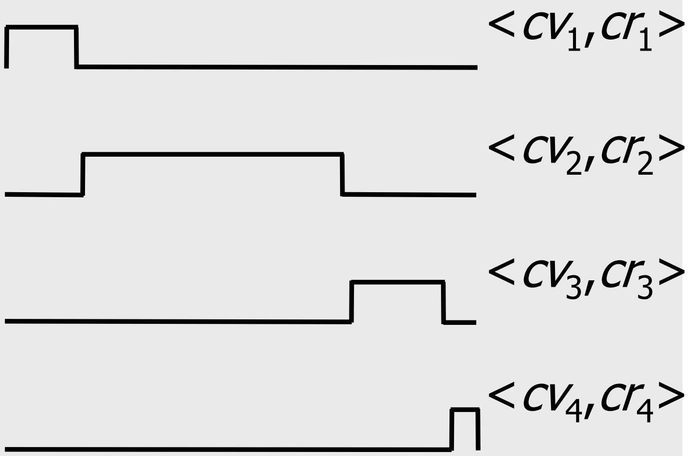
- High quality of APCA noted by many researchers
- Can be indexed*!

Unfortunately, it is non-trivial to understand and implement and thus has only been re-implemented once or twice

*K. Chakrabarti, E. J. Keogh, S. Mehrotra, M. J. Pazzani: Locally adaptive dimensionality reduction for indexing large time series databases. ACM Trans. Database Syst. 27(2): 188-228 (2002)

Adaptive Piecewise Constant Approximation (APCA)





Pros and Cons of APCA as a time series representation

Pros:

- Fast to calculate O(*n*)
- *More* efficient than other approaches
- Supports queries of arbitrary lengths
- Supports non Euclidean measures
- Support fast exact queries, and even faster approximate queries on the same data structure

Cons:

- Somewhat complex implementation
- If visualized directly, looks ascetically unpleasing

Symbolic ApproXimation (SAX)

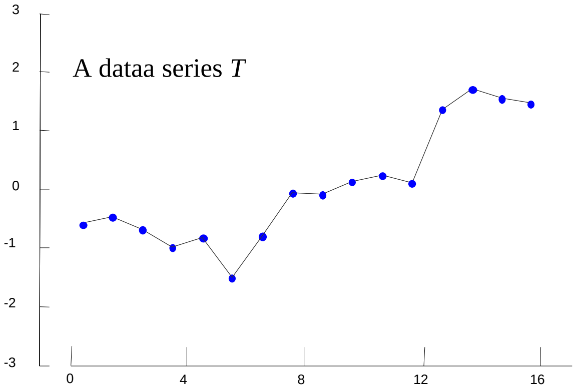
- similar in principle to PAA
  - uses segments to represent data series

Symbolic ApproXimation (SAX)

- similar in principle to PAA
  - uses segments to represent data series
- represents segments with symbols (rather than real numbers)
  - small memory footprint

SAX Representation

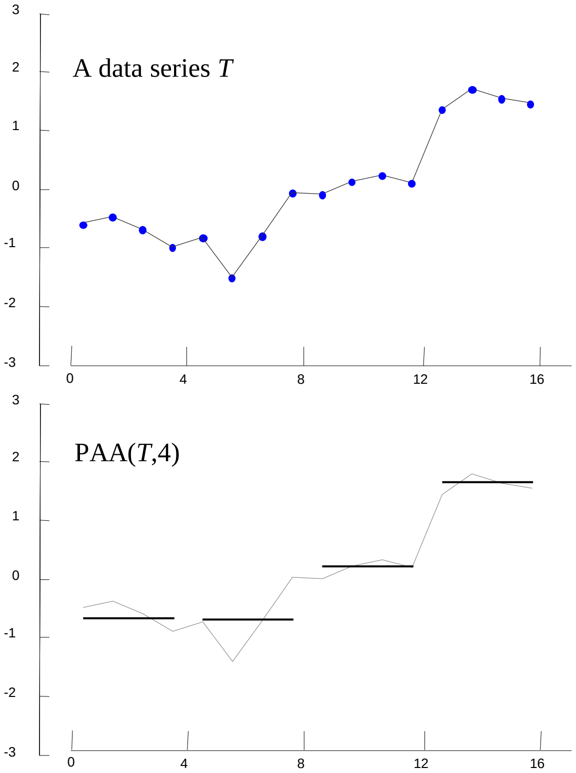
- **S**ymbolic **A**ggregate appro**X**imation (SAX)
  - **(1)** Represent data series *T* of length ***n*** with ***w*** segments using Piecewise Aggregate Approximation (PAA)



SAX Representation

- **S**ymbolic **A**ggregate appro**X**imation (SAX)
  - **(1)** Represent data series *T* of length ***n*** with ***w*** segments using Piecewise Aggregate Approximation (PAA)
    - *T* typically normalized to $\mu = 0, \sigma = 1$
    - $\mathrm{PAA}(T,w) = \overline{T} = \bar{t}_1,\ldots,\bar{t}_w$

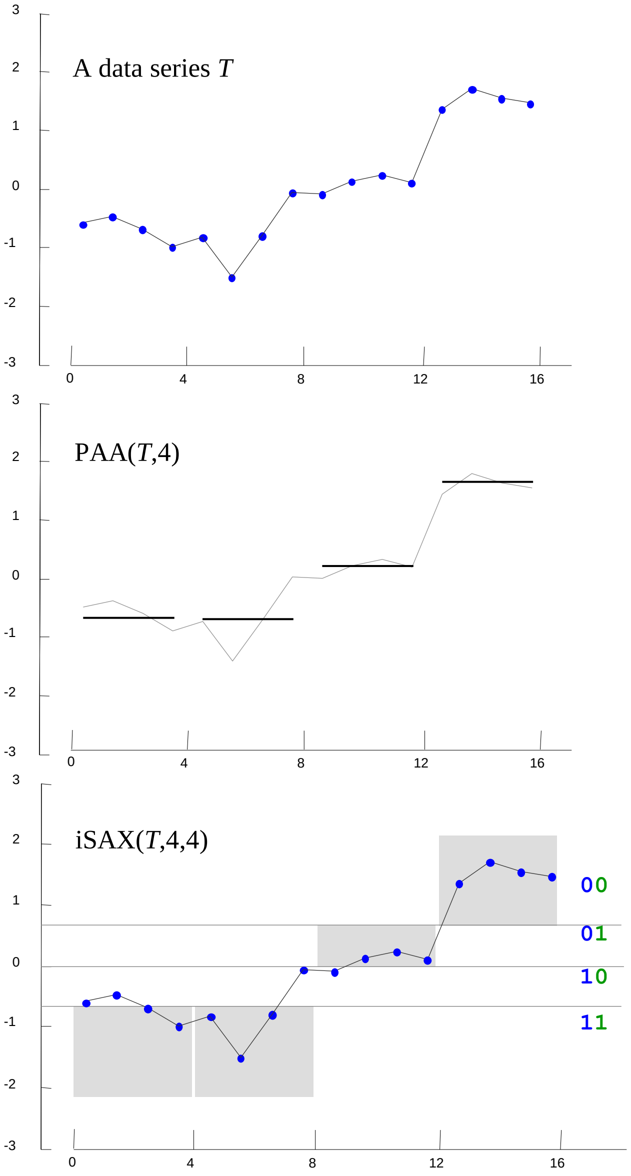
      where $\bar{t}_i = \frac{w}{n}\sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} T_j$



SAX Representation

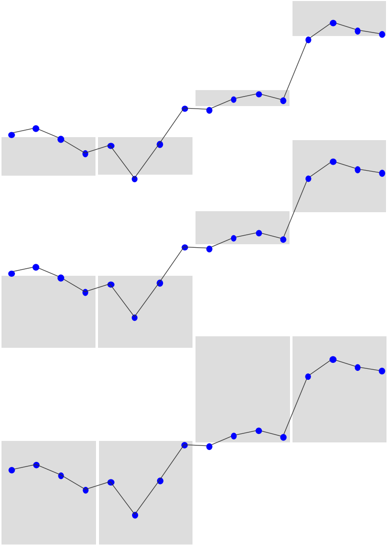
- **S**ymbolic **A**ggregate appro**X**imation (SAX)
  - □ (1) Represent data series T of length $\boldsymbol{n}$ with $\boldsymbol{w}$ segments using Piecewise Aggregate Approximation (PAA)
    - T typically normalized to $\mu = 0$, $\sigma = 1$
    - $\text{PAA}(T,w) = \overline{T} = \overline{t}_1, \ldots, \overline{t}_w$

      where $$\overline{t}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} T_j$$
  - □ (2) Discretize into a vector of symbols
    - Breakpoints map to small alphabet $\boldsymbol{a}$ of symbols



*i*SAX Representation

- *i*SAX offers a bit-aware, quantized, multi-resolution representation with variable granularity



= { 6, 6, 3, 0} = {110 ,110 ,0111 ,000}

= { 3, 3, 1, 0} = {11 ,11 ,011 ,00 }

= { 1, 1, 0, 0} = {1 ,1 ,0 ,0 }

Comparison of Representations

- which representation is the most effective?
  - used same amount of memory for all approaches
  - measured using root mean squared error
  - averaged over 40 datasets and 100 experiment repetitions
  - normalized by best score

Comparison of Representations

- which representation is the most effective?
  - used same amount of memory for all approaches
  - measured using root mean squared error
  - averaged over 40 datasets and 100 experiment repetitions

DFT	DCT	PAA	DWT (Haar)	DWT (Daub12)	APCA	PLA	PQA
0.951	0.923	0.948	0.948	0.902	0.893	0.940	0.927

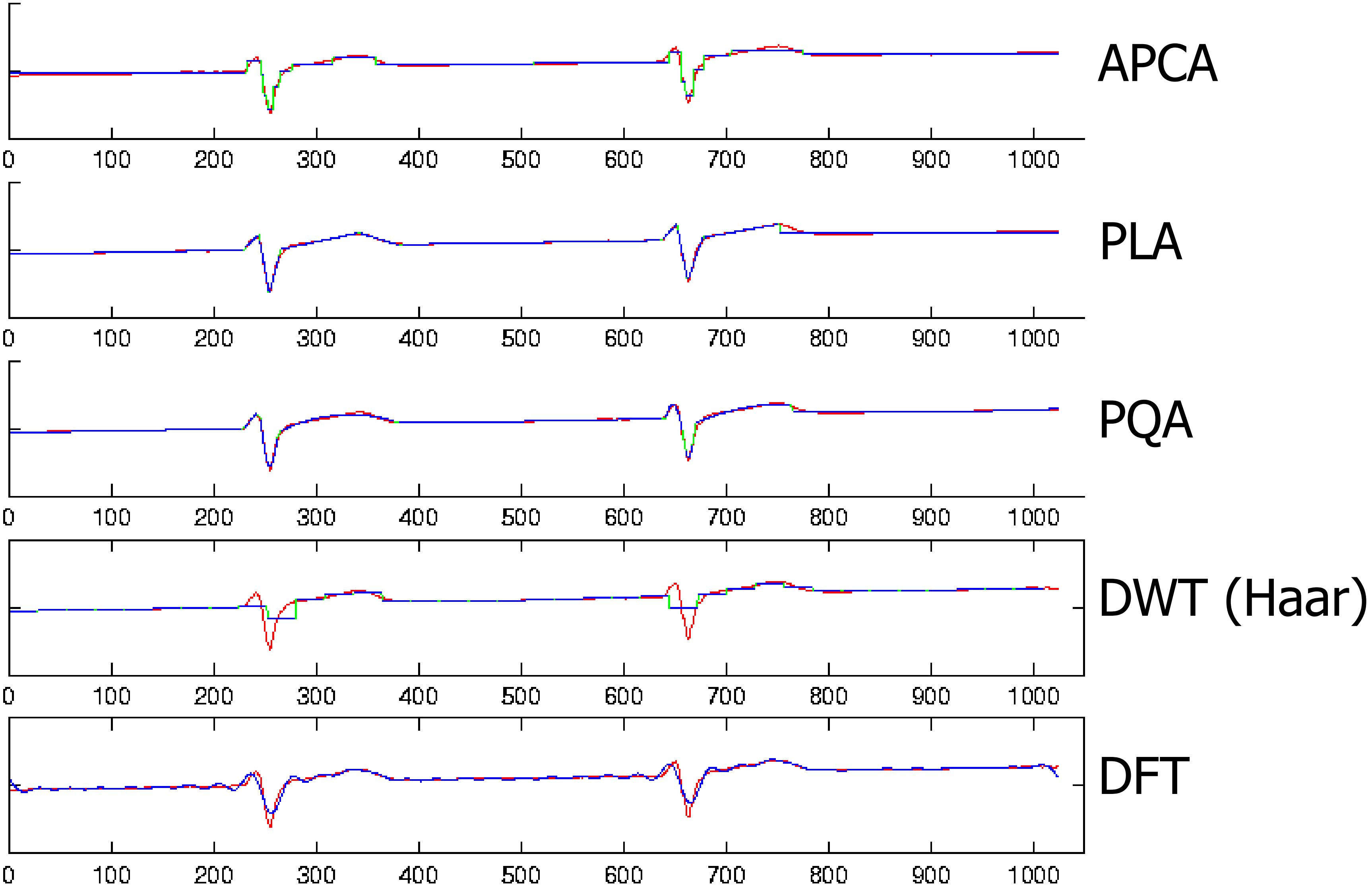
Comparison of Representations

- which representation is the most effective?
  - used same amount of memory for all approaches
  - measured using root mean squared error
  - averaged over 40 datasets and 100 experiment repetitions
  - normalized by best score

DFT	DCT	**PAA**	**DWT (Haar)**	DWT (Daub12)	APCA	PLA	PQA
0.951	0.923	0.948	0.948	0.902	0.893	0.940	0.927

- DFT, PAA, DWT (Haar) slightly better
- no big differences overall (on average!)

Comparison of Representations

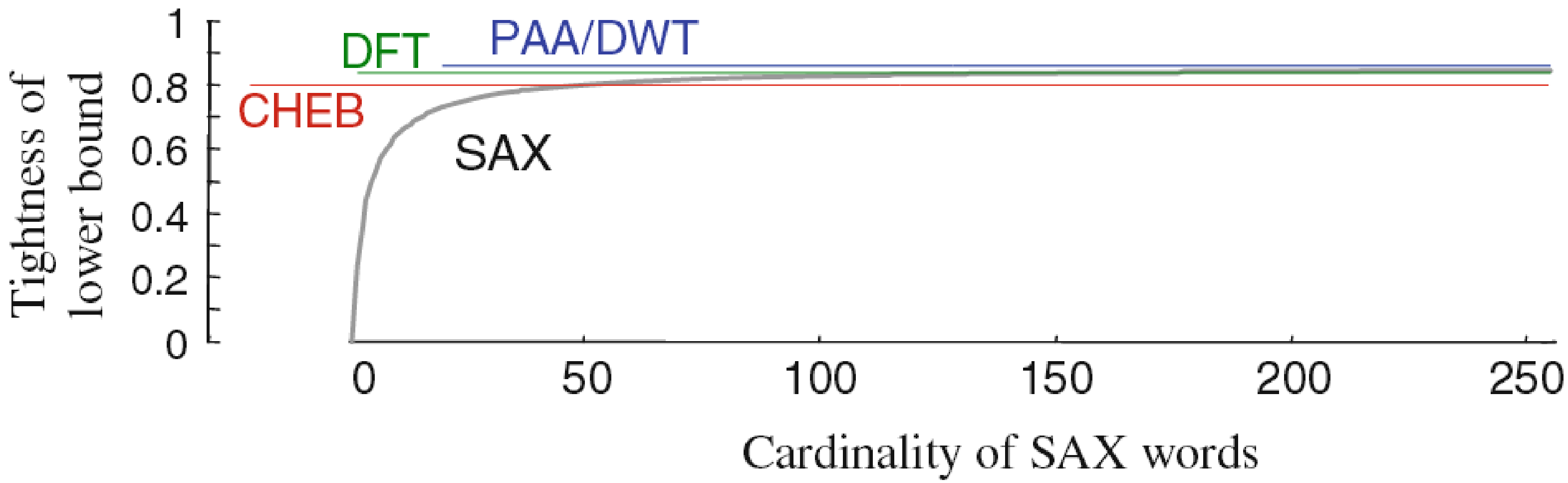


Comparison to SAX

- SAX leads to considerable memory savings at the expense of increased information loss
- how many symbols to use?
  - less symbols -> less memory
  - more symbols -> more accuracy

Comparison to SAX

- SAX leads to considerable memory savings at the expense of increased information loss
- how many symbols to use?



- ach
  - with 256 symbols (8-bits per segment): as good as best approach
- SAX still uses a fraction of the memory of the other approaches

Amnesic Representation

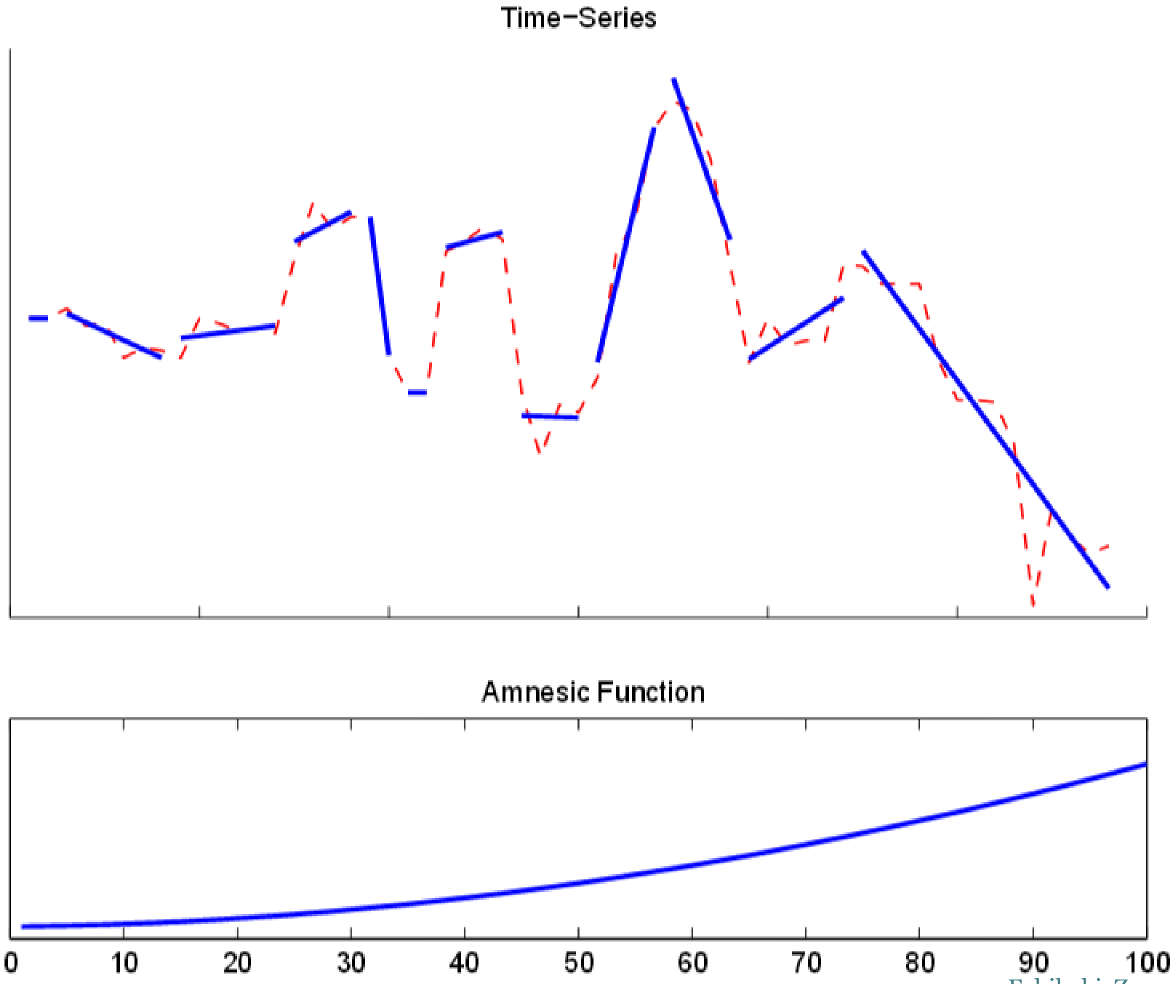
- all previous representations all data series points equally
- for some applications, approximation fidelity is time variant
  - e.g., more recent points should be represented more accurately

Amnesic Representation

- all previous representations all data series points equally
- for some applications, approximation fidelity is time variant
    - e.g., more recent points should be represented more accurately

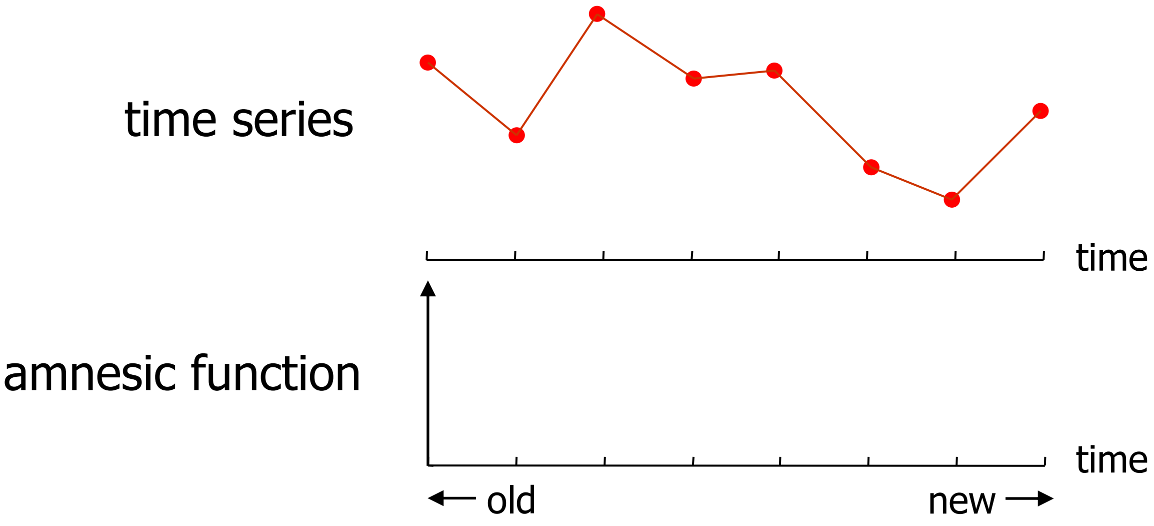
- amnesic representation does exactly that
    - amnesic function defines representation accuracy over time
        - should be non-decreasing

Amnesic Representation



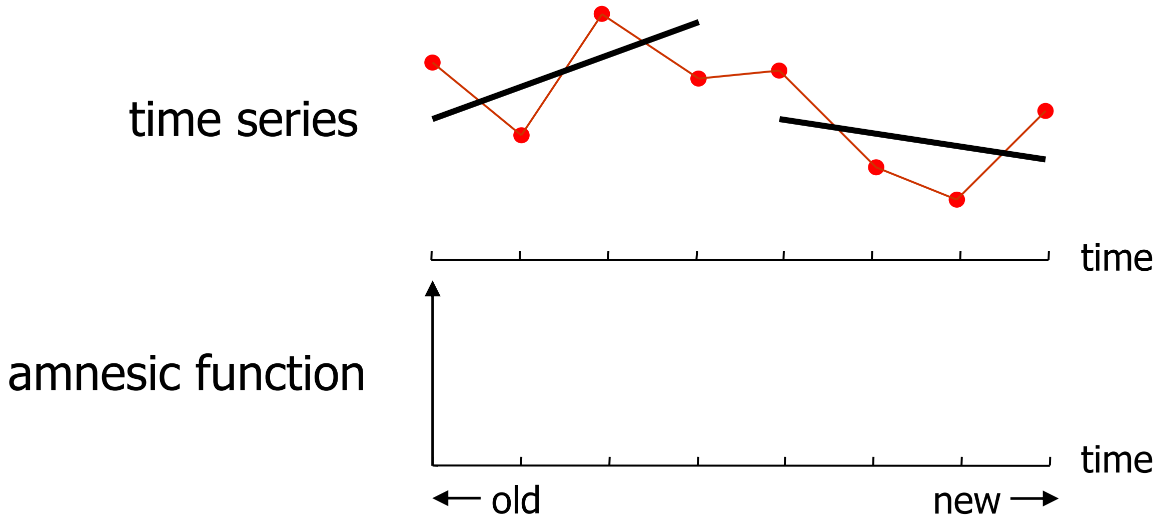
How do we forget?

- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



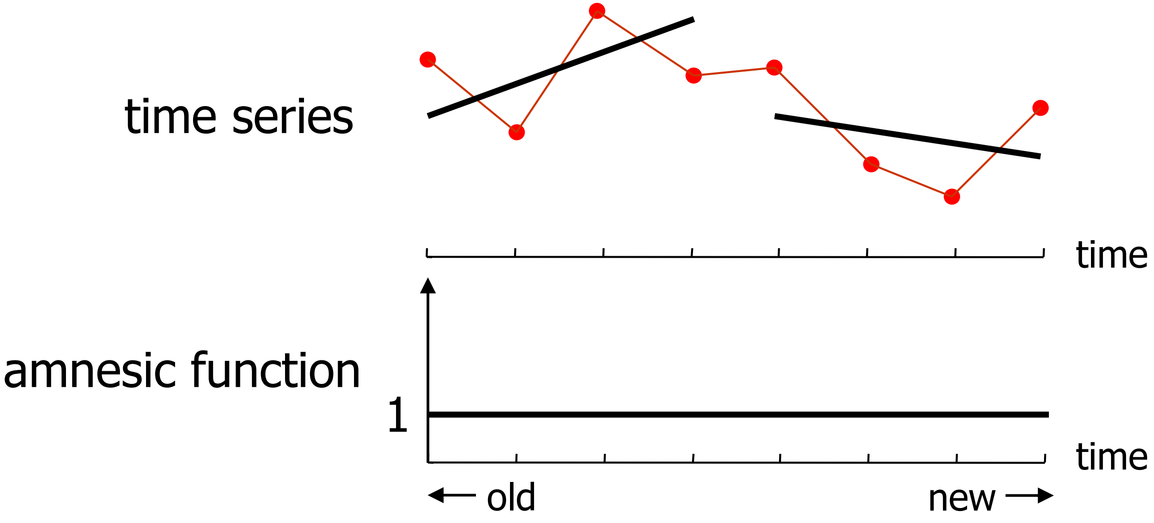
How do we forget?

- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



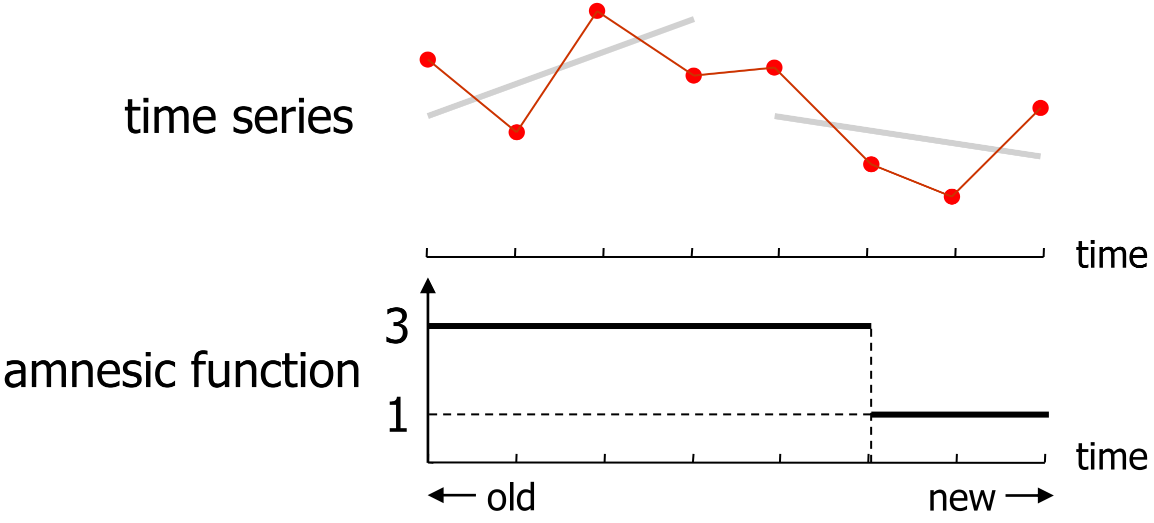
How do we forget?

- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



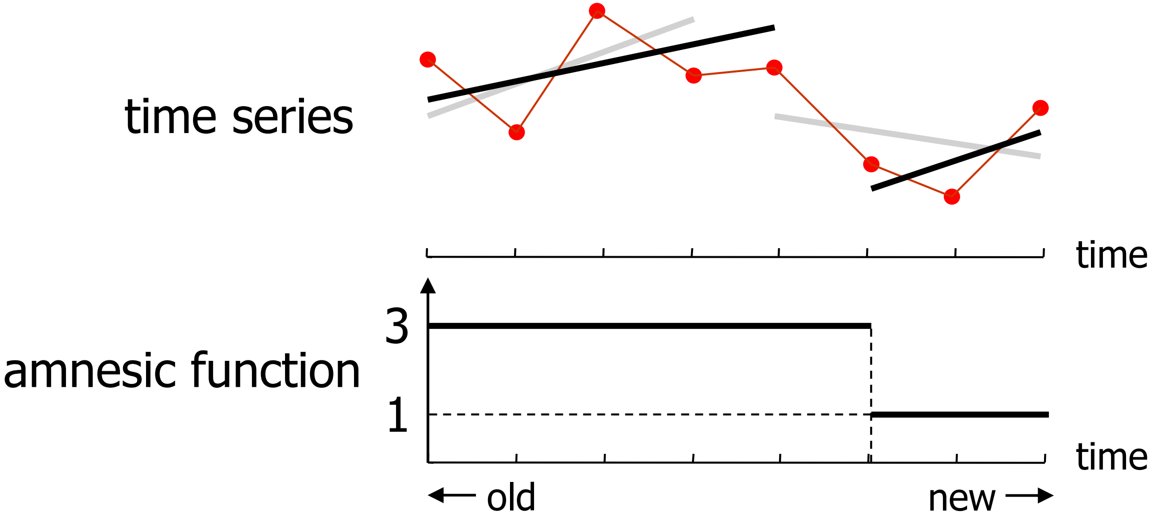
How do we forget?

- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



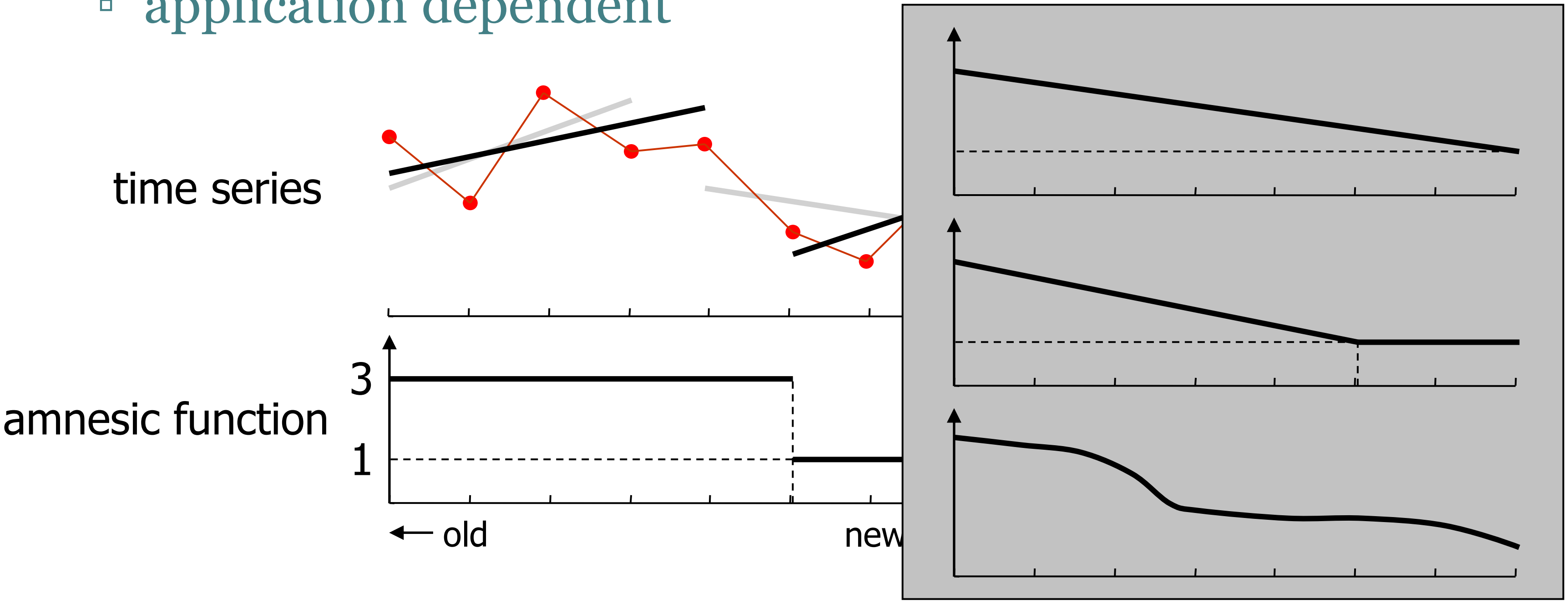
How do we forget?

- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



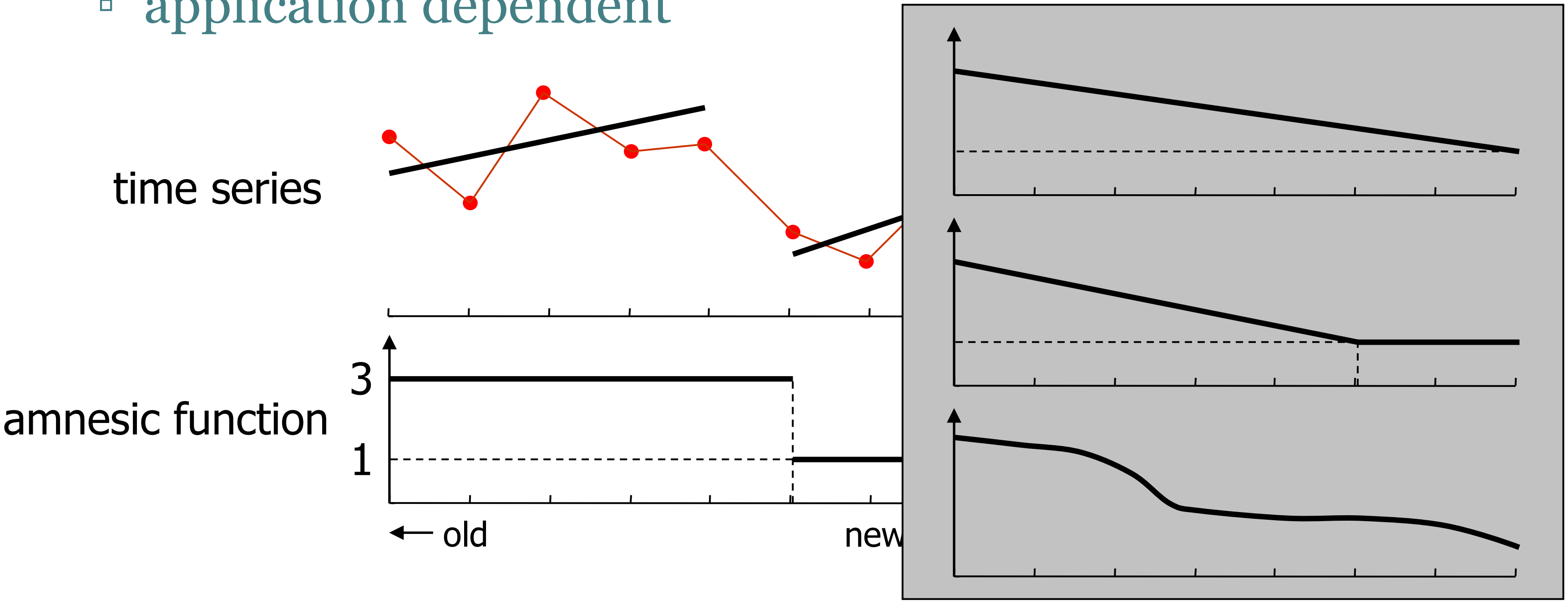
How do we forget?

- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



How do we forget?

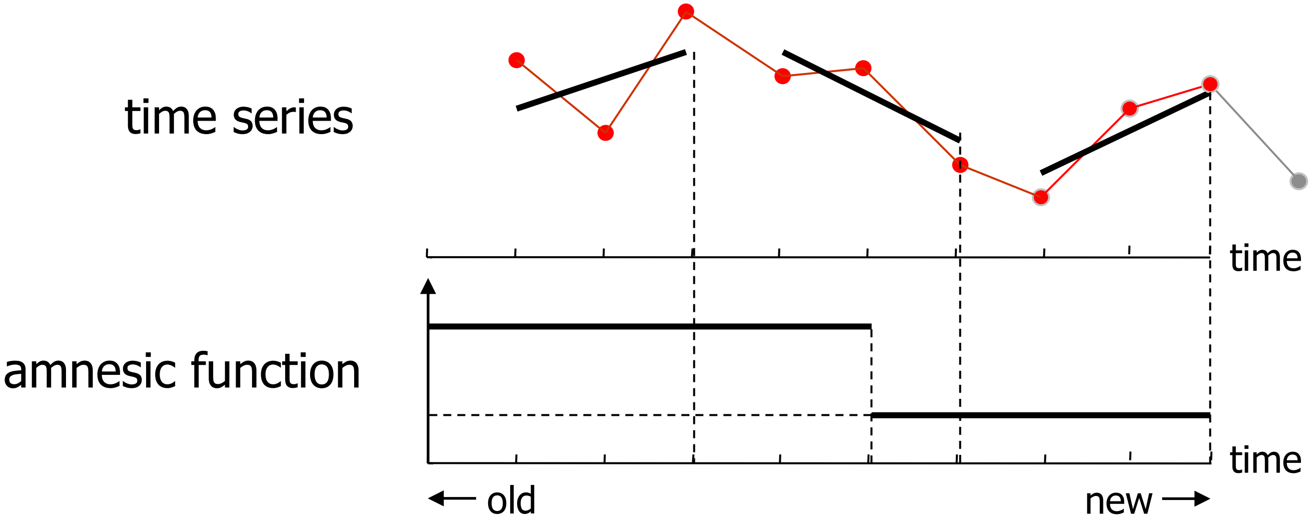
- amnesic functions
  - specify allowed approximation error for each line segment
  - application dependent



- have to satisfy the monotonicity property
  - the allowed error should only be increasing with time

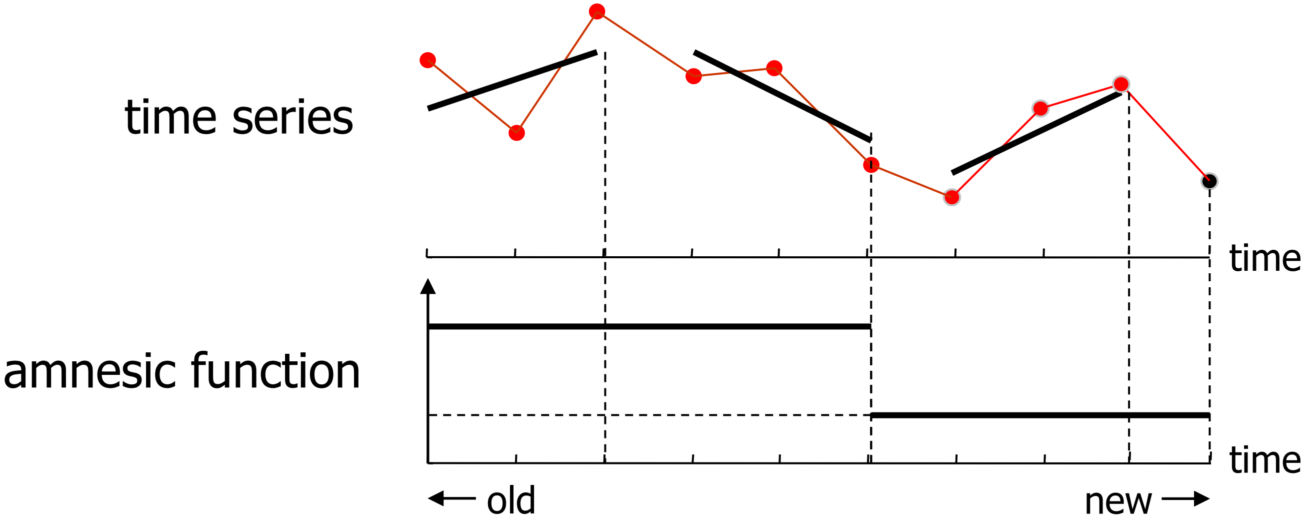
Streaming Data

- data values continuously come in
- at each time instance, have to decide which segments to merge



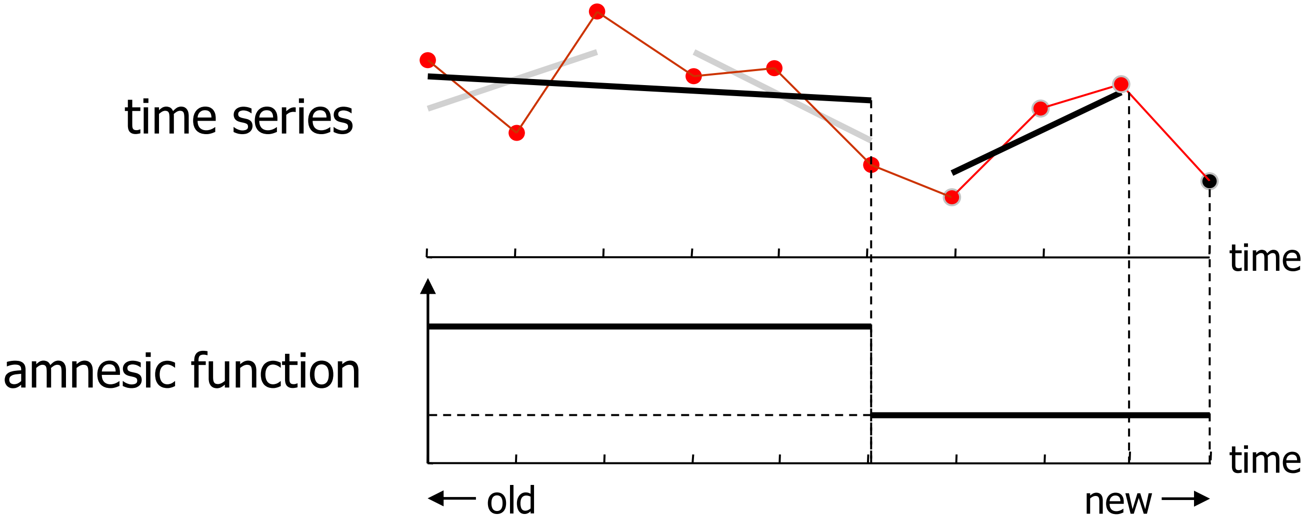
Streaming Data

- data values continuously come in
- at each time instance, have to decide which segments to merge



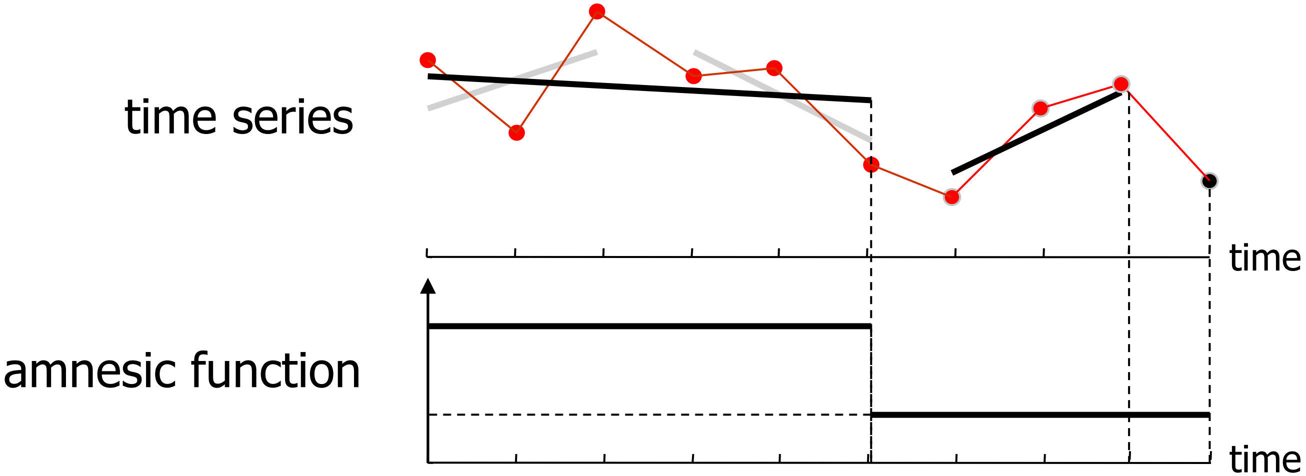
Streaming Data

- data values continuously come in
- at each time instance, have to decide which segments to merge



Streaming Data

- data values continuously come in
- at each time instance, have to decide which segments to merge



- challenges to overcome
  - merge segments in constant time
  - error for segments changes over time
    - relative ordering of segments (based on error) changes

Amnesic Representation

- amnesic functions can be
  - relative
    - determines relative approximation error tolerated for every point in data series (e.g., specify that when we approximate a point twice as old, we accept twice as much error)
  - absolute
    - specifies maximum allowable error for the approximation, for every point in data series

Amnesic Representation

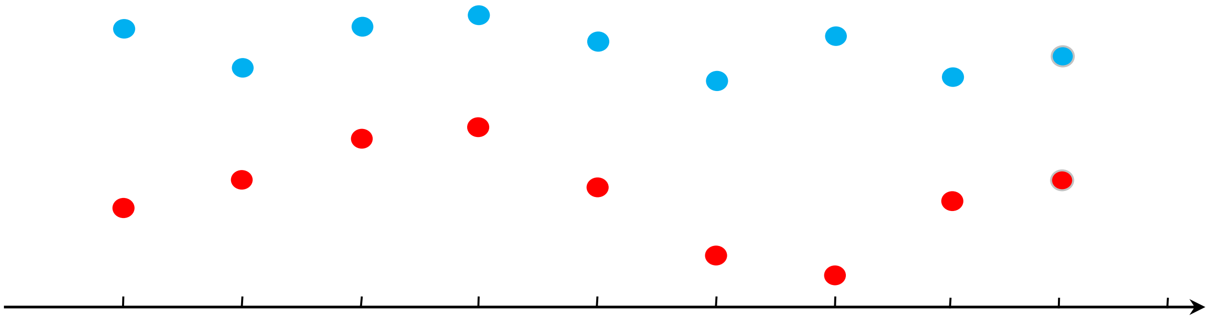
- problems using landmark window (from beginning of time will now):
  - Landmark Window with Relative Amnesic Function (URA)
    - given memory budget M and a relative amnesic function, construct amnesic approximation using memory at most M that minimizes approximation error of data points inside the window
  - Landmark Window with Absolute Amnesic Function (UAA)
    - given an absolute amnesic function, construct amnesic approximation that minimizes required memory

Amnesic Representation

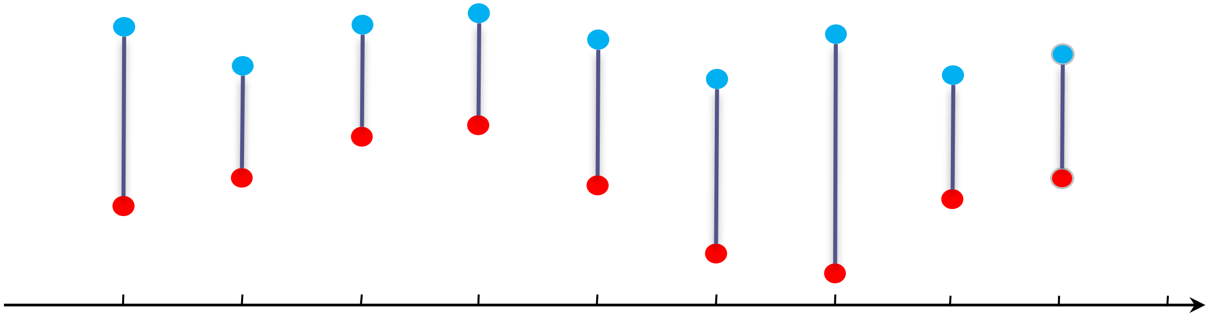
- problems using sliding window (last k values/time points):
  - Sliding Window with Relative Amnesic Function (SRA)
    - given sliding window W, memory budget M, and a relative amnesic function, construct amnesic approximation using at most memory M that minimizes approximation error of data series within window W
  - Sliding Window with Absolute Amnesic Function (SAA)
    - given sliding window W and an absolute amnesic function, construct amnesic approximation for data series within window W that minimizes required memory

Similarity Search

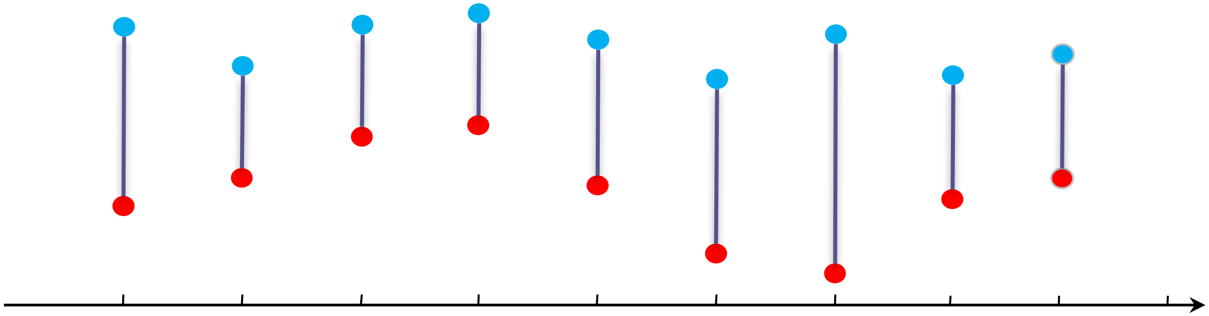
Euclidean Distance



Euclidean Distance



Euclidean Distance



- Euclidean distance
  - pair-wise point distance

$$ED(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Correlation

- measures the degree of relationship between data series
  - indicates the degree and direction of relationship

Correlation

- measures the degree of relationship between data series
  - indicates the degree and direction of relationship
- direction of change
  - positive correlation
    - values of two data series change in same direction
  - negative correlation
    - values of two data series change in opposite directions

Correlation

- measures the degree of relationship between data series
  - indicates the degree and direction of relationship
- direction of change
  - positive correlation
    - values of two data series change in same direction
  - negative correlation
    - values of two data series change in opposite directions
- linear correlation
  - amount of change in one data series bears constant ratio of change in the other data series

Correlation

- measures the degree of relationship between data series
  - indicates the degree and direction of relationship
- direction of change
  - positive correlation
    - values of two data series change in same direction
  - negative correlation
    - values of two data series change in opposite directions
- linear correlation
  - amount of change in one data series bears constant ratio of change in the other data series

- useful in several applications

Pearson’s Correlation Coefficient

- used to see linear dependency between values of data series of equal length, n

$$PC = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s_x}\right)\left(\frac{y_i - \bar{y}}{s_y}\right)$$

Pearson’s Correlation Coefficient

- used to see linear dependency between values of data series of equal length, n

$$PC = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s_x}\right)\left(\frac{y_i - \bar{y}}{s_y}\right)$$

- where $\bar{x}$ is the mean: $\bar{x} = \frac{1}{n-1}\sum_{i=1}^{n} x_i$

- and s_x is the standard deviation: $s_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$

Pearson’s Correlation Coefficient

- used to see linear dependency between values of data series of equal length, n

$$PC = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s_x}\right)\left(\frac{y_i - \bar{y}}{s_y}\right)$$

- takes values in [-1,1]
  - 0 – no correlation
  - -1, 1 – inverse/direct correlation
- there is a statistical test connected to PC, where null hypothesis is the no correlation case (correlation coefficient = 0)
  - test is used to ensure that the correlation similarity is not caused by a random process

PC and ED

- Euclidean distance: $ED = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2},$
- In case of Z-normalized data series (mean = 0, stddev = 1):

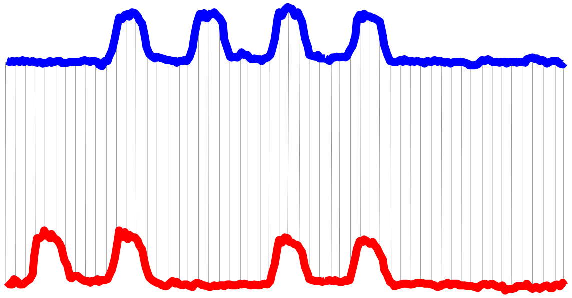
$$PC = \frac{1}{n-1}\sum_{i=1}^{n} x_i \cdot y_i \quad \text{and} \quad ED^2 = 2n(n-1) - 2\sum_{i=1}^{n} x_i y_i$$

so the following formula is true: $ED^2 = 2(n-1)(n - PC)$

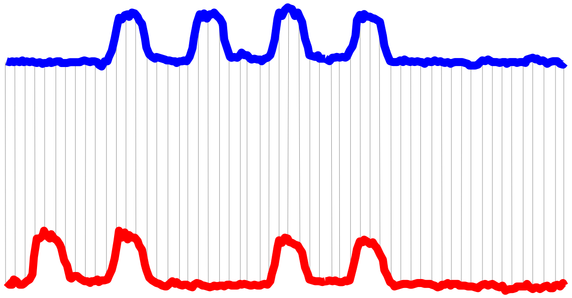
- direct connection between ED and PC for Z-normalized data series
  - if ED is calculated for normalized data series, it can be directly used to calculate the p-value for statistical test of Pearson's correlation instead of actual PC value.

Distance Measures: LCSS against Euclidean, DTW

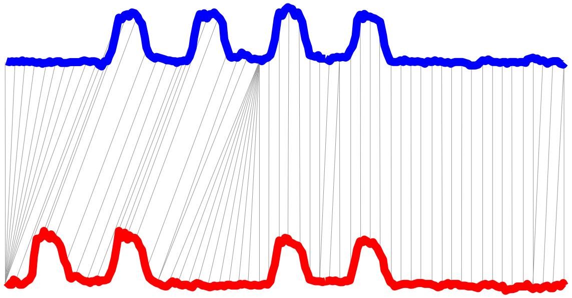
- Euclidean
  - rigid



Distance Measures: LCSS against Euclidean, DTW



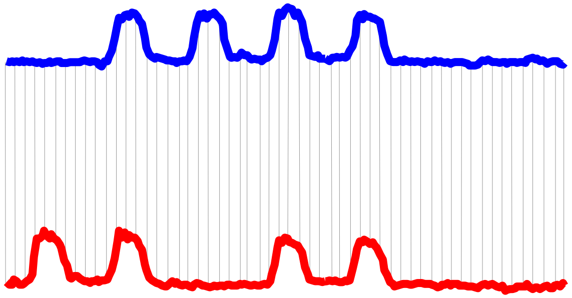
- Euclidean
  - rigid



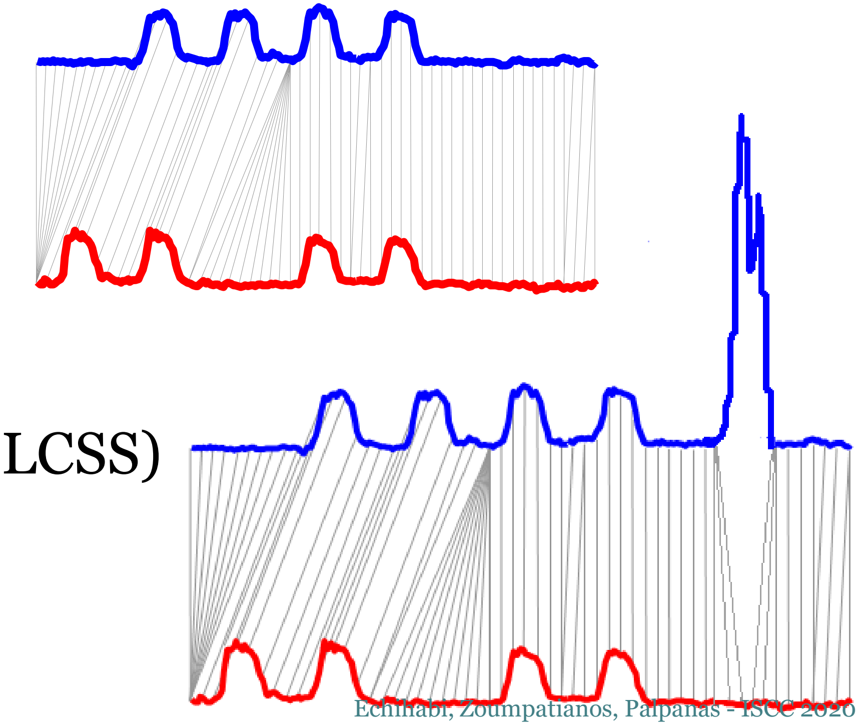
- Dynamic Time Warping (DTW)
  - allows local scaling

Distance Measures:
LCSS against Euclidean, DTW

- Euclidean
  - rigid



- Dynamic Time Warping (DTW)
  - allows local scaling



- Longest Common SubSequence (LCSS)
  - allows local scaling
  - ignores outliers

Similarity Matching

- given a data series collection D and a query data series q, return the data series from D that are the most similar to q
    - there exist different flavors of this basic operation

- basis for most data series analysis tasks

Similarity Matching
Nearest Neighbor (NN) Search

- given a data series collection D and a query data series q, return the data series from D that has the smallest distance to q

Similarity Matching
Nearest Neighbor (NN) Search

- given a data series collection D and a query data series q, return the data series from D that has the smallest distance to q

- result set contains one data series

Similarity Matching
Nearest Neighbor (NN) Search

- serial scan
    - compute the distance between q and every $d_i \in D$
    - return d_i with the smallest distance to q

Similarity Matching
Nearest Neighbor (NN) Search

- serial scan
  - bsf = Inf   // best so far distance
  - for every $d_i \in D$
    - compute distance, dist, between d_i and q
    - if this dist less than bsf then bsf=dist
  - return d_i corresponding to bsf

Similarity Matching
k-Nearest Neighbors (kNN) Search

- given a data series collection D and a query data series q, return the k data series from D that have the k smallest distances to q

Similarity Matching
k-Nearest Neighbors (kNN) Search

- given a data series collection D and a query data series q, return the k data series from D that have the k smallest distances to q

- result set contains k data series

Similarity Matching
k-Nearest Neighbors (kNN) Search

- serial scan
  - compute the distance between q and every $d_i \in D$
  - return the k d_i with the k smallest distances to q

Similarity Matching
k-Nearest Neighbors (kNN) Search

- serial scan
  - kbsf = Null // best so far max-heap of k elements
  - for every $d_i \in D$
    - compute distance, dist, between d_i and q
    - if this dist less than max of kbsf then insert dist in kbsf
  - return k d_i corresponding to k elements in kbsf

Similarity Matching
ε-Range Search

- given a data series collection D and a query data series q, return all data series from D that are within distance ε from q

Similarity Matching
ε-Range Search

- given a data series collection D and a query data series q, return all data series from D that are within distance ε from q

- result set contains [?] data series

Similarity Matching
ε-Range Search

- serial scan
  - compute the distance between q and every $d_i \in D$
  - return all d_i with distance less than ε to q

Similarity Matching
ε-Range Search

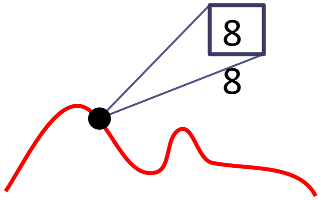
- serial scan
  - res = {} // empty result set
  - for every $d_i \in D$
    - compute distance, dist, between d_i and q
    - if this dist less than ε then insert dist in res
  - return all d_i corresponding to elements in res

Problem Variations

Series

Problem Variations

Series

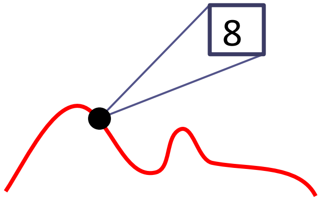


Univariate

each point represents one value (e.g., temperature)

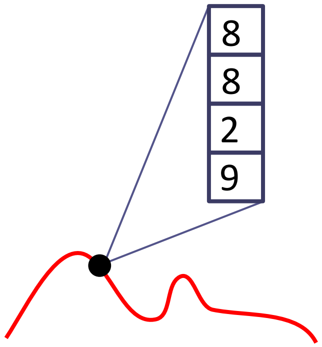
Problem Variations

Series



Univariate

each point represents one value (e.g., temperature)

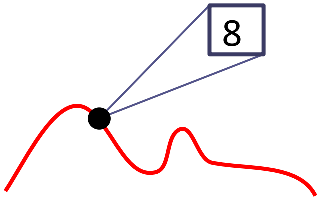


Multivariate

each point represents many values (e.g., temperature, humidity, pressure, wind, etc.)

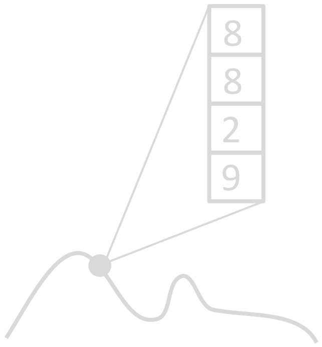
Problem Variations

Series



Univariate

each point represents one value (e.g., temperature)

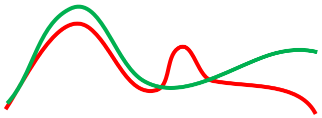


Multivariate

each point represents many values (e.g., temperature, humidity, pressure, wind, etc.)

Problem Variations

Queries



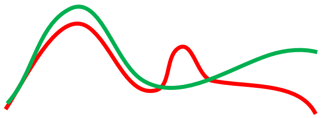
Whole matching

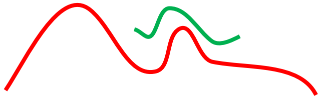
Entire query

Entire candidate

Problem Variations

Queries





Whole matching

Entire query

Entire candidate

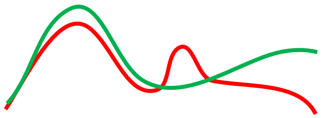
Subsequence matching

Entire query

A subsequence of a candidate

Problem Variations

Queries



Whole matching

Entire query

Entire candidate

Subsequence matching

Entire query

A subsequence of a candidate

Problem Variations

Distances

Euclidean Distance (ED)
Dynamic Time Warping (DTW)
Longest Common Subsequence (LCSS)
Edit Distance
And more…

Publications

PVLDB‘20

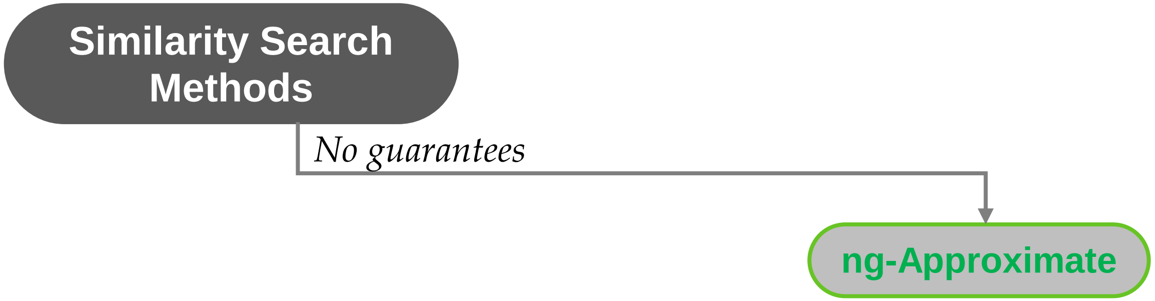
Methods

Similarity Search Methods

Publications

PVLDB‘20

Methods

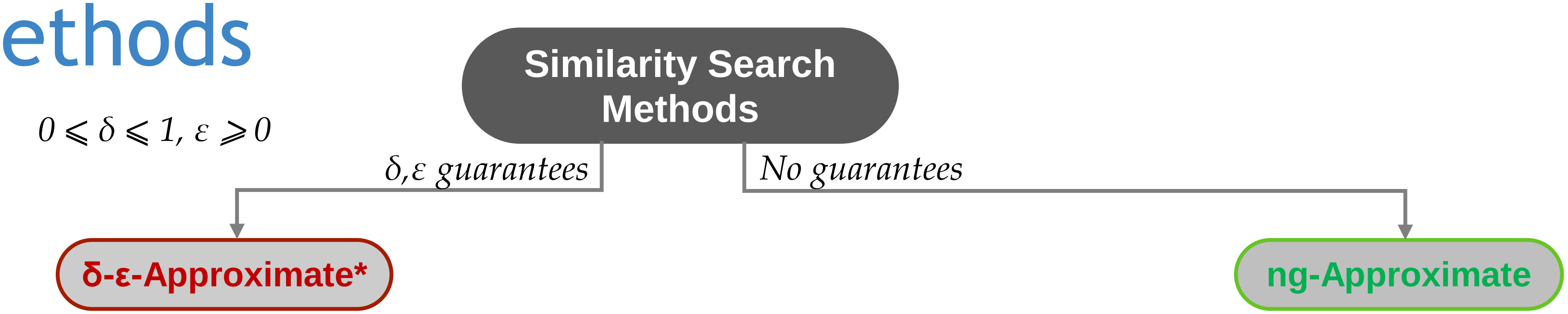


Publications

PVLDB'20

Methods

$0 \leqslant \delta \leqslant 1, \varepsilon \geqslant 0$

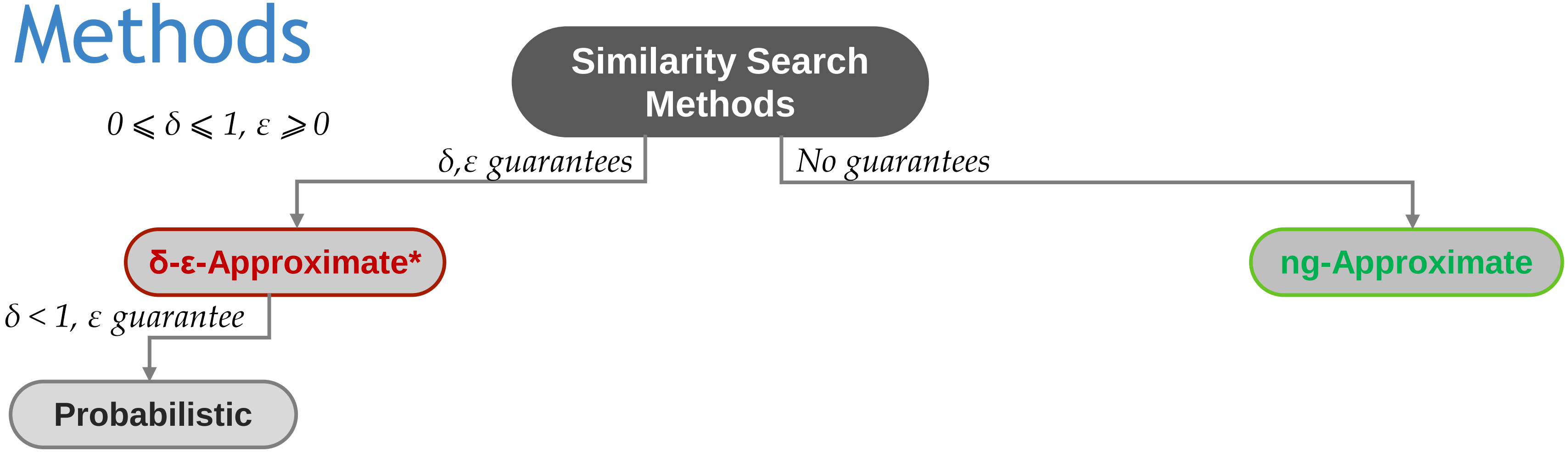


* Result is within distance (1+ ε) of the exact answer with probability δ

Publications

PVLDB‘20

Methods



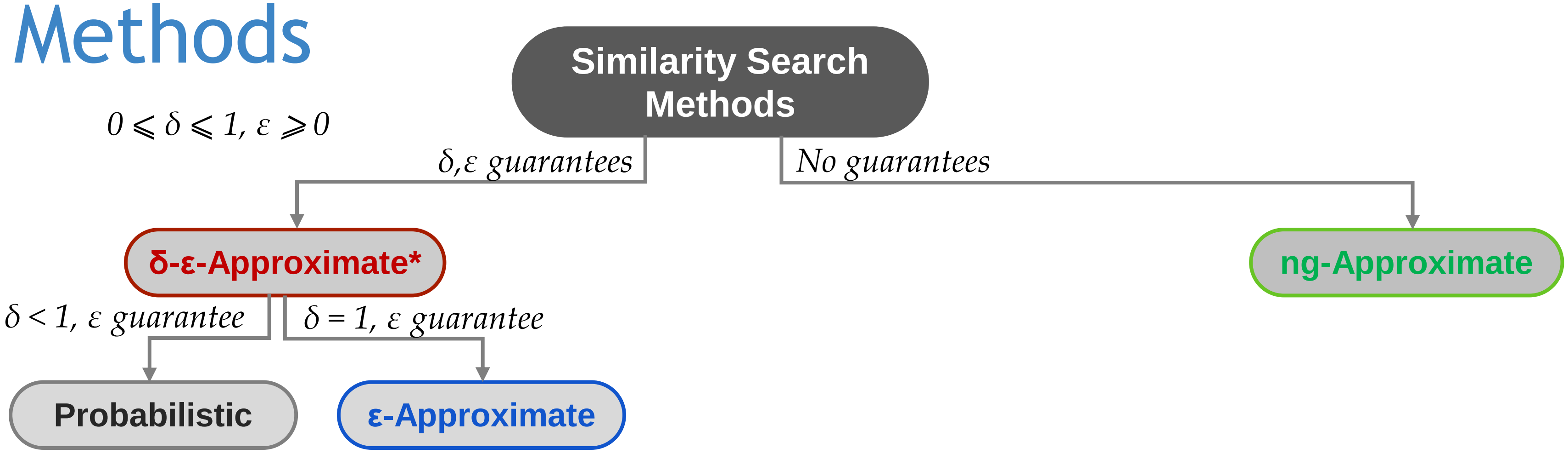
* Result is within distance (1+ ε) of the exact answer with probability δ

Publications

PVLDB'20

Methods

$0 \leqslant \delta \leqslant 1, \varepsilon \geqslant 0$



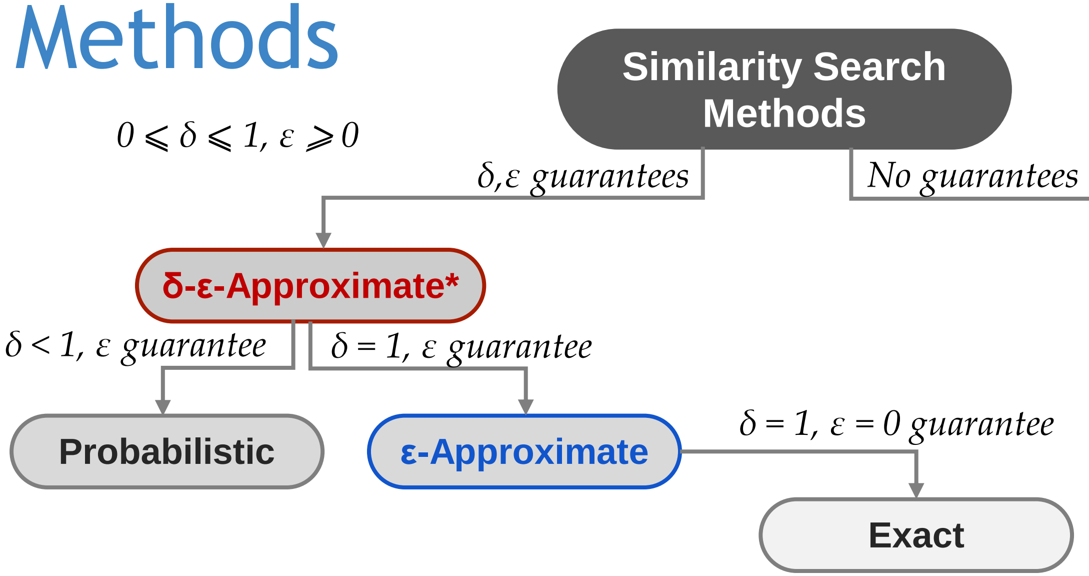
* Result is within distance (1+ ε) of the exact answer with probability δ

Publications

PVLDB'20

Methods

$0 \leqslant \delta \leqslant 1, \varepsilon \geqslant 0$



*** Result is within distance (1+ ε) of the exact answer with probability δ**

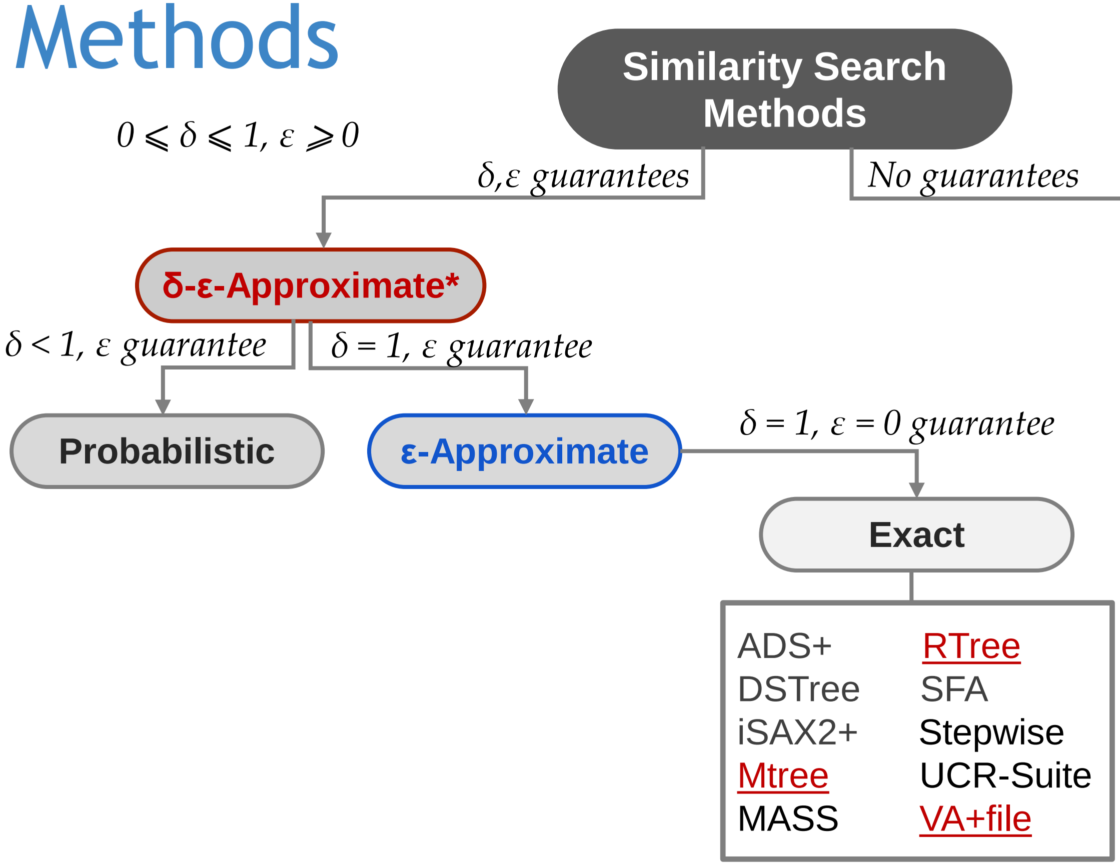
Publications

PVLDB‘20

Methods

Techniques for data Series

Techniques for High-D vectors



Similarity Search Methods

$0 \leqslant \delta \leqslant 1, \varepsilon \geqslant 0$

δ,ε guarantees

No guarantees

δ-ε-Approximate*

ng-Approximate

$\delta < 1$, *ε guarantee*

$\delta = 1$, *ε guarantee*

Probabilistic

ε-Approximate

$\delta = 1, \varepsilon = 0$ *guarantee*

Exact

ADS+	RTree
DSTree	SFA
iSAX2+	Stepwise
Mtree	UCR-Suite
MASS	VA+file

*** Result is within distance (1+ ε) of the exact answer with probability δ**

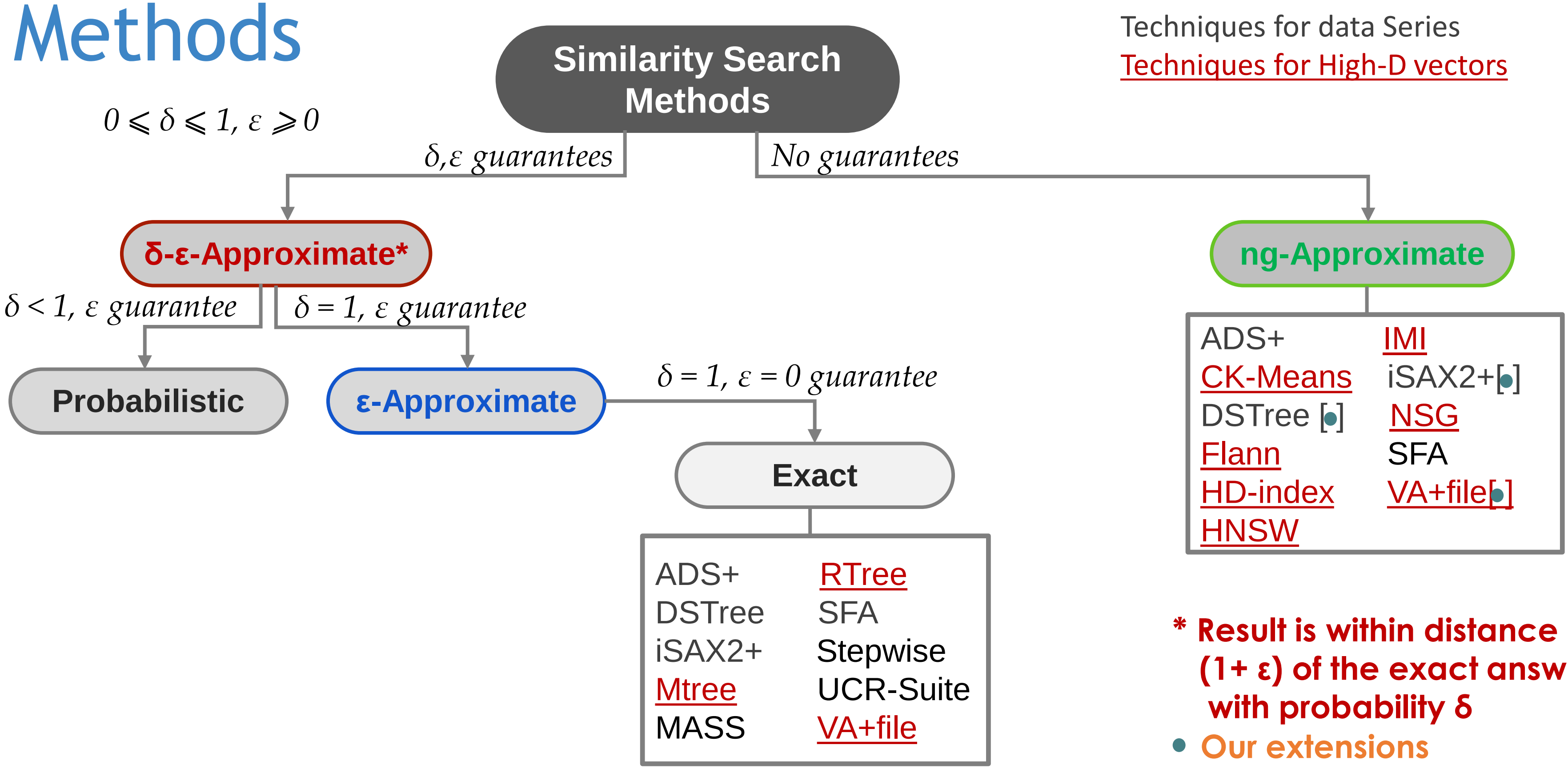
Publications

PVLDB‘20

Methods

Techniques for data Series

Techniques for High-D vectors



Similarity Search Methods
$0 \leqslant \delta \leqslant 1,\ \varepsilon \geqslant 0$
δ,ε guarantees
No guarantees
δ-ε-Approximate*
ng-Approximate
$\delta < 1,\ \varepsilon$ guarantee
$\delta = 1,\ \varepsilon$ guarantee
Probabilistic
ε-Approximate
$\delta = 1,\ \varepsilon = 0$ guarantee
Exact
ADS+
DSTree
iSAX2+
Mtree
MASS
RTree
SFA
Stepwise
UCR-Suite
VA+file
ADS+
CK-Means
DSTree [●]
Flann
HD-index
HNSW
IMI
iSAX2+[●]
NSG
SFA
VA+file[●]

* Result is within distance (1+ ε) of the exact answer with probability δ

● Our extensions

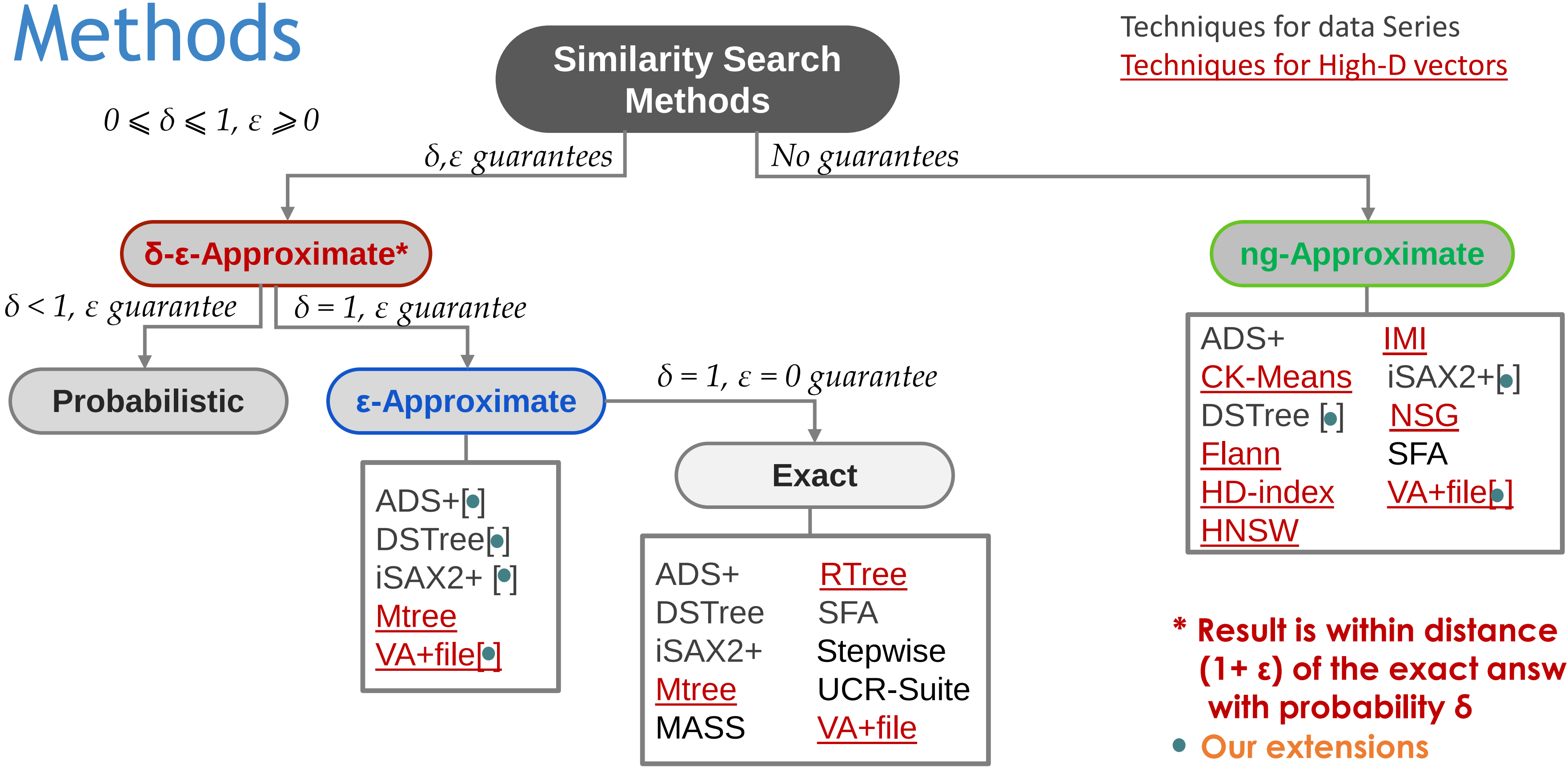
Publications

PVLDB'20

Methods

Techniques for data Series

Techniques for High-D vectors



Similarity Search Methods

$0 \leqslant \delta \leqslant 1, \varepsilon \geqslant 0$

δ,ε guarantees

No guarantees

δ-ε-Approximate*

δ < 1, ε guarantee

δ = 1, ε guarantee

Probabilistic

ε-Approximate

- ADS+[●]
- DSTree[●]
- iSAX2+ [●]
- Mtree
- VA+file[●]

δ = 1, ε = 0 guarantee

Exact

ADS+	RTree
DSTree	SFA
iSAX2+	Stepwise
Mtree	UCR-Suite
MASS	VA+file

ng-Approximate

ADS+	IMI
CK-Means	iSAX2+[●]
DSTree [●]	NSG
Flann	SFA
HD-index	VA+file[●]
HNSW	

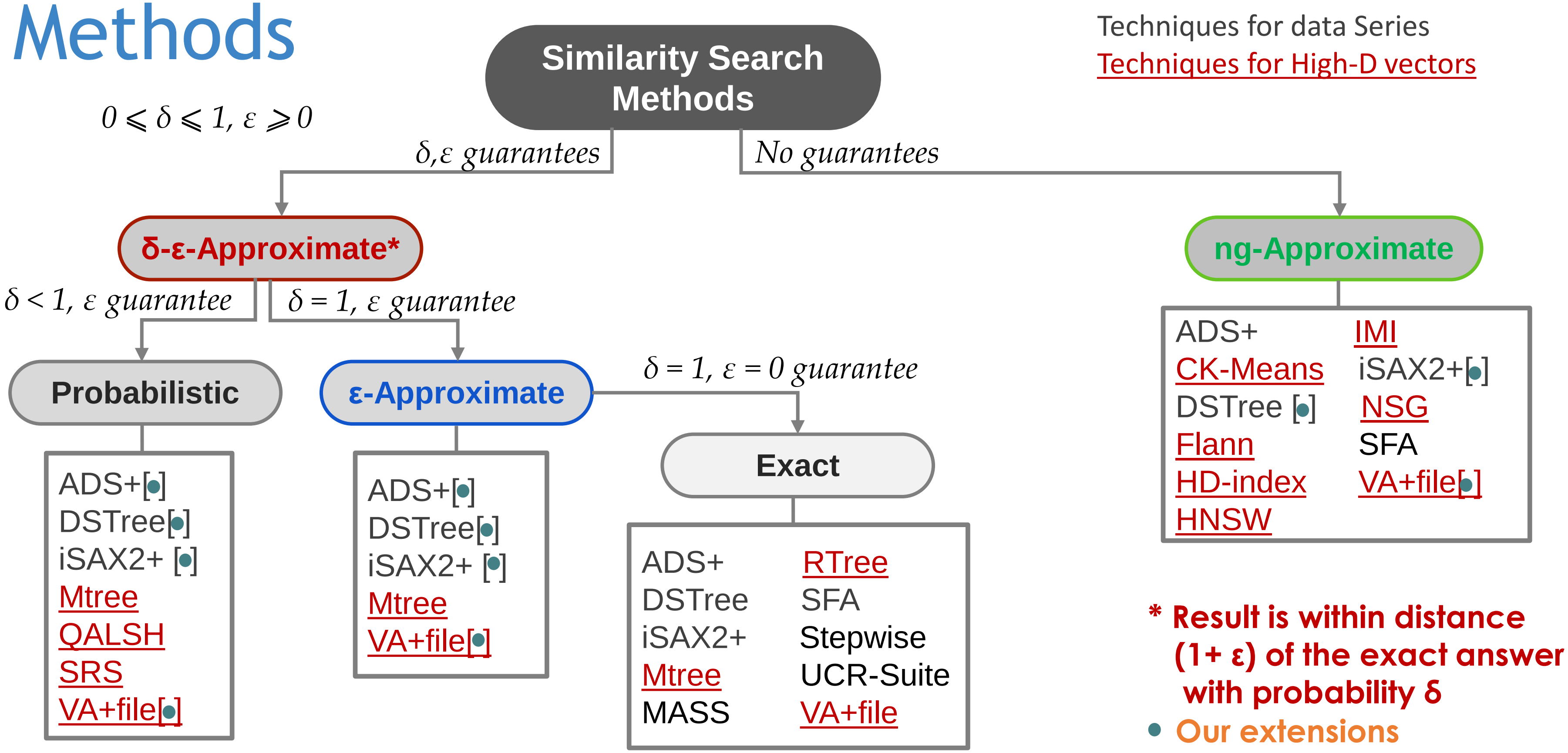
* Result is within distance (1+ ε) of the exact answer with probability δ

● Our extensions

Publications

PVLDB‘20

Methods



Techniques for data Series

Techniques for High-D vectors

Similarity Search Methods

$0 \leqslant \delta \leqslant 1, \varepsilon \geqslant 0$

δ, ε *guarantees*

No guarantees

δ-ε-Approximate*

$\delta < 1, \varepsilon$ *guarantee*

$\delta = 1, \varepsilon$ *guarantee*

Probabilistic

ADS+[●]
DSTree[●]
iSAX2+ [●]
Mtree
QALSH
SRS
VA+file[●]

ε-Approximate

ADS+[●]
DSTree[●]
iSAX2+ [●]
Mtree
VA+file[●]

$\delta = 1, \varepsilon = 0$ *guarantee*

Exact

ADS+
DSTree
iSAX2+
Mtree
MASS
RTree
SFA
Stepwise
UCR-Suite
VA+file

ng-Approximate

ADS+
CK-Means
DSTree [●]
Flann
HD-index
HNSW
IMI
iSAX2+[●]
NSG
SFA
VA+file[●]

*** Result is within distance (1+ ε) of the exact answer with probability δ**

● Our extensions

Similarity Matching
Fast Euclidean Distance

- similarity matching requires many distance computations
  - can significantly slow down processing
    - because of large number of data series in the collection
    - because of high dimensionality of each data series

Similarity Matching
Fast Euclidean Distance

- similarity matching requires many distance computations
  - can significantly slow down processing
    - because of large number of data series in the collection
    - because of high dimensionality of each data series

- in case of Euclidean Distance, we can speedup processing by
  - smart implementation of distance function
  - early abandoning

Similarity Matching
Fast Euclidean Distance

- similarity matching requires many distance computations
  - can significantly slow down processing
    - because of large number of data series in the collection
    - because of high dimensionality of each data series

- in case of Euclidean Distance, we can speedup processing by
  - smart implementation of distance function
  - early abandoning

- result in considerable performance improvement

Similarity Matching
Fast Euclidean Distance

- smart implementation of distance function
  - do not compute the square root (of the Euclidean Distance)

$$ED(X,Y) = \sum_{i=1}^{n}(x_i - y_i)^2$$

Similarity Matching
Fast Euclidean Distance

- smart implementation of distance function
  - do not compute the square root (of the Euclidean Distance)

$$ED(X,Y) = \sum_{i=1}^{n} (x_i - y_i)^2$$

- does not alter the results
- saves precious CPU cycles

Similarity Matching
Fast Euclidean Distance

- early abandoning
  - stop the distance computation as soon as it exceeds the value of bsf

$$ED(X,Y) = \sum_{i=1}^{m} (x_i - y_i)^2 , \qquad m \leq n$$

Similarity Matching
Fast Euclidean Distance

- early abandoning
  - stop the distance computation as soon as it exceeds the value of bsf

$$ED(X,Y) = \sum_{i=1}^{m} (x_i - y_i)^2 , \qquad m \leq n$$

- does not alter the results
- avoids useless computations

Process Overview

- Raw data: original full-dimensional space
- Summarization: reduced dimensionality space
- Searching in original space *costly*
- Searching in reduced space *faster*:
  - Less data, indexing techniques available, lower bounding

Process Overview

- Raw data: original full-dimensional space
- Summarization: reduced dimensionality space
- Searching in original space *costly*
- Searching in reduced space *faster*:
  - Less data, indexing techniques available, lower bounding
- Lower bounding enables us to
  - *prune search space:* throw away data series based on reduced dimensionality representation
  - *guarantee correctness* of answer
    - no false negatives
    - false positives filtered out based on raw data

Similarity Retrieval

- Range Query
  - Find all time series S where $D(Q,S) \leq \varepsilon$
- Nearest Neighbor query
  - Find all the k most similar time series to Q
- A method to answer the above queries
  - Linear scan … very slow

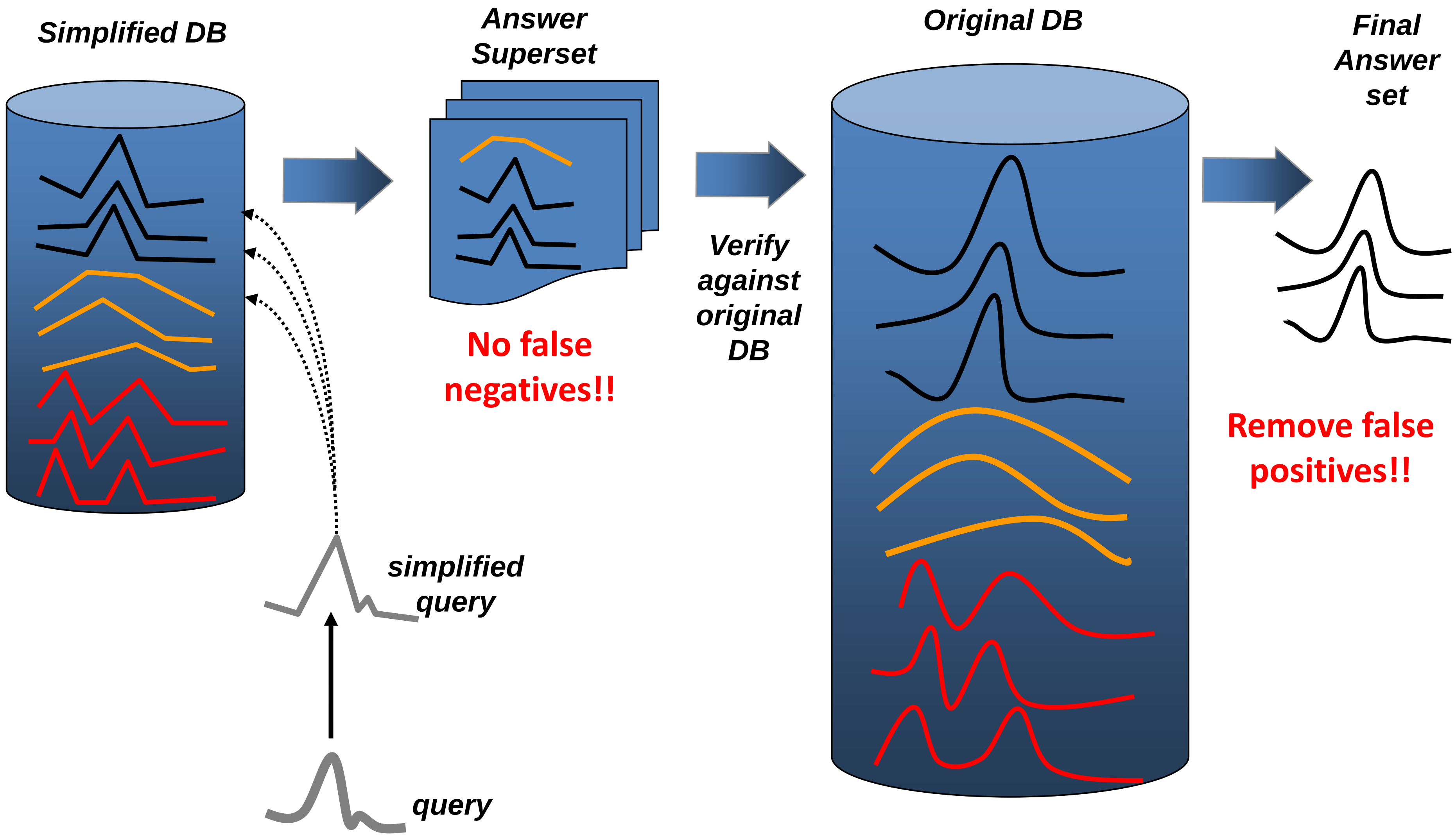
- A better approach GEMINI

GEMINI

Solution: Quick filter-and-refine:

- extract *m* features (numbers, e.g., average)
- map to point in *m*-dimensional feature space
- organize points
- retrieve the answer using a NN query
- discard false positives

Generic Search using Lower Bounding



GEMINI: contractiveness

- GEMINI works when:

$$D_{feature}(F(x), F(y)) <= D(x, y)$$

- *Note that, the closer the feature distance to the actual one, the better*

Lower Bounding

We can speed up similarity search by using a lower bounding function

- D: distance measure
- LB: lower bounding function s.t.: **$LB(Q, S_i) \le D(Q,S_i)$**

Intuition

✓ Try to use a cheap lower bounding calculation as often as possible

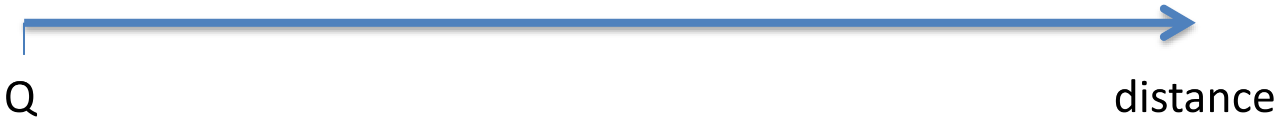
✓ Do the expensive, full calculations when absolutely necessary

```
1-NN Search Using LB

➢ Set best = ∞
➢ For each $S_i$:
      →if $LB(S_i, Q) < best$
          if $D(S_i, Q) < best$
              best = $D(S_i, Q)$
```

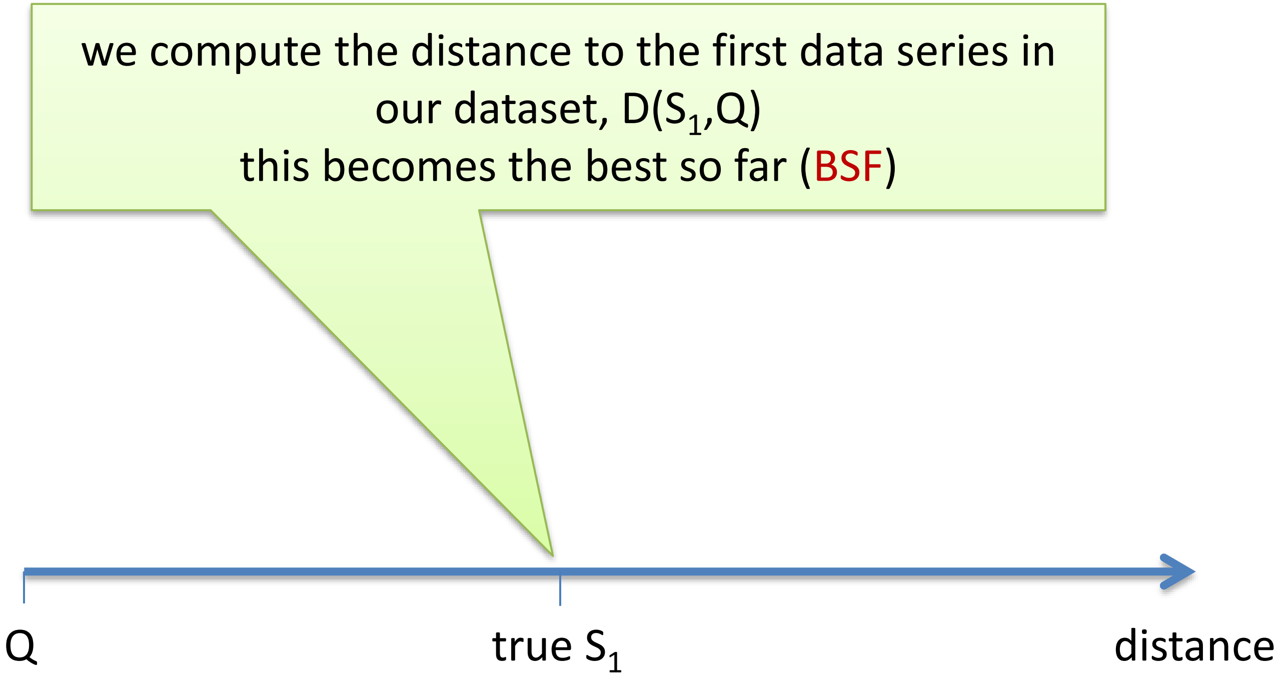
```
Range Query Using LB

For each $S_i$:
    →if $LB(S_i, Q) \le \varepsilon$
        if $D(S_i, Q) < \varepsilon$
            report $S_i$
```

Lower Bounding

we want to find the 1-NN to our query data series, Q

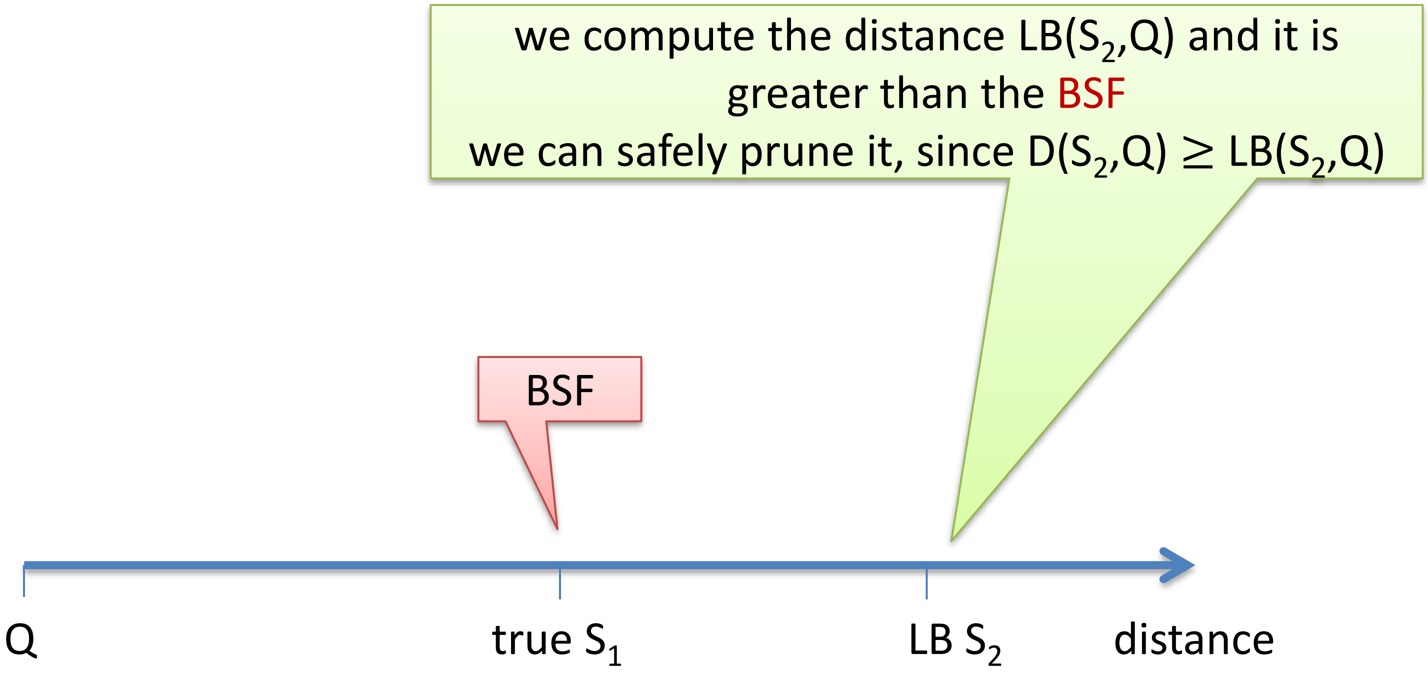


Lower Bounding



we compute the distance to the first data series in our dataset, D(S_1,Q)
this becomes the best so far (BSF)

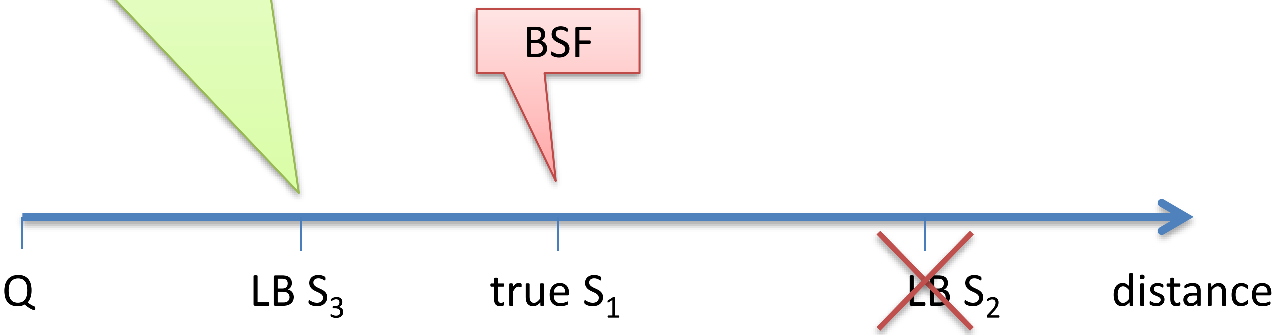
Lower Bounding



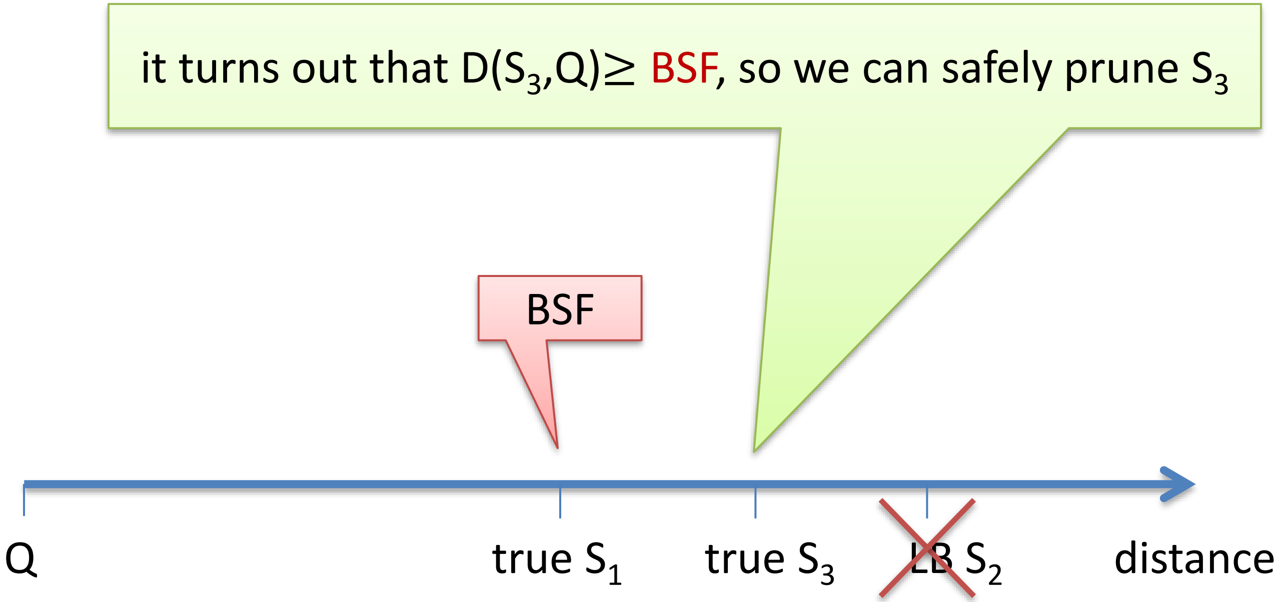
we compute the distance LB(S_2,Q) and it is greater than the BSF
we can safely prune it, since D(S_2,Q) $\geq$ LB(S_2,Q)

Lower Bounding

we compute the distance LB(S_3,Q) and it is smaller than the BSF we have to compute D(S_3,Q)≥ LB(S_3,Q), since it may still be smaller than BSF



Lower Bounding



it turns out that $D(S_3,Q) \geq$ BSF, so we can safely prune S_3

BSF

Q

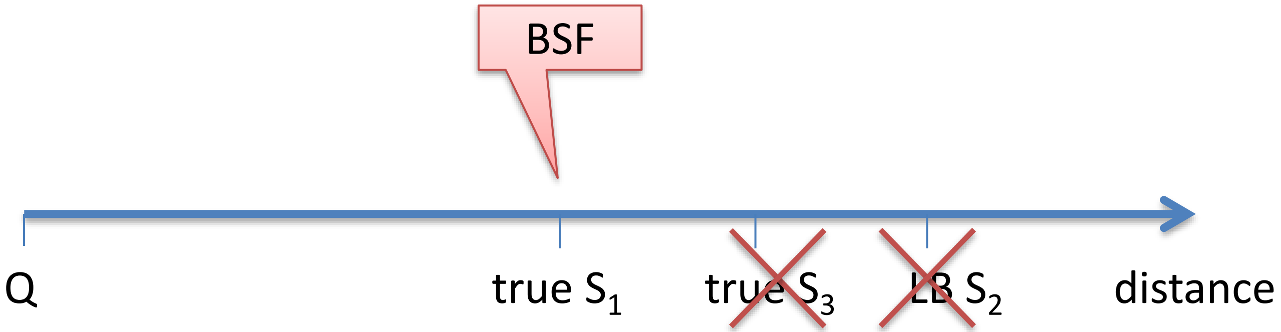
true S_1

true S_3

~~LB~~ S_2

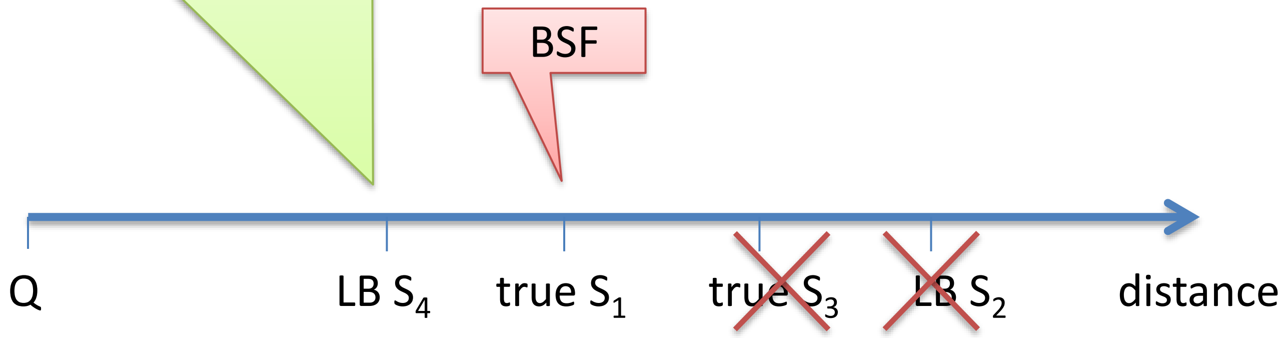
distance

Lower Bounding

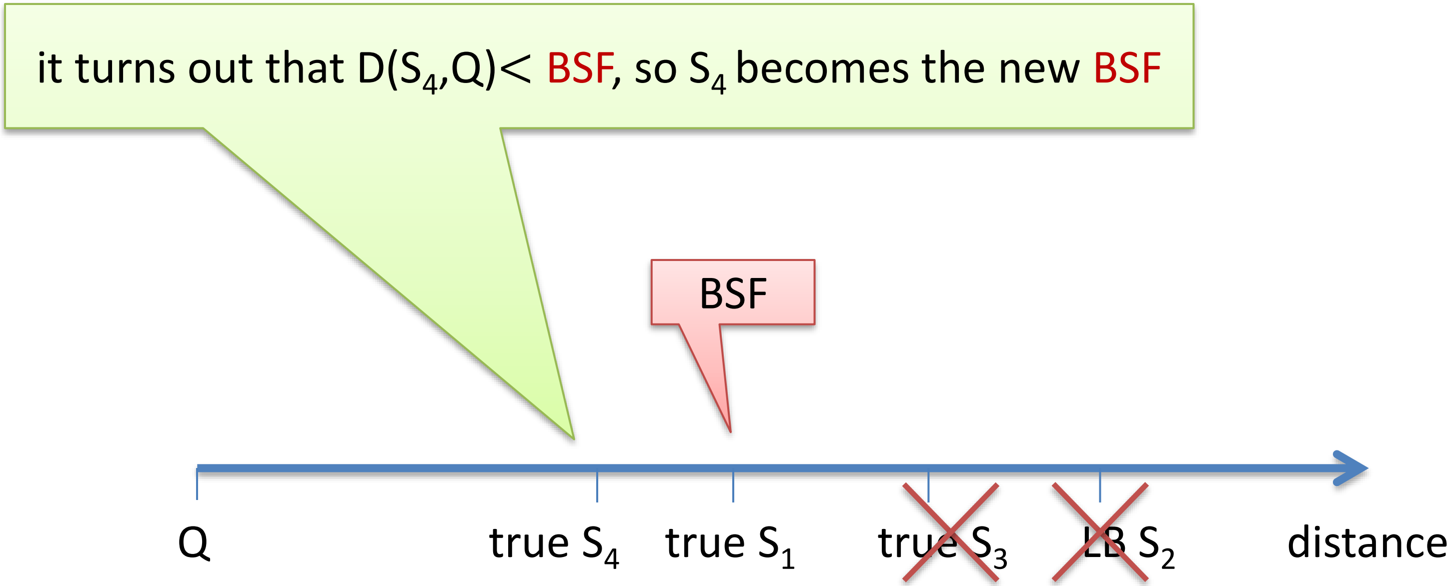


Lower Bounding

we compute the distance LB(S_4,Q) and it is smaller than the BSF
we have to compute D(S_4,Q)≥ LB(S_4,Q), since it may still be smaller than BSF

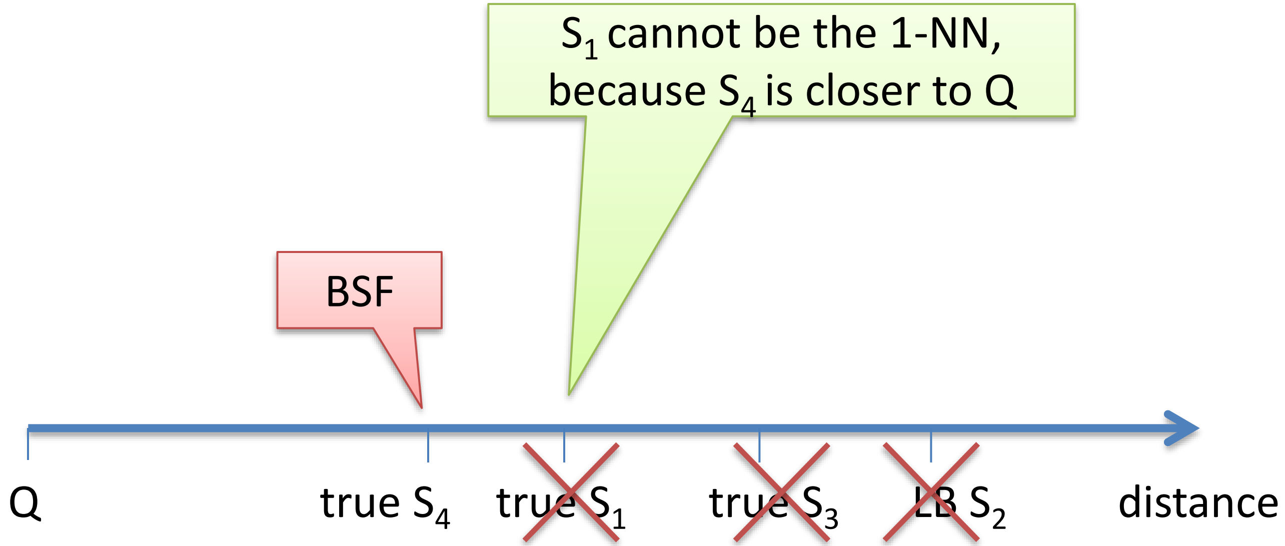


Lower Bounding



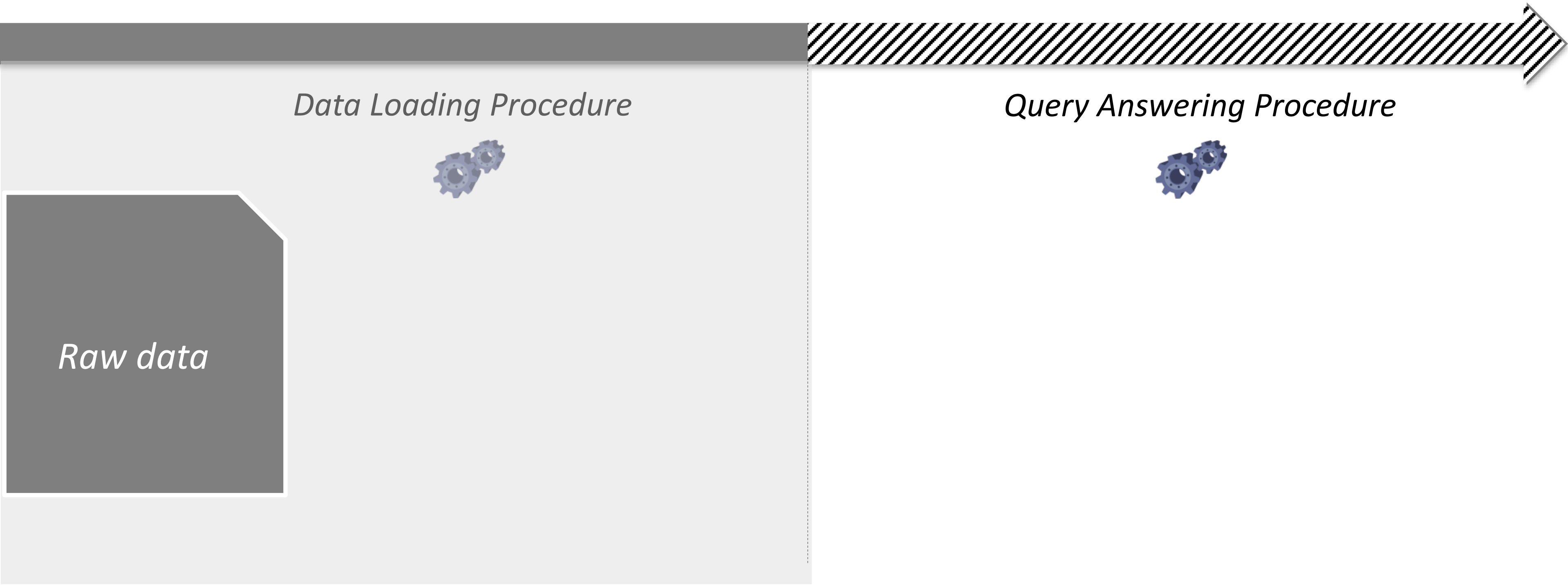
it turns out that $D(S_4,Q) <$ BSF, so S_4 becomes the new BSF

Lower Bounding

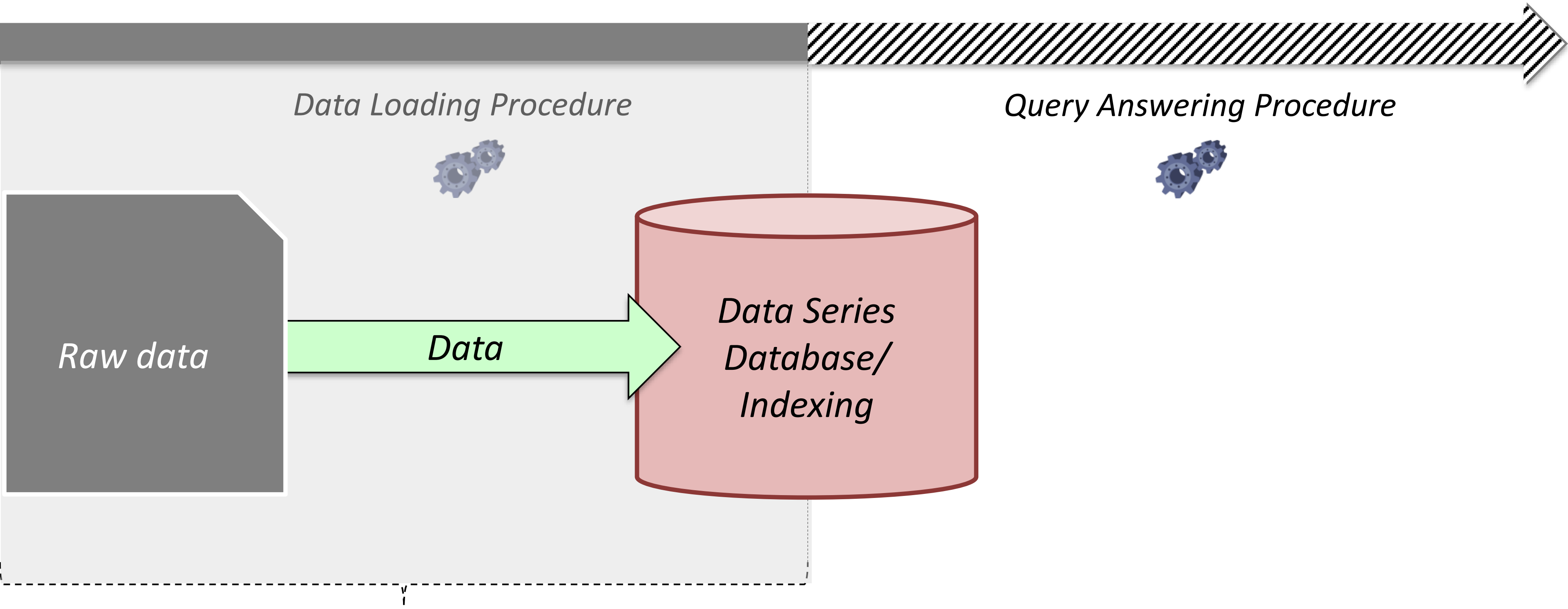


Query Answering

Query answering process

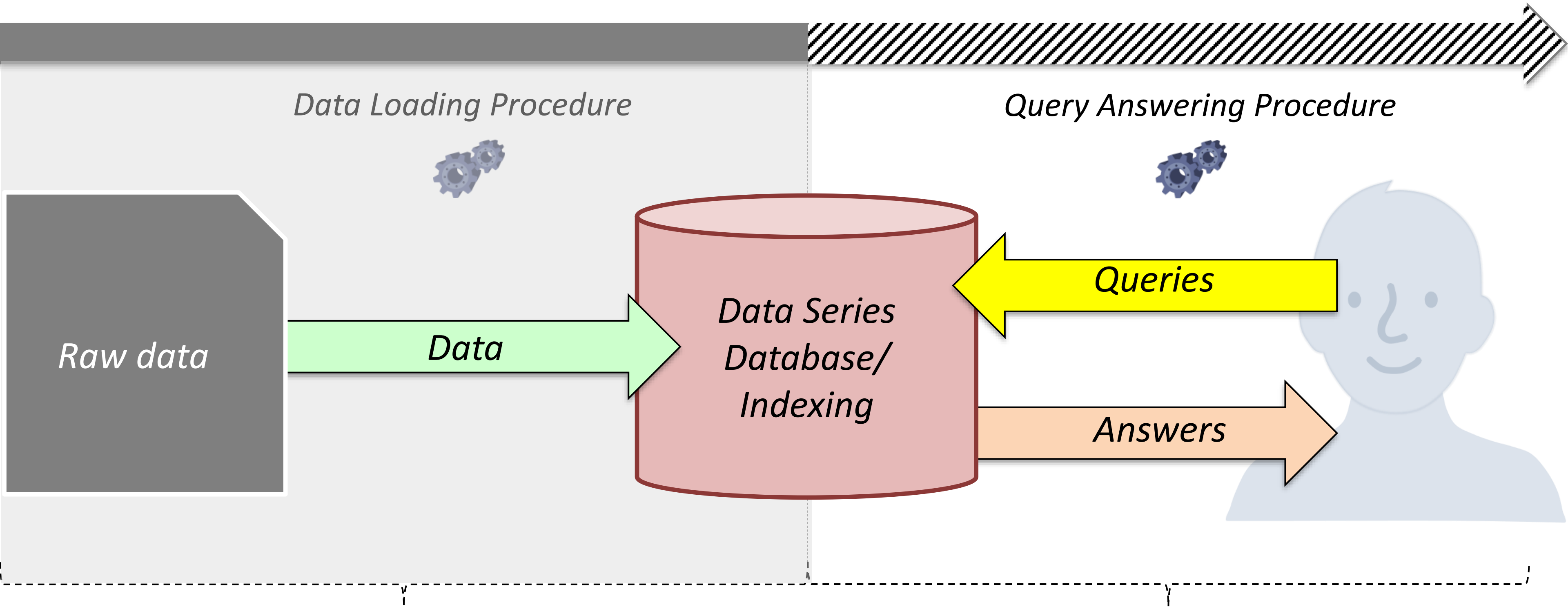


Query answering process



data-to-query time

Query answering process



Data Loading Procedure

Query Answering Procedure

Raw data

Data

Data Series Database/ Indexing

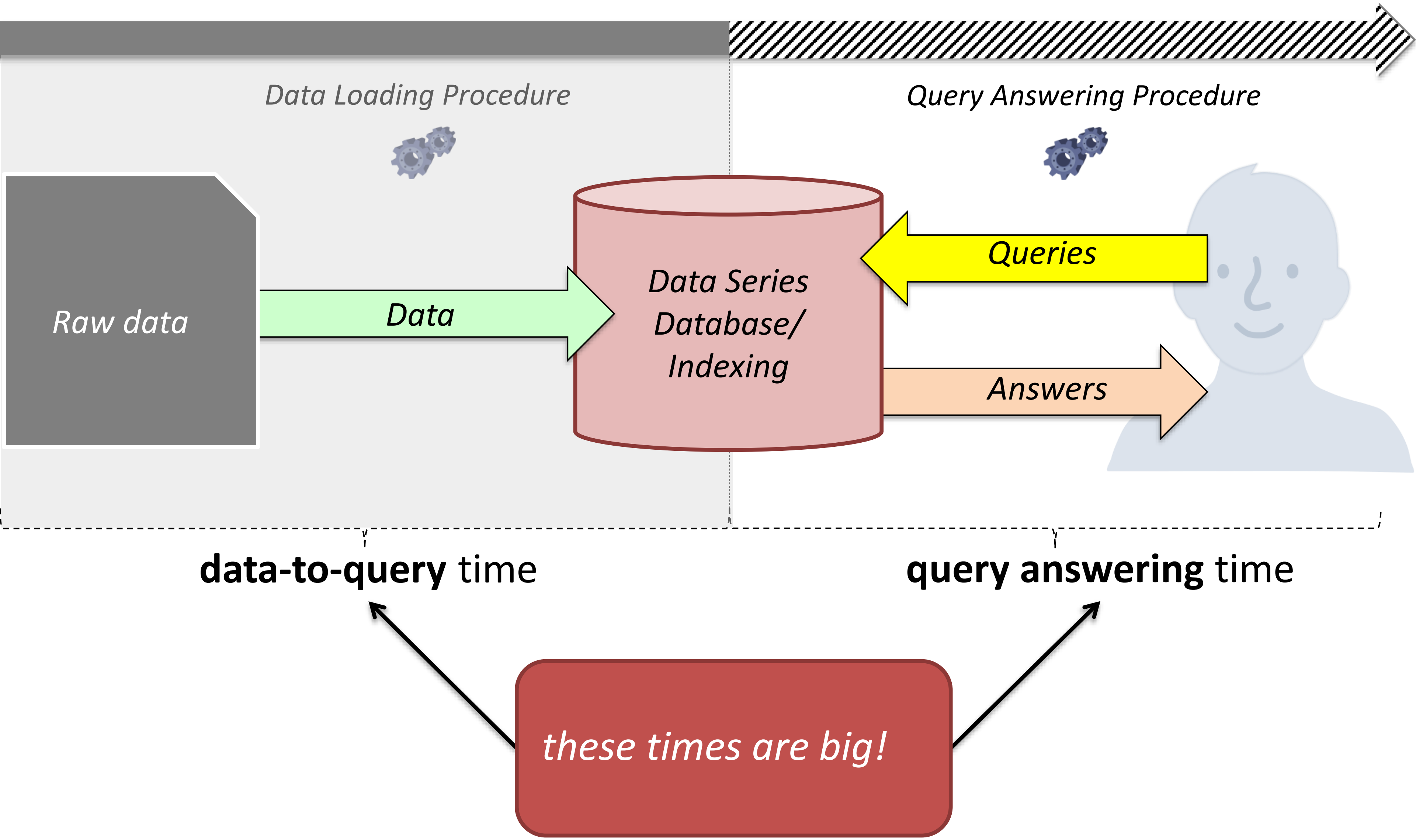
Queries

Answers

data-to-query time

query answering time

Query answering process



Data Loading Procedure

Query Answering Procedure

Raw data

Data

Data Series Database/ Indexing

Queries

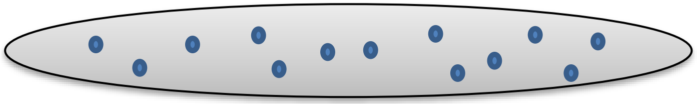
Answers

data-to-query time

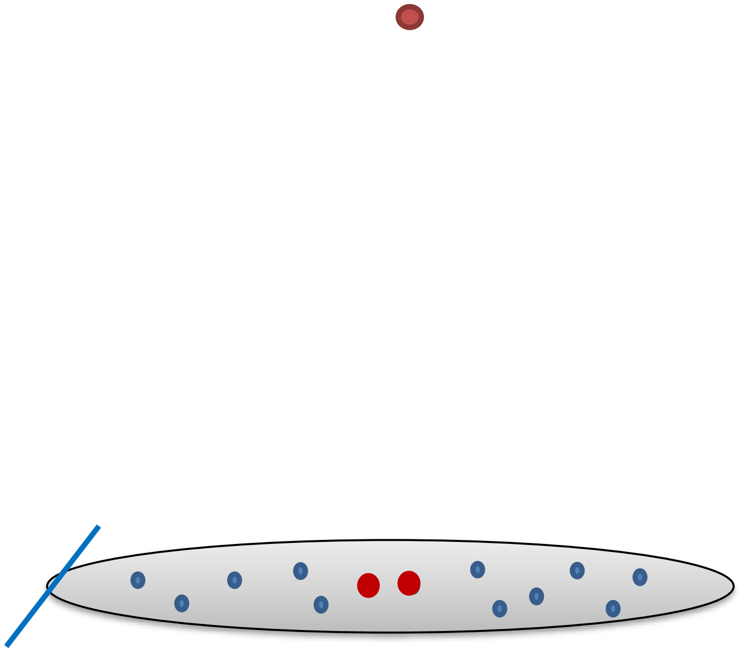
query answering time

these times are big!

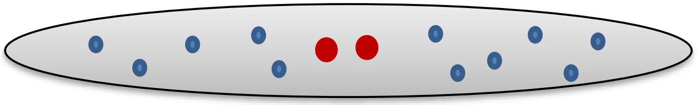
Similarity Search via
Serial Scan



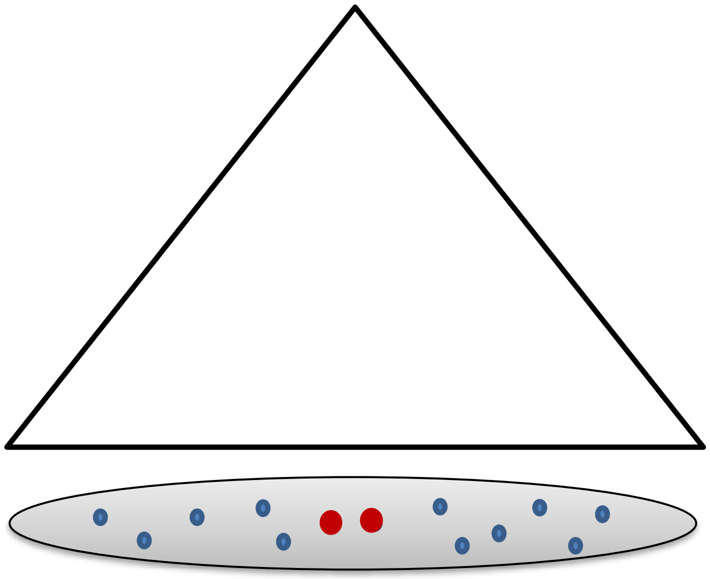
Similarity Search via
Serial Scan



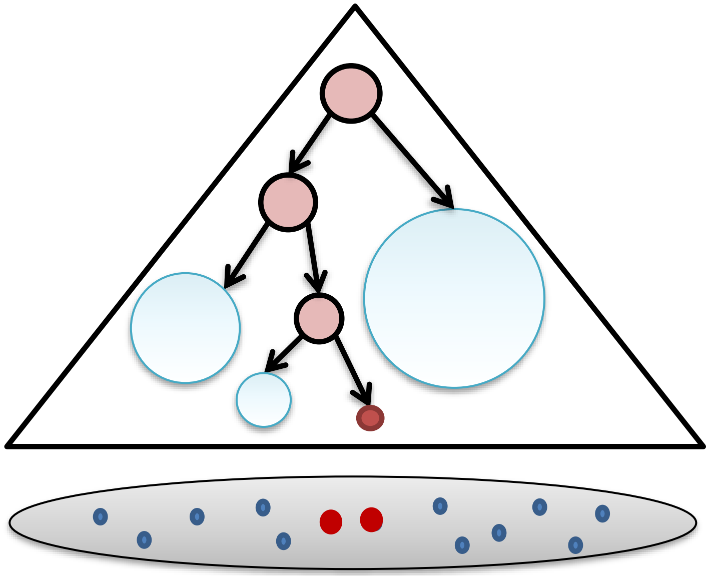
Similarity Search via
Serial Scan



Similarity Search via
Indexing

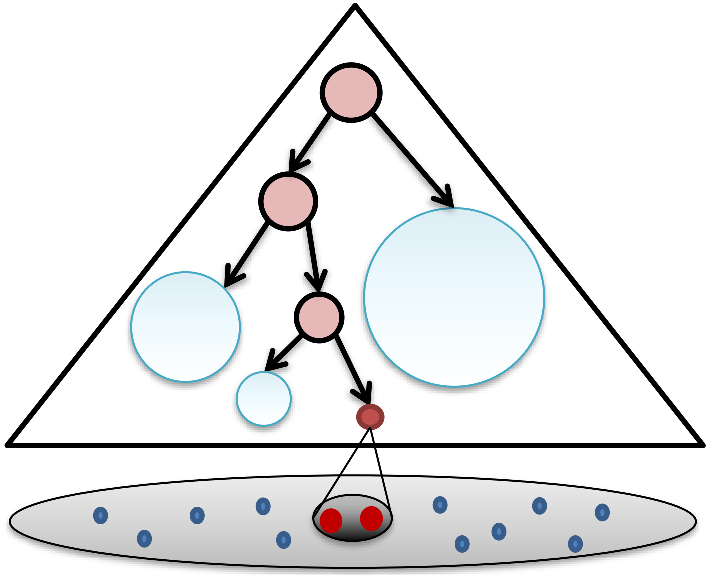


Similarity Search via
Indexing

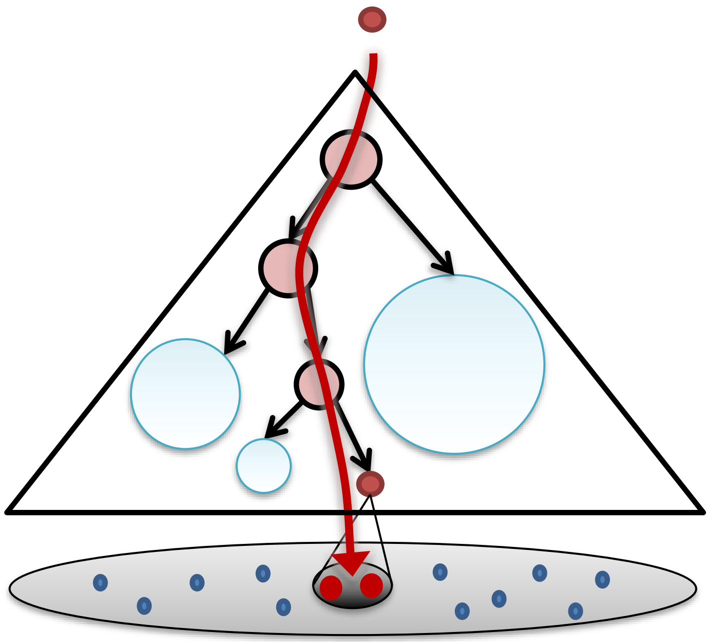


Similarity Search via

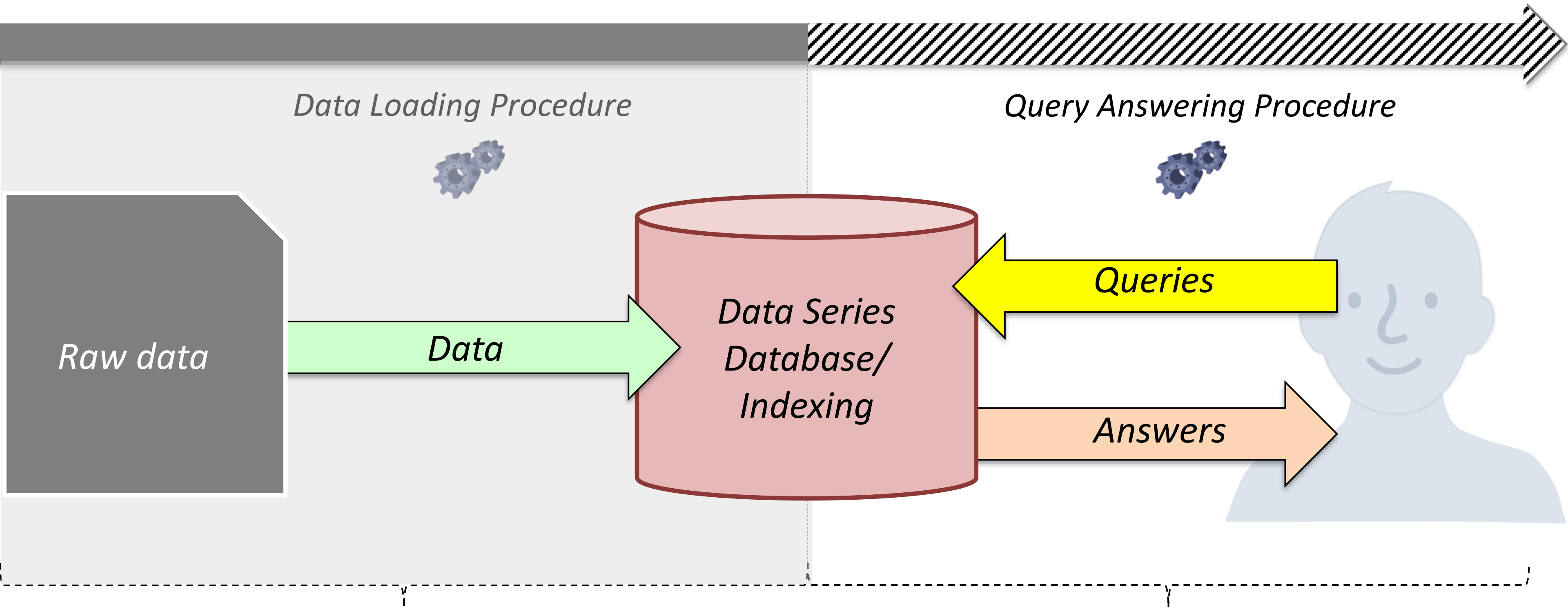
Indexing



Similarity Search via
Indexing



Query answering process



data-to-query time

query answering time

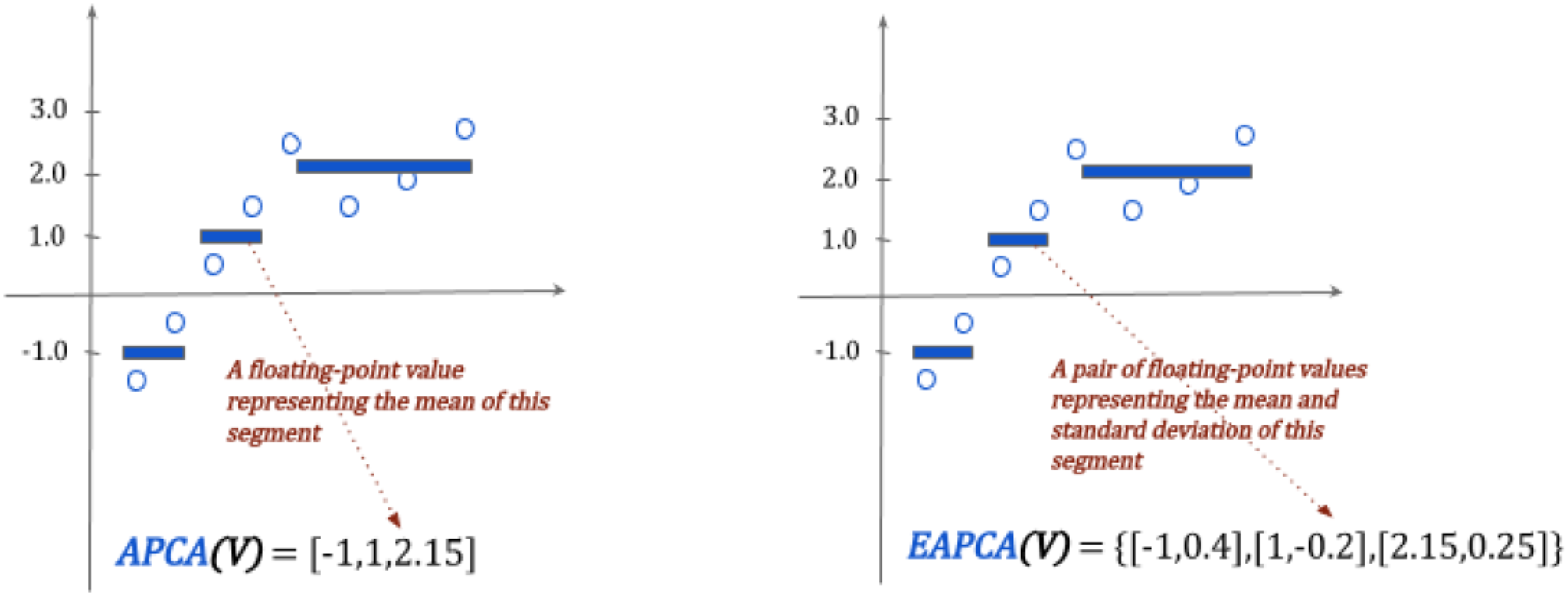
we need solutions for both problems!

Data Series Indexing

DSTree

Summarization

V = [-1.5,-0.5, 0.5, 1.5, 2.5, 1.5,2, 2.6]



(a) APCA

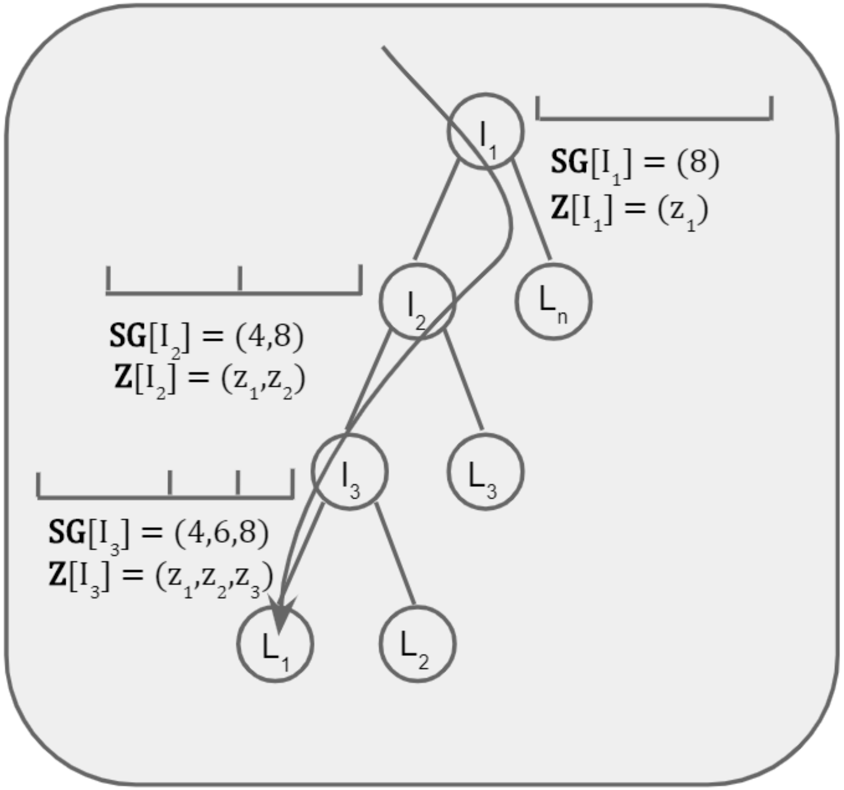
(b) EAPCA

Intertwined with indexing

The APCA and EAPCA representations

DSTree
Indexing

V = [-1.5,-0.5, 0.5, 1.5, 2.5, 1.5,2, 2.6]


I_1
$\mathbf{SG}[I_1] = (8)$
$\mathbf{Z}[I_1] = (z_1)$
I_2
L_n
$\mathbf{SG}[I_2] = (4,8)$
$\mathbf{Z}[I_2] = (z_1,z_2)$
I_3
L_3
$\mathbf{SG}[I_3] = (4,6,8)$
$\mathbf{Z}[I_3] = (z_1,z_2,z_3)$
L_1
L_2

Each node contains

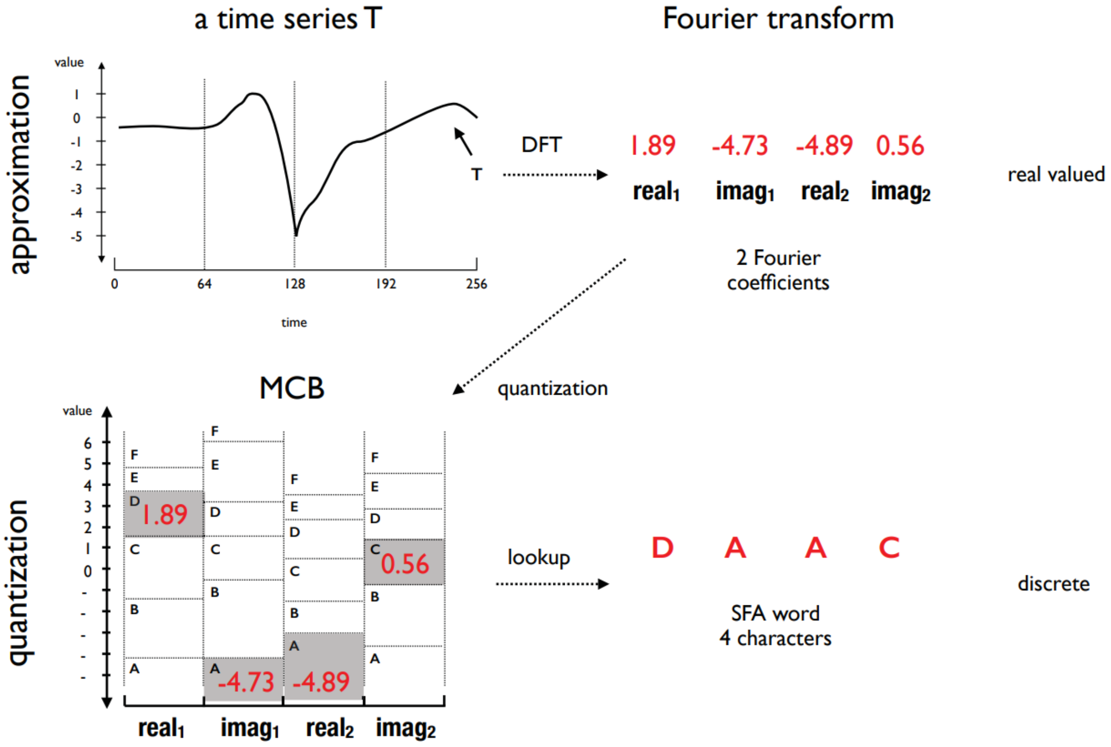
- ❑ # vectors
- ❑ segmentation **SG**
- ❑ synopsis **Z**

Each Leaf node also :

- ❑ stores its raw vectors in a separate disk file

Symbolic Fourier Approximation (SFA)

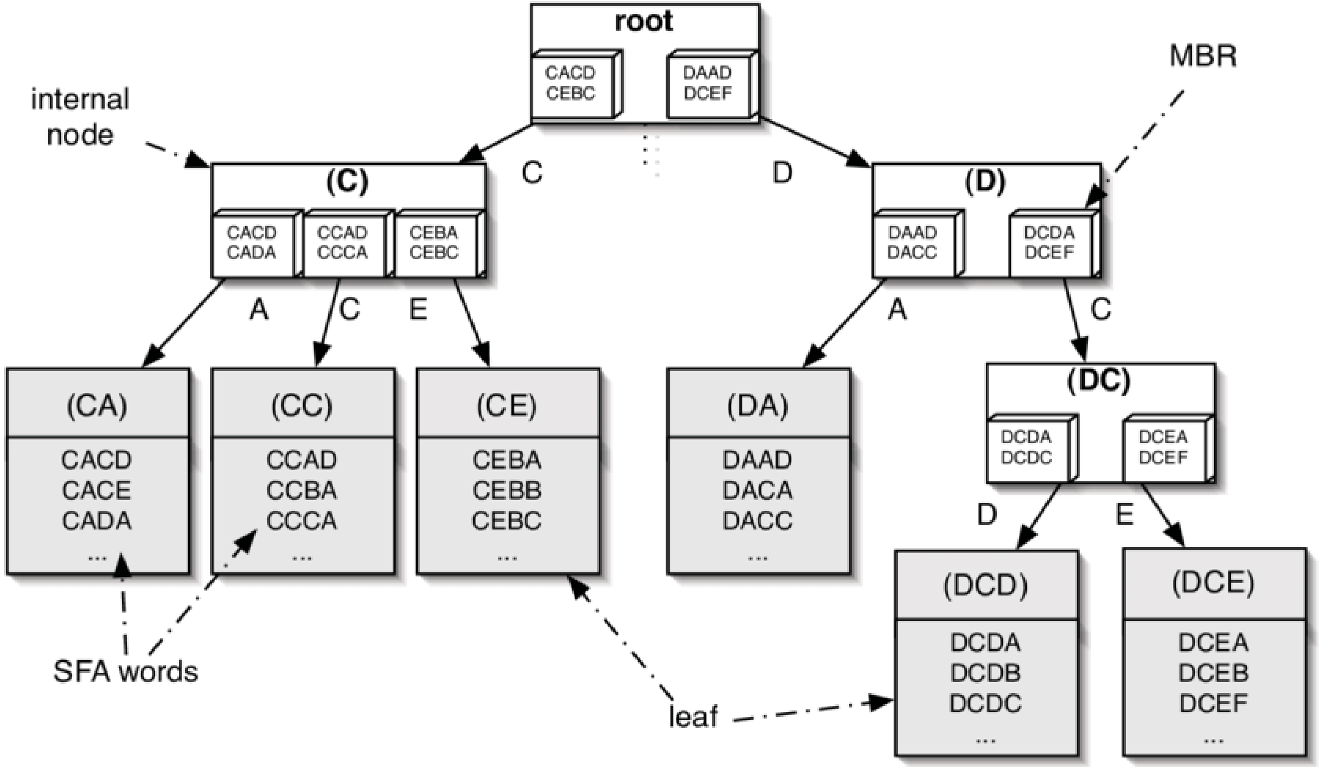
Summarization



The SFA representation*

*https://www2.informatik.hu-berlin.de/~schaefpa/talks/scalable_classification.pptx

SFA
Indexing



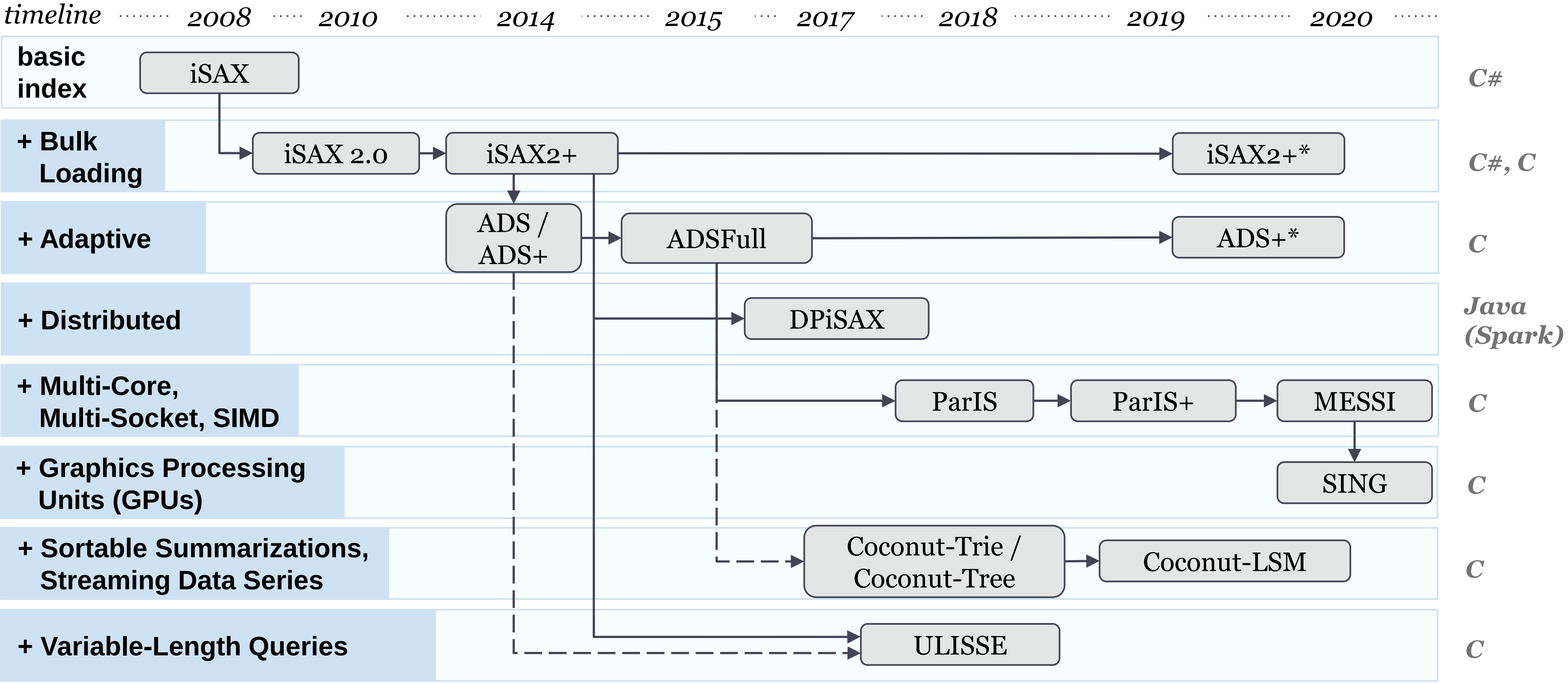
The SFA Trie*

*https://www2.informatik.hu-berlin.de/~schaefpa/talks/scalable_classification.pptx

iSAX Index Family

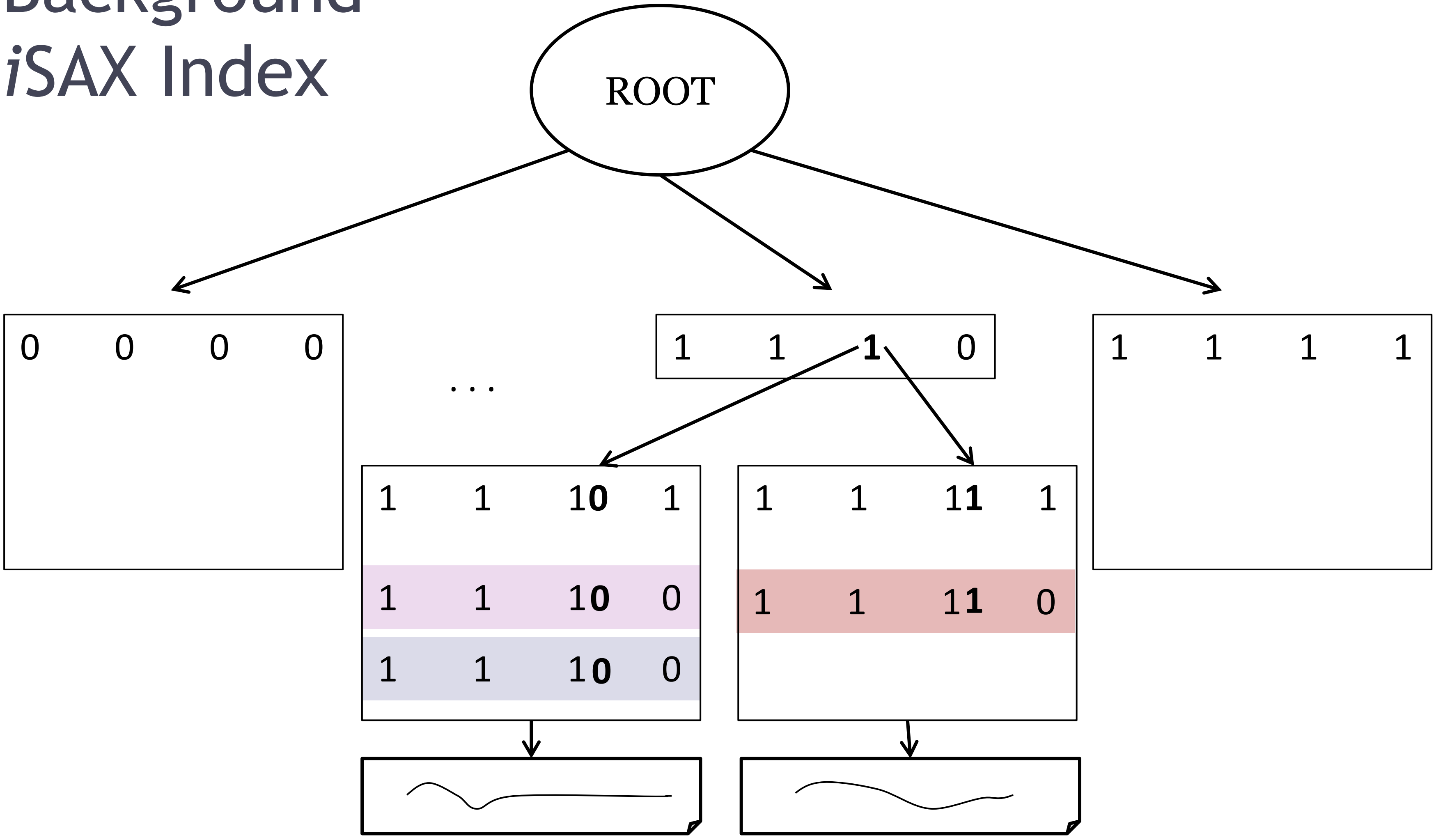
Publications

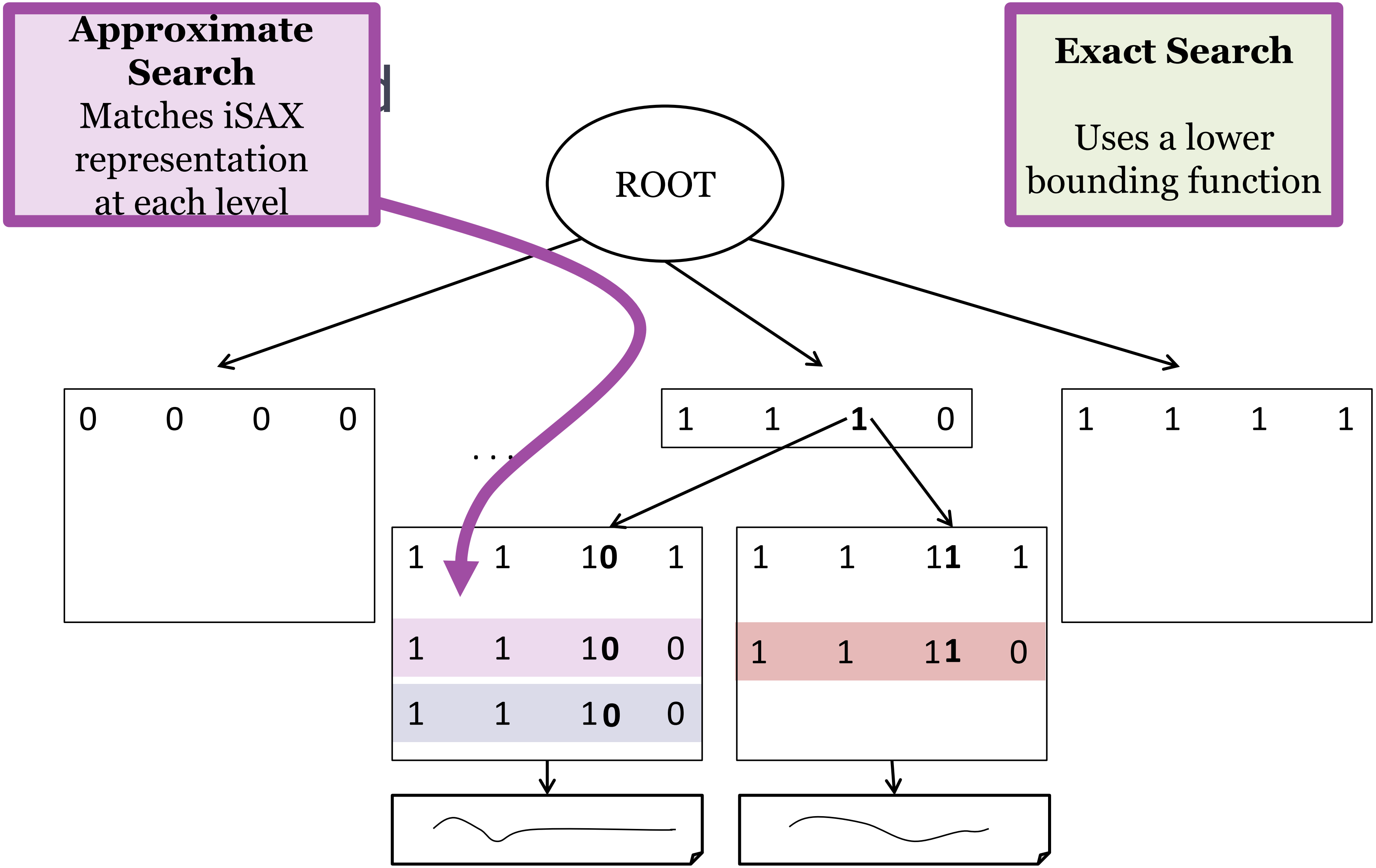
CCIS'19



Timeline depicted on top; implementation languages marked on the right. Solid arrows denote inheritance of index design; dashed arrows denote inheritance of some of the design features; two new versions of iSAX2+/ADS+ marked with asterisk support approximate similarity search with deterministic and probabilistic quality guarantees.

Background
*i*SAX Index





Approximate Search
Matches iSAX representation at each level

Exact Search

Uses a lower bounding function

iSAX 2.0
Bulk Loading Algorithm

- design principles:
  - take advantage of available main memory
  - maximize sequential disk accesses

iSAX 2.0
Bulk Loading Algorithm

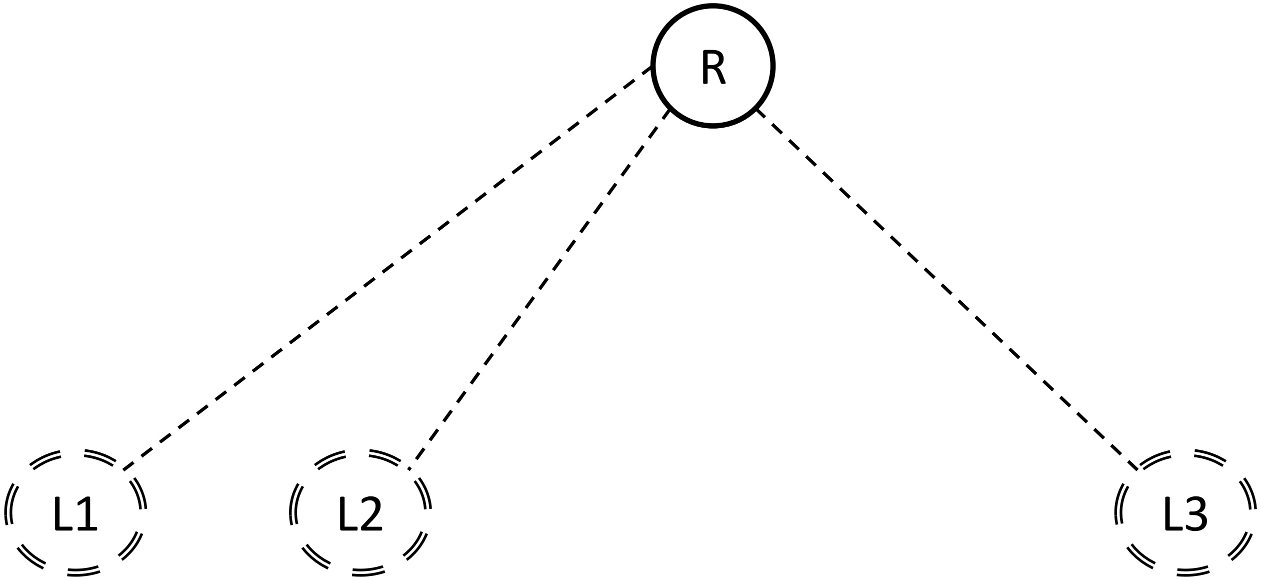
- intuition for proposed solution:
  - for each leaf node, collect as many data series that belong to it as possible before materializing the leaf node
    - the raw values of data series in leaf nodes are written to disk

iSAX 2.0
Bulk Loading Algorithm

Publications

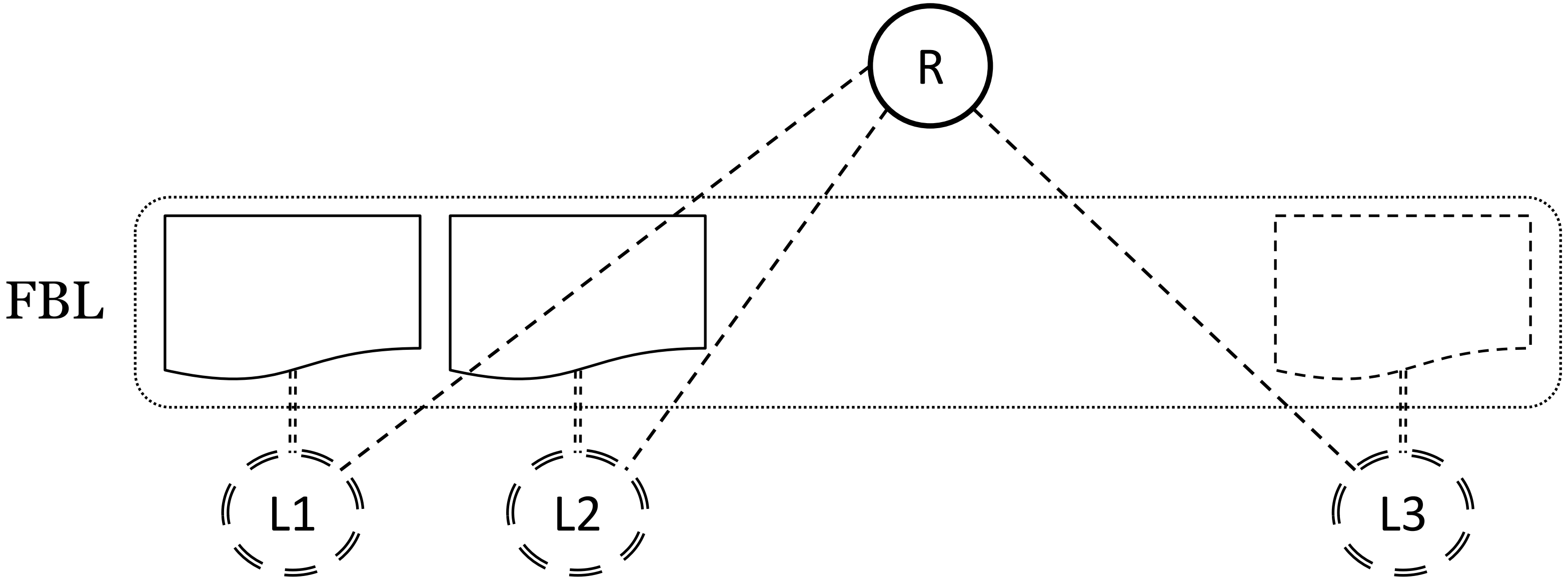
ICDM'10

- iterate between two phases (till all data series are indexed):
  - Phase 1
    - read data series and group them according to first-level nodes
    - use all available main memory
  - Phase 2
    - grow index by processing the subtree rooted at each one of the first-level nodes one at-a-time
    - flush leaf node contents to disk using sequential accesses



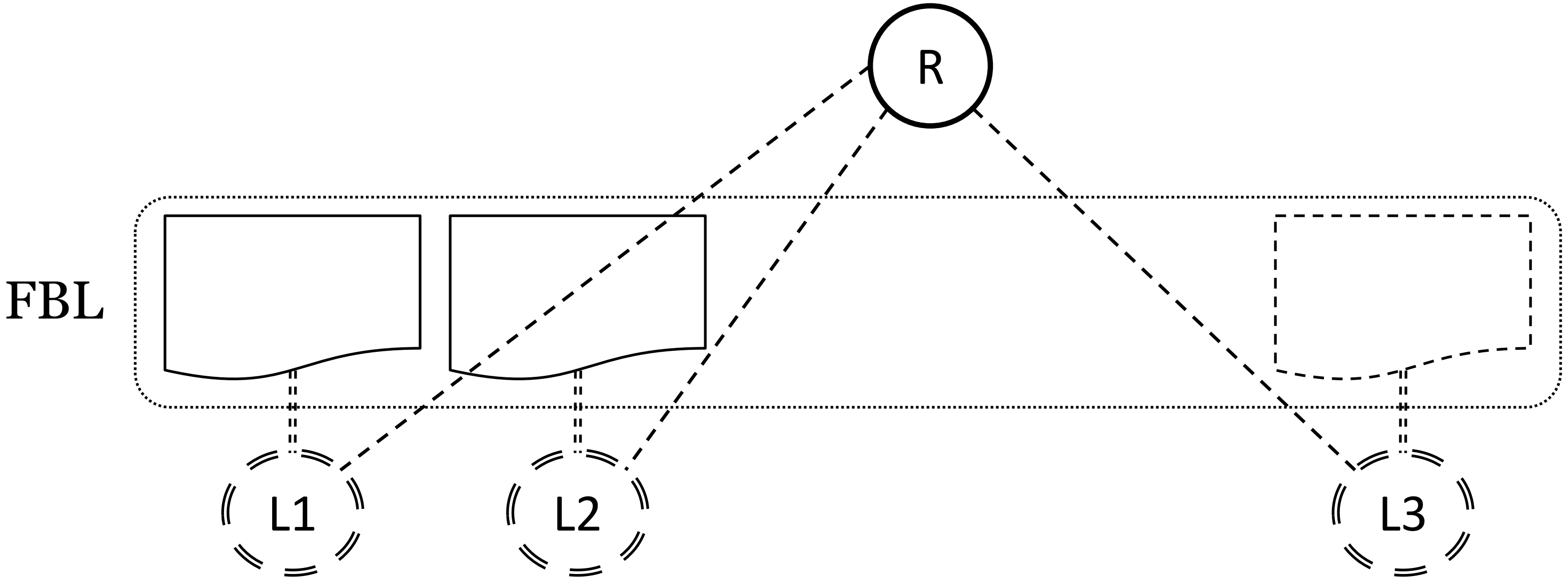
main memory

disk



main memory

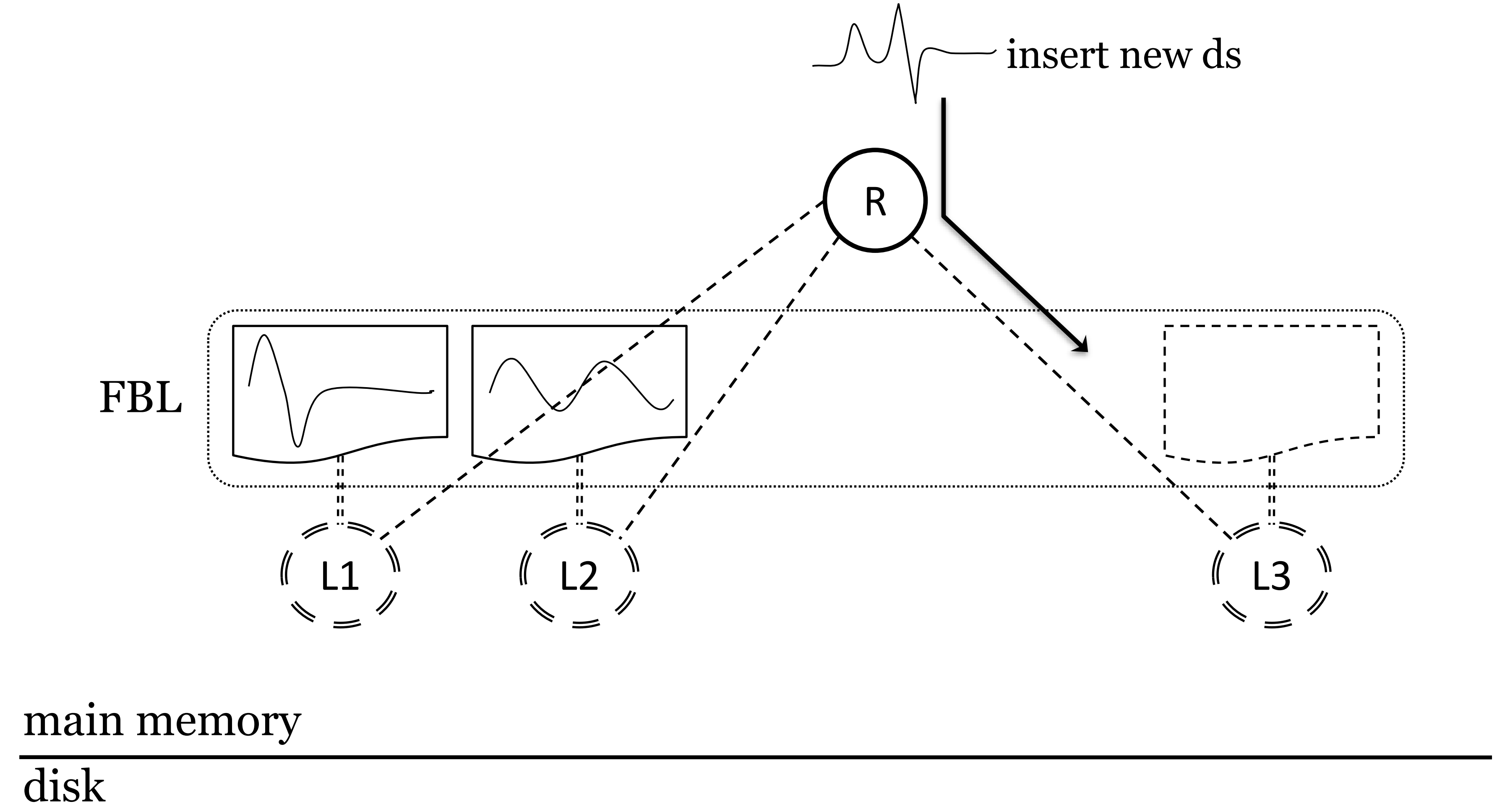
disk

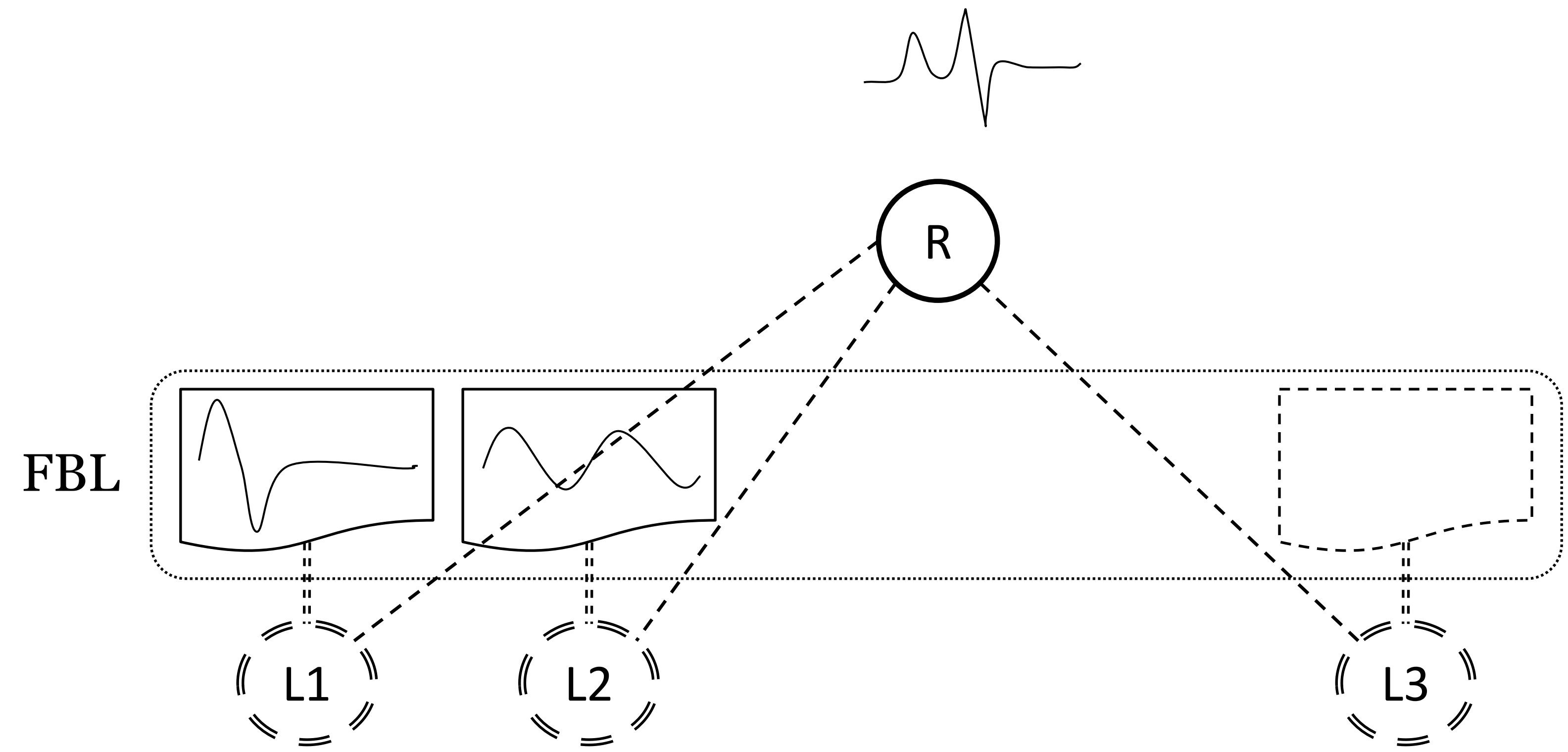


no limit in the size of FBLs!

main memory

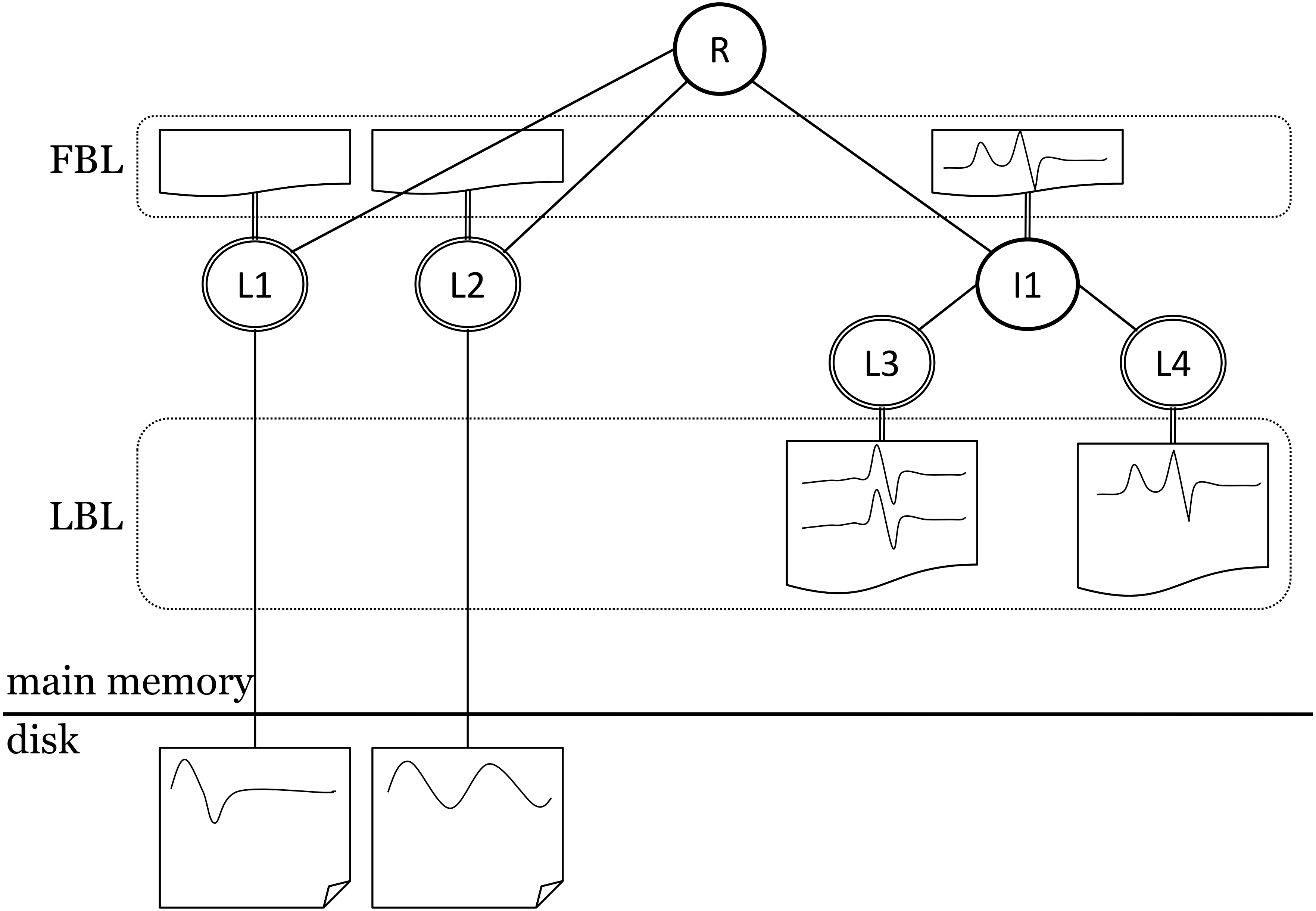
disk

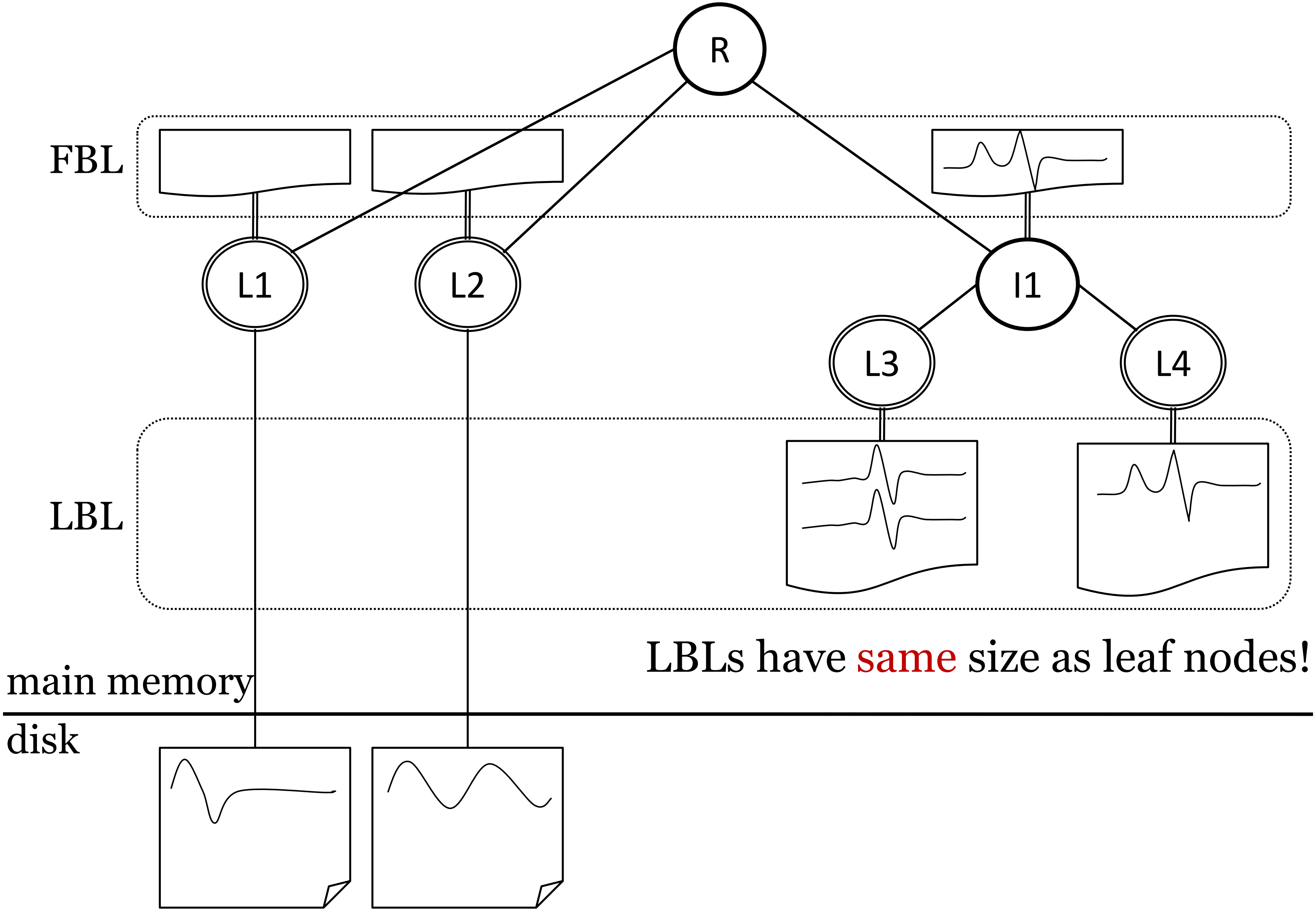

insert new ds
R
FBL
L1
L2
L3
main memory
disk



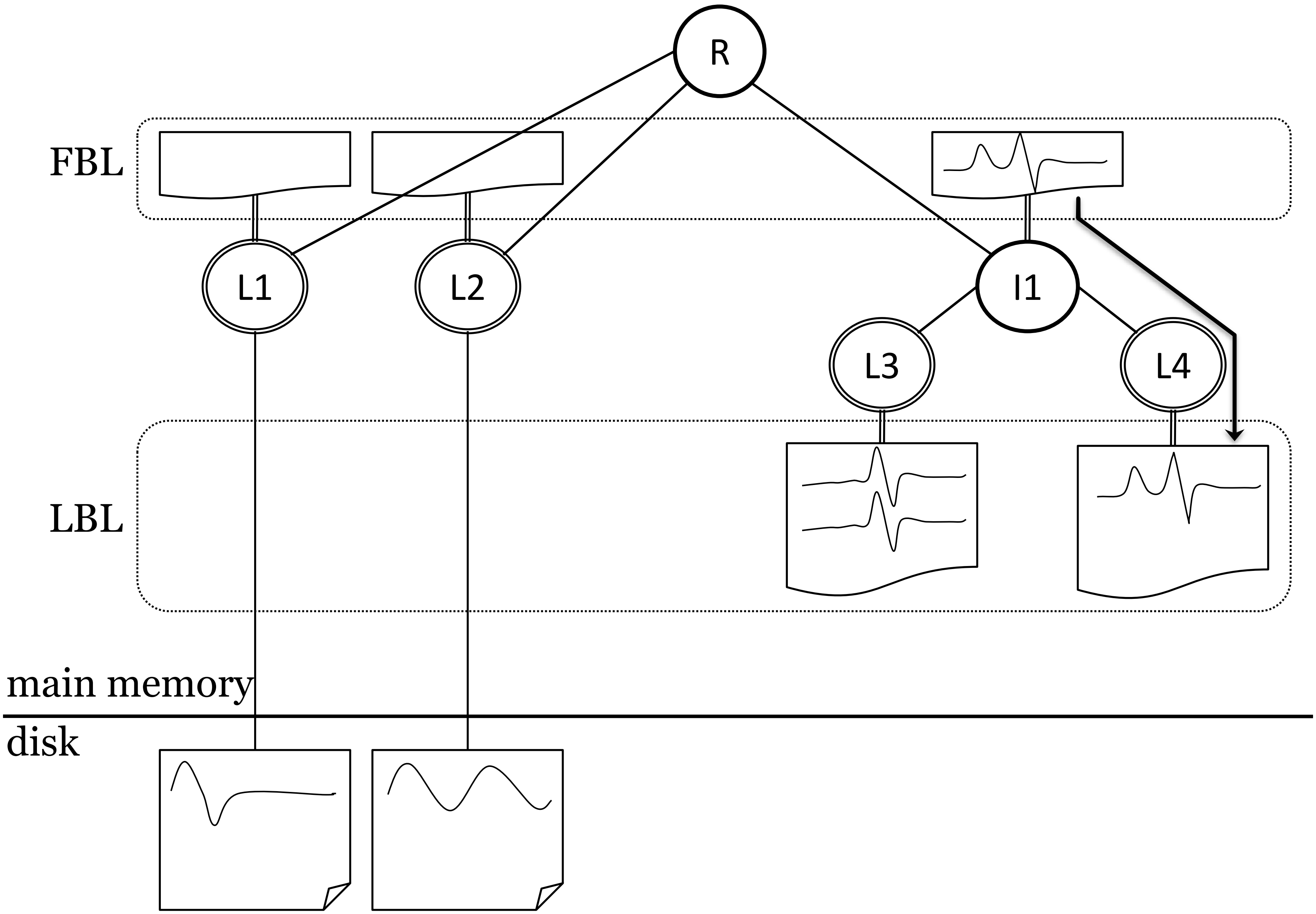
main memory

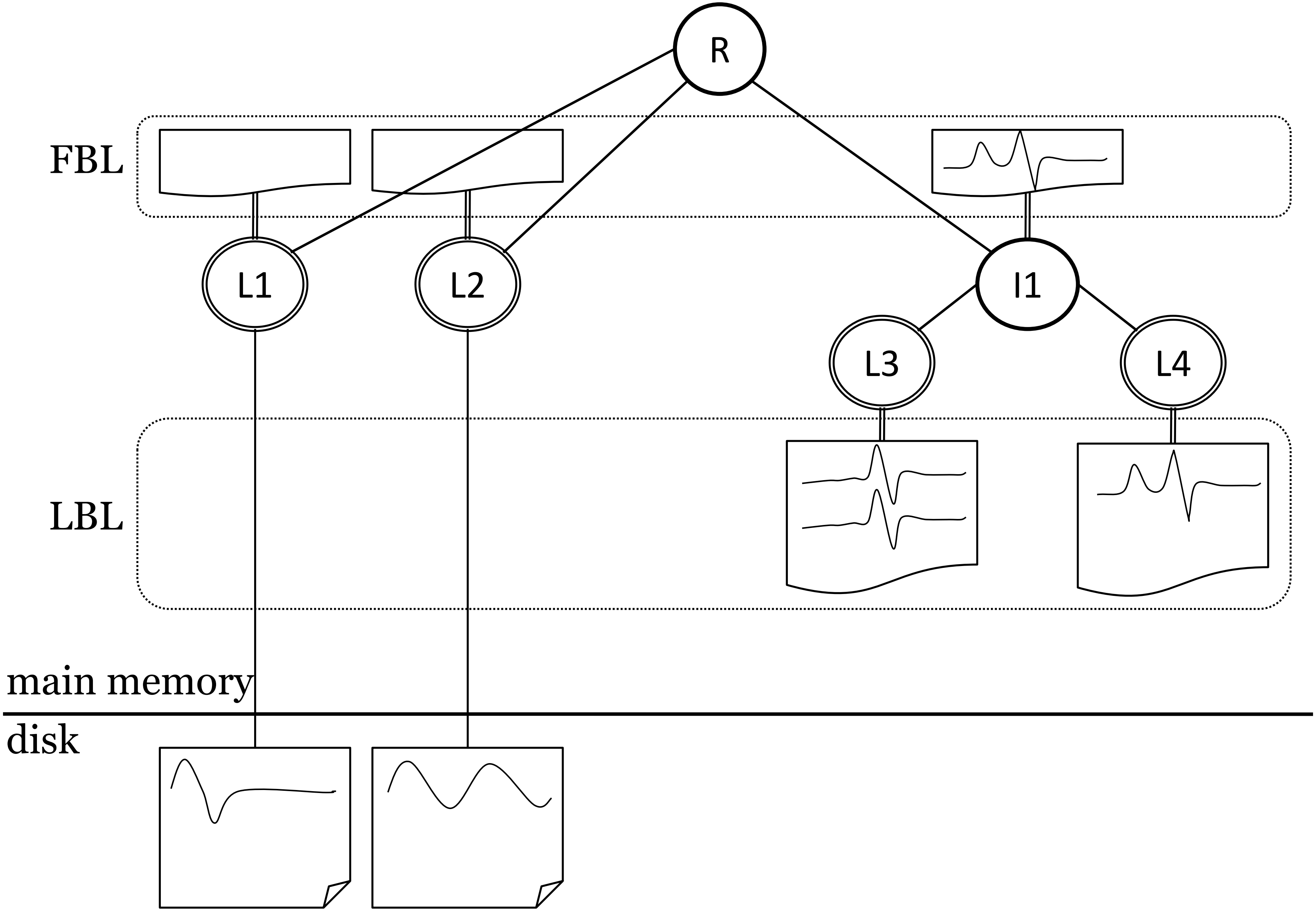
disk

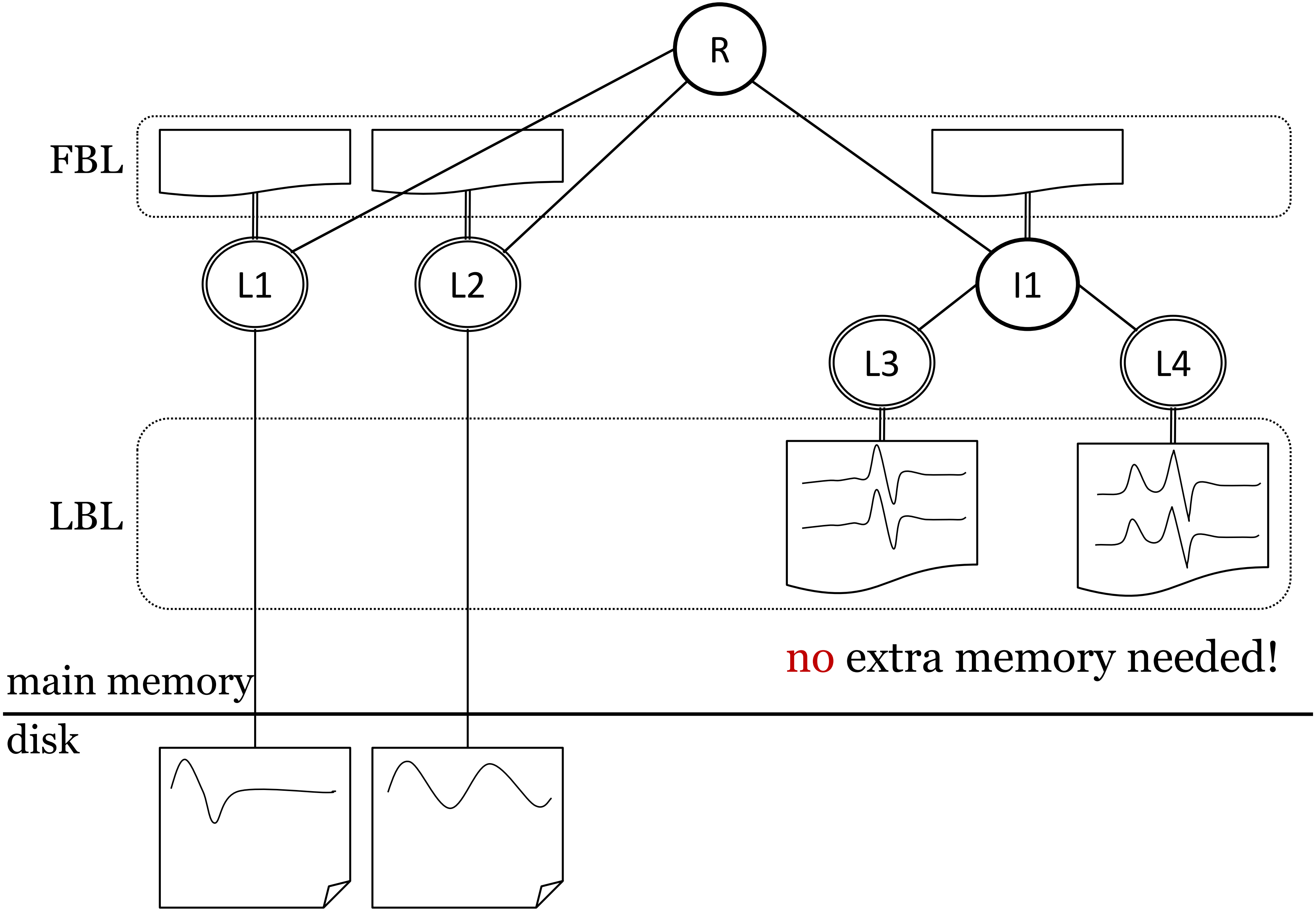

R
FBL
L1
L2
I1
L3
L4
LBL
main memory
disk



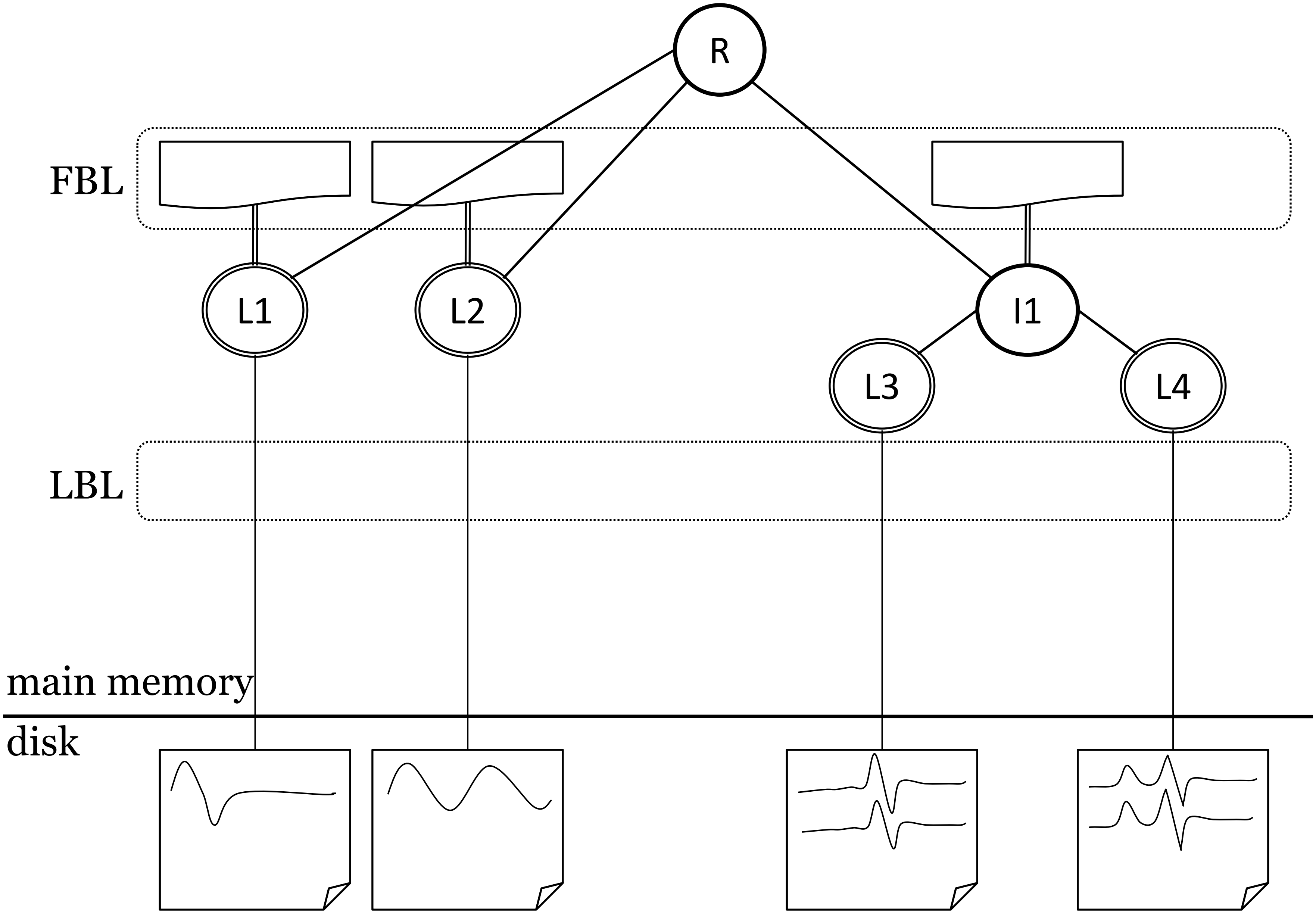
LBLs have same size as leaf nodes!

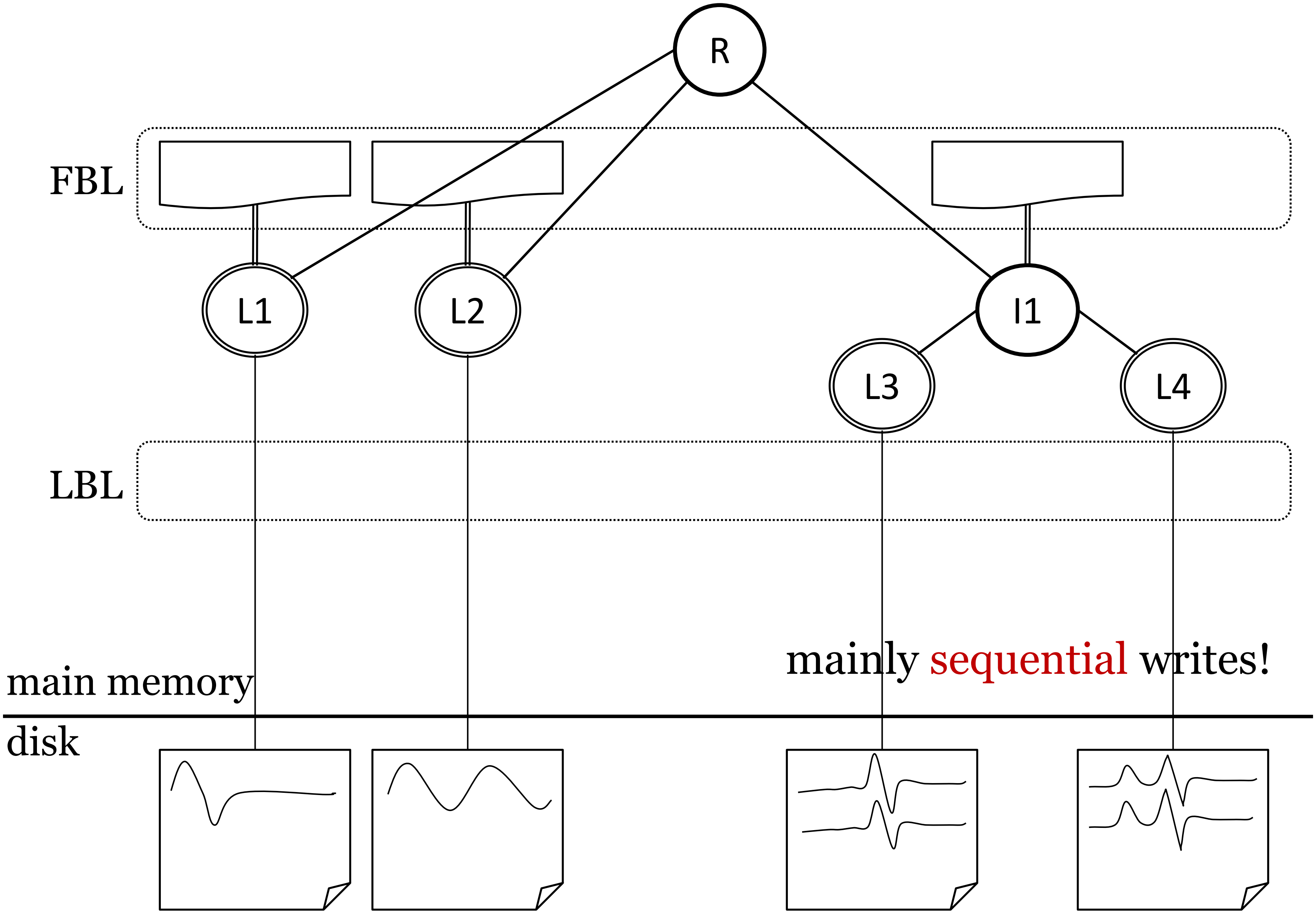

R
FBL
L1
L2
I1
L3
L4
LBL
main memory
disk


R
FBL
L1
L2
I1
L3
L4
LBL
main memory
disk



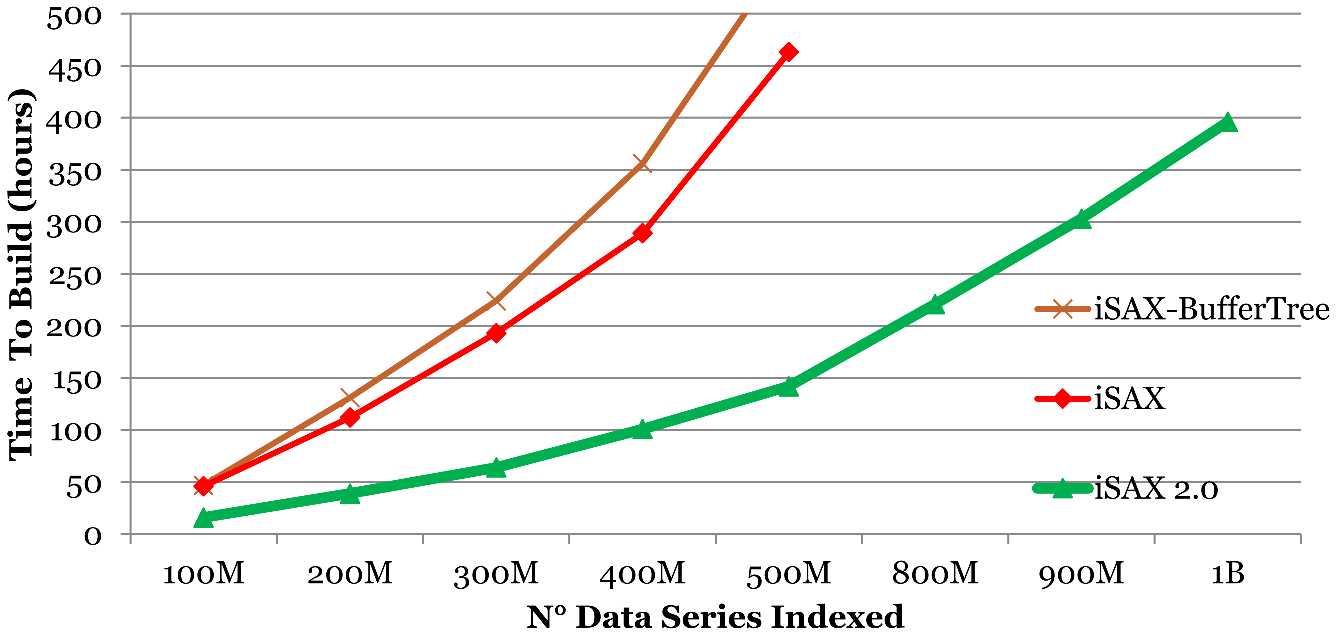
no extra memory needed!


R
FBL
L1
L2
I1
L3
L4
LBL
main memory
disk



mainly sequential writes!

Experimental Evaluation
Bulk Loading



- 1 Billion data series indexed in 16 days: 72% less time
- indexing time per data series: 0.001 sec

iSAX2+

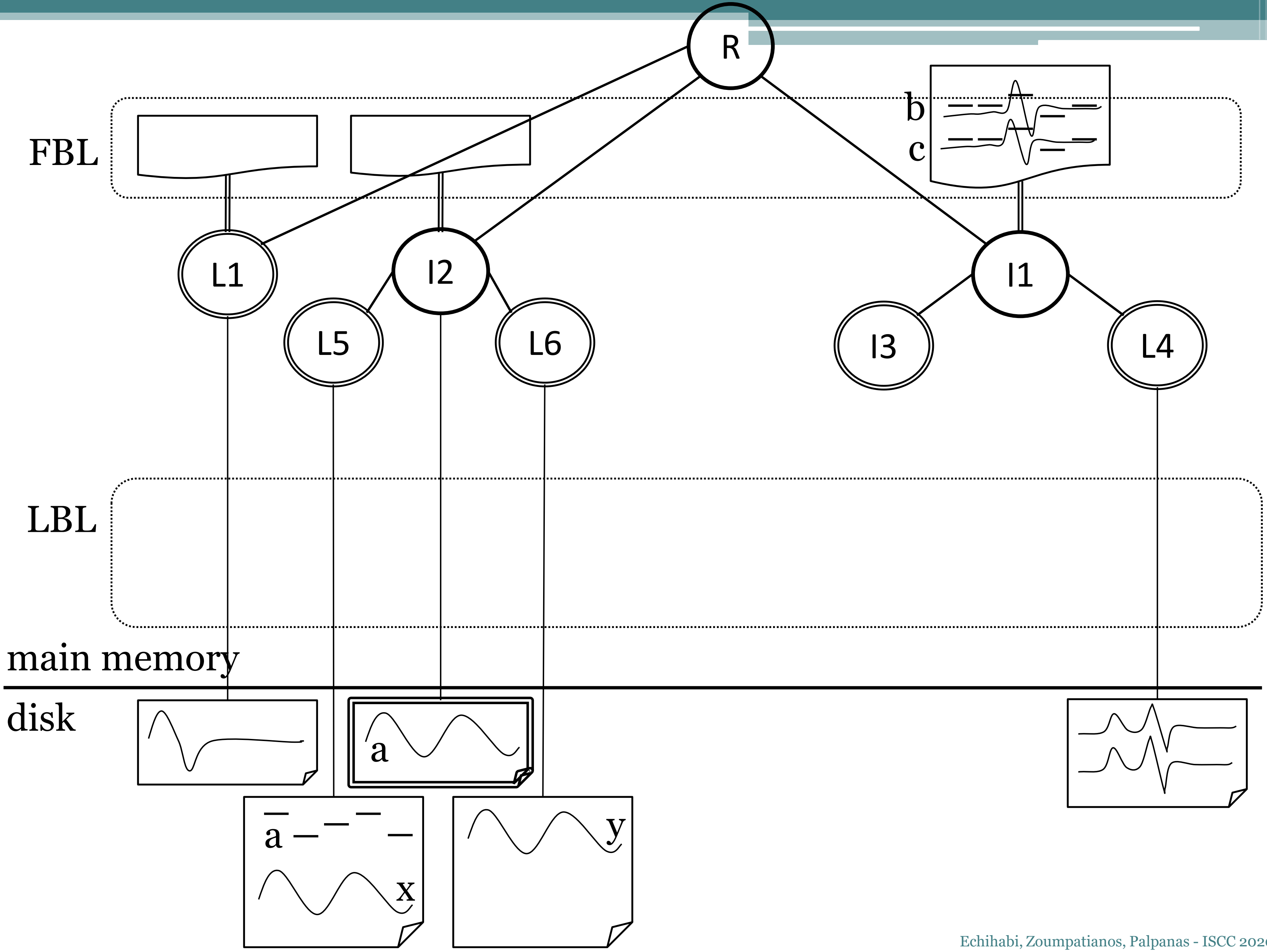
- design principle:
  - do not move around (read/write) raw data of data series and its approximation unless necessary

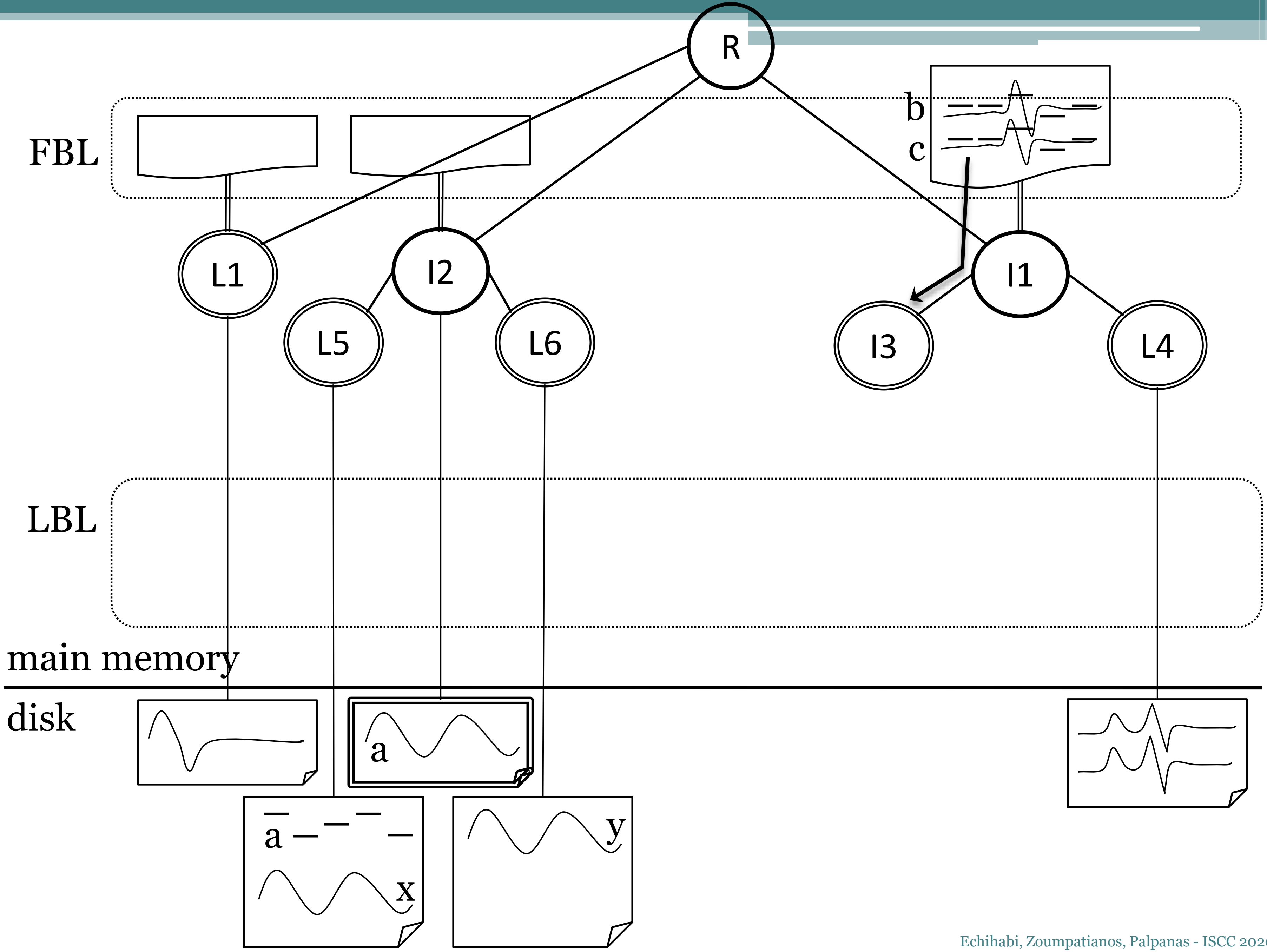
Publications

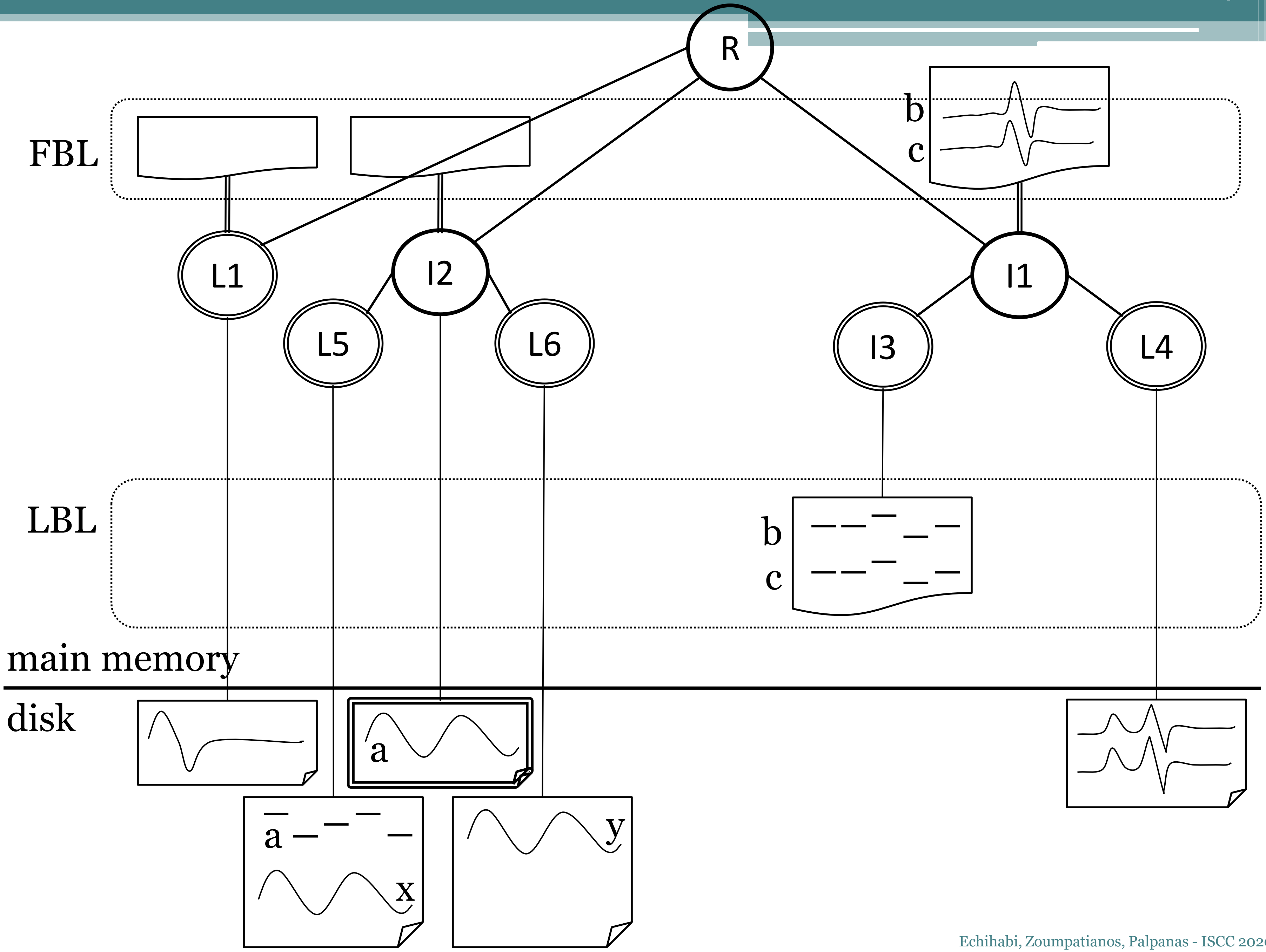
KAIS‘14

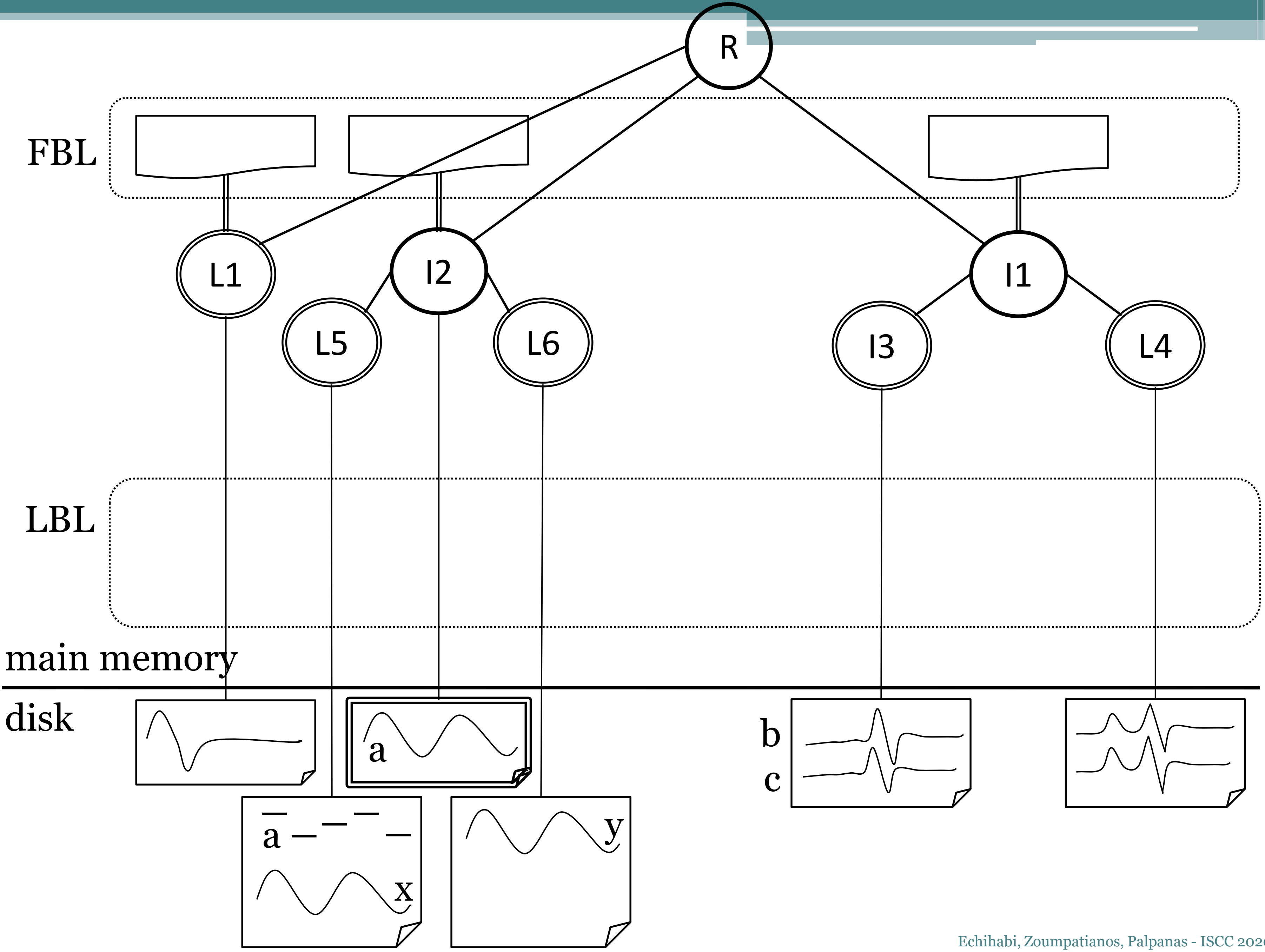
*i*SAX2+

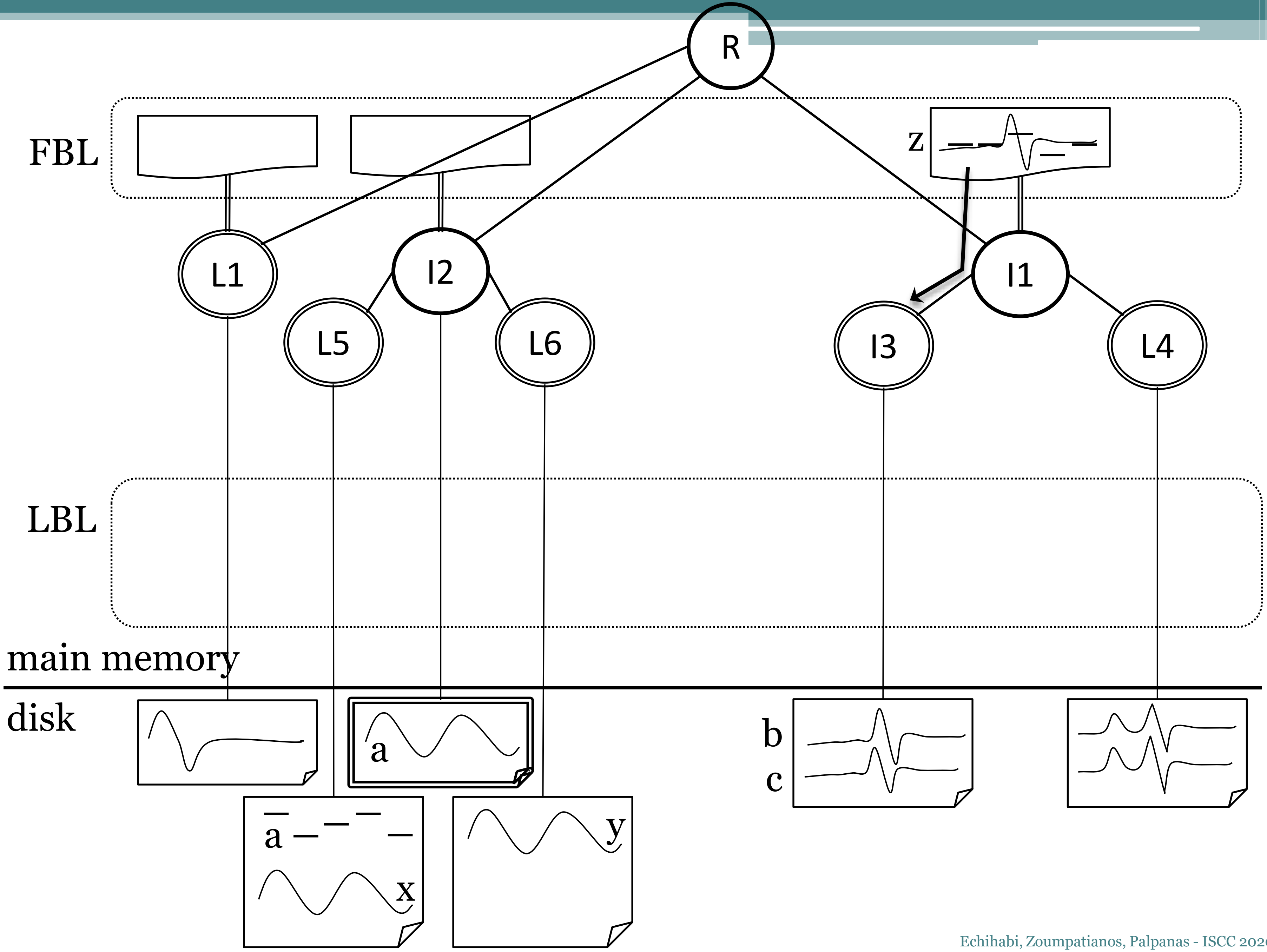
- intuition for proposed solution:
  - *i*SAX grows fast at the beginning of bulk loading, its shape stabilizing well before the end of the process
  - several data series end up in leaf nodes that never need to split
  - implement lazy splitting:
    - move raw data to leaf node the first time
    - if leaf node splits, do not move raw data until the end of index building process

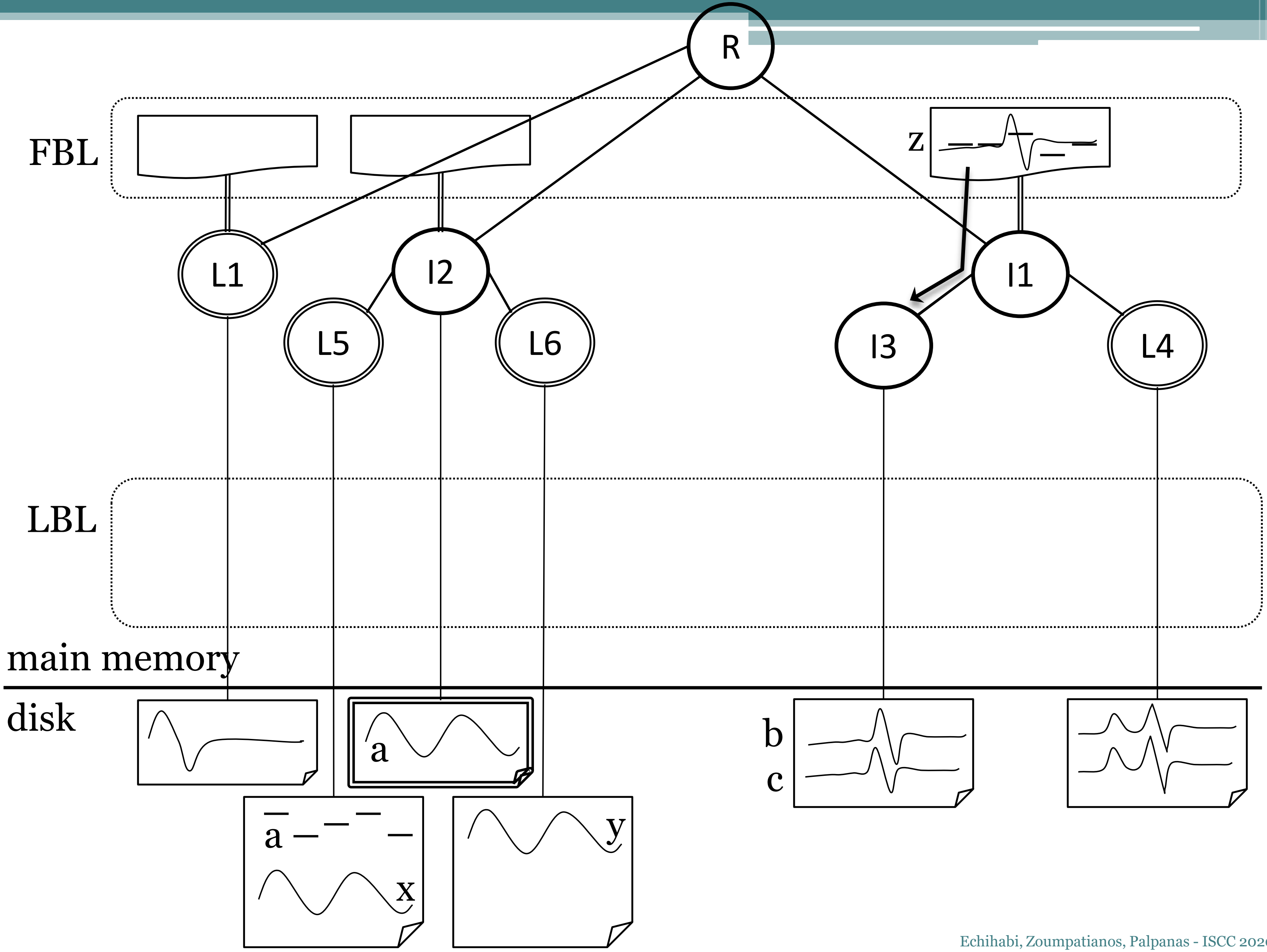

R
FBL
b
c
L1
I2
I1
L5
L6
I3
L4
LBL
main memory
disk
a
a
x
y

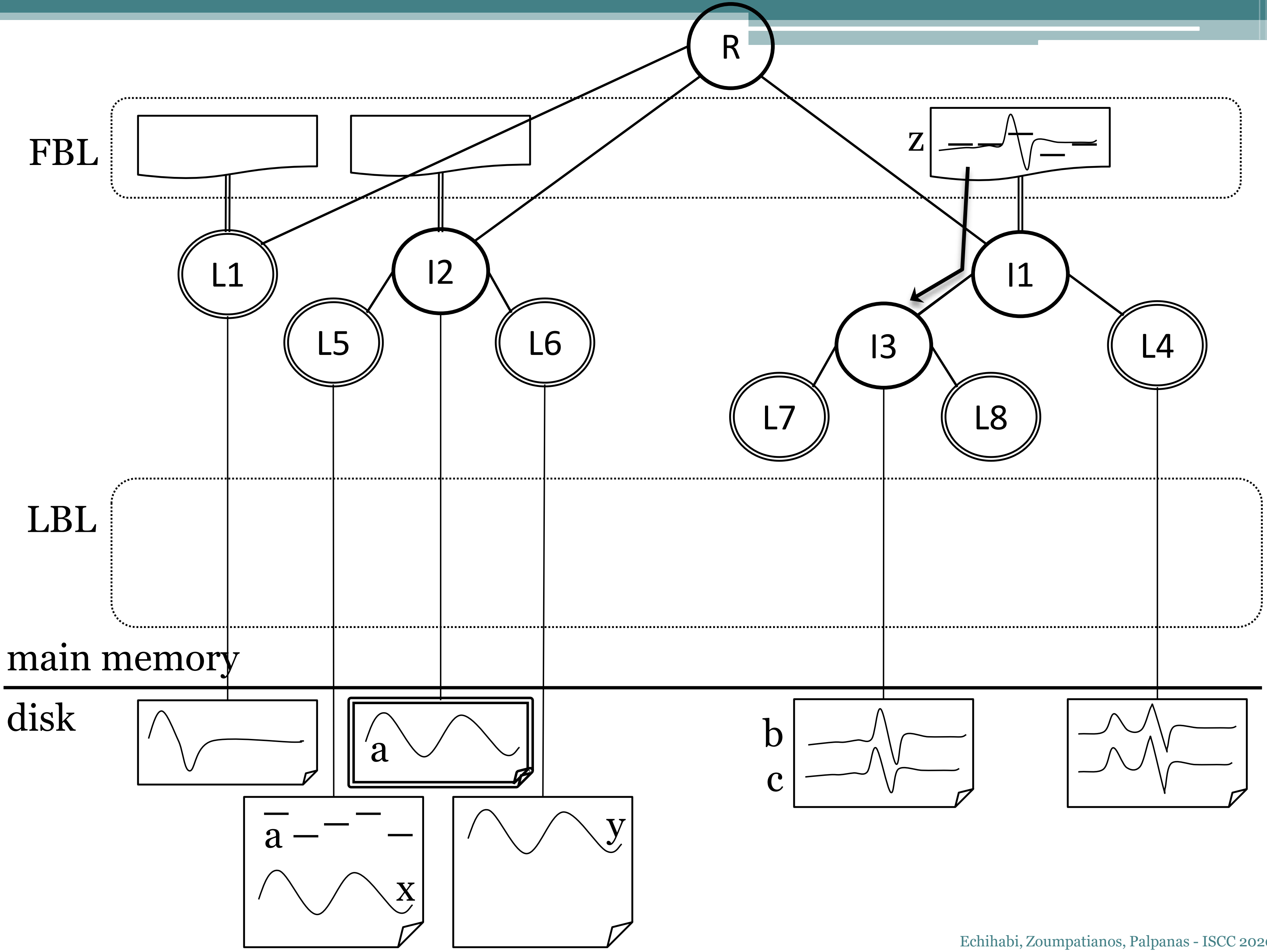

R
FBL
b
c
L1
I2
L5
L6
I3
I1
L4
LBL
main memory
disk
a
a
x
y

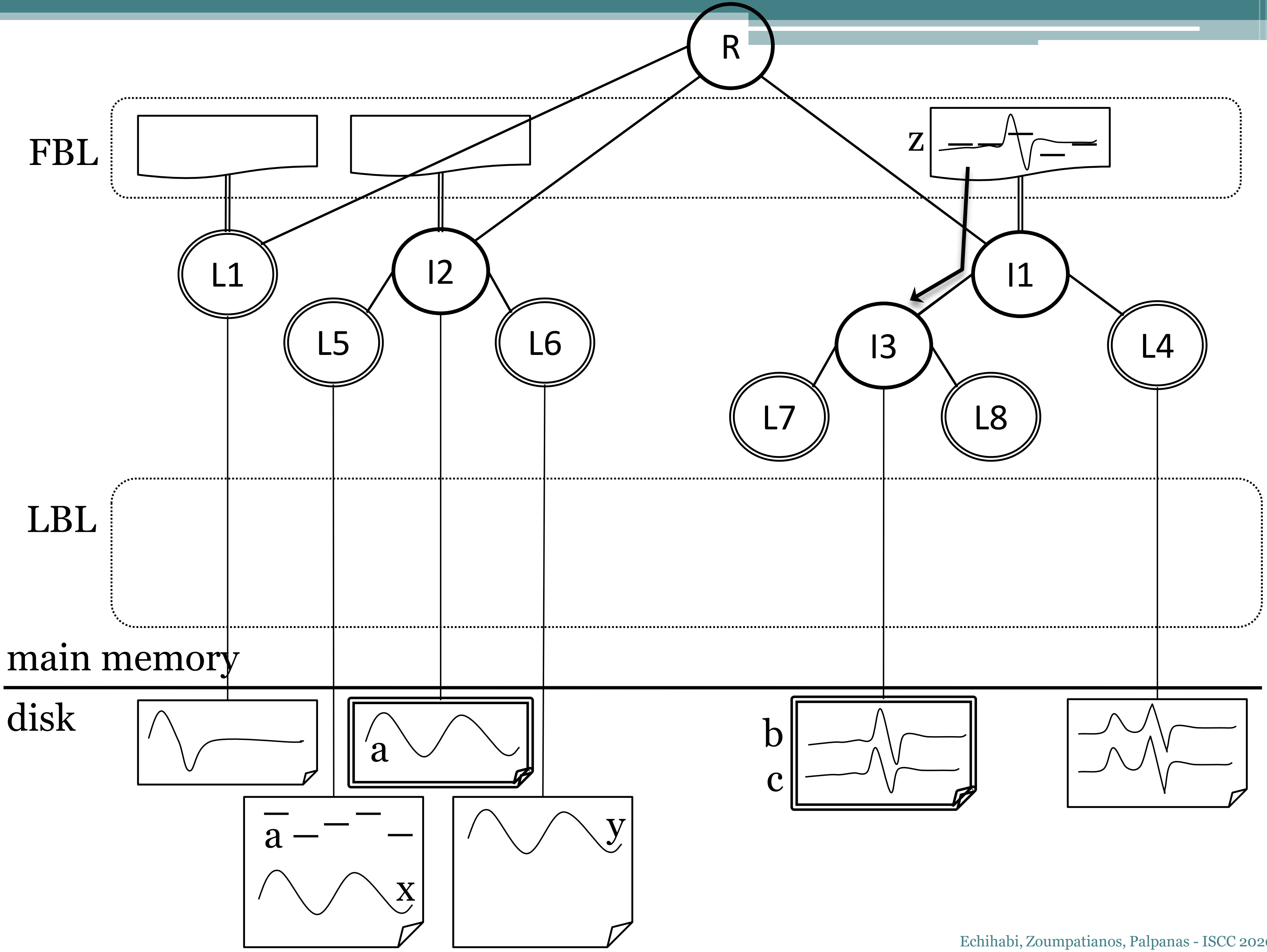

R
FBL
b
c
L1
I2
I1
L5
L6
I3
L4
LBL
b
c
main memory
disk
a
a
x
y

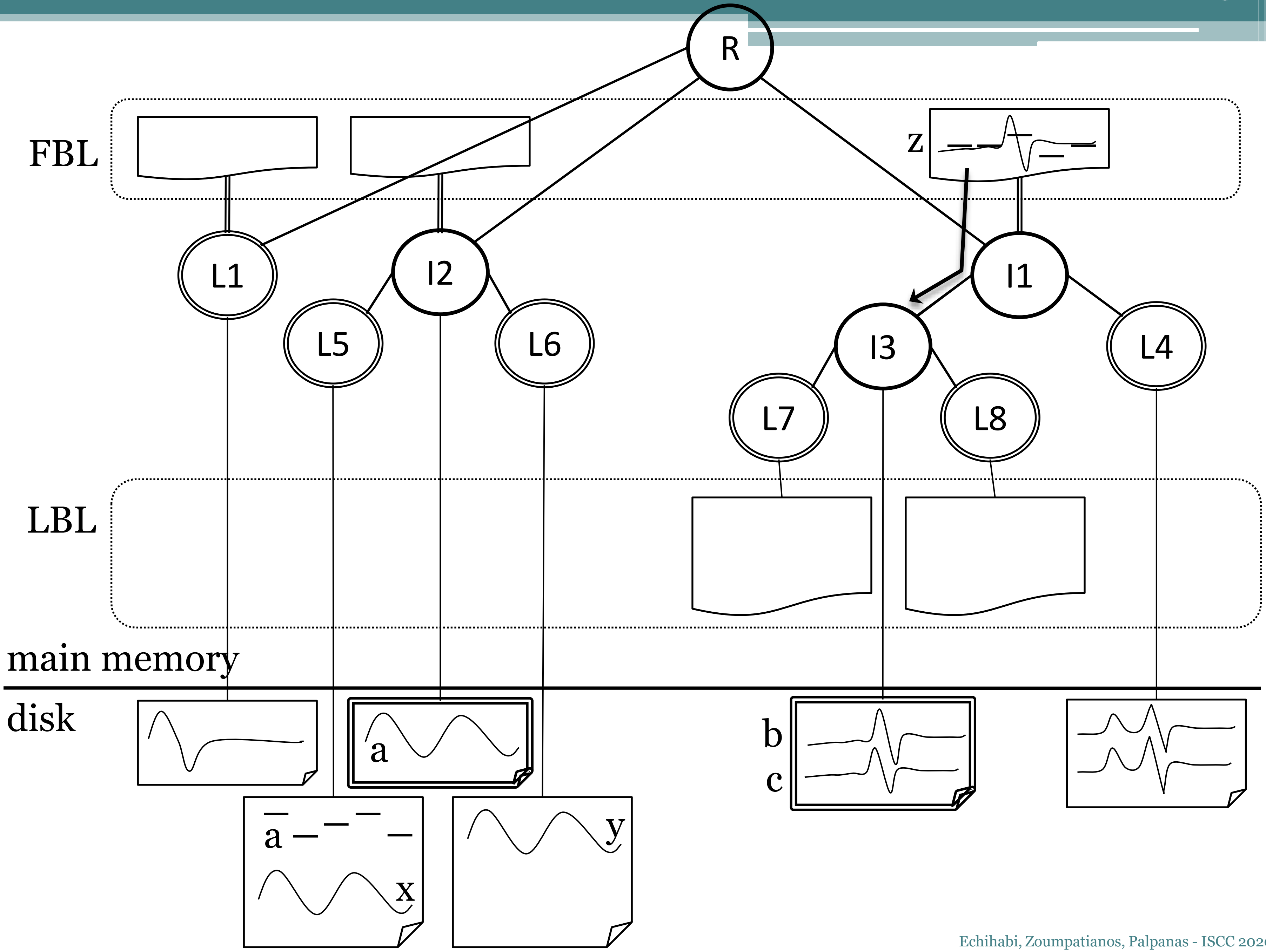

R
FBL
L1
I2
L5
L6
I1
I3
L4
LBL
main memory
disk
a
a
x
y
b
c

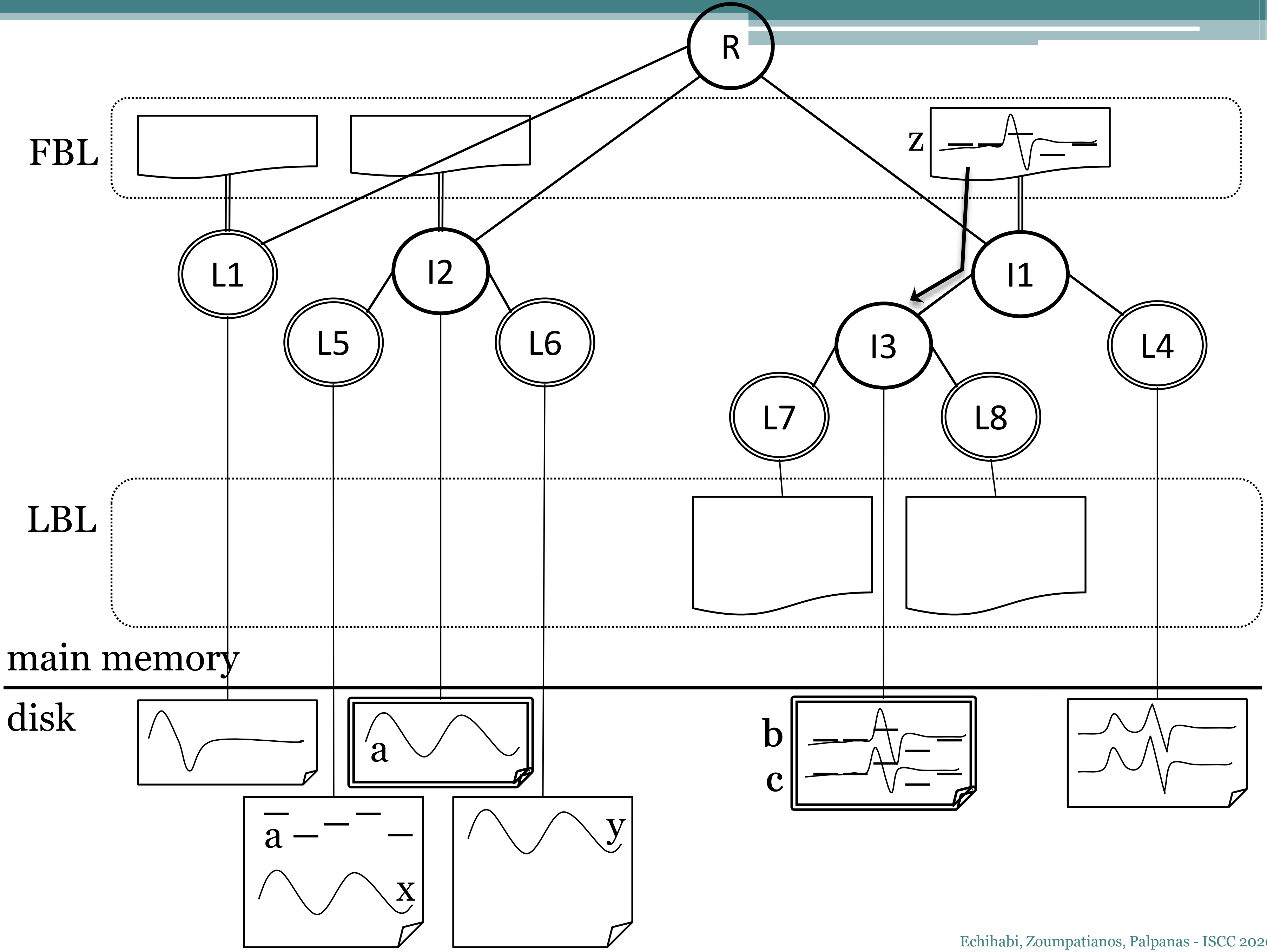

R
FBL
z
L1
I2
L5
L6
I3
I1
L4
LBL
main memory
disk
a
b
c
a
x
y

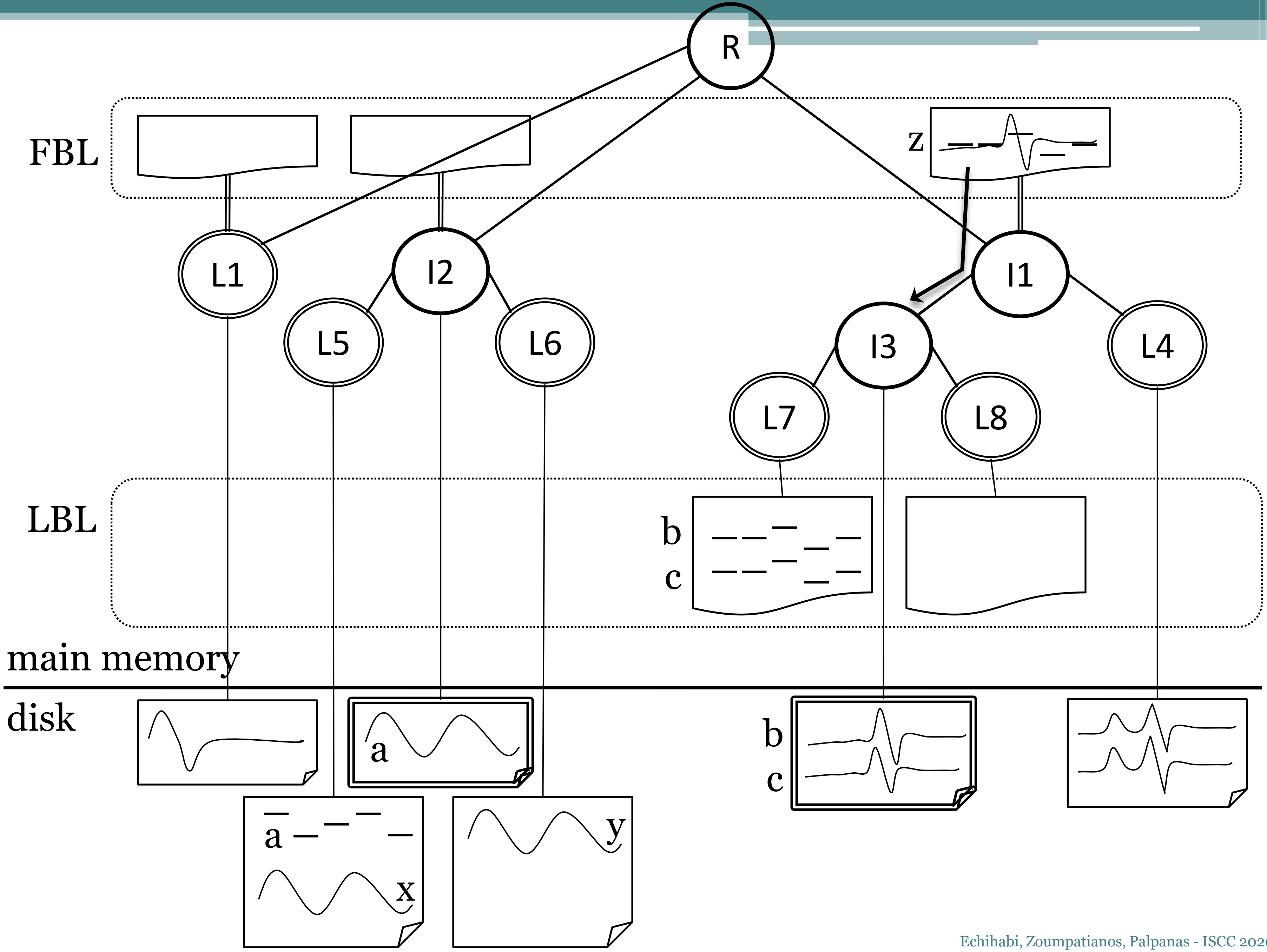

R
FBL
z
L1
I2
I1
L5
L6
I3
L4
LBL
main memory
disk
a
b
c
a
x
y

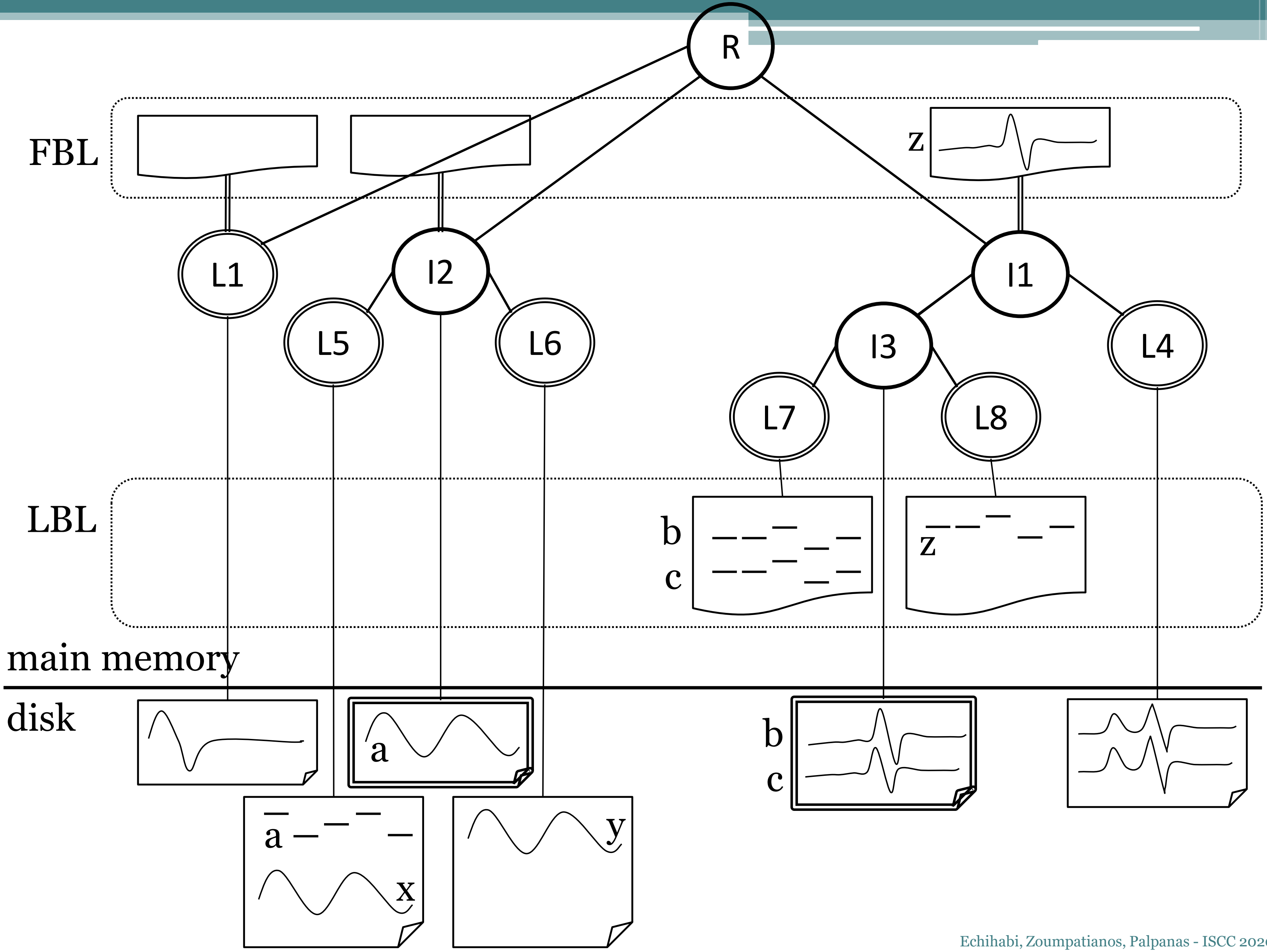

R
FBL
z
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
b
c
a
x
y

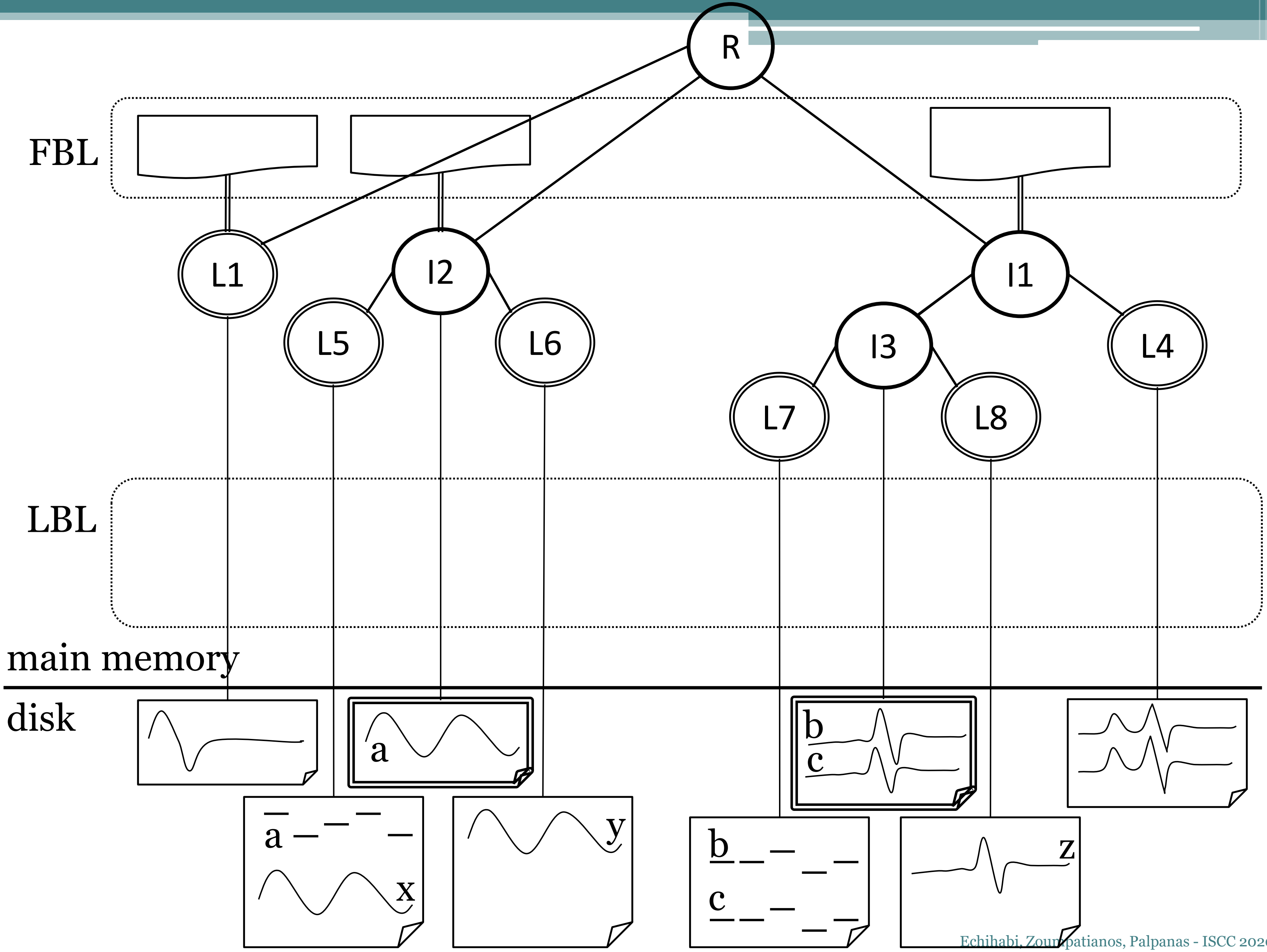

R
FBL
z
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
a
x
y
b
c

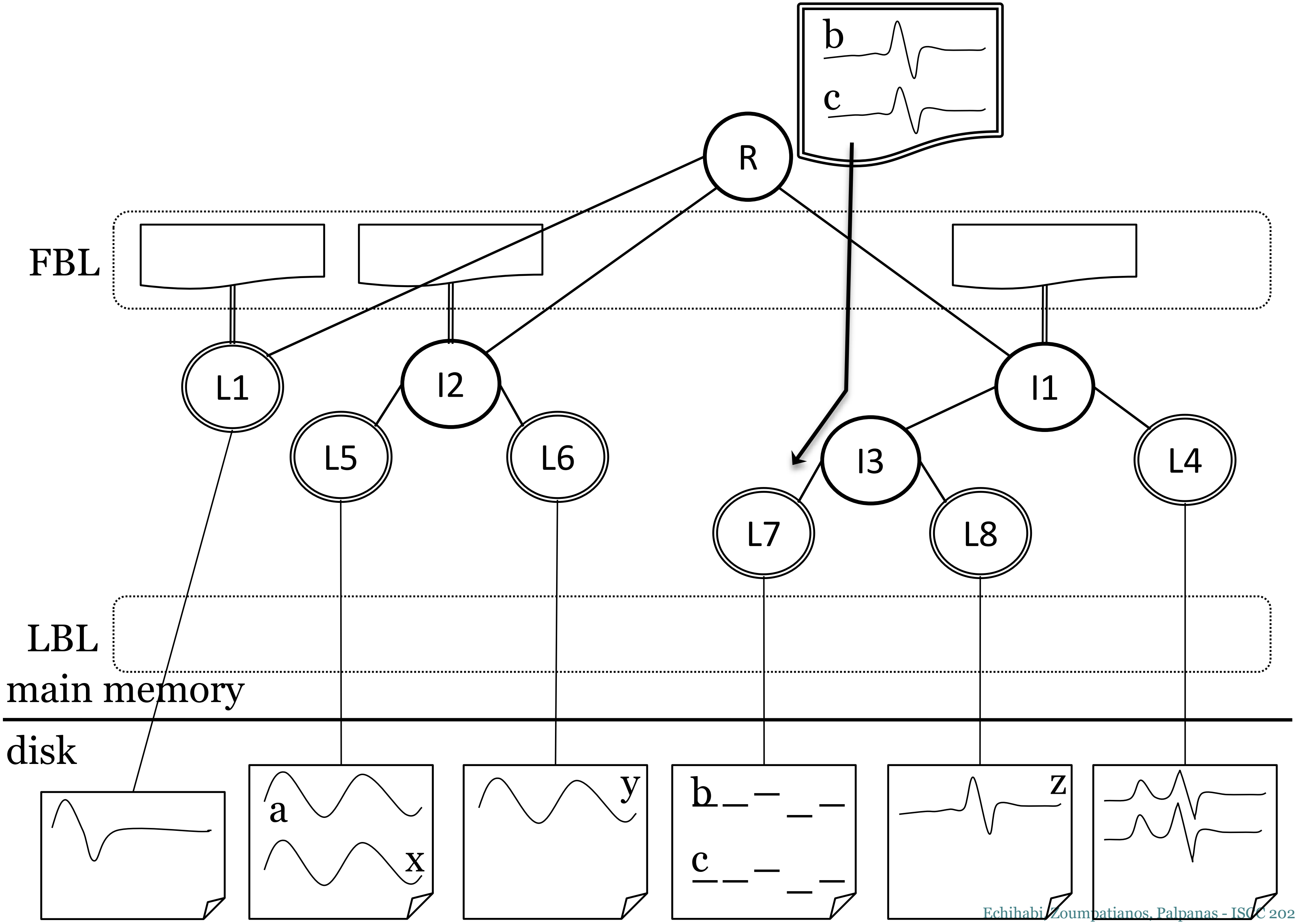

R
FBL
z
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
b
c
a
x
y

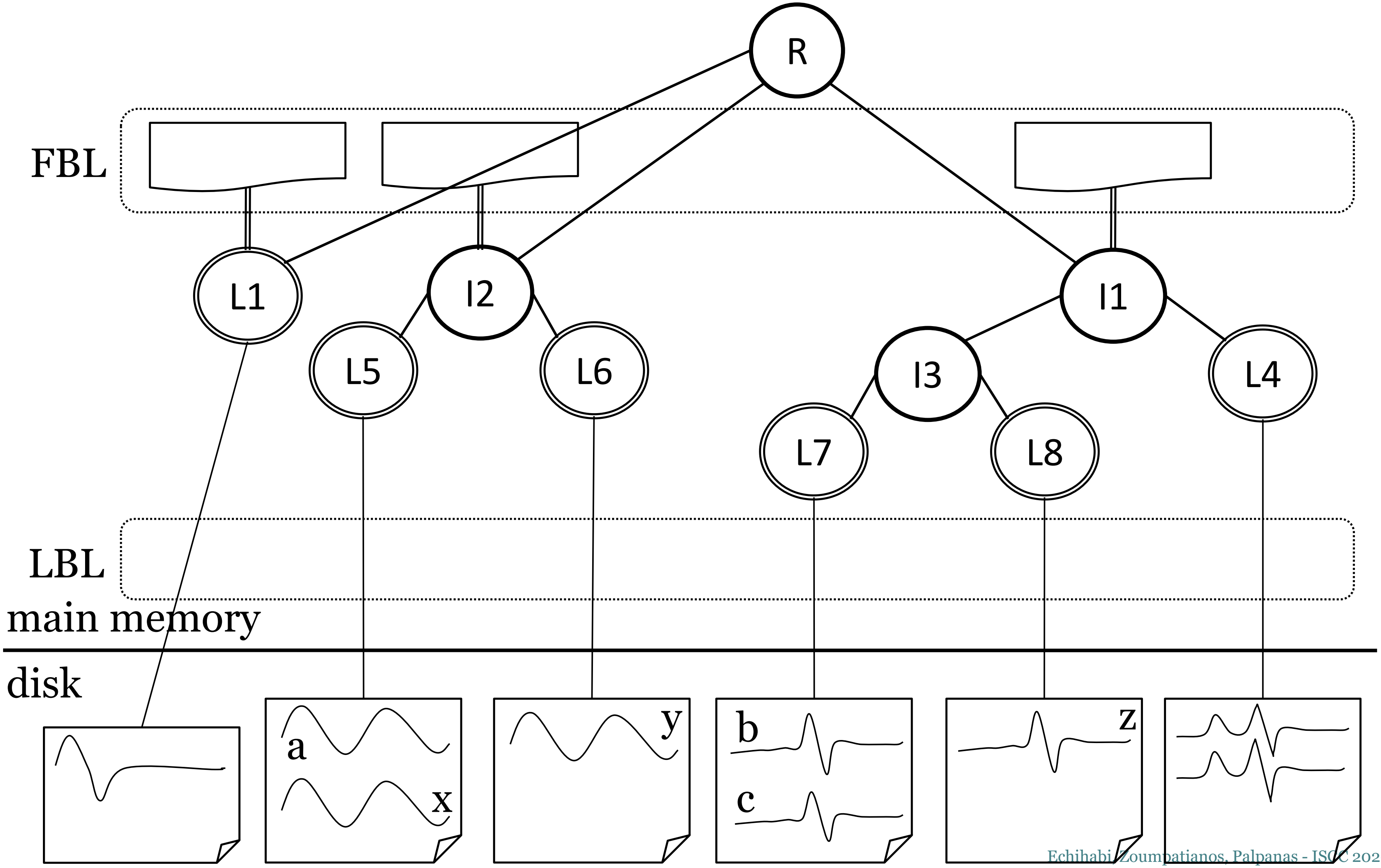

R
FBL
z
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
b
c
a
x
y


R
FBL
z
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
b
c
main memory
disk
a
b
c
a
x
y

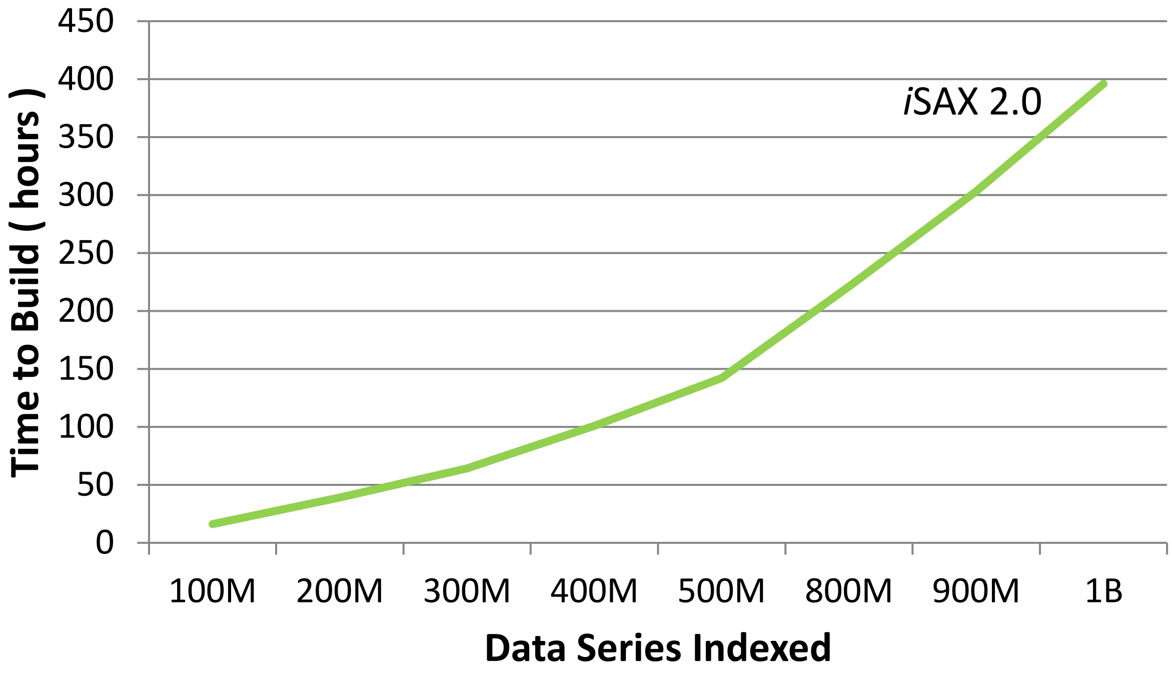

R
FBL
z
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
b
c
z
main memory
disk
a
b
c
a
x
y


R
FBL
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
a
x
y
b
c
b
c
z

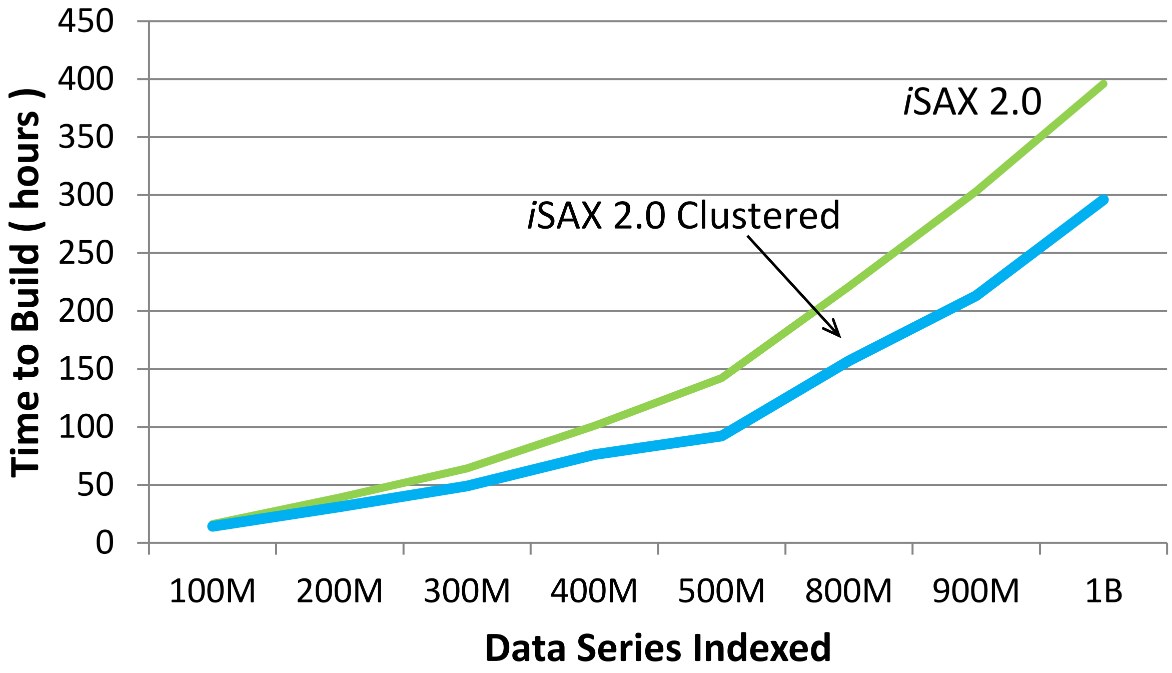

b
c
R
FBL
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
x
y
b
c
z


R
FBL
L1
I2
I1
L5
L6
I3
L4
L7
L8
LBL
main memory
disk
a
x
y
b
c
z

Experimental Evaluation
Bulk Loading

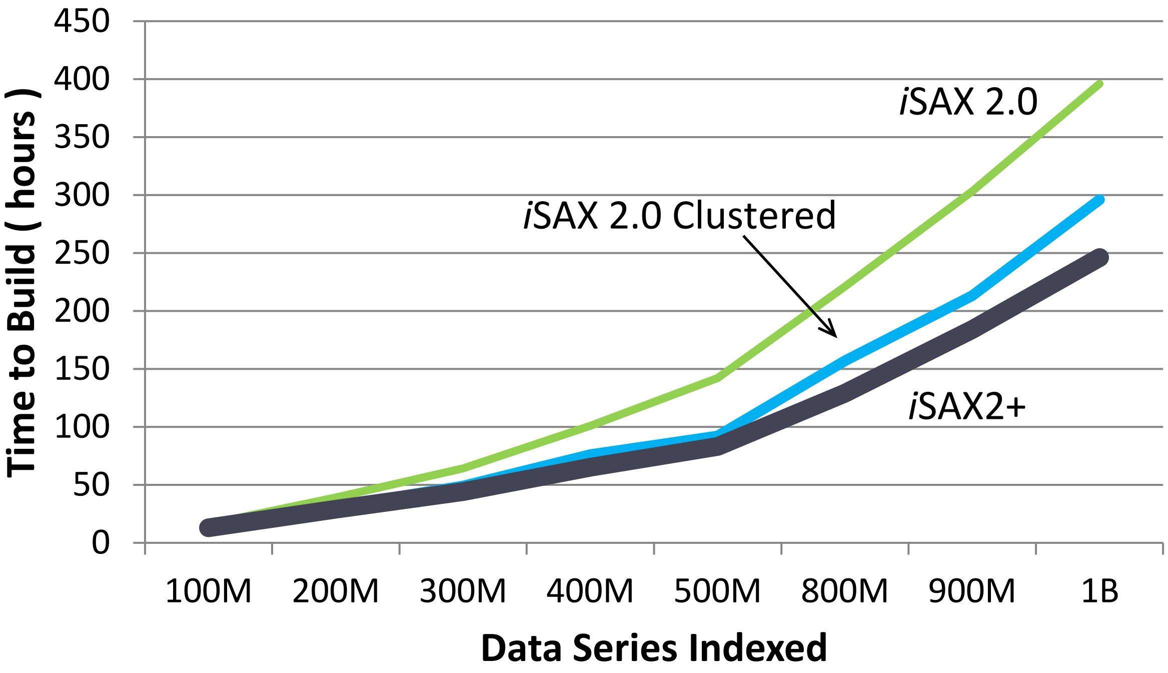


Experimental Evaluation
Bulk Loading



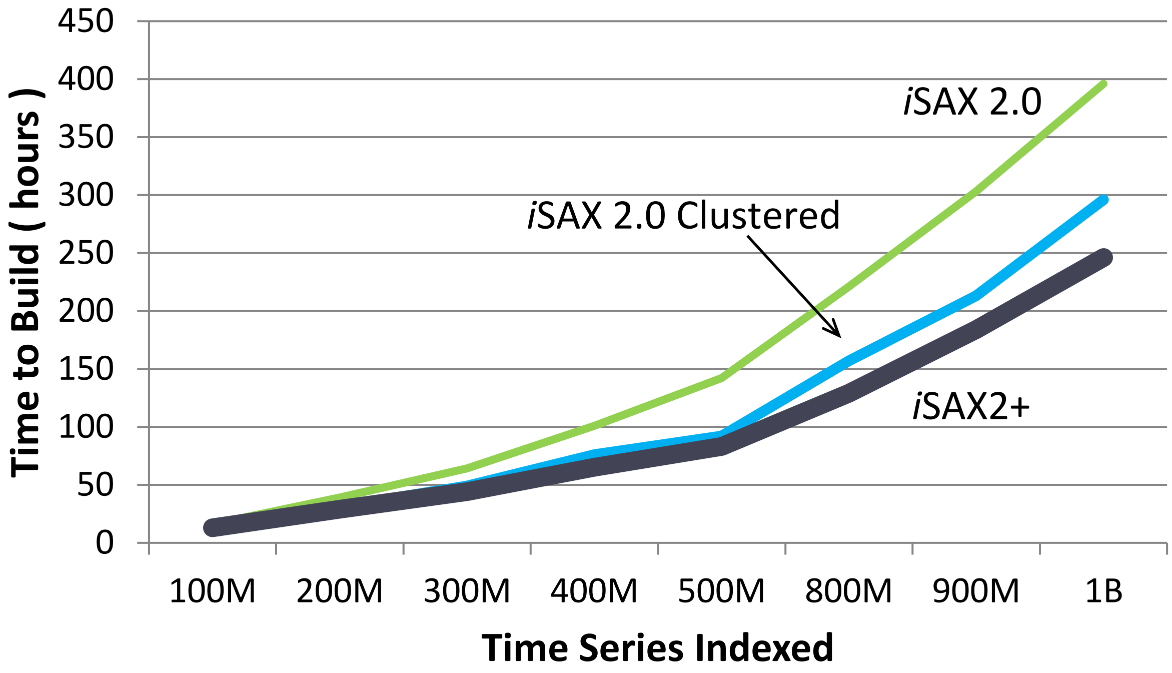
- *i*SAX 2.0 Clustered needs 30% less time than *i*SAX 2.0

Experimental Evaluation
Bulk Loading



- *i*SAX 2.0 Clustered needs 30% less time than *i*SAX 2.0
- *i*SAX2+ needs 40% less time than *i*SAX 2.0

Experimental Evaluation
Bulk Loading



- 1 Billion data series indexed in 10 days: 82% less time than *i*SAX
- indexing time per data series: 0.8 milliseconds

Adaptive Data Series Index: ADS+

- novel paradigm for building a data series index
  - do not build entire index and then answer queries
  - start answering queries by building the part of the index needed by those queries

- still guarantee correct answers

Adaptive Data Series Index: ADS+

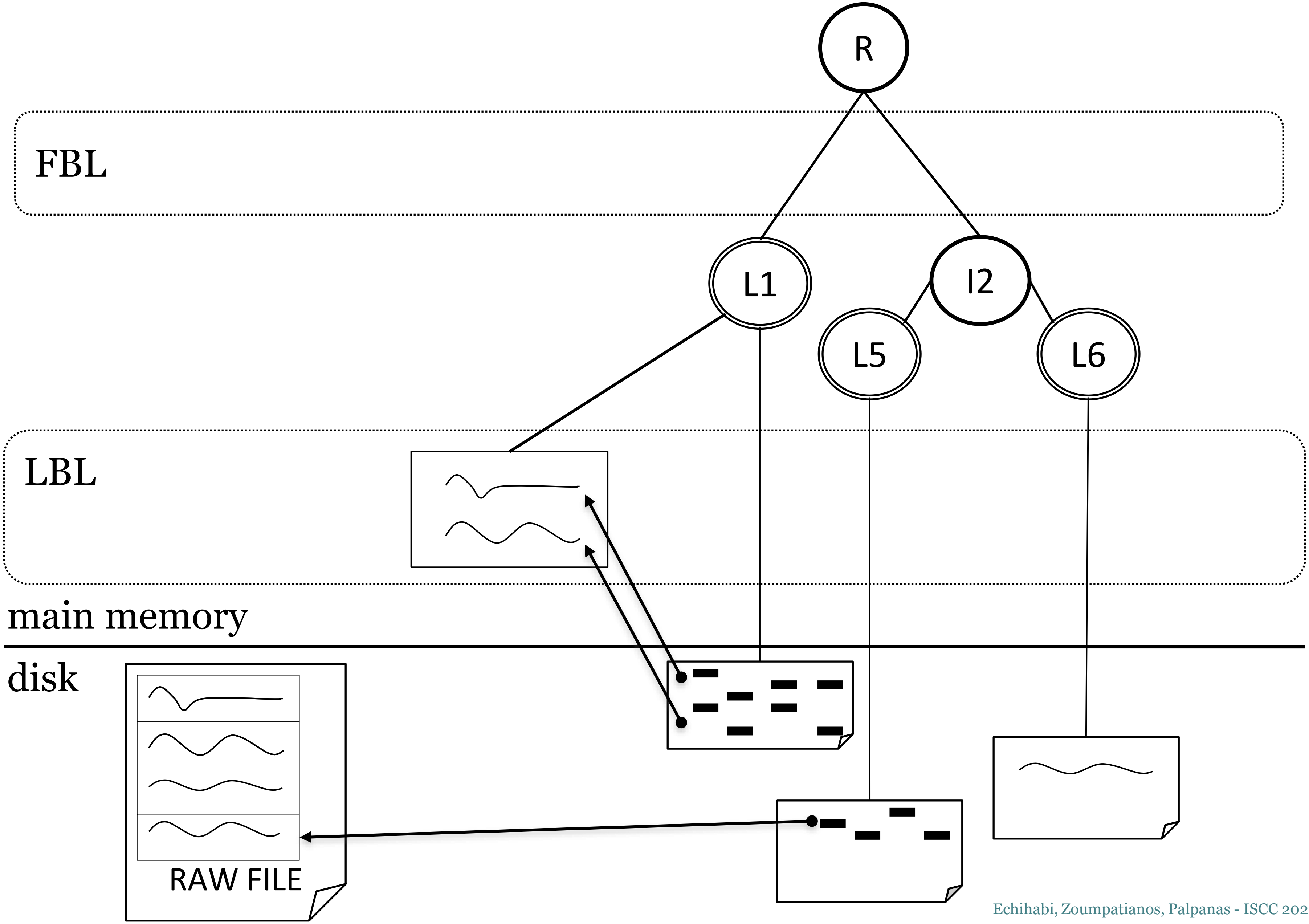
Publications

SIGMOD'14

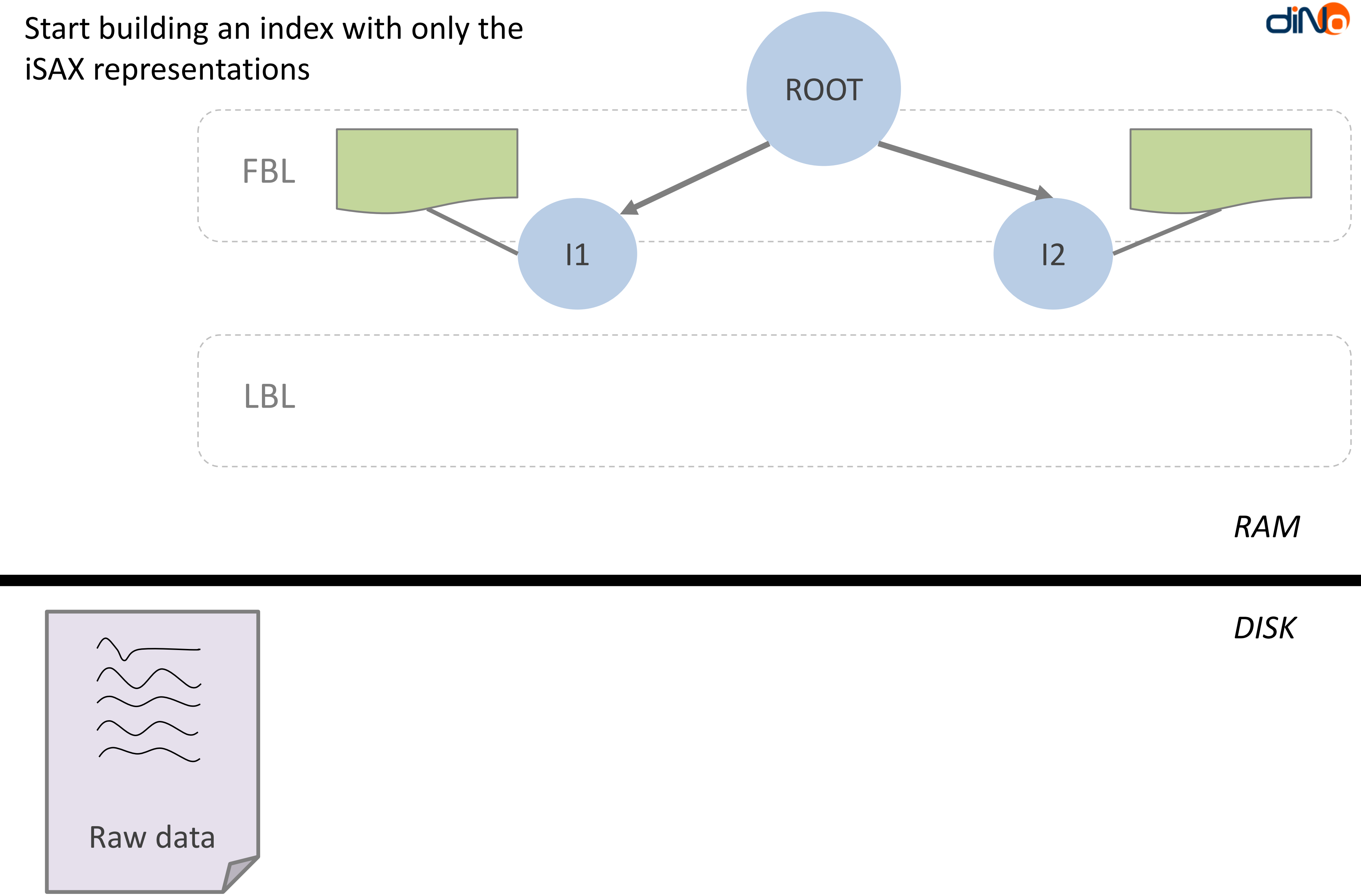
PVLDB'15

VLDBJ'16

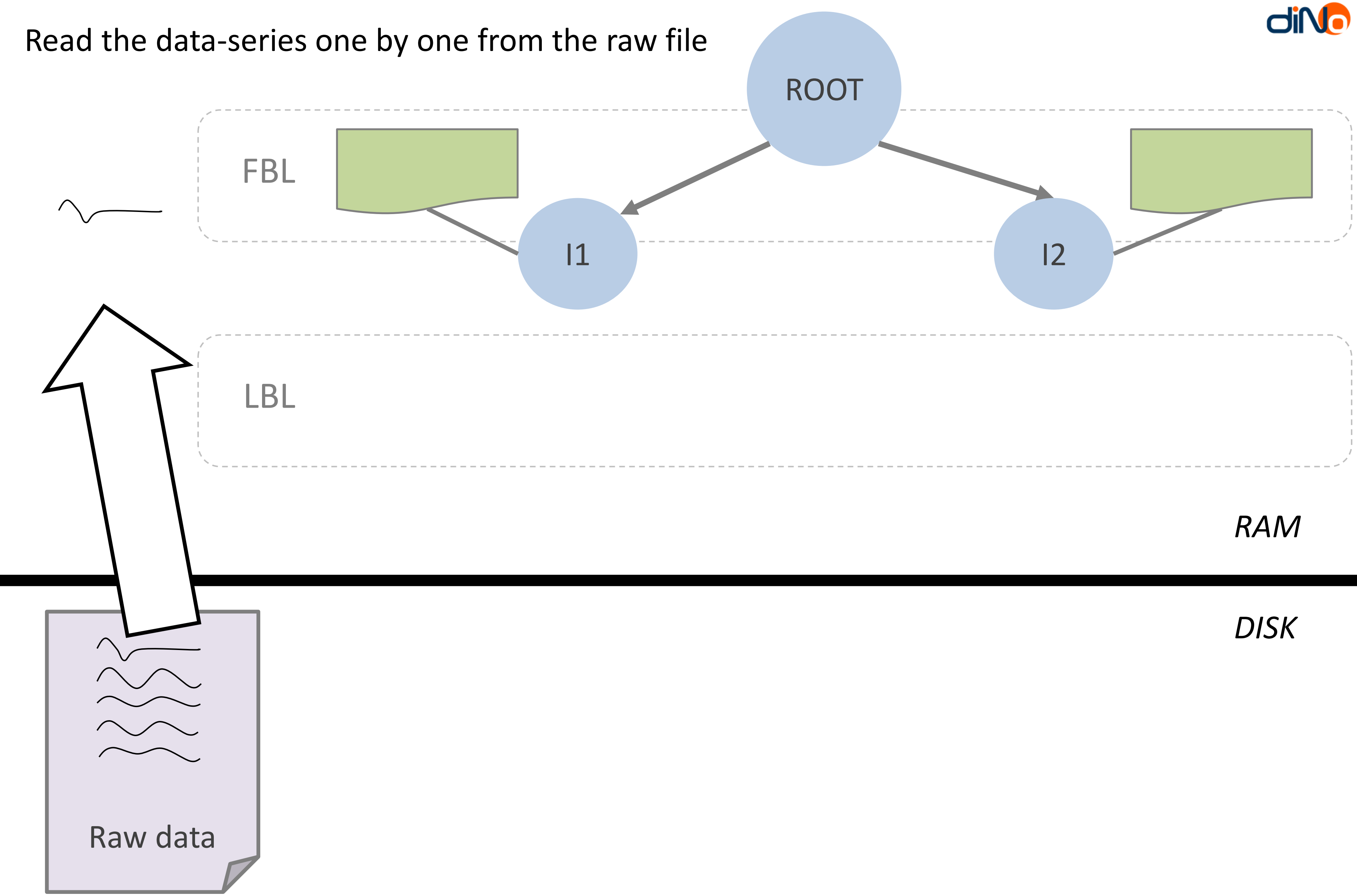
- intuition for proposed solution
    - build the iSAX index using the iSAX representations
        - just like iSAX2+
    - but start with a large leaf size
        - minimize initial cost
    - postpone leaf materialization to query time
        - only materialize (at query time) leaves needed by queries
    - parts that are queried more are refined more
        - use smaller leaf sizes (reduced leaf materialization and query answering costs)


R
FBL
L1
I2
L5
L6
LBL
main memory
disk
RAW FILE

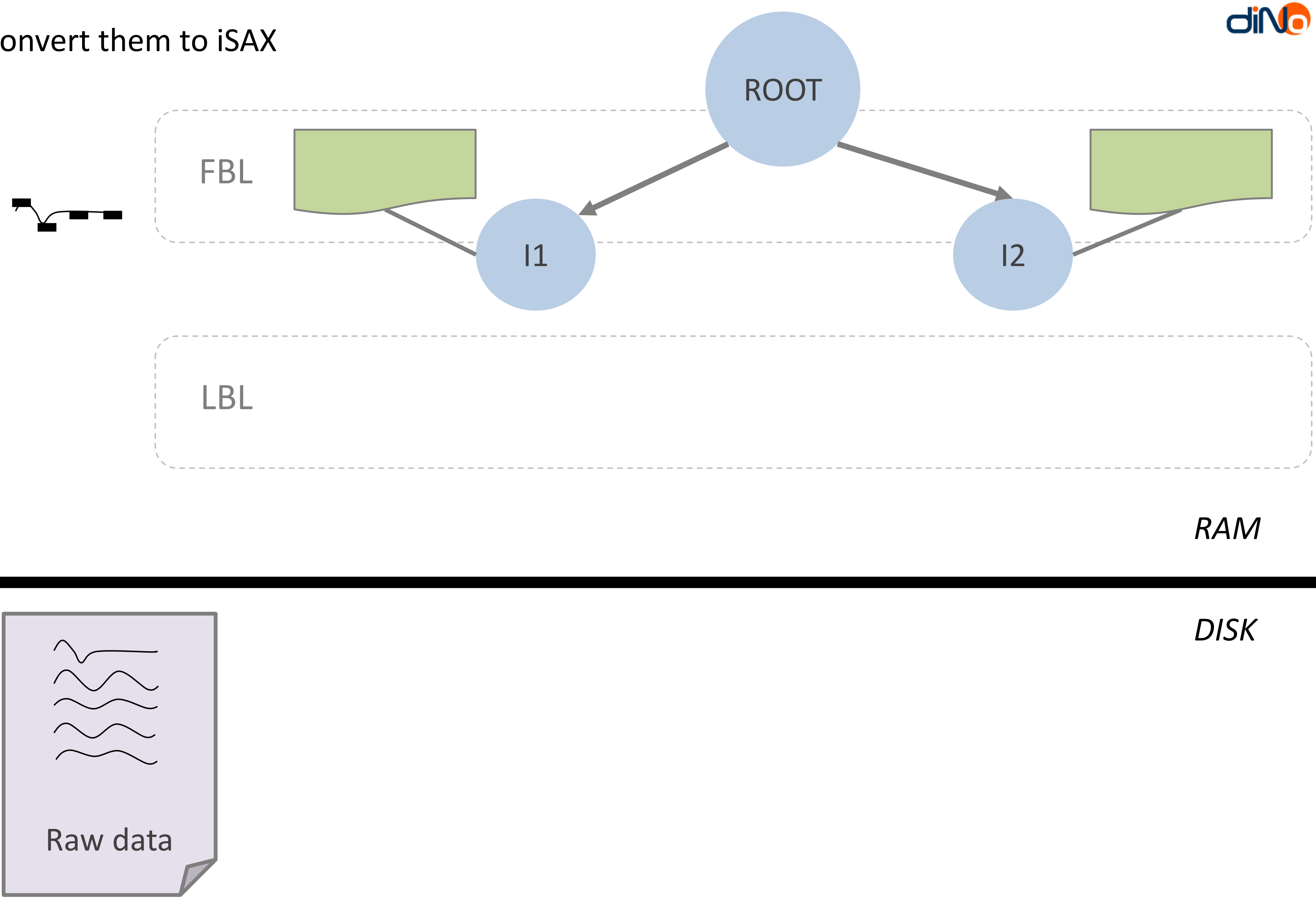
Start building an index with only the iSAX representations



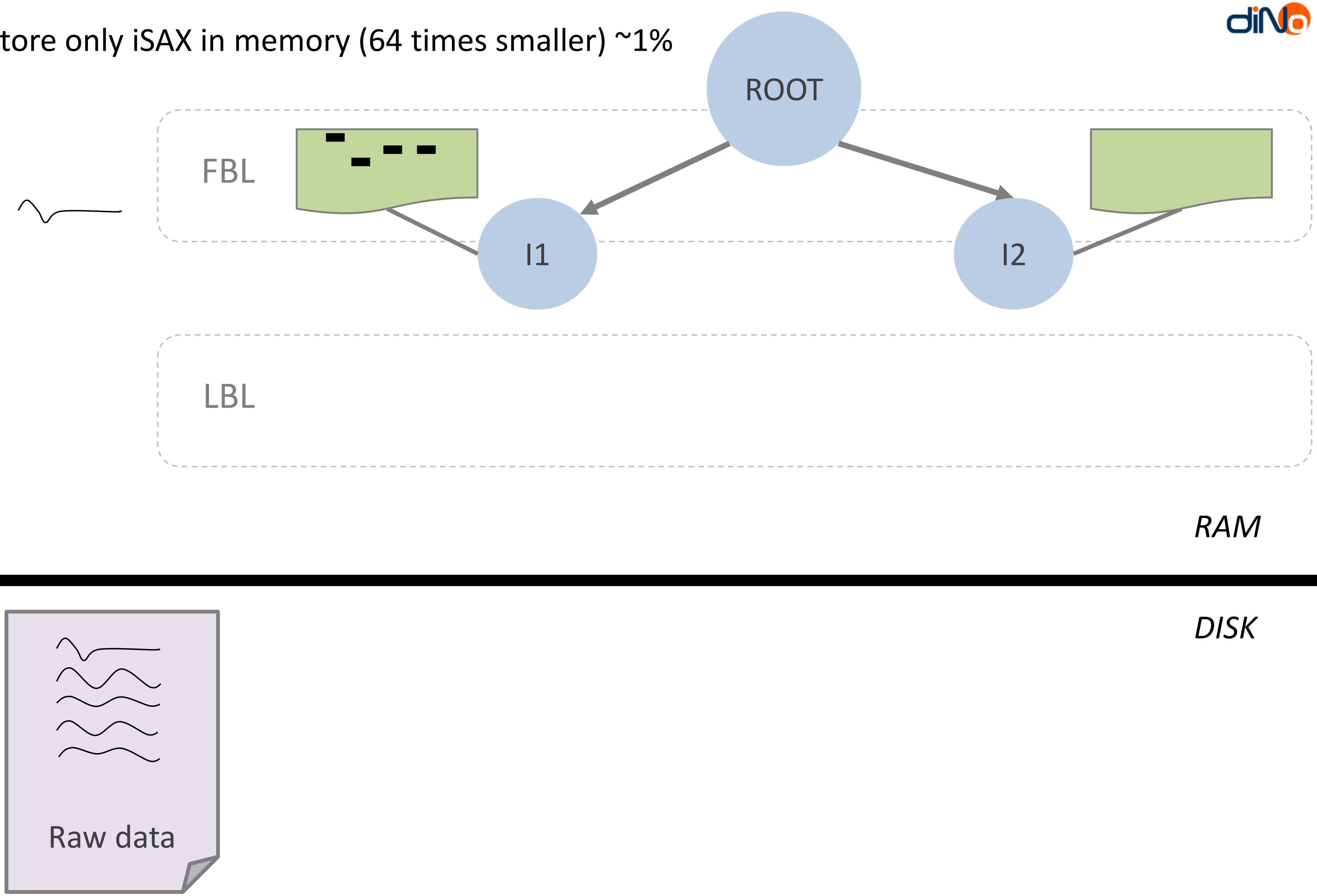
Read the data-series one by one from the raw file



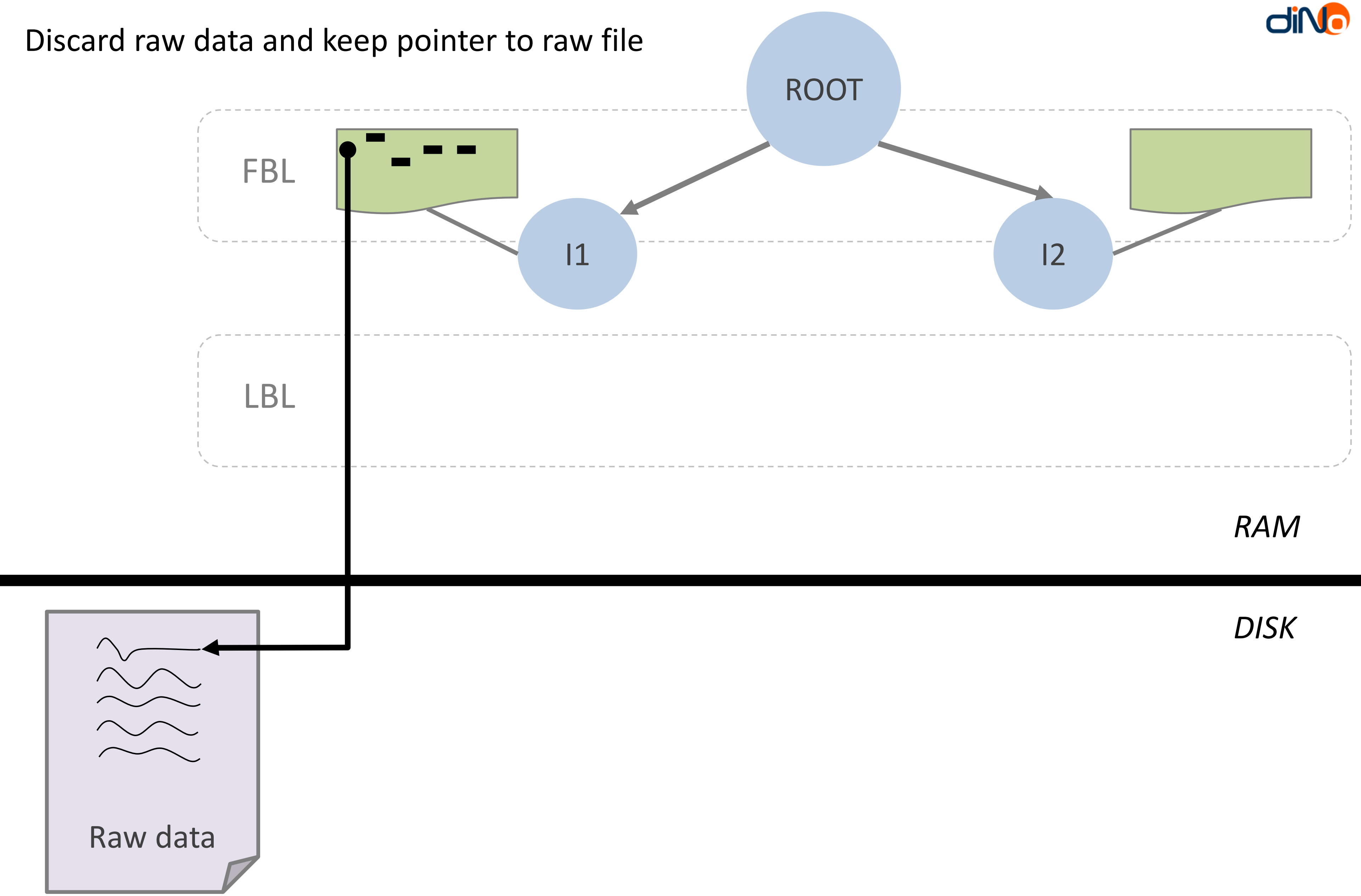
Convert them to iSAX



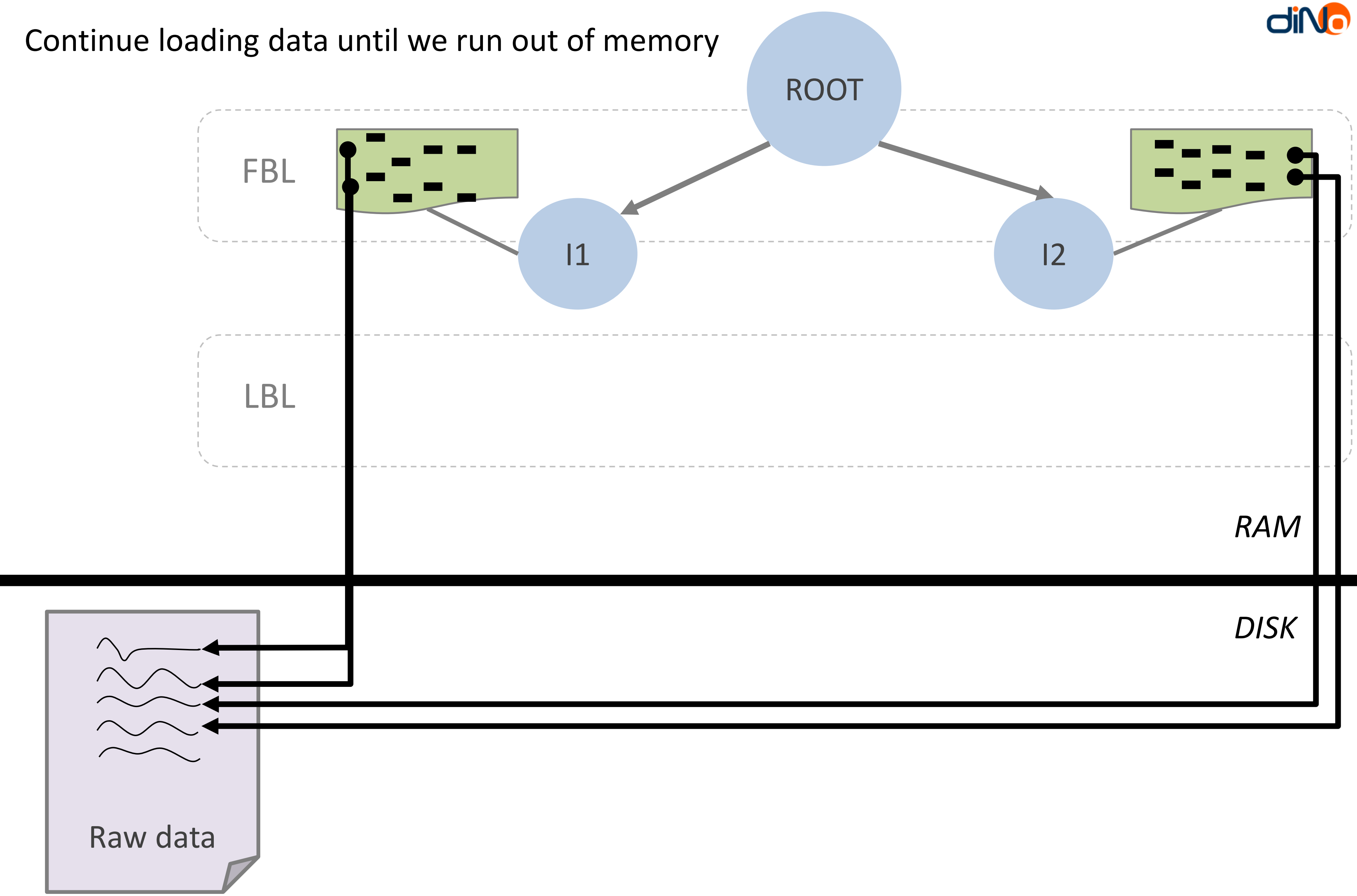
Store only iSAX in memory (64 times smaller) ~1%



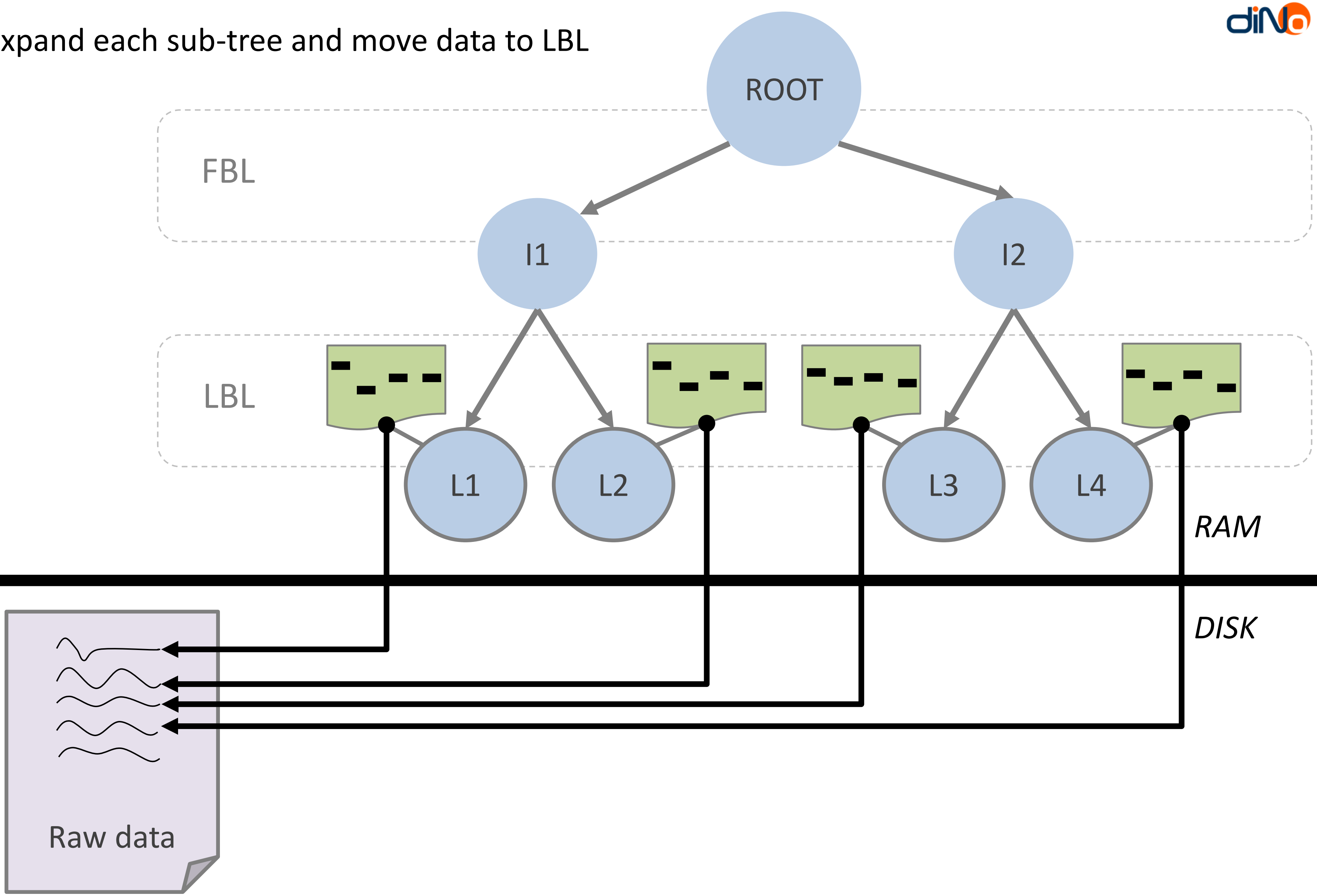
Discard raw data and keep pointer to raw file



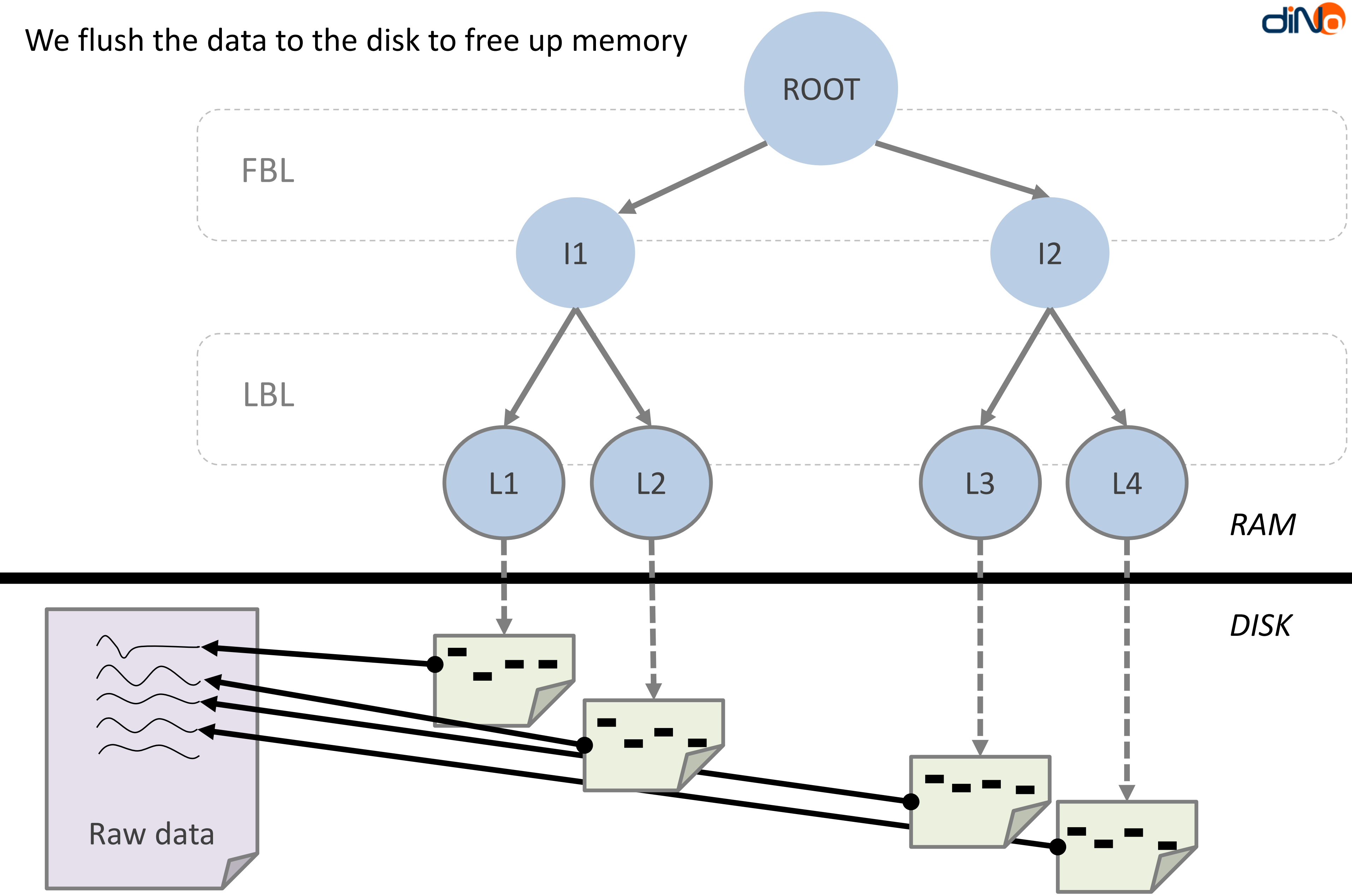
Continue loading data until we run out of memory



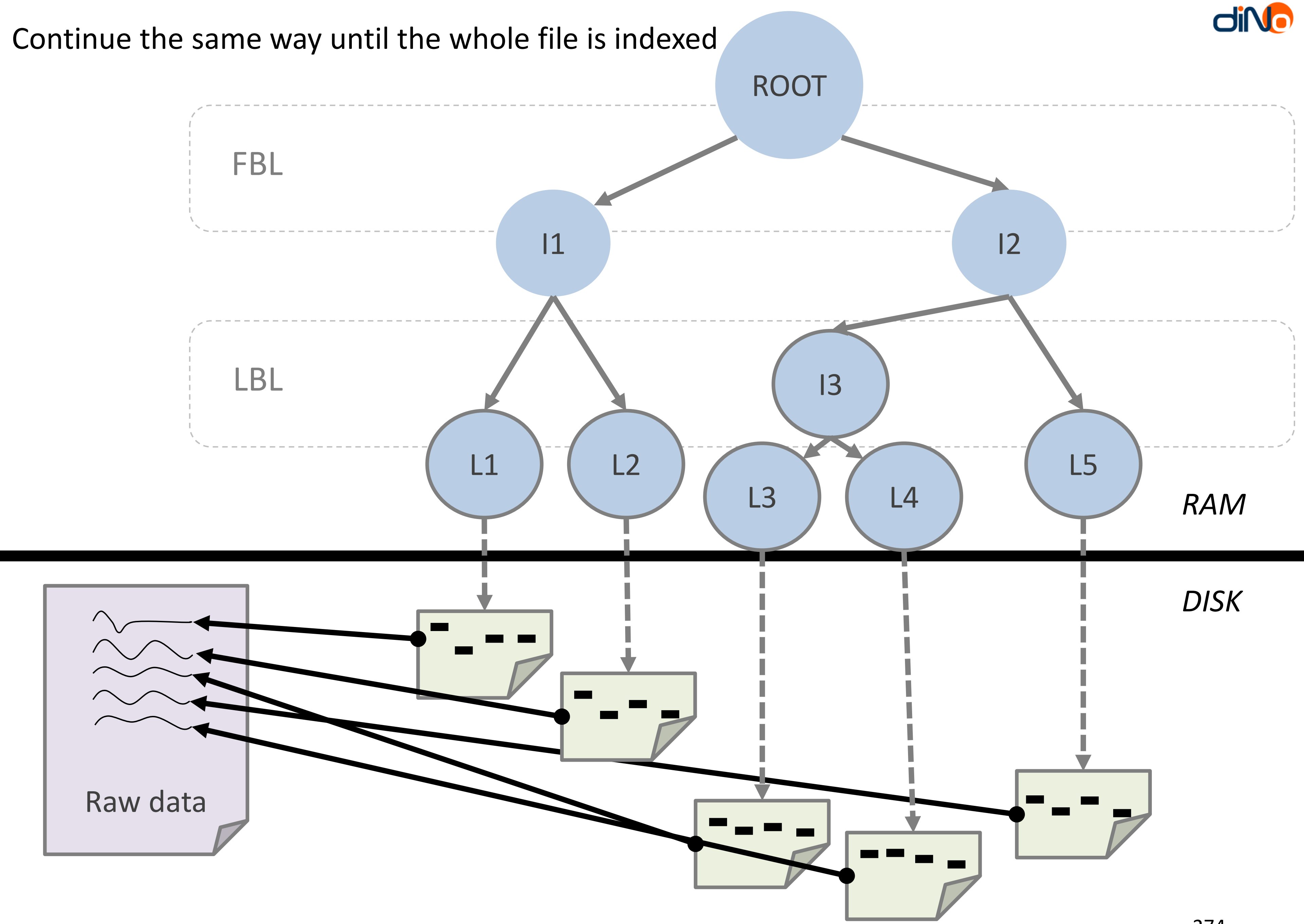
Expand each sub-tree and move data to LBL



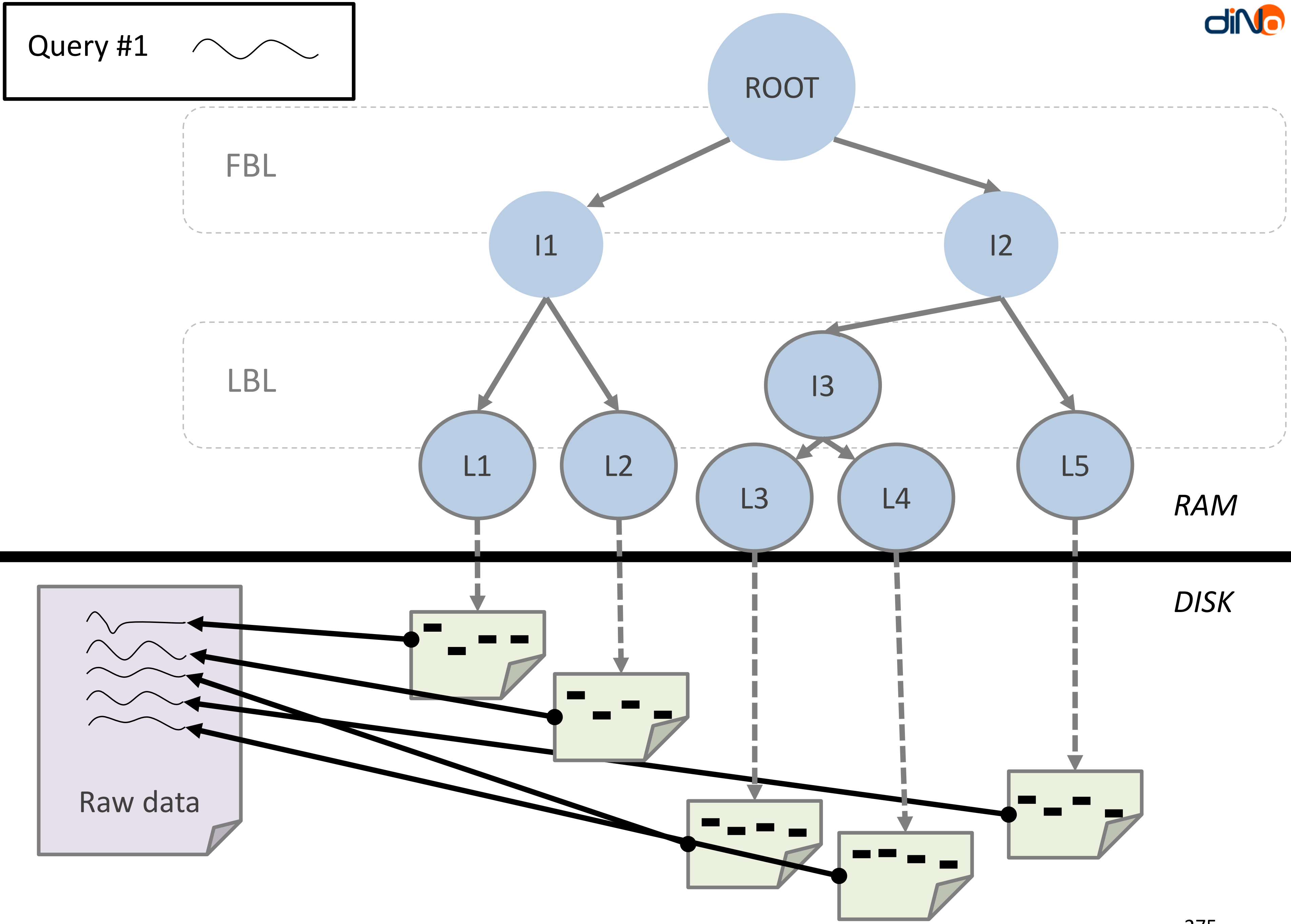
We flush the data to the disk to free up memory



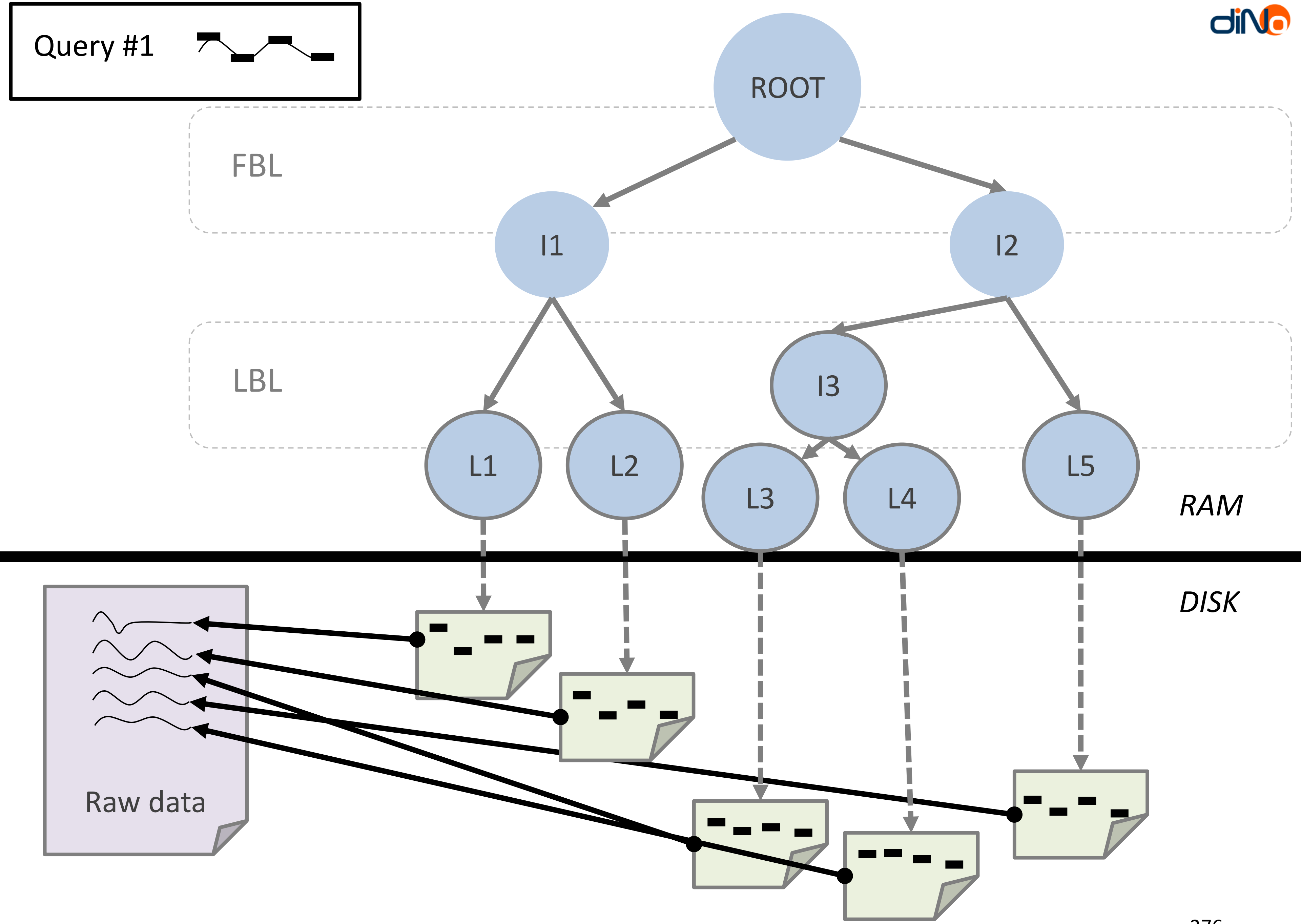
Continue the same way until the whole file is indexed



Query #1

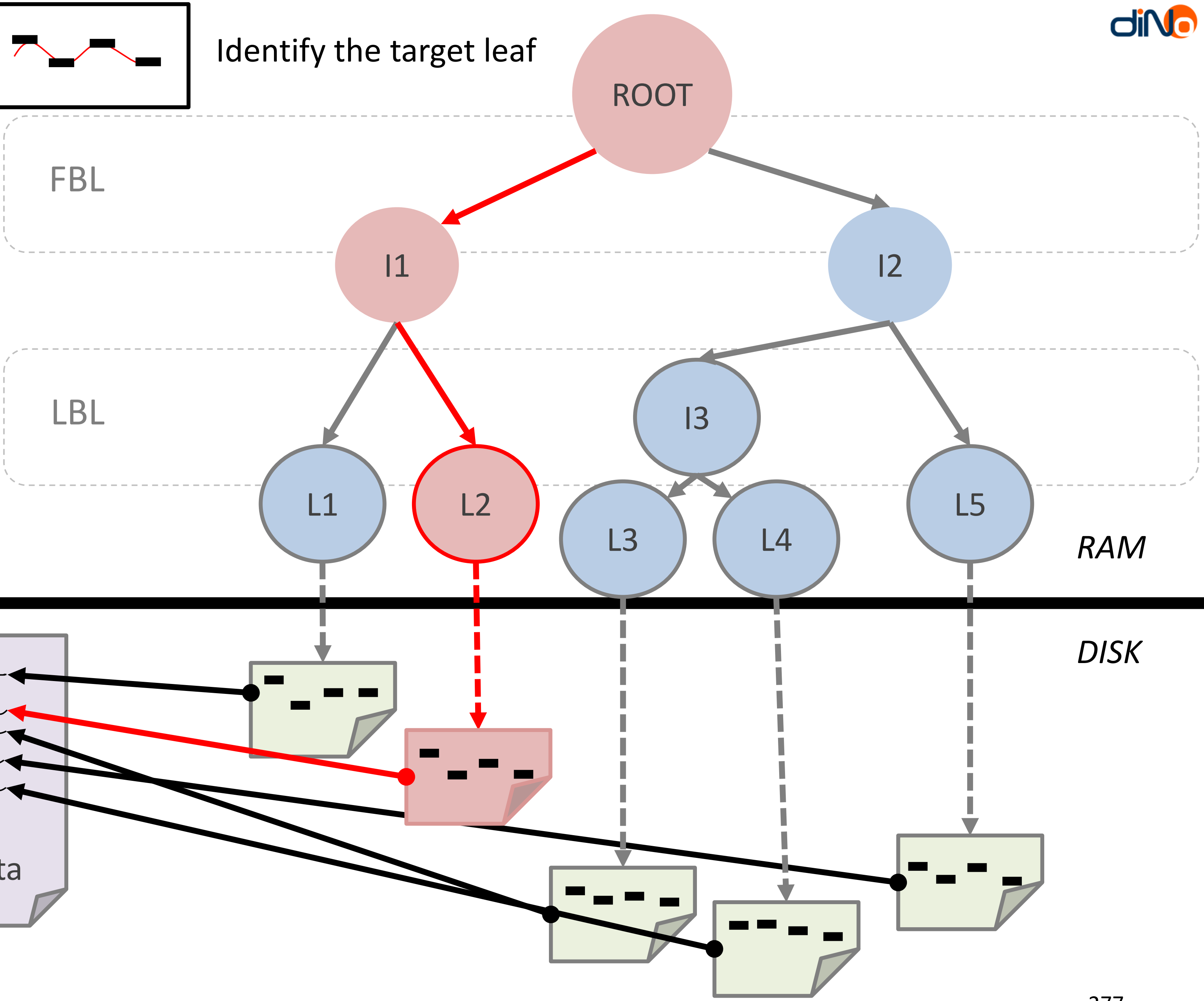


Query #1



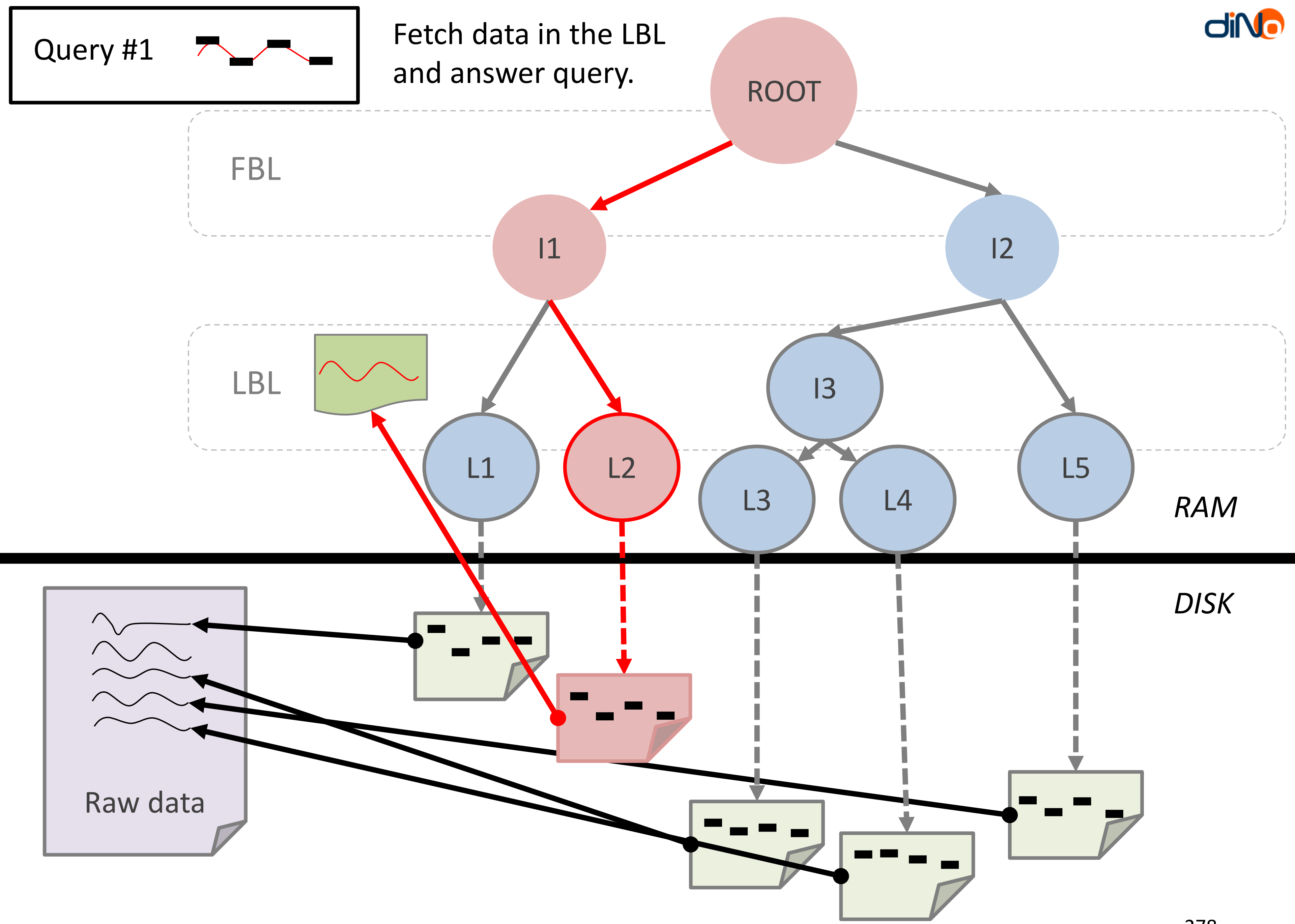
Query #1

Identify the target leaf

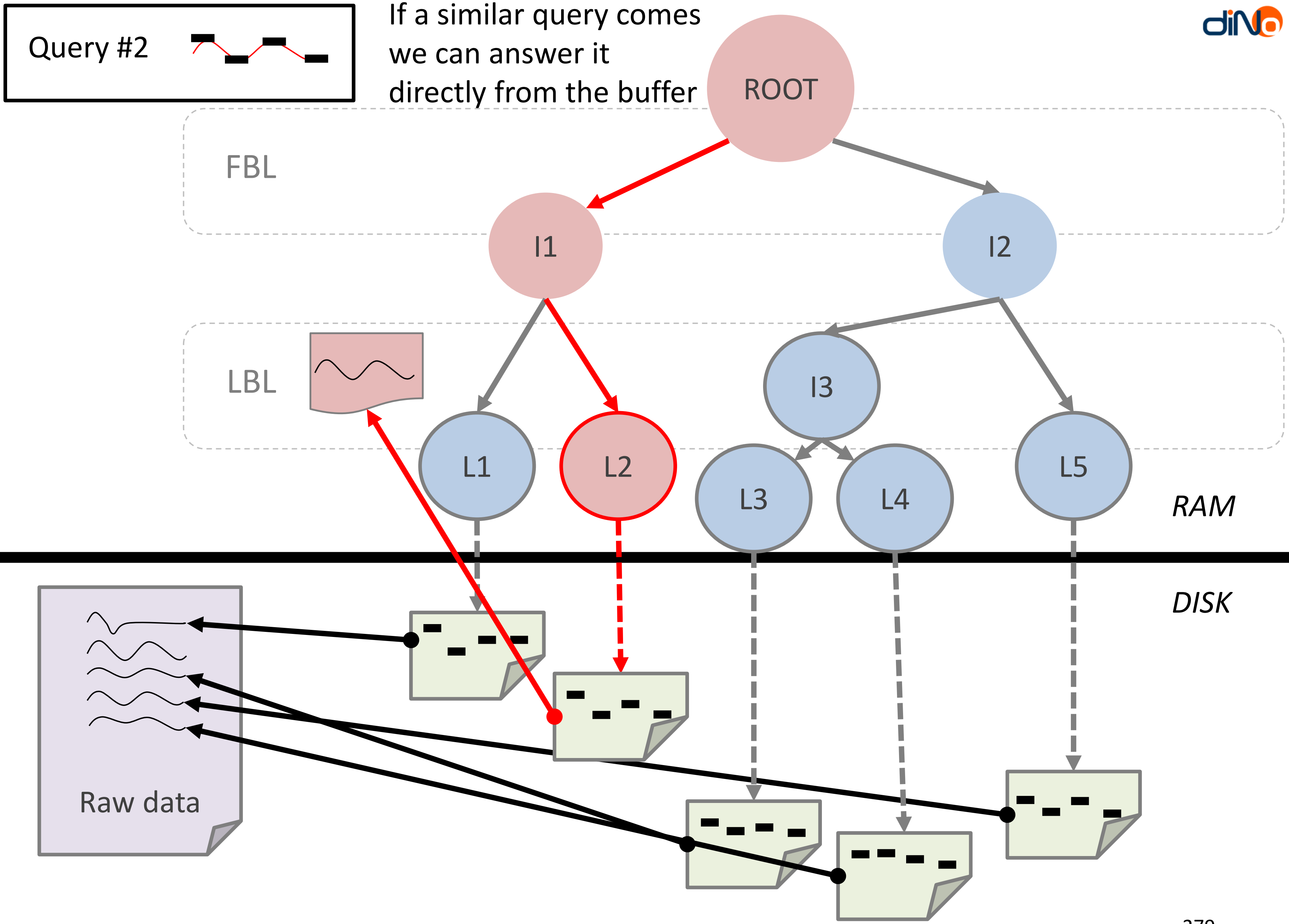


Query #1

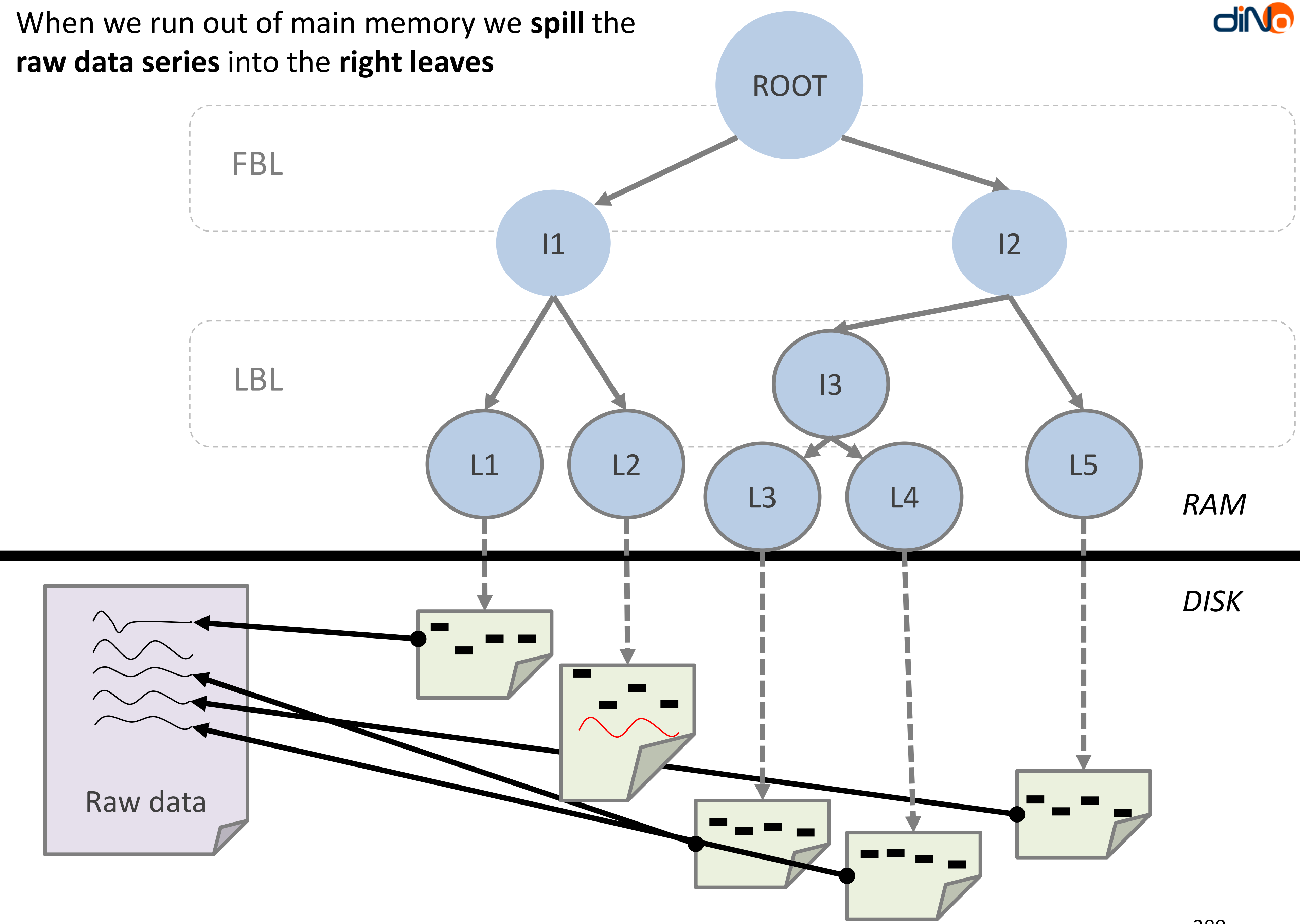
Fetch data in the LBL and answer query.



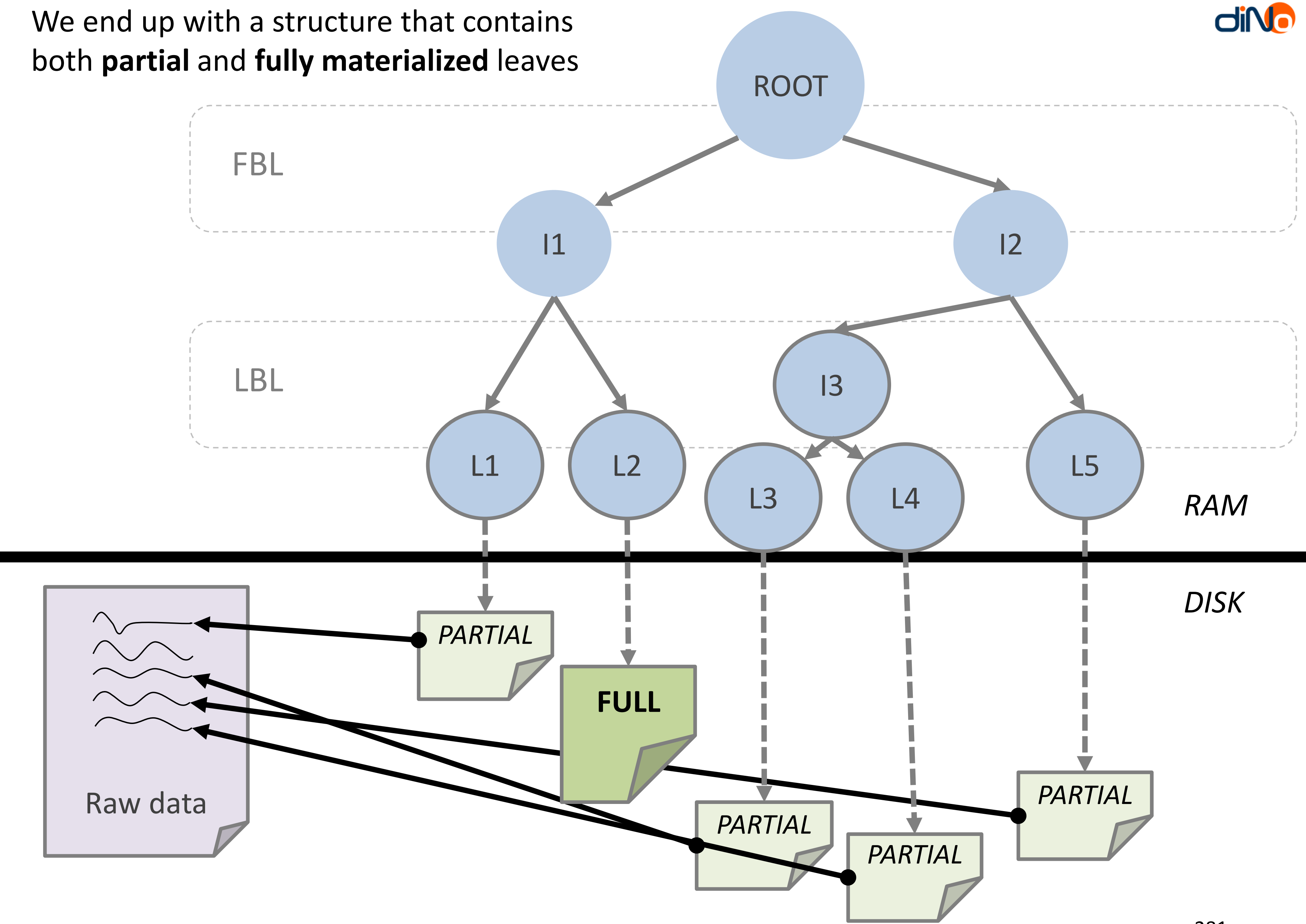
If a similar query comes we can answer it directly from the buffer

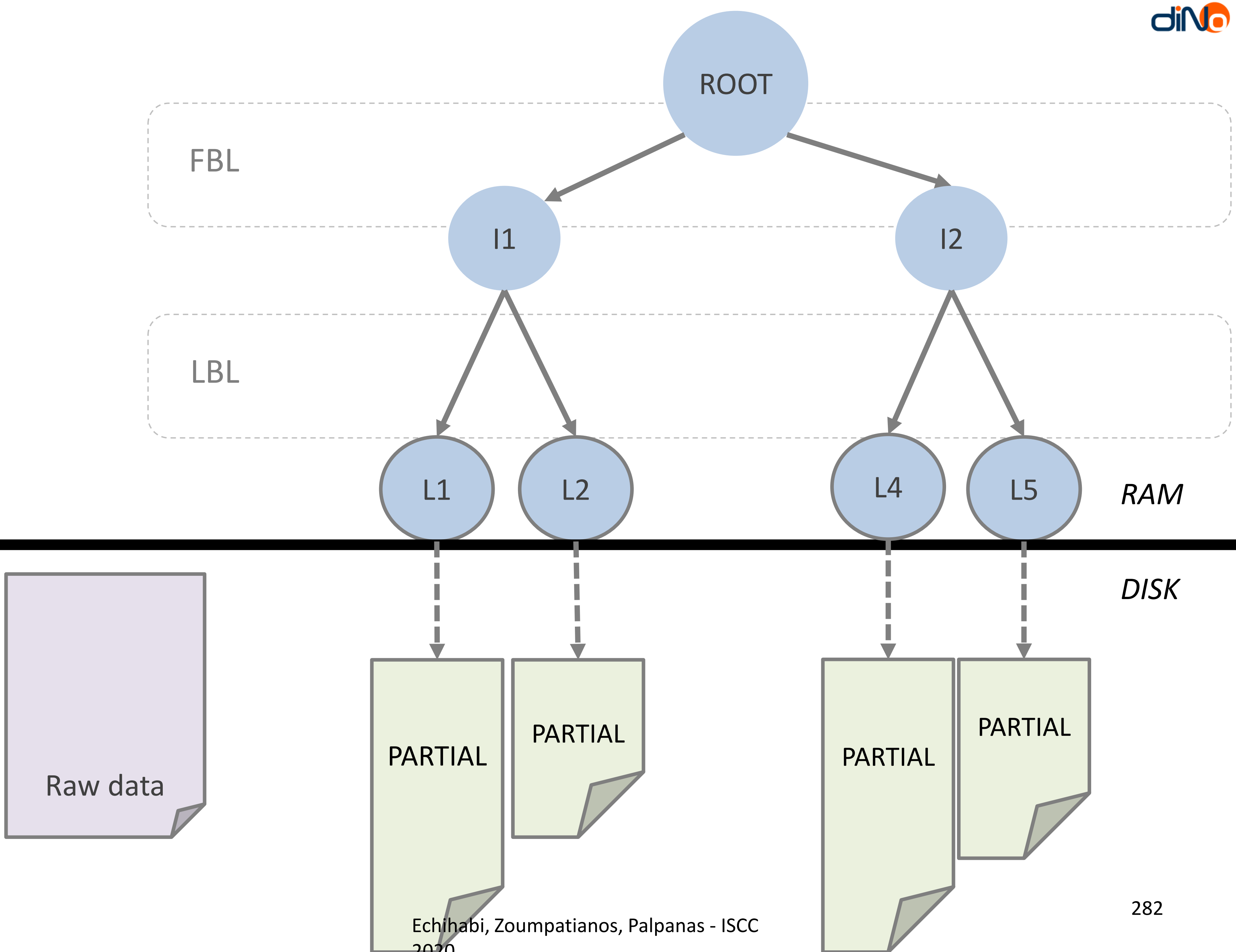


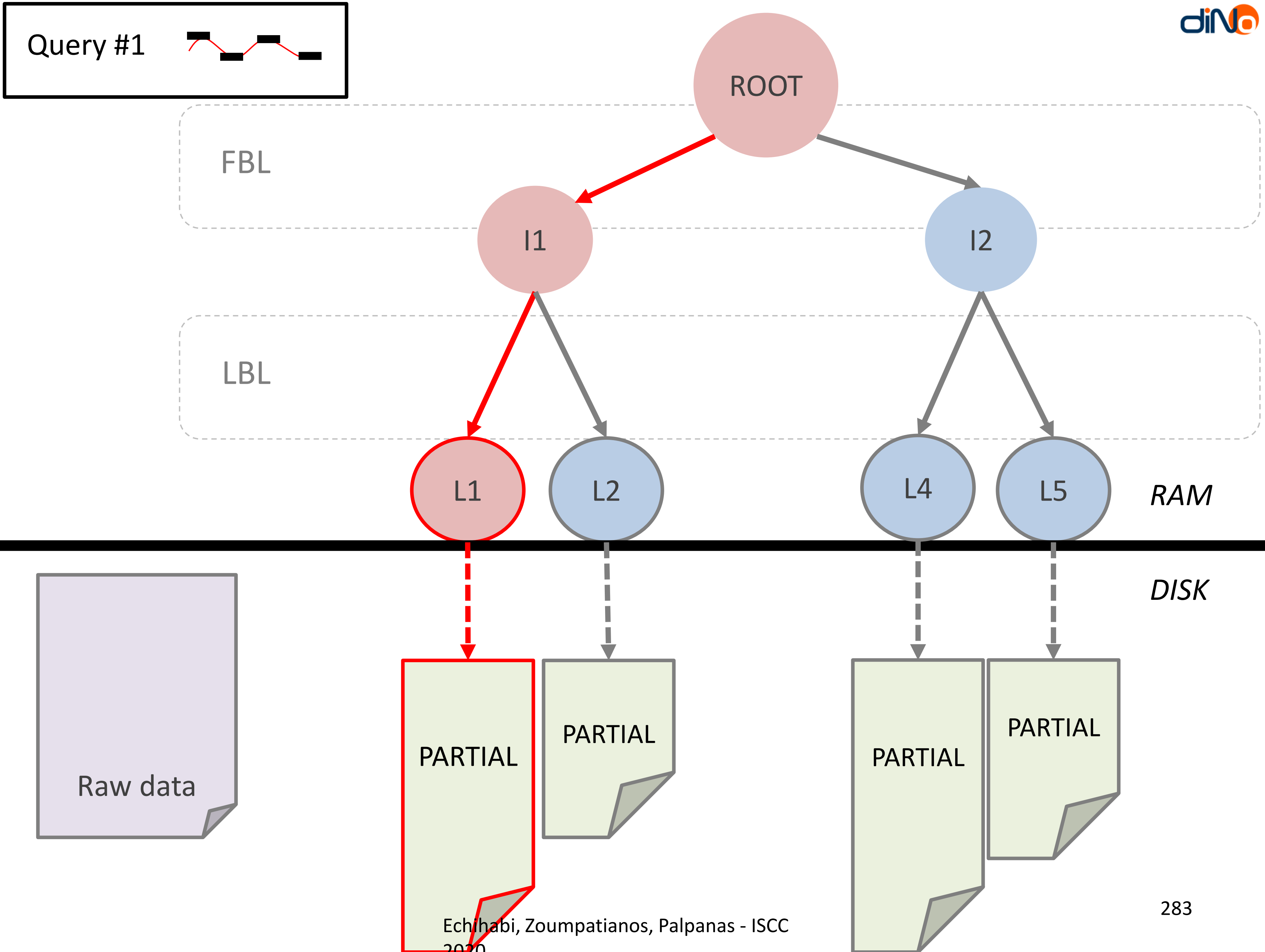
When we run out of main memory we **spill** the **raw data series** into the **right leaves**



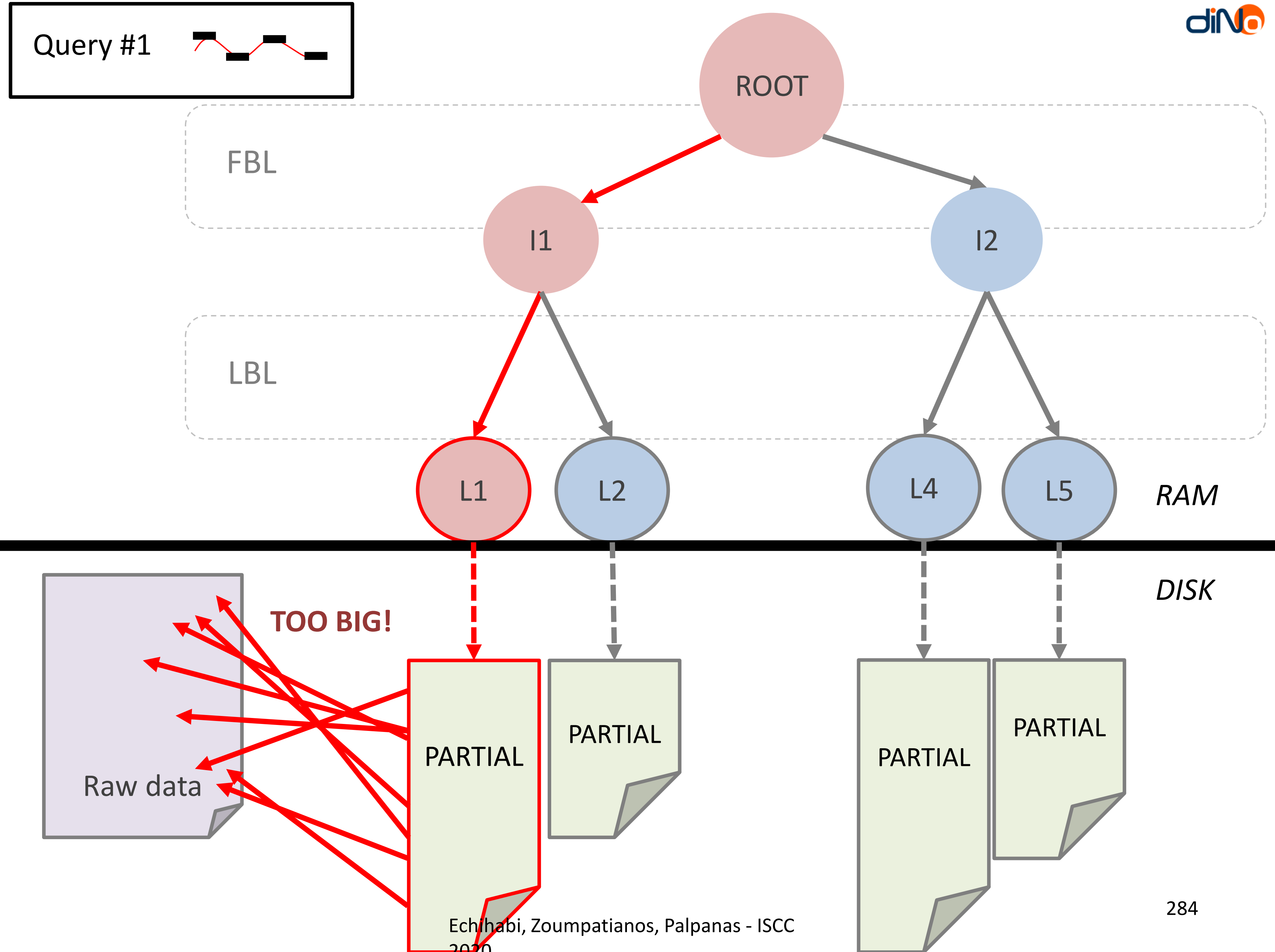
We end up with a structure that contains both **partial** and **fully materialized** leaves



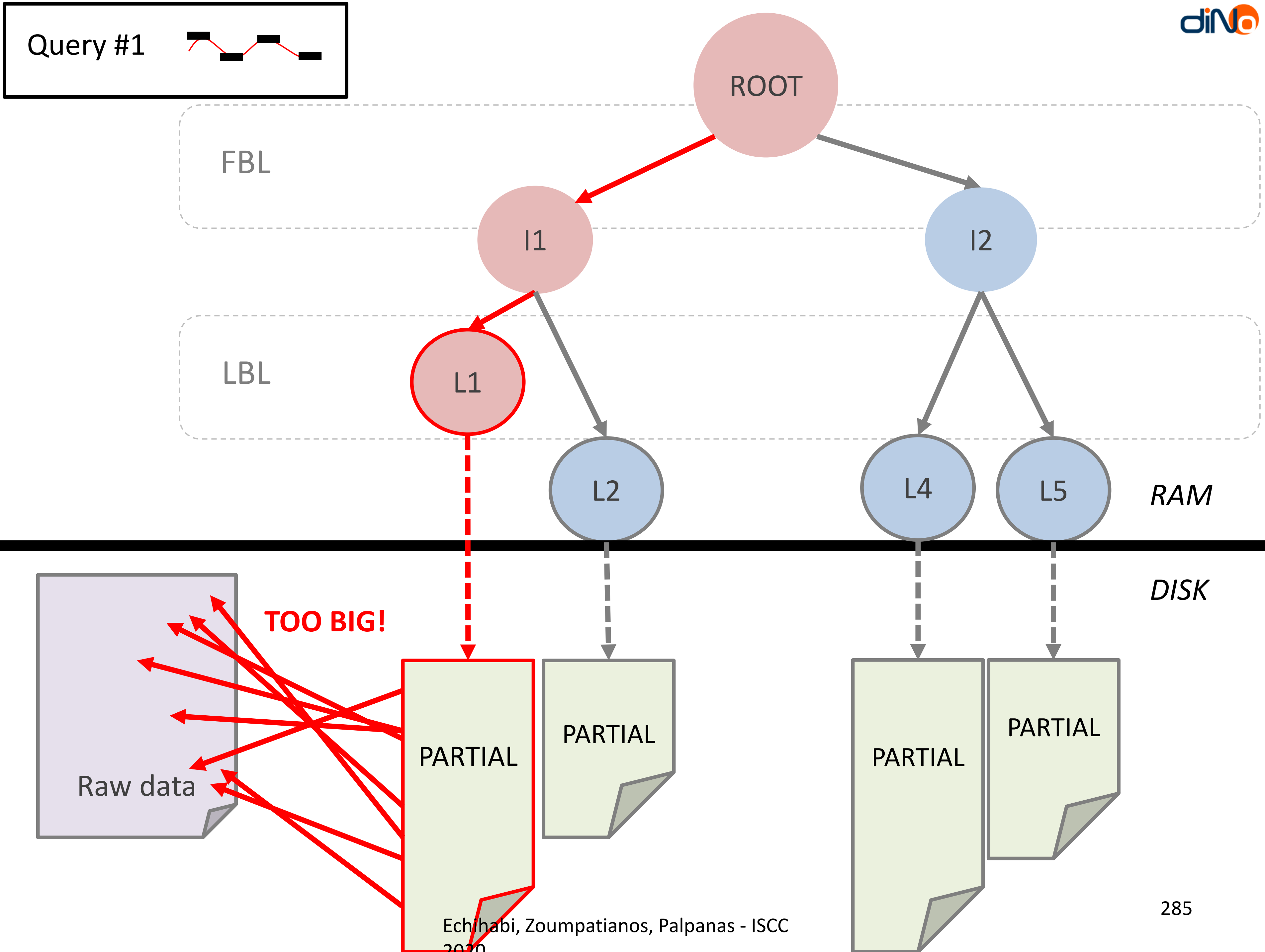

ROOT
FBL
I1
I2
LBL
L1
L2
L4
L5
RAM
DISK
Raw data
PARTIAL
PARTIAL
PARTIAL
PARTIAL


Query #1
ROOT
FBL
I1
I2
LBL
L1
L2
L4
L5
RAM
DISK
Raw data
PARTIAL
PARTIAL
PARTIAL
PARTIAL

Query #1


ROOT
FBL
I1
I2
LBL
L1
L2
L4
L5
RAM
DISK
TOO BIG!
Raw data
PARTIAL
PARTIAL
PARTIAL
PARTIAL

Query #1



ROOT

FBL

I1

I2

LBL

L1

L2

L4

L5

RAM

DISK

TOO BIG!

Raw data

PARTIAL

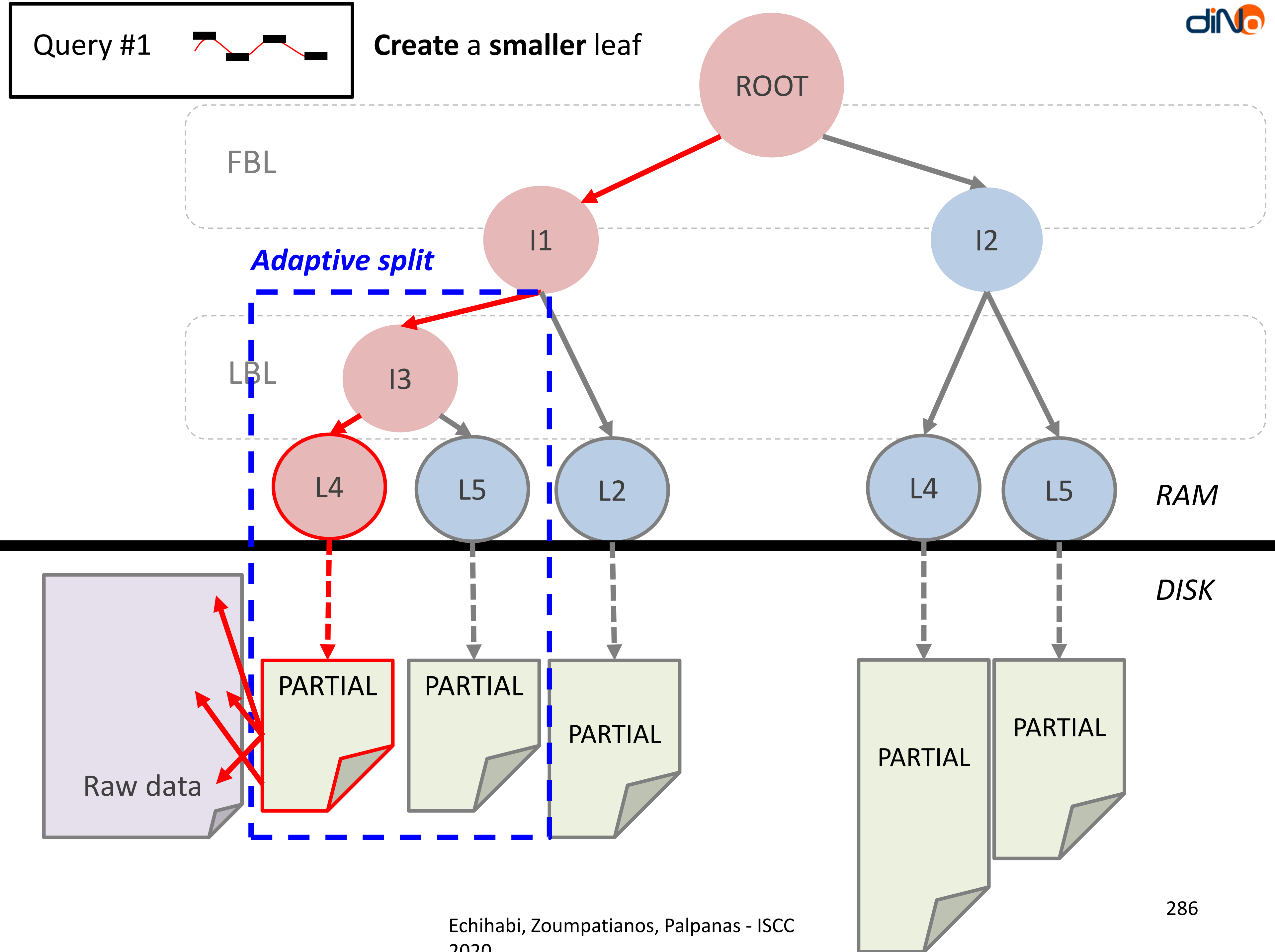
PARTIAL

PARTIAL

PARTIAL

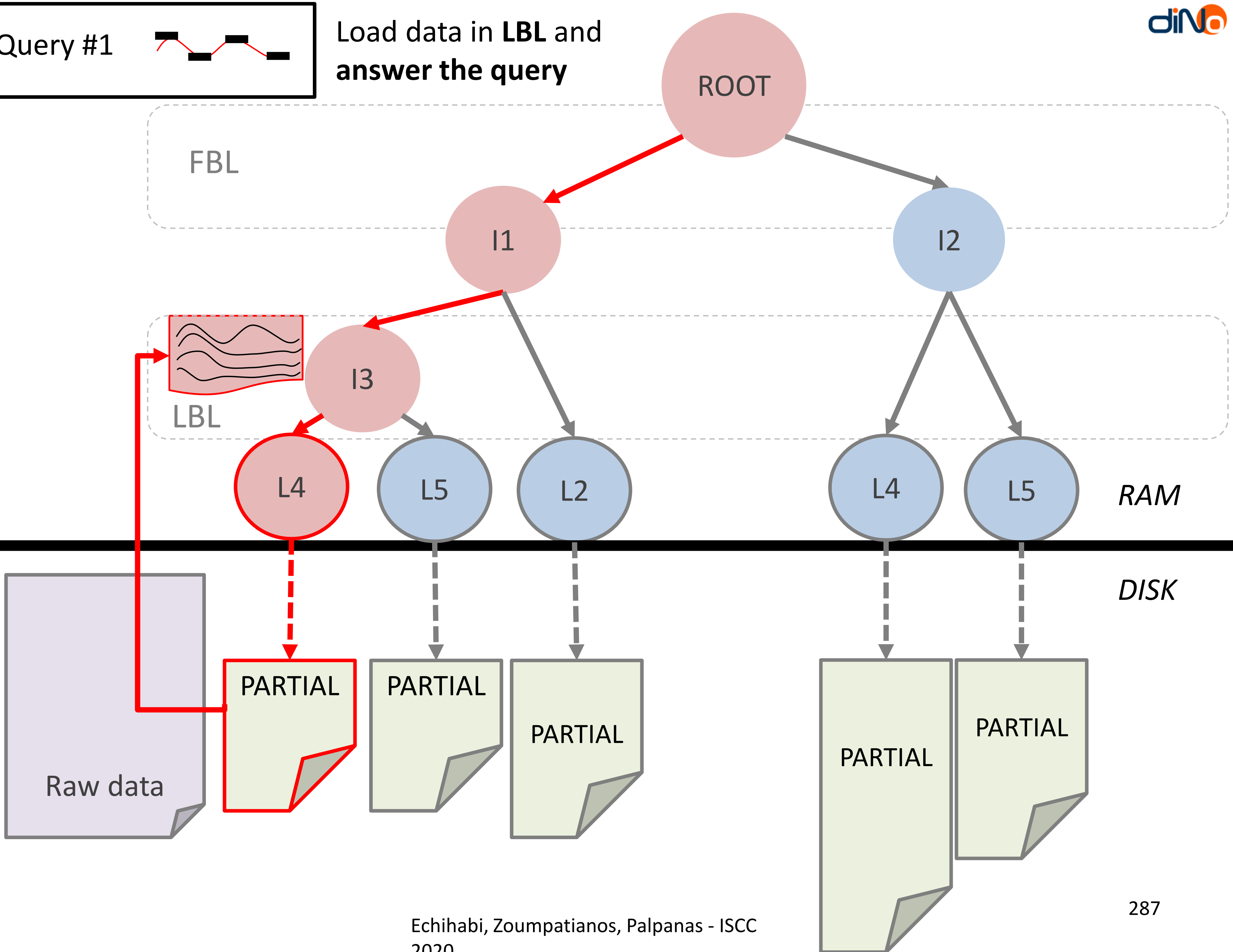
Query #1

Create a **smaller** leaf

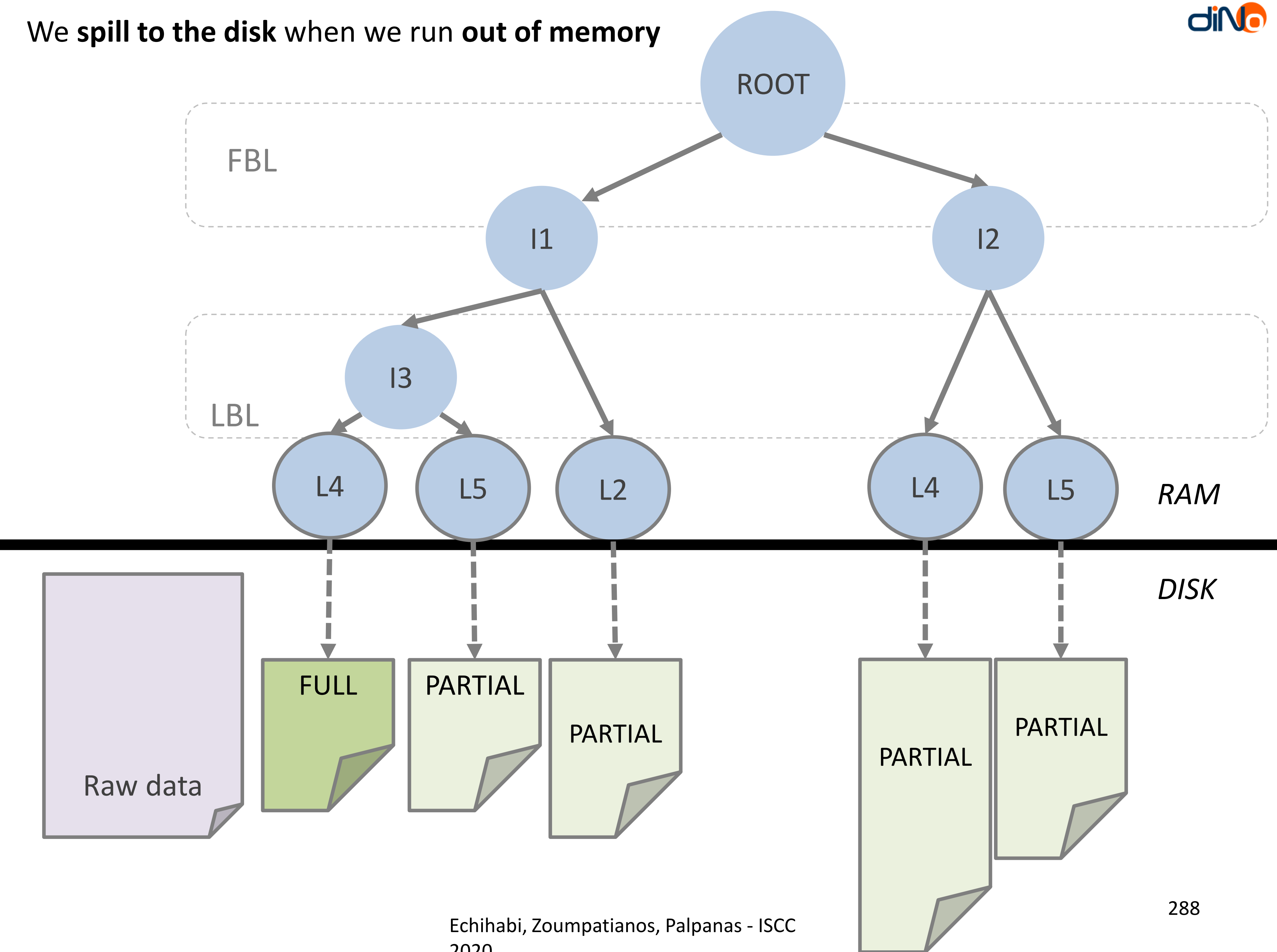


Query #1

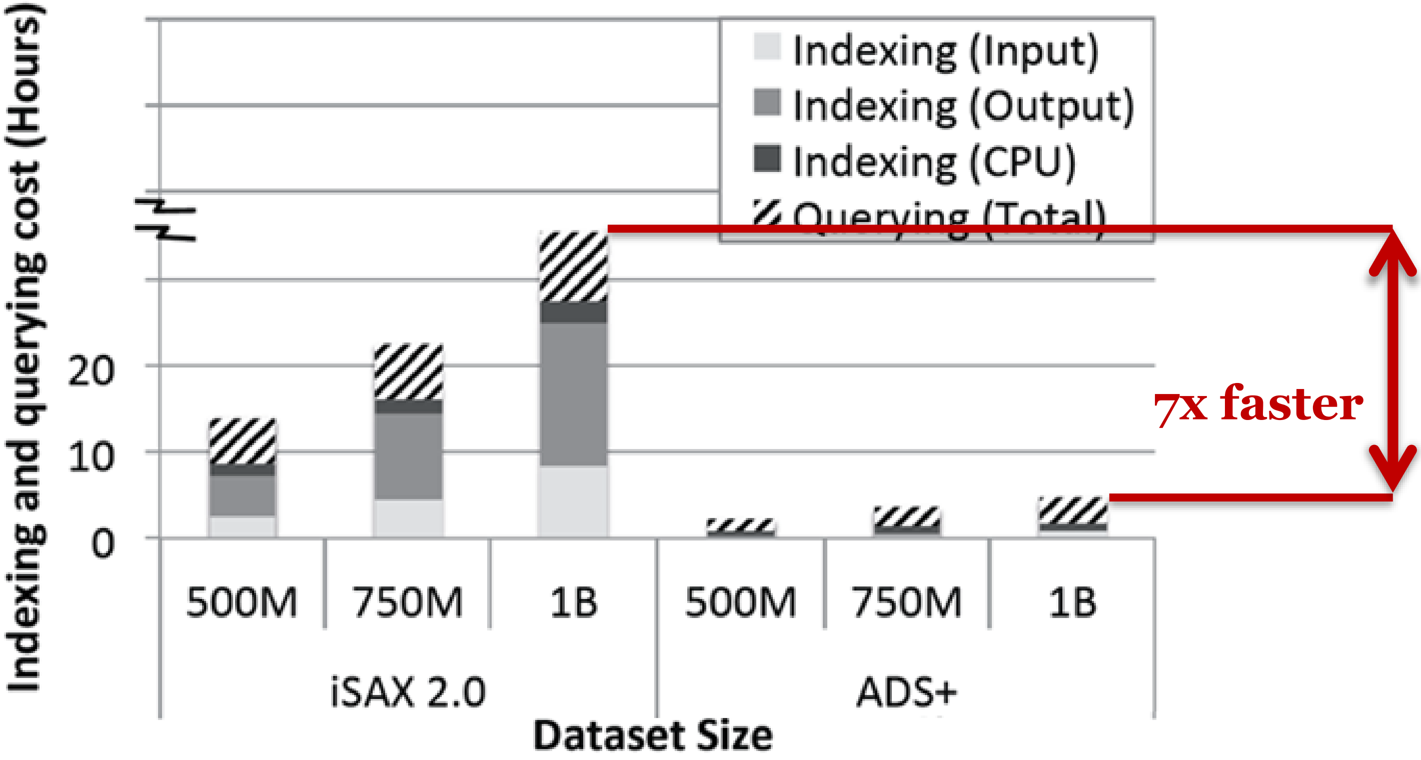
Load data in **LBL** and **answer the query**



We **spill to the disk** when we run **out of memory**



Experimental Evaluation



- iSAX 2.0 needs more than 35 hours to answer 100K approximate queries
- ADS+ answers 100K approximate queries in less than 5 hours

Publications

PVLDB‘18

SIGMOD’19

VLDBJ’20

Extensions…

- Coconut: current solution for limited memory devices and streaming time series
  - bottom-up, succinct index construction based on sortable summarizations

Extensions…

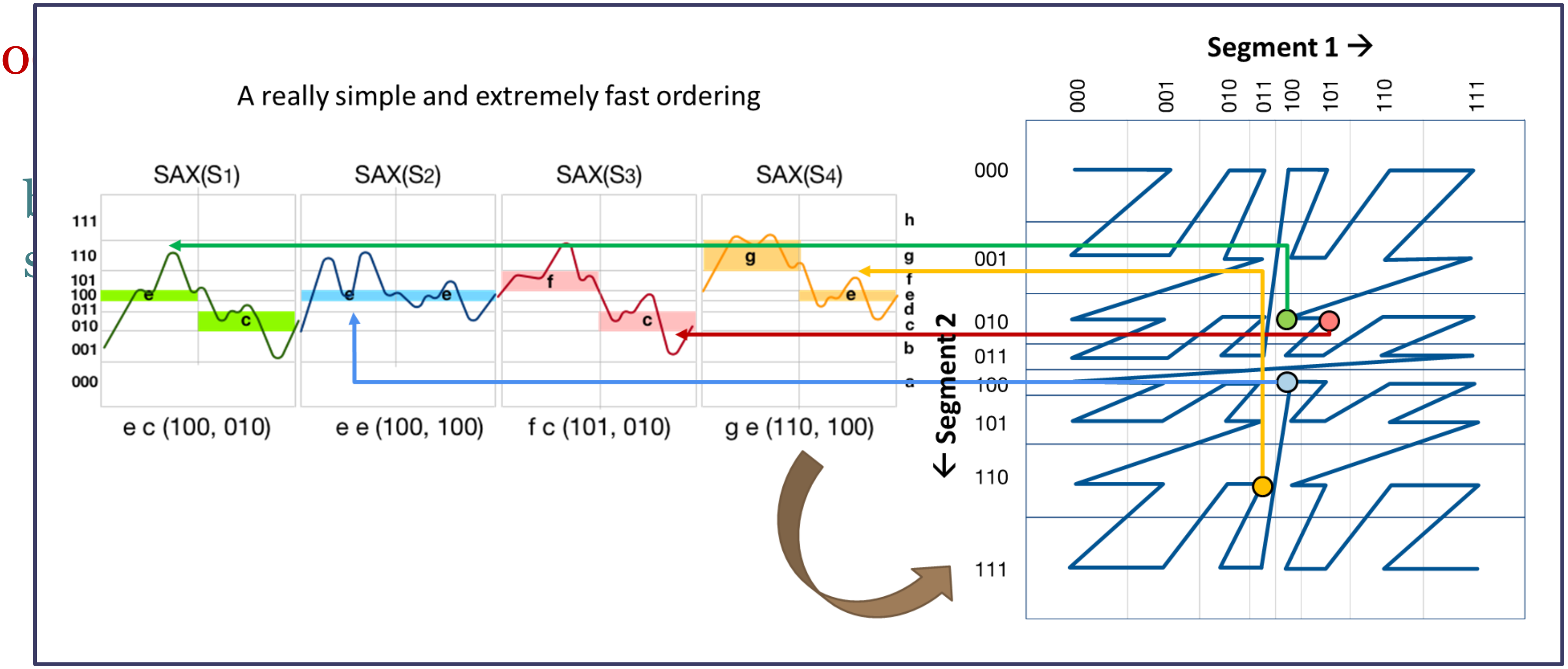
Publications

PVLDB'18

SIGMOD'19

VLDBJ'20

- Co
    - l
      s



A really simple and extremely fast ordering

Publications

PVLDB‘18

SIGMOD’19

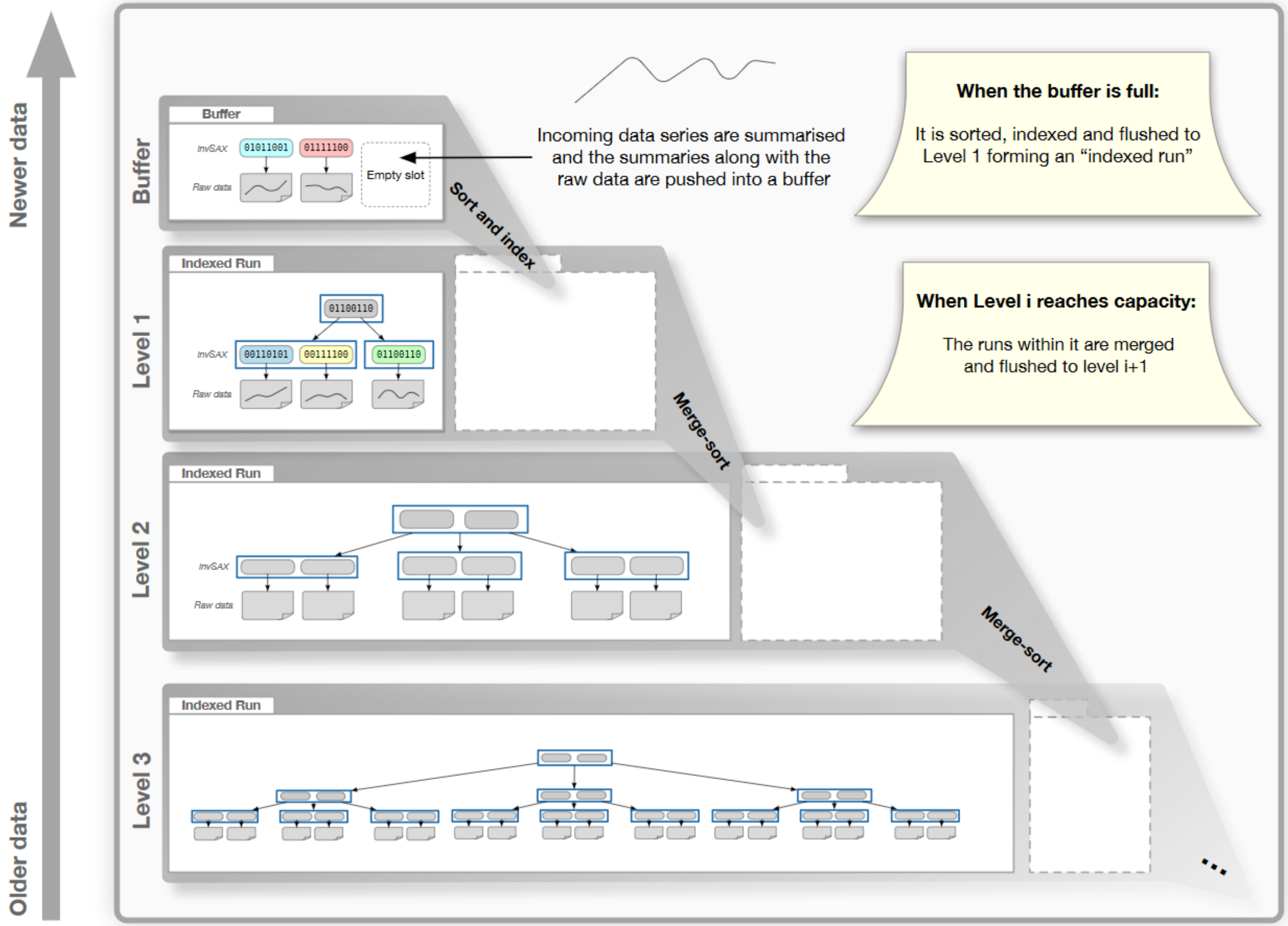
VLDBJ’20

Extensions…

- Coconut: current solution for limited memory devices and streaming time series
  - bottom-up, succinct index construction based on sortable summarizations
  - outperforms state-of-the-art in terms of index space, index construction time, and query answering time

Coconut-LSM

Extensions…



Incoming data series are summarised and the summaries along with the raw data are pushed into a buffer

When the buffer is full:

It is sorted, indexed and flushed to Level 1 forming an "indexed run"

When Level i reaches capacity:

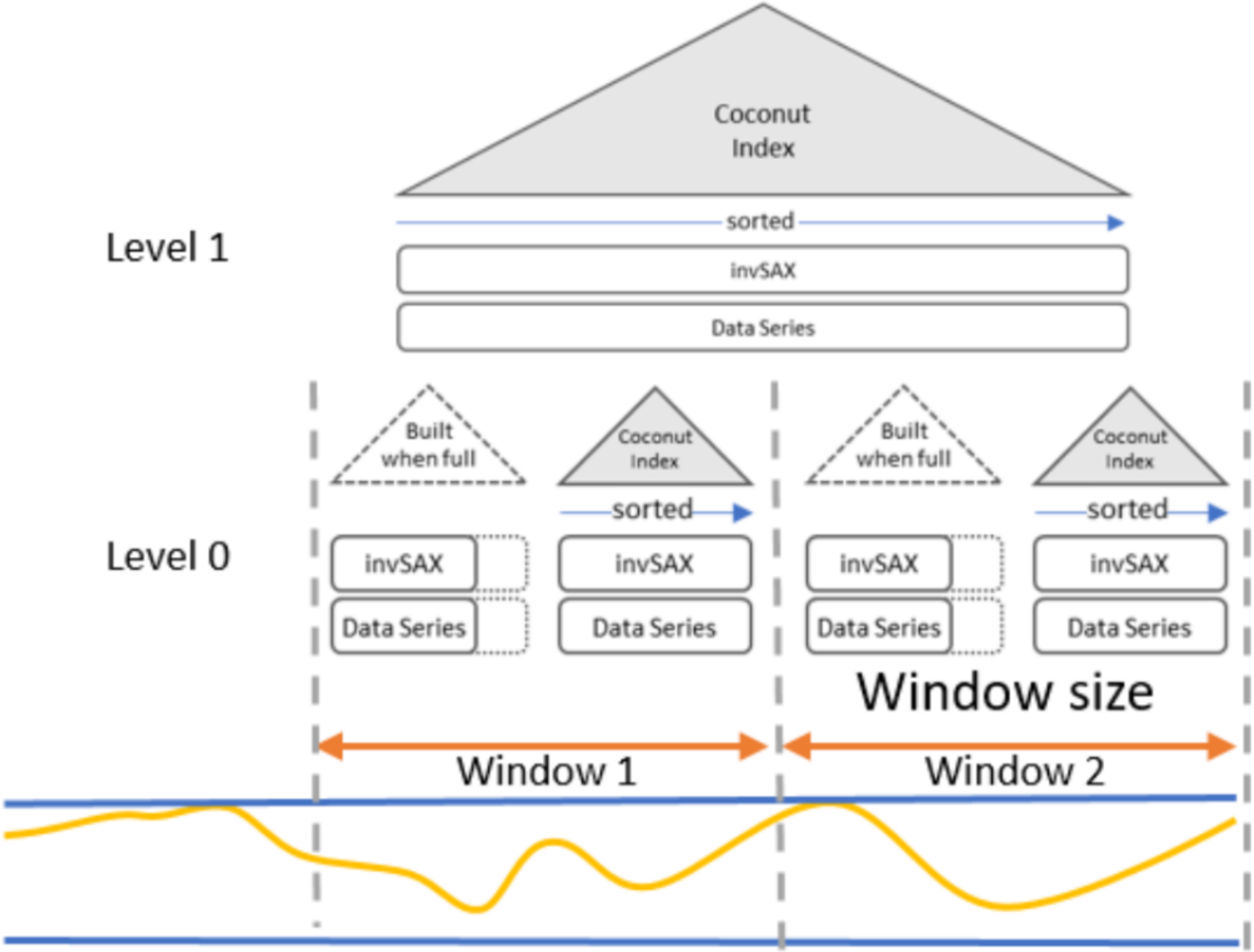
The runs within it are merged and flushed to level i+1

Publications

- PVLDB'18
- SIGMOD'19
- VLDBJ'20

Coconut-LSM

Extensions…



Publications

- PVLDB‘18
- SIGMOD’19
- VLDBJ’20

Extensions…

Publications

PVLDB'18

SIGMOD'19

VLDBJ'20

ICDE'18

PVLDB'19

- Coconut: current solution for limited memory devices and streaming time series
  - bottom-up, succinct index construction based on sortable summarizations
  - outperforms state-of-the-art in terms of index space, index construction time, and query answering time
- ULISSE: current solution for variable-length queries
  - single-index support of queries of variable lengths

Publications

PVLDB'18

SIGMOD'19

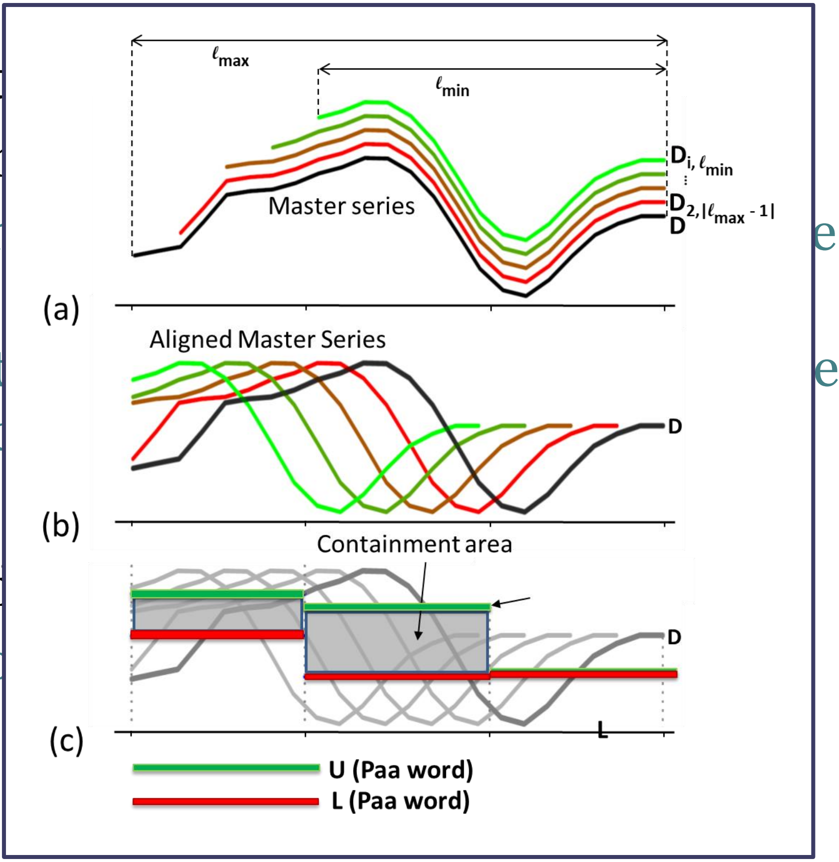
VLDBJ'20

ICDE'18

PVLDB'19

Extensions…

- Coconut: current solut
  and streamin
  - bottom-up, succinct in e
    summarizations
  - outperforms state-of-t ex
    construction time, anc



- ULISSE: current soluti
  - single-index support c

Publications

PVLDB'18

SIGMOD'19

VLDBJ'20

ICDE'18

PVLDB'19

Extensions...

- Coconut: current solution for limited memory devices and streaming time series
  - bottom-up, succinct index construction based on sortable summarizations
  - outperforms state-of-the-art in terms of index space, index construction time, and query answering time
- ULISSE: current solution for variable-length queries
  - single-index support of queries of variable lengths
  - orders of magnitude faster than competing approaches

Publications

ICDM'17

TKDE'18

PKDD'19

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (Spark)
  - balances work of different worker nodes

Parallelization/Distribution

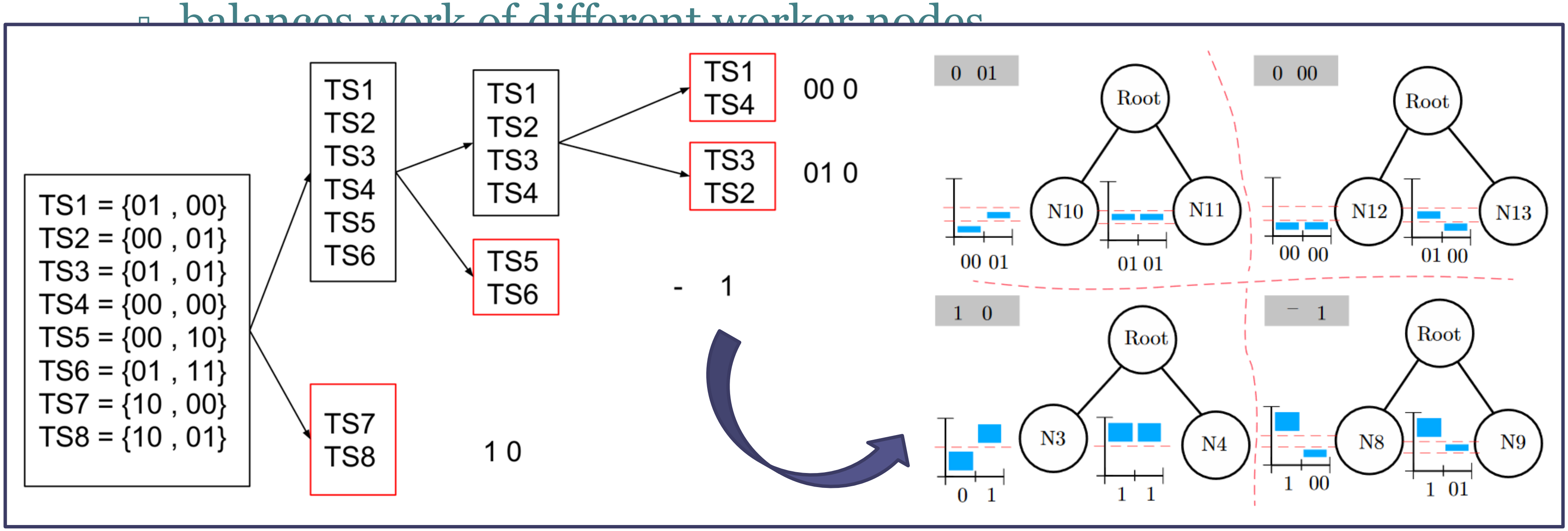
Publications

ICDM‘17

TKDE’18

PKDD’19

- **DPiSAX**: current solution for distributed processing (Spark)
  - balances work of different worker nodes



TS1 = {01 , 00}
TS2 = {00 , 01}
TS3 = {01 , 01}
TS4 = {00 , 00}
TS5 = {00 , 10}
TS6 = {01 , 11}
TS7 = {10 , 00}
TS8 = {10 , 01}

TS1 TS2 TS3 TS4 TS5 TS6

TS7 TS8

1 0

TS1 TS2 TS3 TS4

TS5 TS6

- 1

TS1 TS4

00 0

TS3 TS2

01 0

Publications

ICDM'17

TKDE'18

PKDD'19

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (Spark)
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized solution

Publications

ICDM'17

TKDE'18

PKDD'19

BigData'18

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (Spark)
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized solution

- ParIS: current solution for modern hardware
  - completely masks out the CPU cost

Publications

ICDM'17

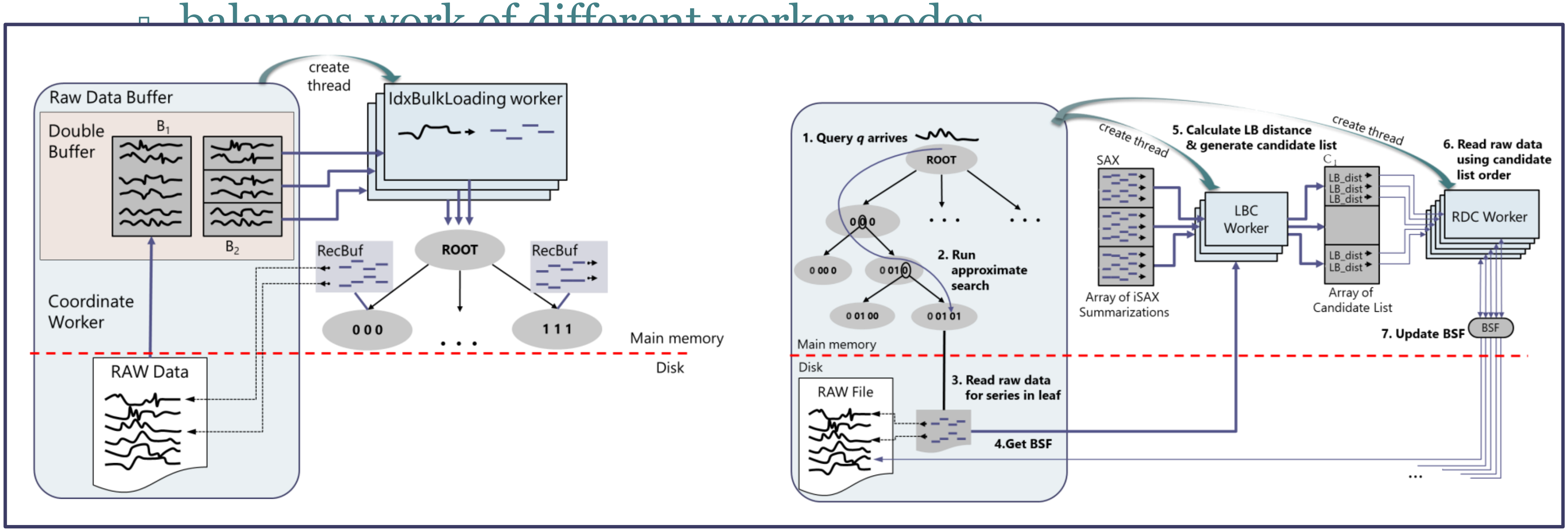
TKDE'18

PKDD'19

BigData'18

Parallelization/Distribution

- **DPiSAX**: current solution for distributed processing (Spark)
  - balances work of different worker nodes



Publications

ICDM'17

TKDE'18

PKDD'19

BigData'18

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (Spark)
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized solution
- ParIS: current solution for modern hardware
  - masks out the CPU cost
  - answers exact queries in the order of a few secs
    - 3 orders of magnitude faster then single-core solutions

Publications

ICDM‘17

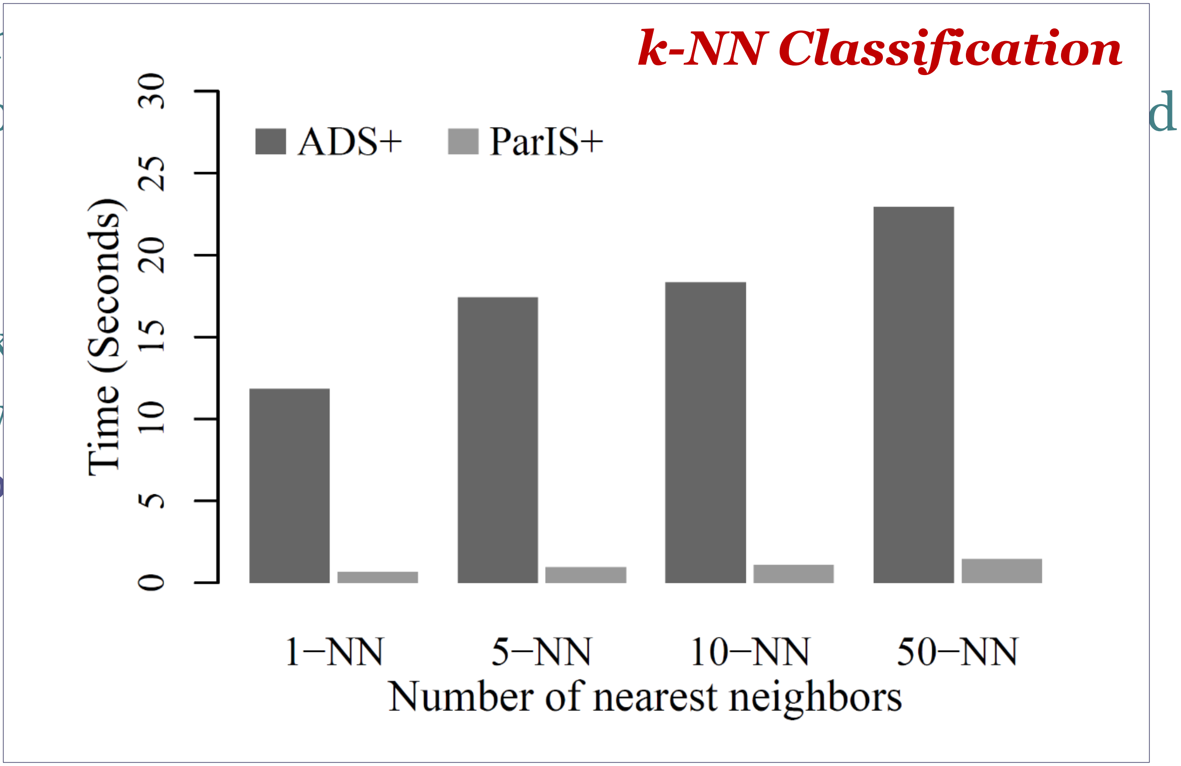
TKDE’18

PKDD’19

BigData’18

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (Spark)
  - balar
  - perfo d solution
- ParIS:
  - mask
  - answ
    - 3 o



Parallelization/Distribution

Publications

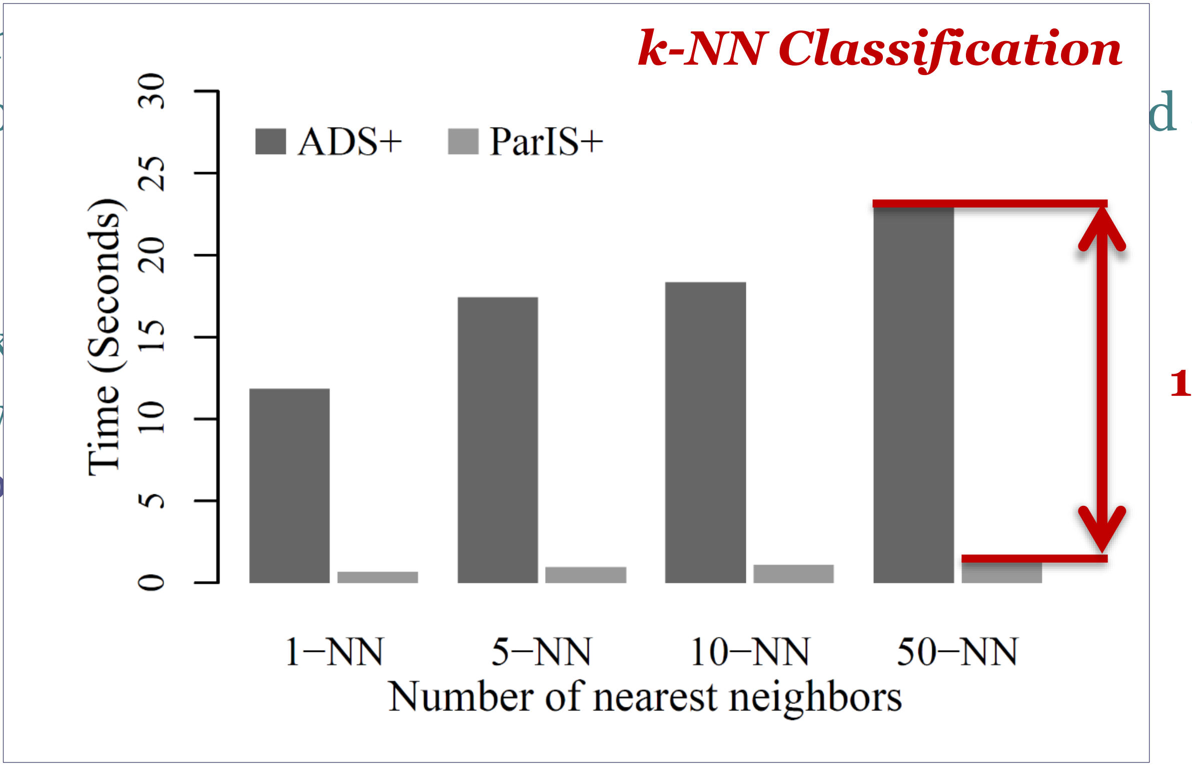
ICDM'17

TKDE'18

PKDD'19

BigData'18

- DPiSAX: current solution for distributed processing (Spark)
  - balar
  - perfo d solution
- ParIS:
  - mask
  - answ
    - 3 o



18x faster

Publications

ICDM'17

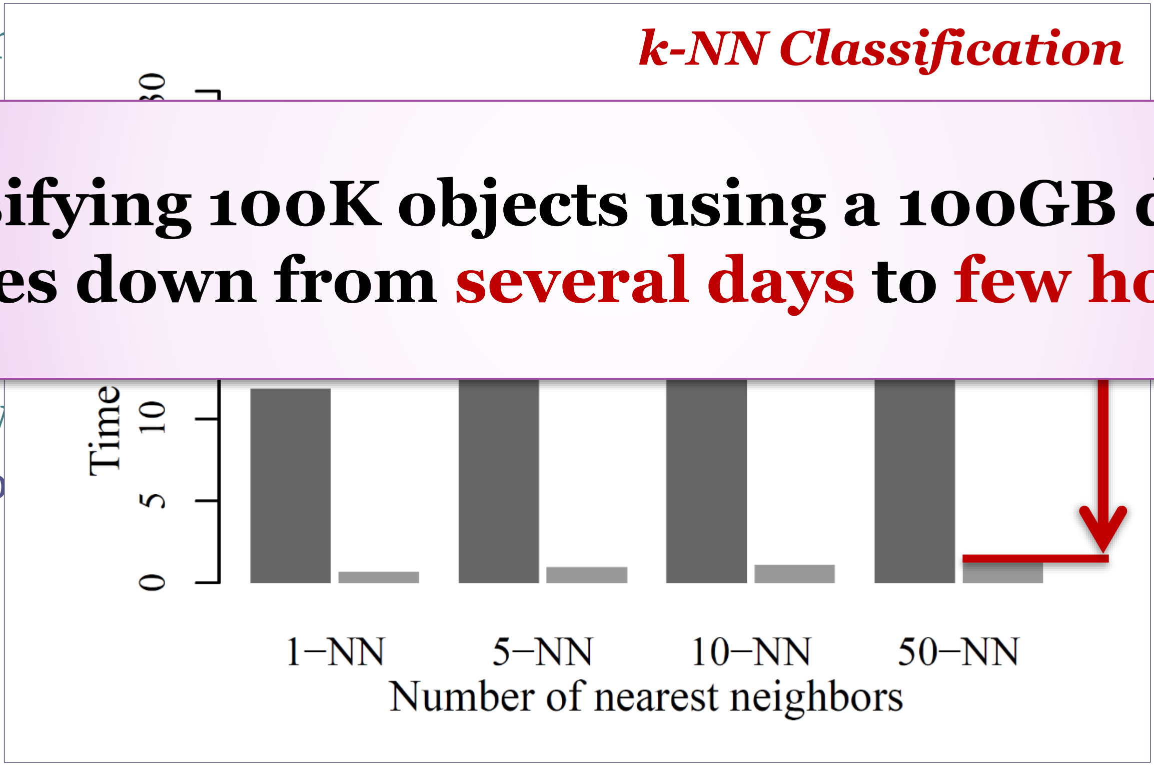
TKDE'18

PKDD'19

BigData'18

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (Spark)
    - balar
    - answ
        - 3 o


k-NN Classification

18x faster

classifying 100K objects using a 100GB dataset goes down from several days to few hours!

Parallelization/Distribution

Publications

- ICDM'17
- TKDE'18
- PKDD'19
- BigData'18
- ICDE'20

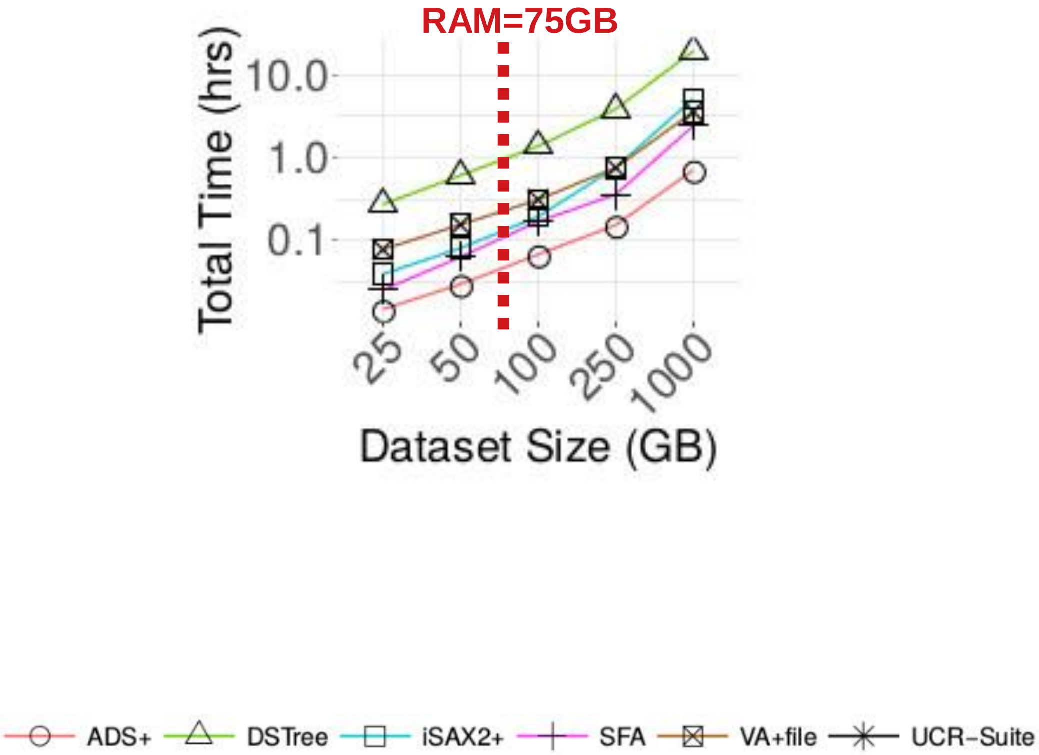
- DPiSAX: current solution for distributed processing (Spar
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized solution
- ParIS: current single-node parallel solution
  - masks out the CPU cost
  - answers exact queries in the order of a few secs
    - >1 order of magnitude faster then single-core solutions
- MESSI: current single-node parallel solution + in-memory data
  - answers exact queries at interactive speeds: ~50msec on 100GB
- SING: current single-node parallel solution + GPU + in-memory data
  - answers exact queries at interactive speeds: ~32msec on 100GB

Experimental Comparison: Exact Query Answering Methods

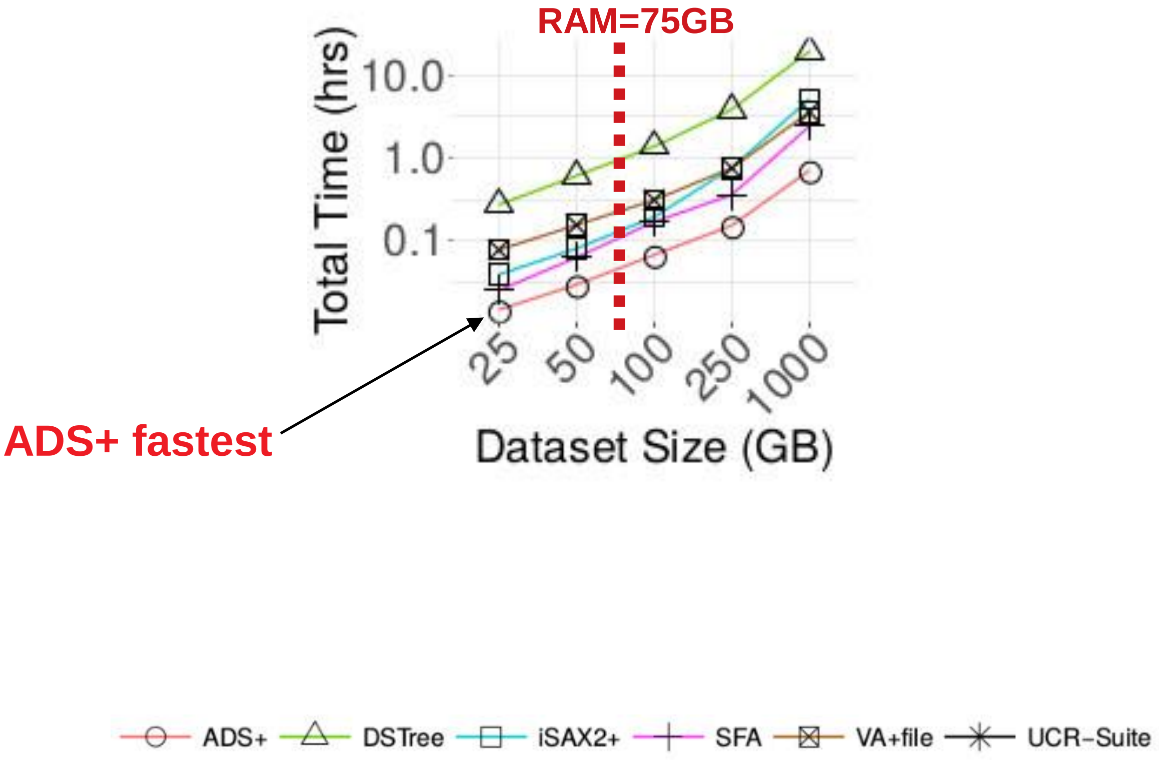
Experimental Framework

- Hardware
  - HDD and SSD
- Datasets
  - Synthetic (25GB to 1TB) and 4 real (100 GB)
- Exact Query Workloads
  - 100 – 10,000 queries
- Performance measures
  - Time, #disk accesses, footprint, pruning, Tightness of Lower Bound (TLB), etc.
- C/C++ methods (4 methods reimplemented from scratch)
- Procedure:
  - Step 1: Parametrization
  - Step 2: Evaluation of individual methods
  - Step 3: Comparison of best methods

Time for Indexing (Idx) vs. Dataset Size



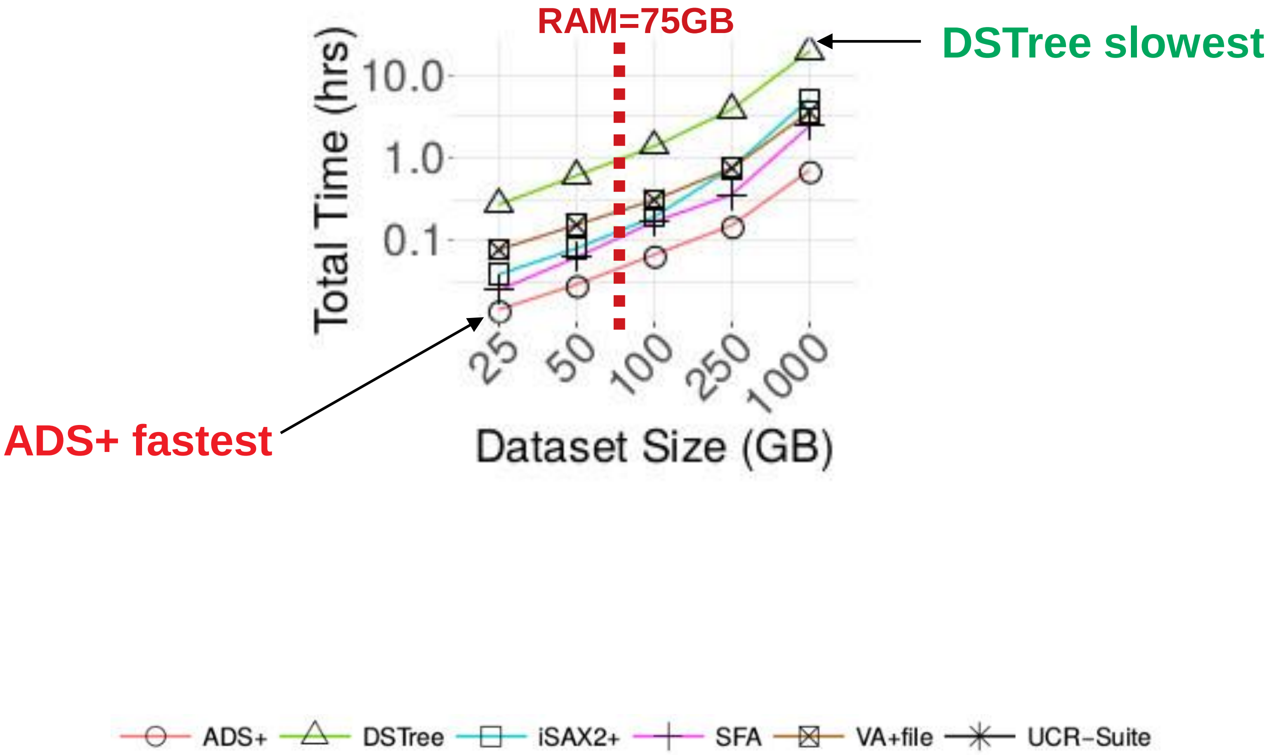
Time for Indexing (Idx) vs. Dataset Size



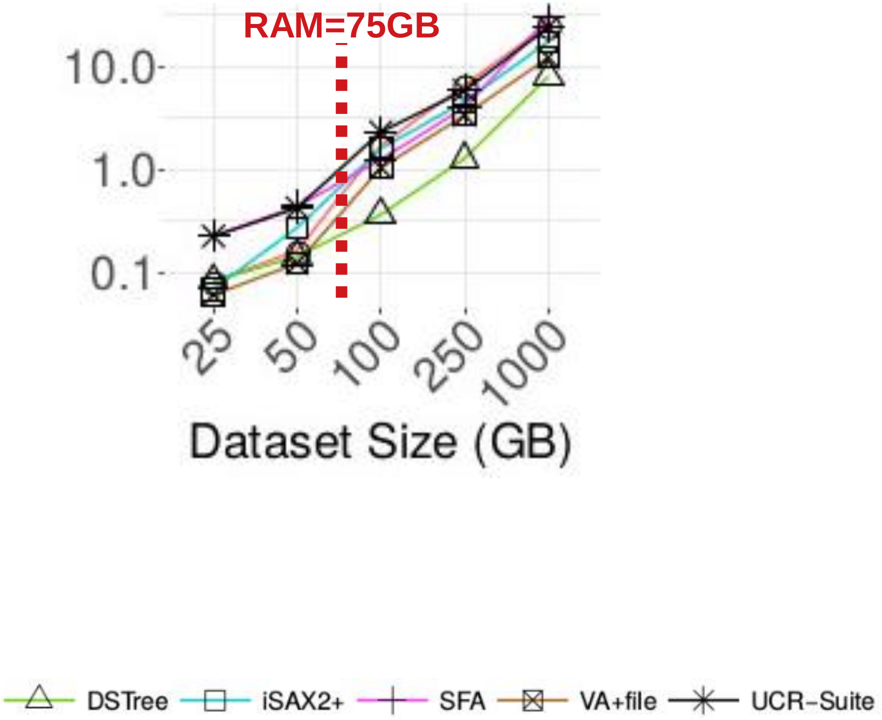
RAM=75GB

ADS+ fastest

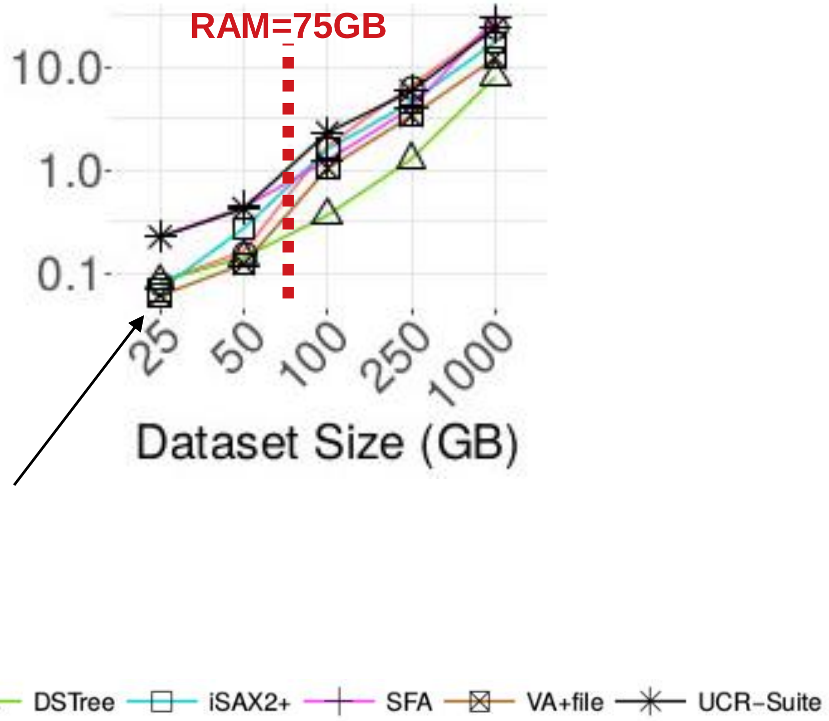
Time for Indexing (Idx) vs. Dataset Size



Time for 100 Exact Queries vs. Dataset size

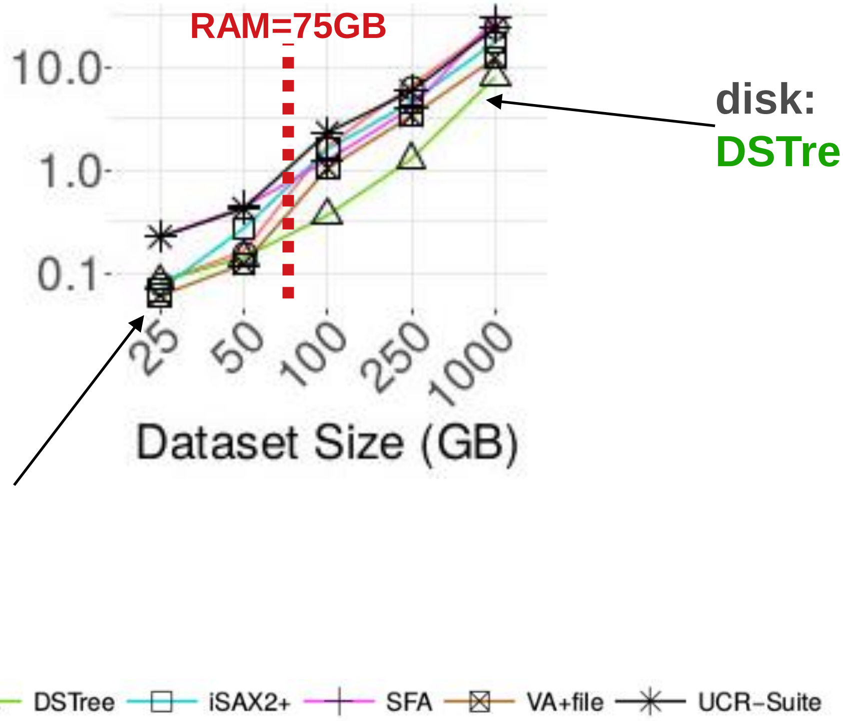


Time for 100 Exact Queries vs. Dataset size



In-memory:
VA+file fastest

Time for 100 Exact Queries vs. Dataset size

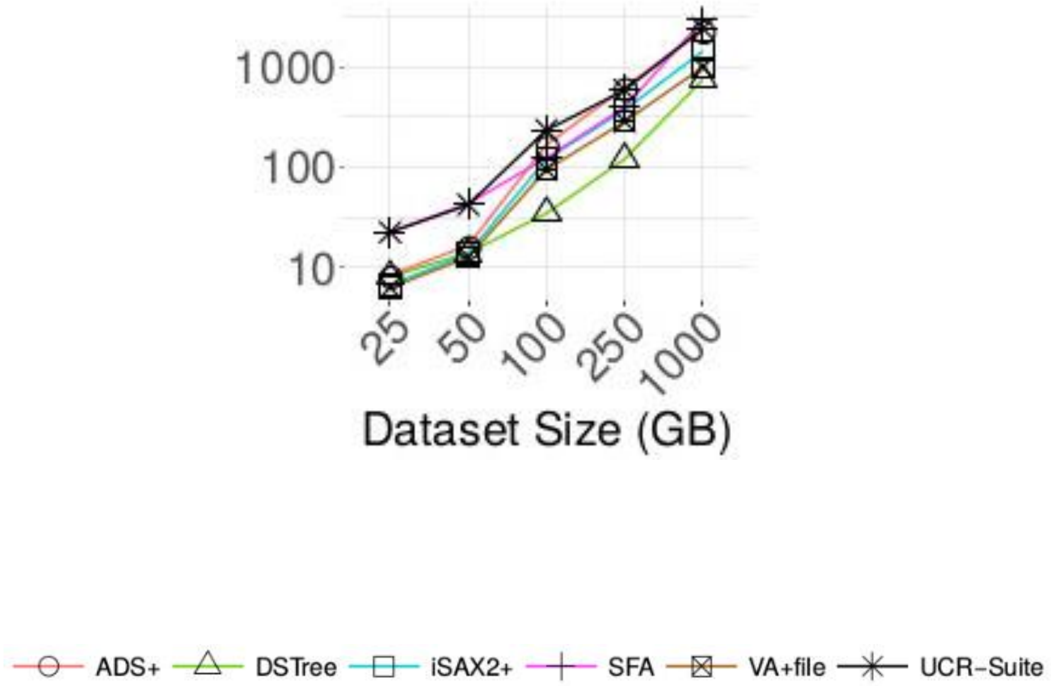


RAM=75GB

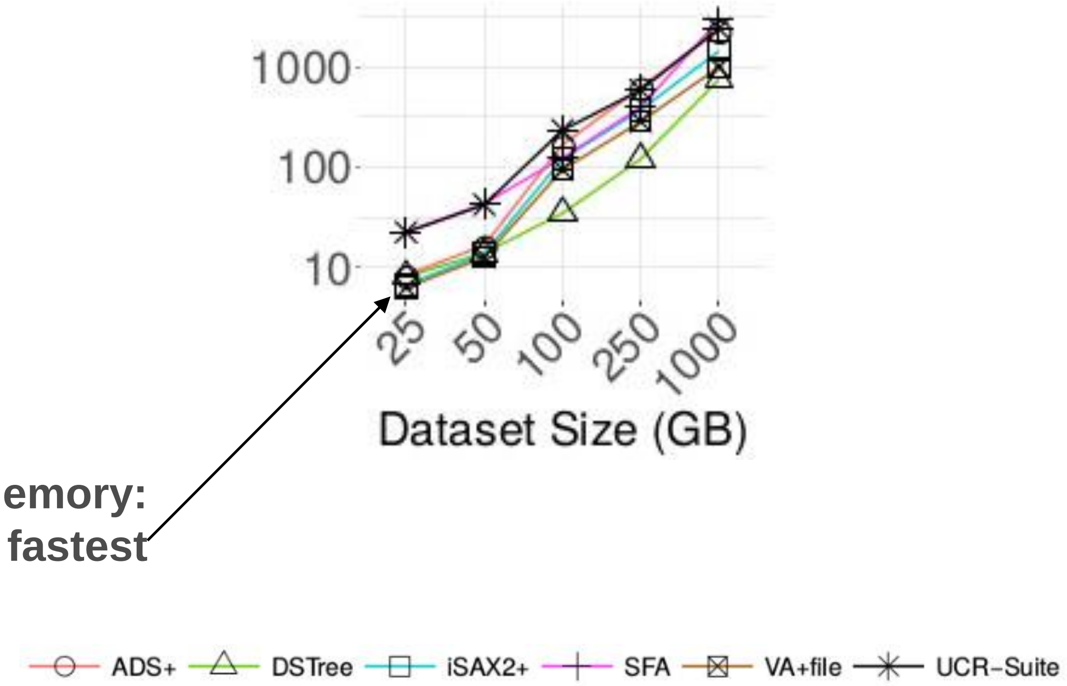
disk:
DSTree fastest

In-memory:
VA+file fastest

Time for Idx + 10K Exact Queries vs. Dataset size

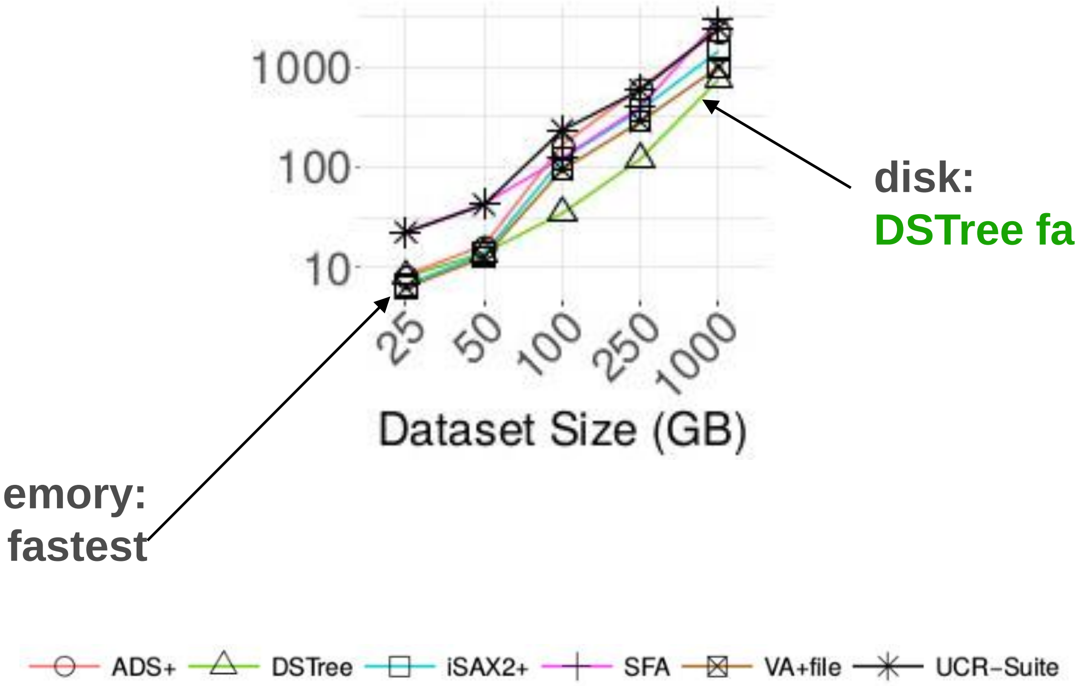


Time for Idx + 10K Exact Queries vs. Dataset size



In-memory:
VA+file fastest

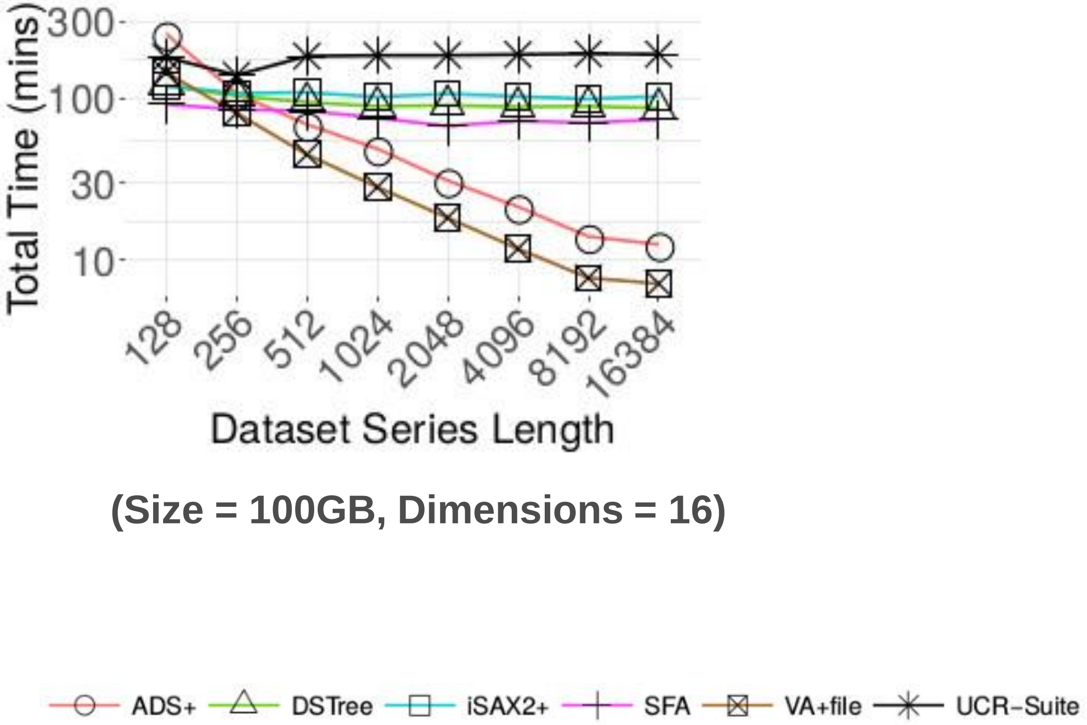
Time for Idx + 10K Exact Queries vs. Dataset size



disk:
DSTree fastest

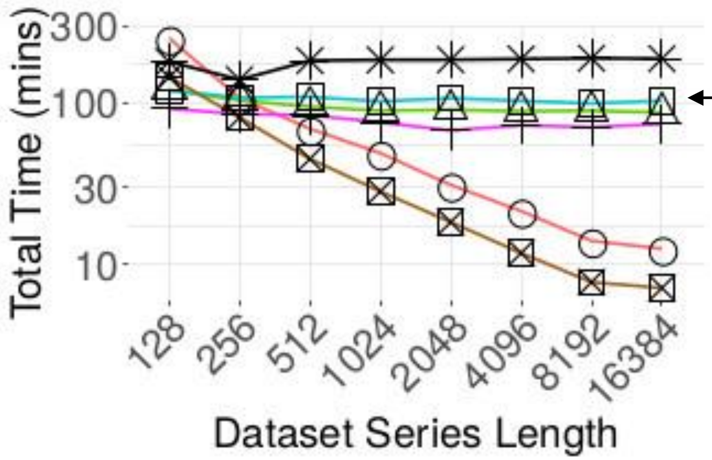
In-memory:
VA+file fastest

Time for Idx + 10K Exact Queries vs. Series Length



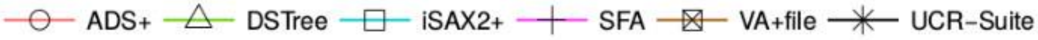
(Size = 100GB, Dimensions = 16)

Time for Idx + 10K Exact Queries vs. Series Length

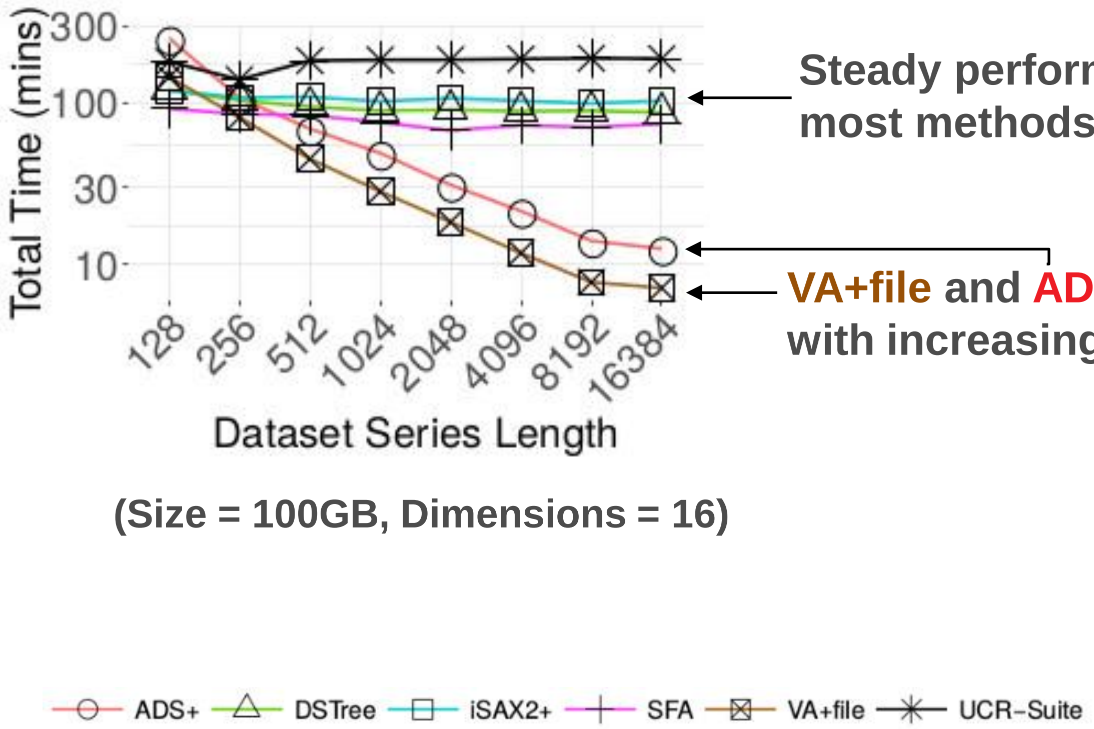


Steady performance for most methods

(Size = 100GB, Dimensions = 16)



Time for Idx + 10K Exact Queries vs. Series Length



Steady performance for most methods

VA+file and ADS+ get faster with increasing length

(Size = 100GB, Dimensions = 16)

Unexpected Results

- Some methods do not scale as expected (or not at all!)
- Brought back to the spotlight two older methods VA+file and DSTree
  - Our reimplementations outperform by far the original ones
- Optimal parameters for some methods are different from the ones reported in the original papers
- Tightness of Lower Bound (TLB) does not always predict performance

TLB does not always predict performance

TLB does not always predict performance

The TLB measures the quality of a summarization (higher is better)

TLB does not always predict performance

The TLB measures the quality of a summarization (higher is better)

$$\text{TLB} = \frac{dist(Query,candidate)\ in\ reduced\ space}{dist(Query,candidate)\ in\ original\ space}$$

TLB does not always predict performance

The TLB measures the quality of a summarization (higher is better)

$$\mathbf{0} \le \quad \text{TLB} = \frac{dist(Query, candidate)\ in\ reduced\ space}{dist(Query, candidate)\ in\ original\ space} \quad \le \mathbf{1}$$

worst

best

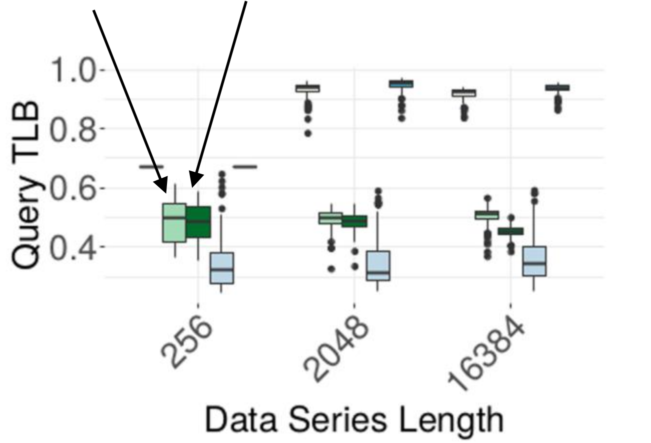
TLB does not always predict performance

The TLB measures the quality of a summarization (higher is better)

$$\mathbf{0} \leq \quad \text{TLB} = \frac{dist(Query,candidate)\ in\ reduced\ space}{dist(Query,candidate)\ in\ original\ space} \quad \leq \mathbf{1}$$

worst best

DSTree and iSAX2+ have similar TLB



TLB does not always predict performance

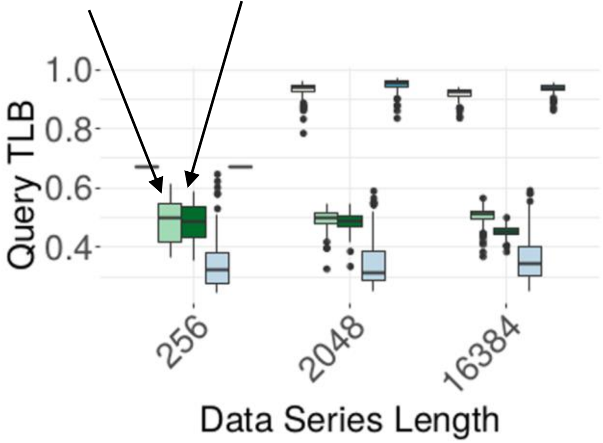
The TLB measures the quality of a summarization (higher is better)

$$0 \leq \quad \text{TLB} = \frac{dist(Query, candidate)\ in\ reduced\ space}{dist(Query, candidate)\ in\ original\ space} \quad \leq 1$$

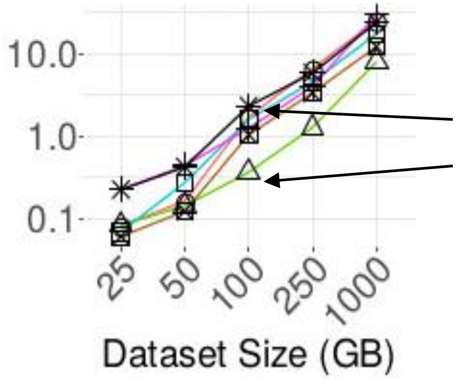
worst

best

DSTree and iSAX2+ have similar TLB



YET



iSAX2+ 5x slower than DSTree

TLB does not always predict performance

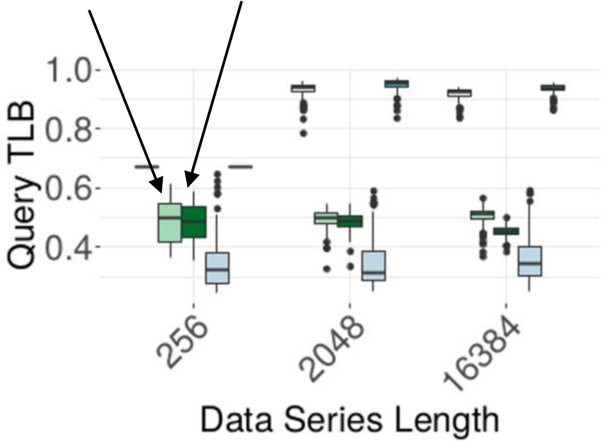
The TLB measures the quality of a summarization (higher is better)

$$0 \leq \quad \text{TLB} = \frac{dist(Query, candidate)\ in\ reduced\ space}{dist(Query, candidate)\ in\ original\ space} \quad \leq 1$$

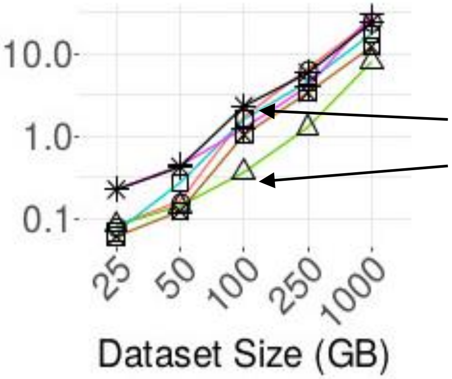
worst

best

DSTree and iSAX2+ have similar TLB



YET



iSAX2+ 5x slower than DSTree

No bias, same data and same implementation framework

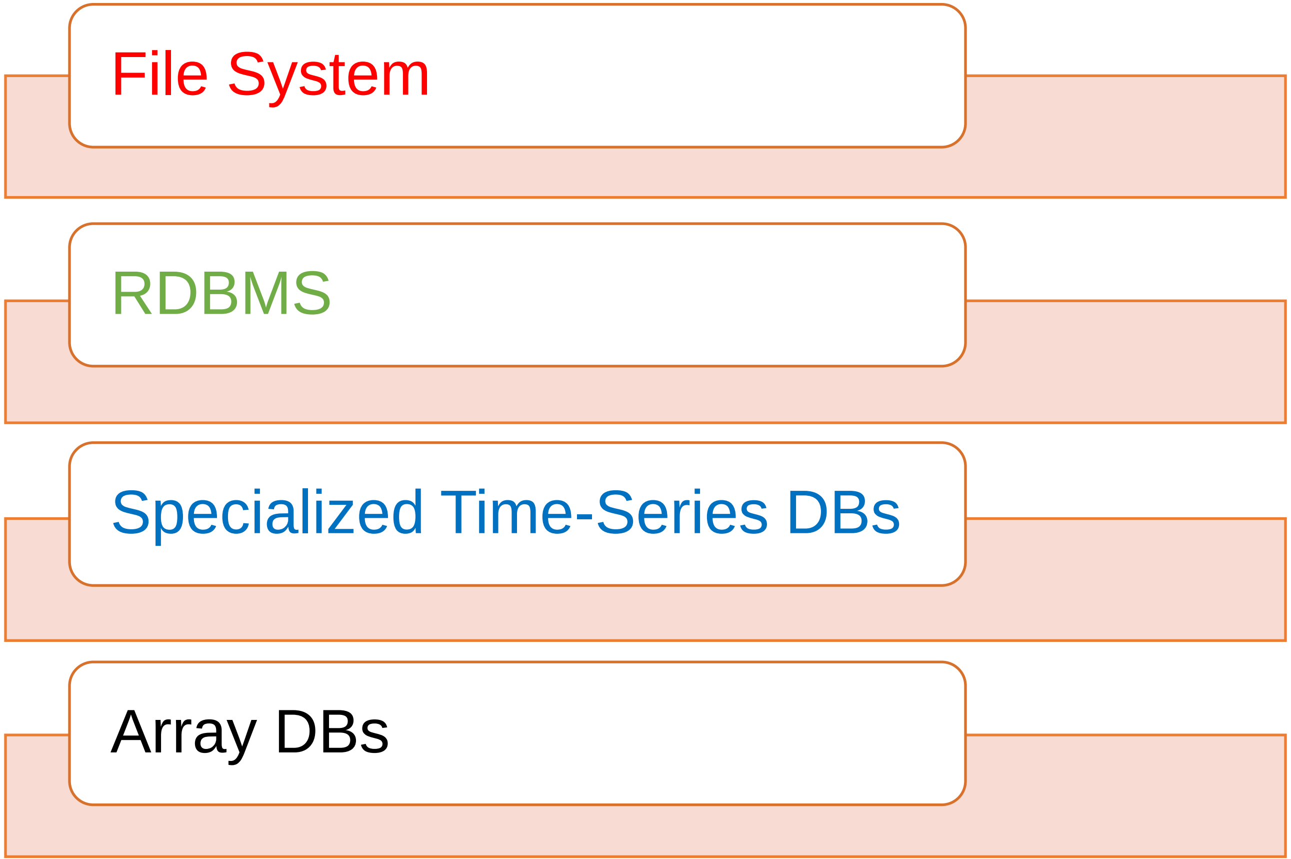
Insights

- Results are sensitive to:
  - Parameter tuning
  - Hardware setup
  - Implementation
  - Workload selection
- Results identify methods that would benefit from modern hardware

Time Series Management Systems

Storing Time-Series

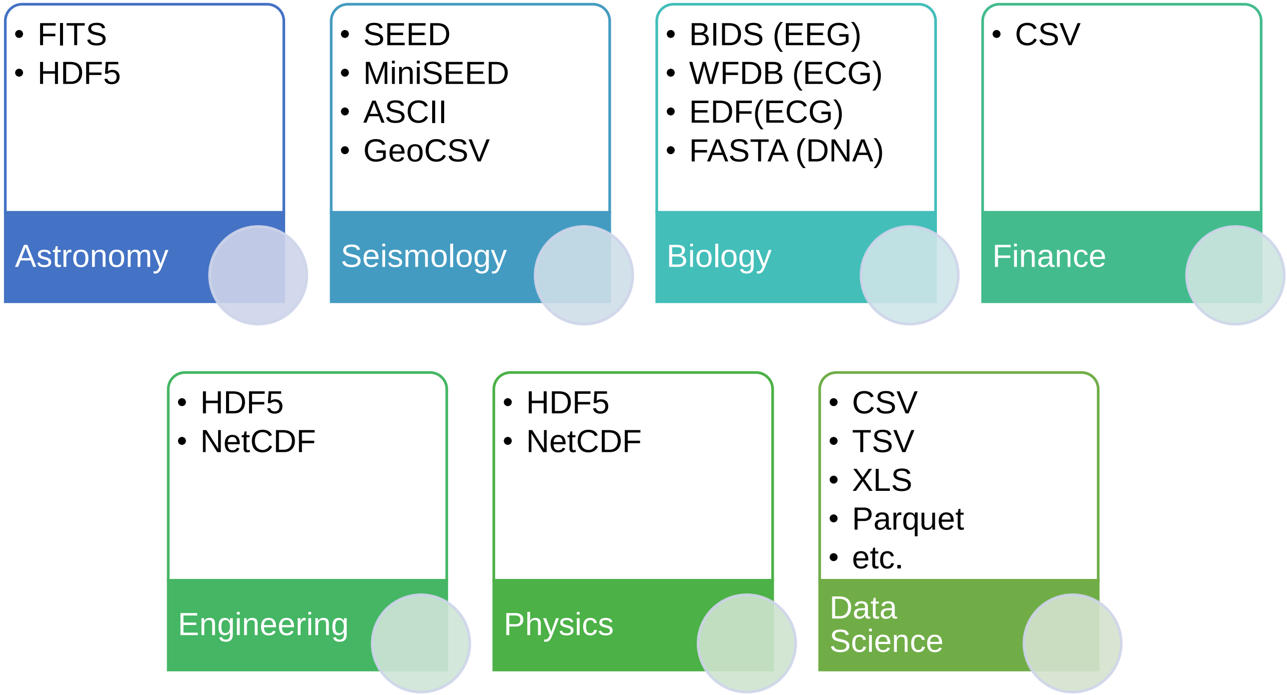
Multiple options. By popularity:



- File System
- RDBMS
- Specialized Time-Series DBs
- Array DBs

Storing Time-Series: **File-System**

Multiple **different formats** implemented for **various applications**



Astronomy
- FITS
- HDF5

Seismology
- SEED
- MiniSEED
- ASCII
- GeoCSV

Biology
- BIDS (EEG)
- WFDB (ECG)
- EDF(ECG)
- FASTA (DNA)

Finance
- CSV

Engineering
- HDF5
- NetCDF

Physics
- HDF5
- NetCDF

Data Science
- CSV
- TSV
- XLS
- Parquet
- etc.

Storing Time-Series: **DBMS**

Commercial System

Illustra (1993) → IBM Informix (Time-Series DataBlade):

- Users need to define a time-series sub-type, which have a datetime as the first column in the definition
- Can encode both regular and irregular time-series (fixed of variable intervals)
- Can describe meta-data
- Supports: running aggregates, prev, next value reasoning, horizontal and vertical mathematical operations, lags, etc.

Academic System

Shore → SEQ

- Custom Time-Series Data Type
- Various time-series operators (order, correlation, etc.)

Oracle:

Commercial System

- Introduced Time-Series functionality in Oracle8
- Now merged into the main product.
- It is in the form of time-series analytics functions (e.g., forecasting)

Storing Time-Series: DBMS

Commercial System

Illustra (1993) → IBM Inform es DataBlade):

- Users need to define a which have a datetime as the first column in the d
- Can encode l of variable interv
- Ca
- Supp zontal and vertica

Most people use DBMSs merely for storing and retrieving time-series.

All analysis is performed externally.

Academic System

Shore → SE

- Custom Tir
- Various time

Commercial System

Oracle:

- Introduced Ti nality
- Now merged into the main product.
- It is in the form of time-series analytics functions (e.g., forecasting)

Storing Time-Series:
Specialized Time-Series DBs

InfluxDB
- Storage: Custom (TSM-Tree)

TimeScaleDB
- Storage: PostgreSQL

Beringei
- Storage: Compressed Arrays on Disk

Druid
- Metadata Storage: DBMS
- Data Storage: HDFS, S3

Prometheus
- Storage: Custom (TSDB Format)

CrateDB
- Storage: Custom (Column-oriented)

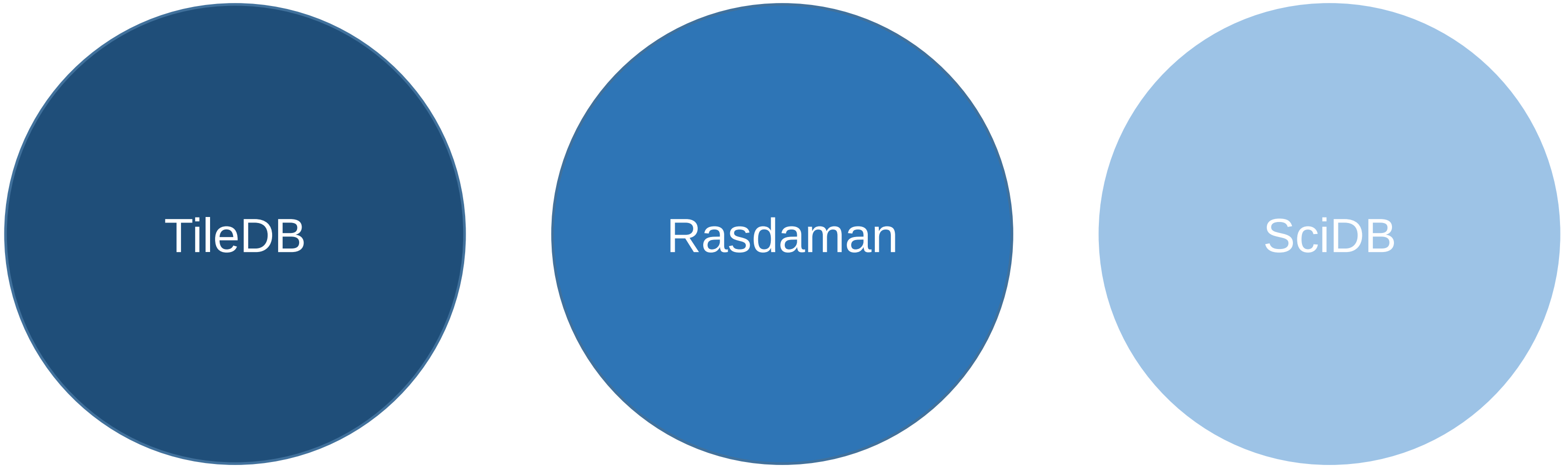
IoTDB
- Storage: Custom: (TsFile – compression + stats)

OpenTSDB
- Storage: HBase

QuasarDB
- Storage: RocksDB

Amazon TimeStream
- Storage: Unknown

Storing Time-Series: **ArrayDBs**

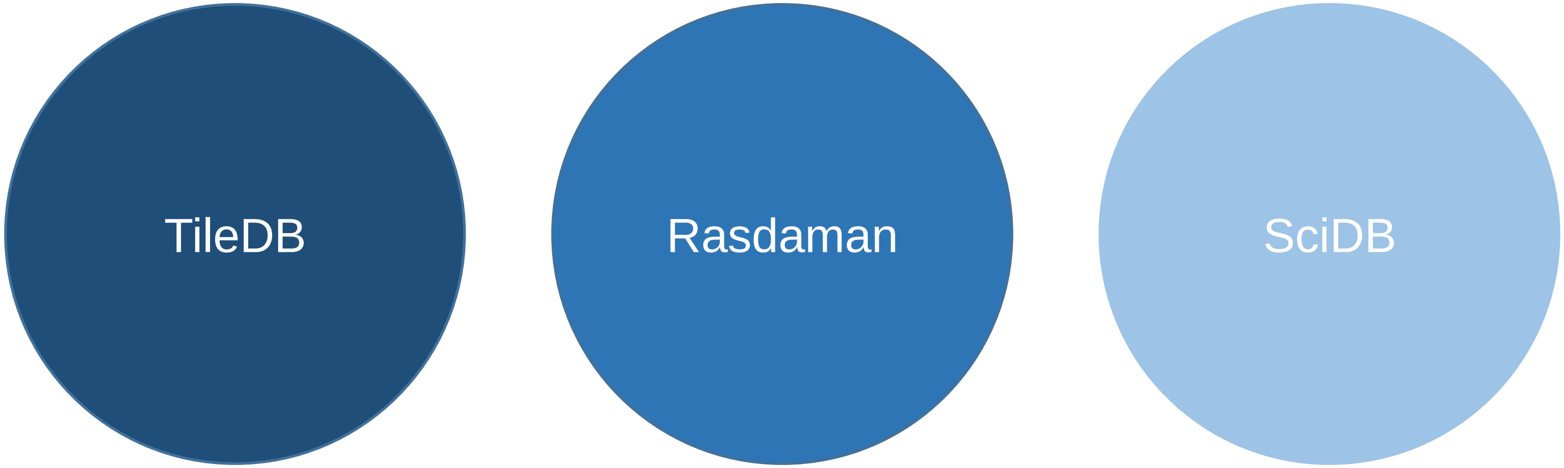


TileDB

Rasdaman

SciDB

Storing Time-Series: **ArrayDBs**



TileDB

Rasdaman

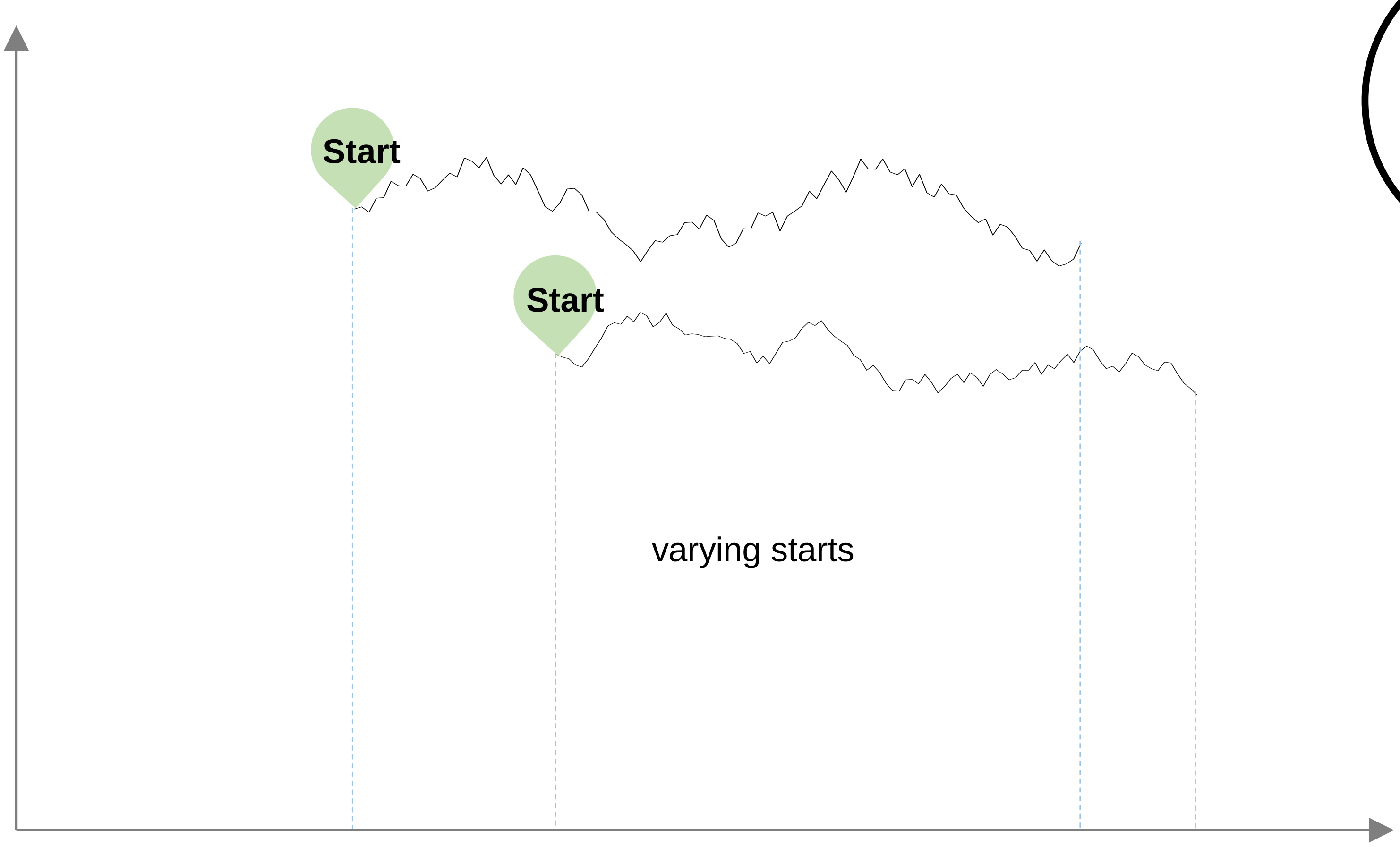
SciDB

Custom Log-Structured storage

Sits on top of existing DBMSs

Custom storage

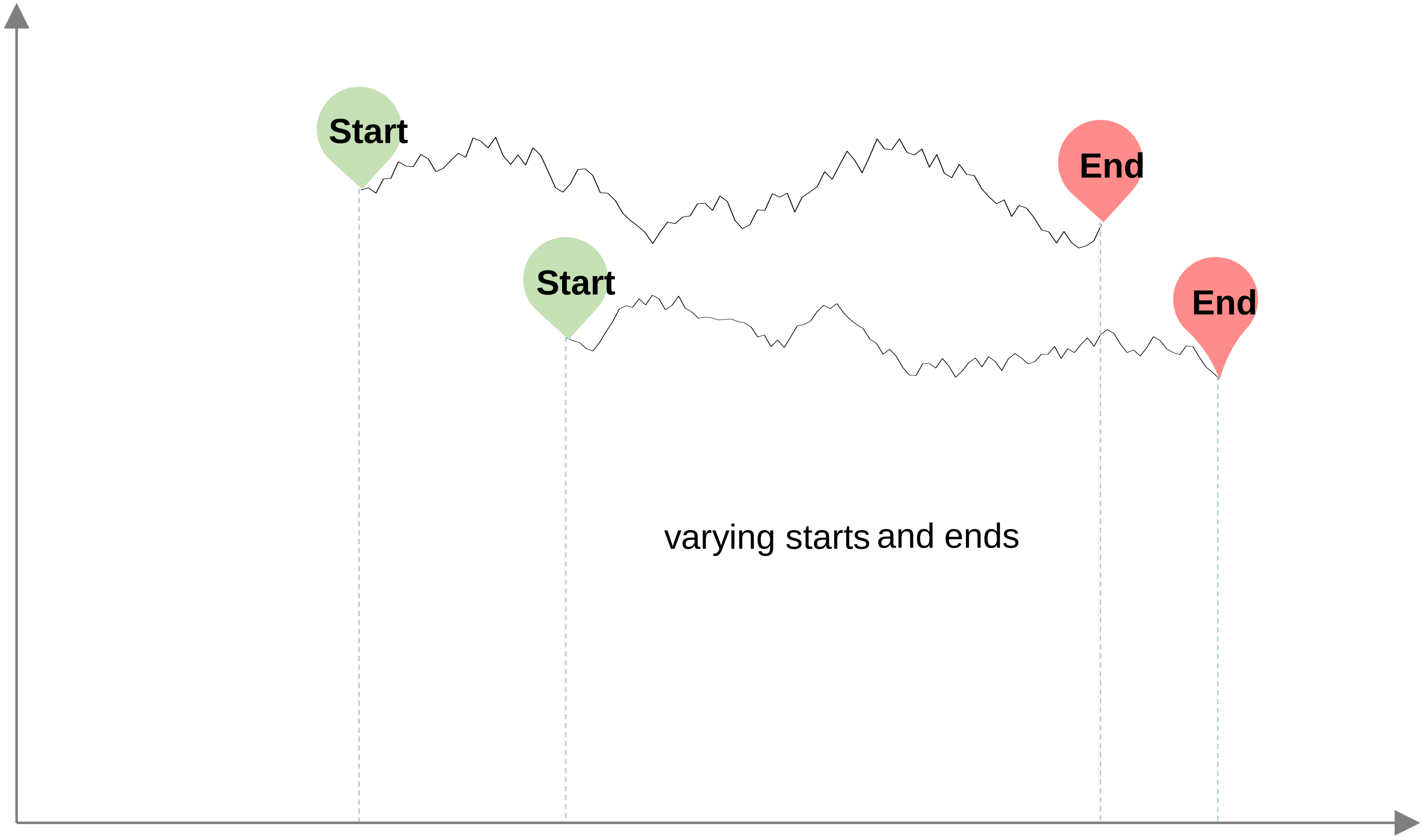
Time-Series Characteristics



The Data-Type Characteristics

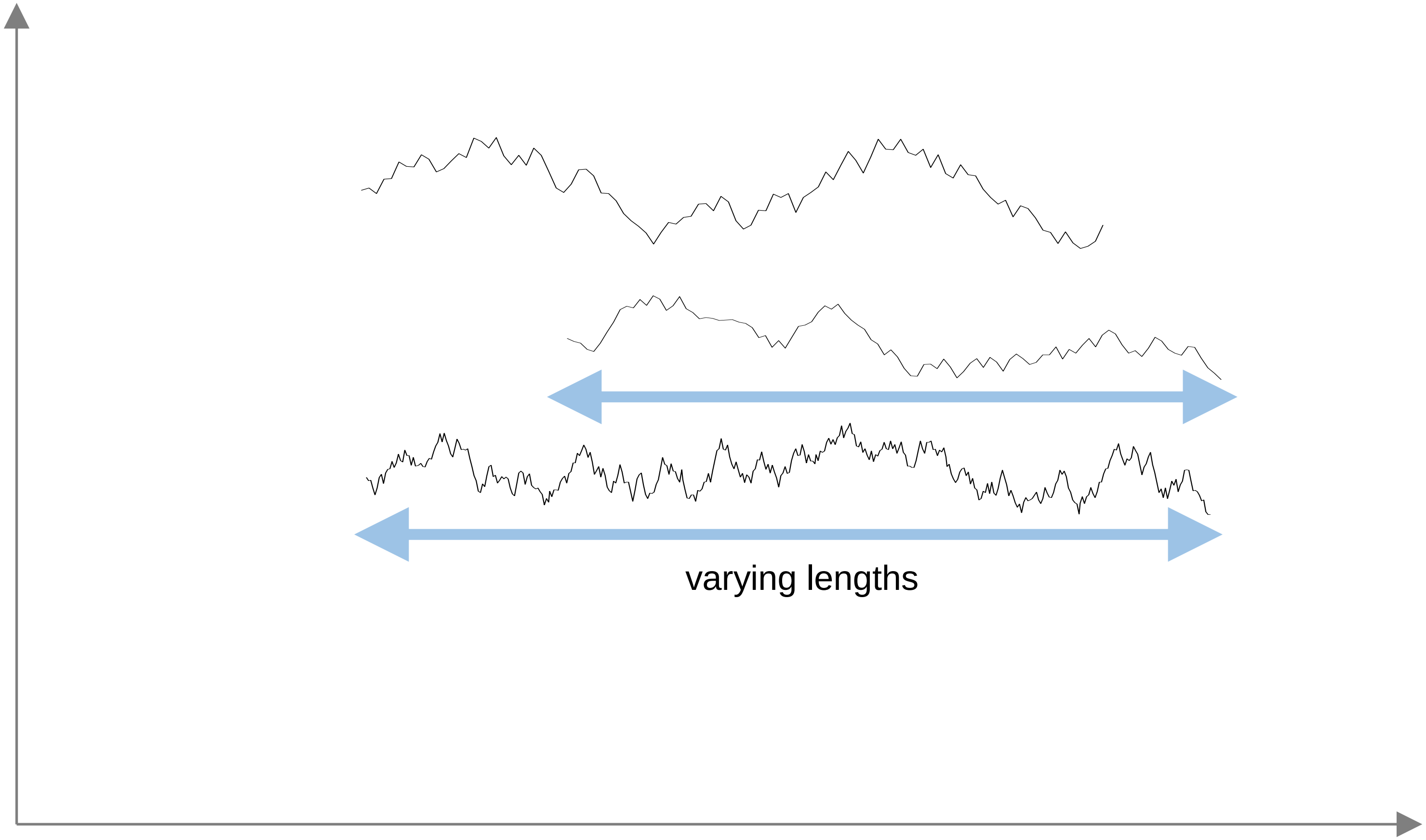
What are the properties of data series?

Time-Series Characteristics

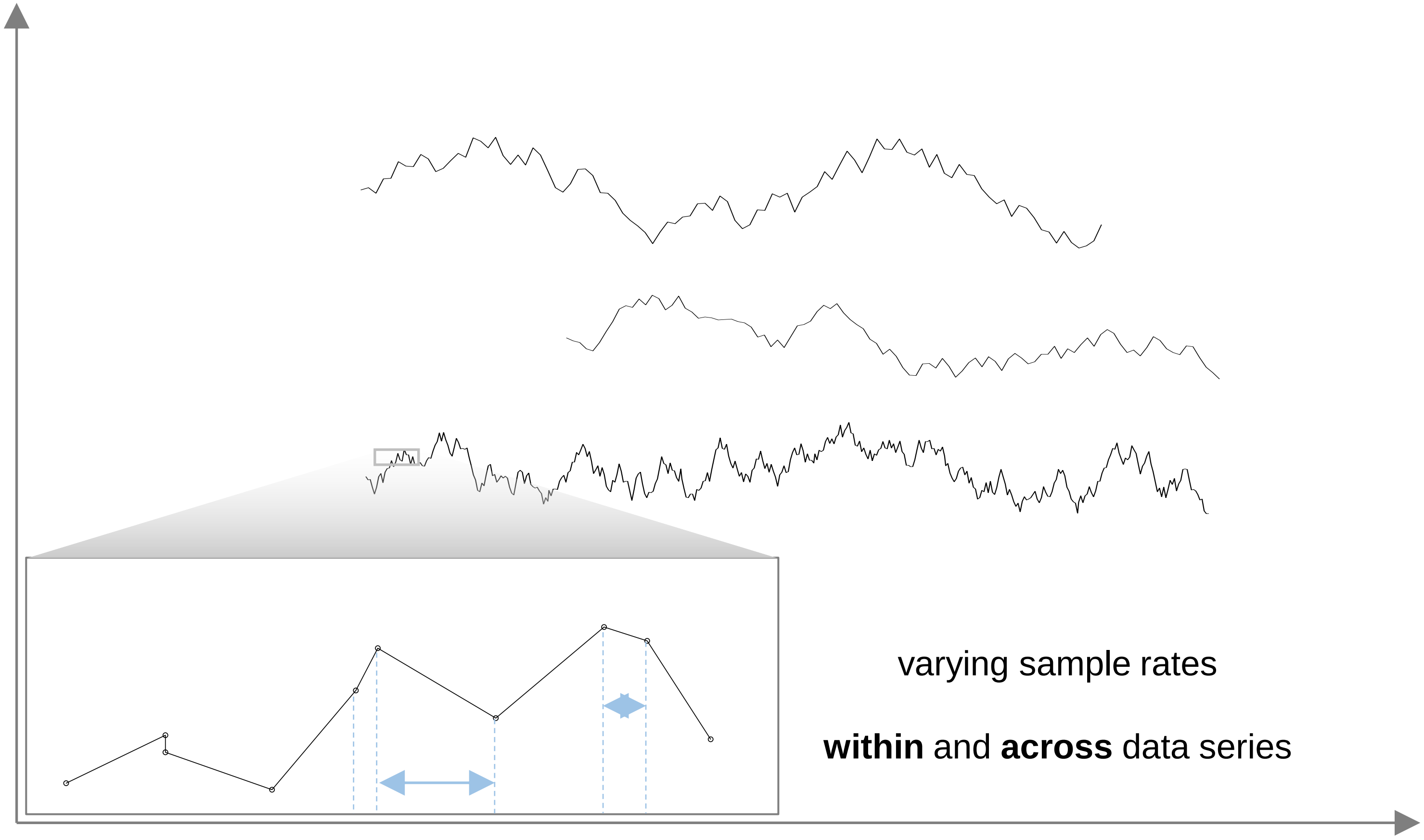


varying starts and ends

Time-Series Characteristics



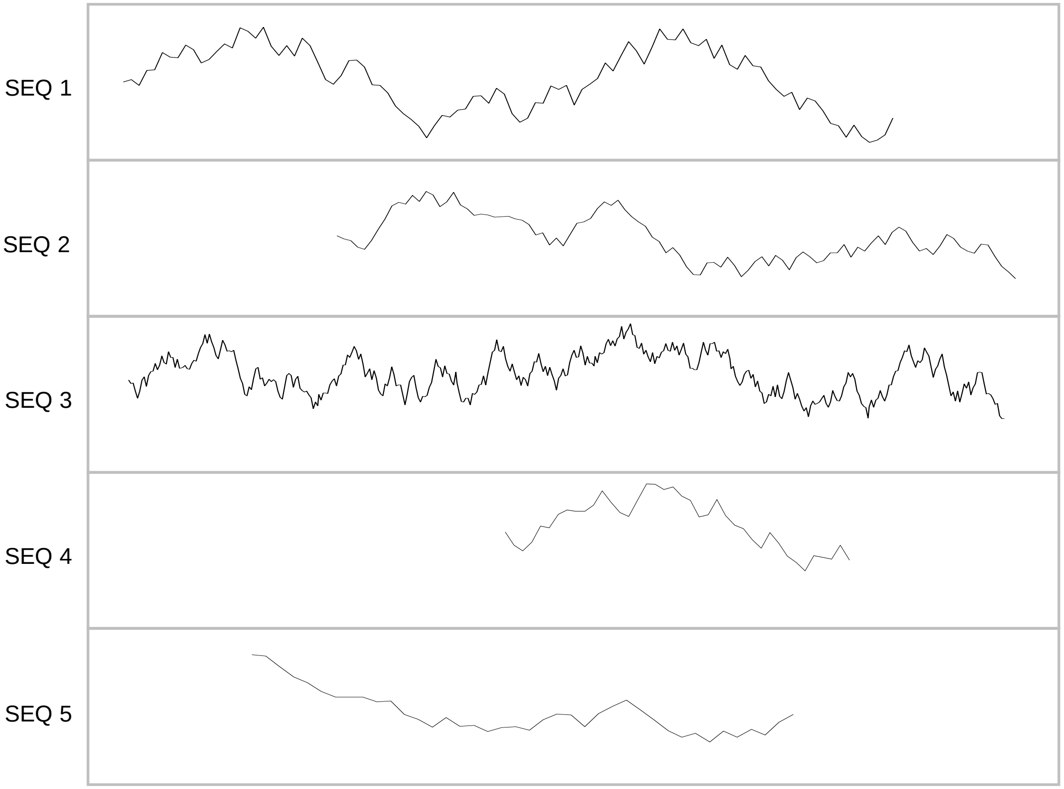
Time-Series Characteristics



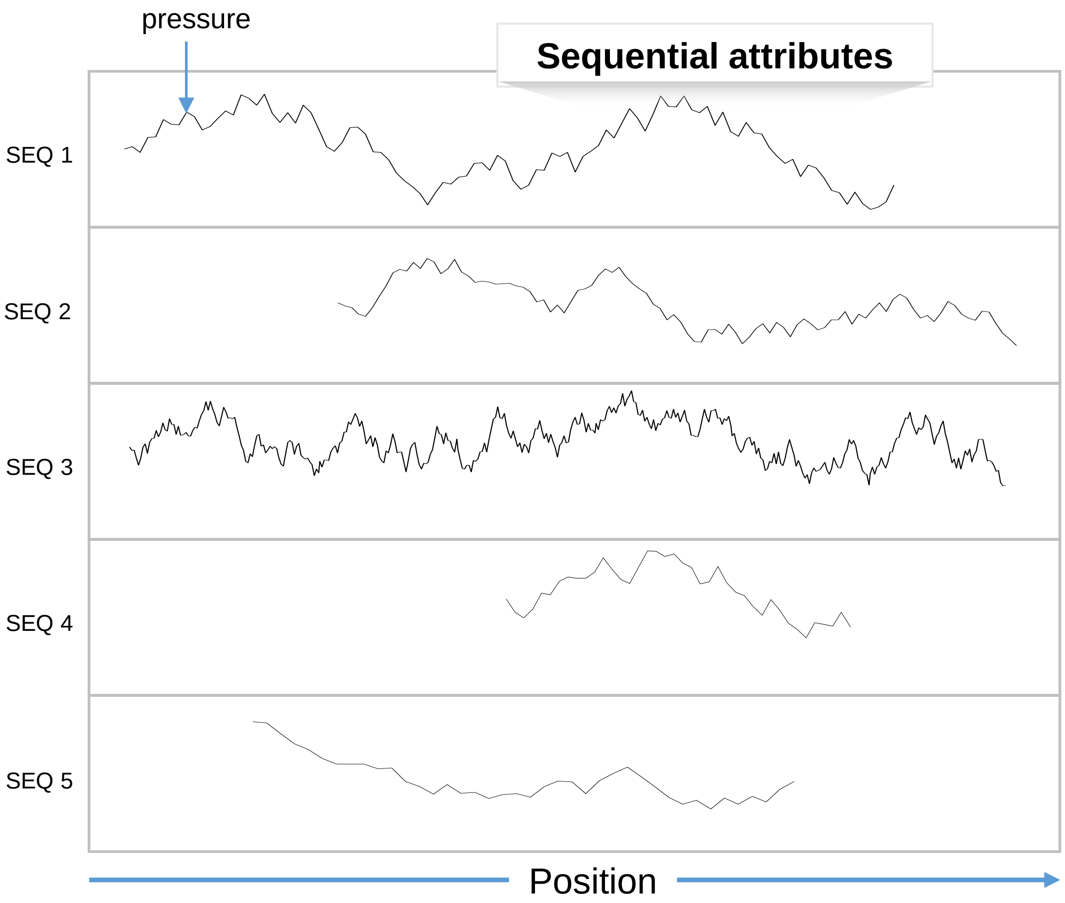
varying sample rates

within and **across** data series

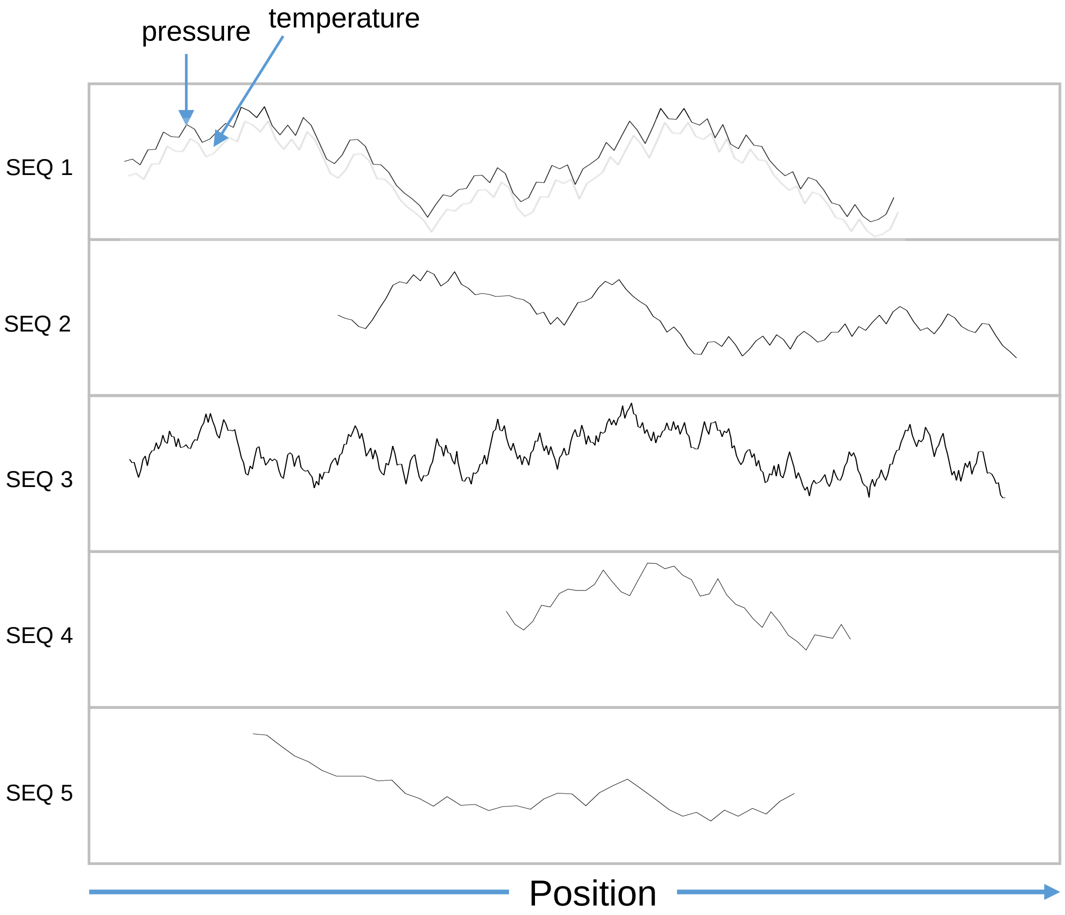
Time-Series Characteristics



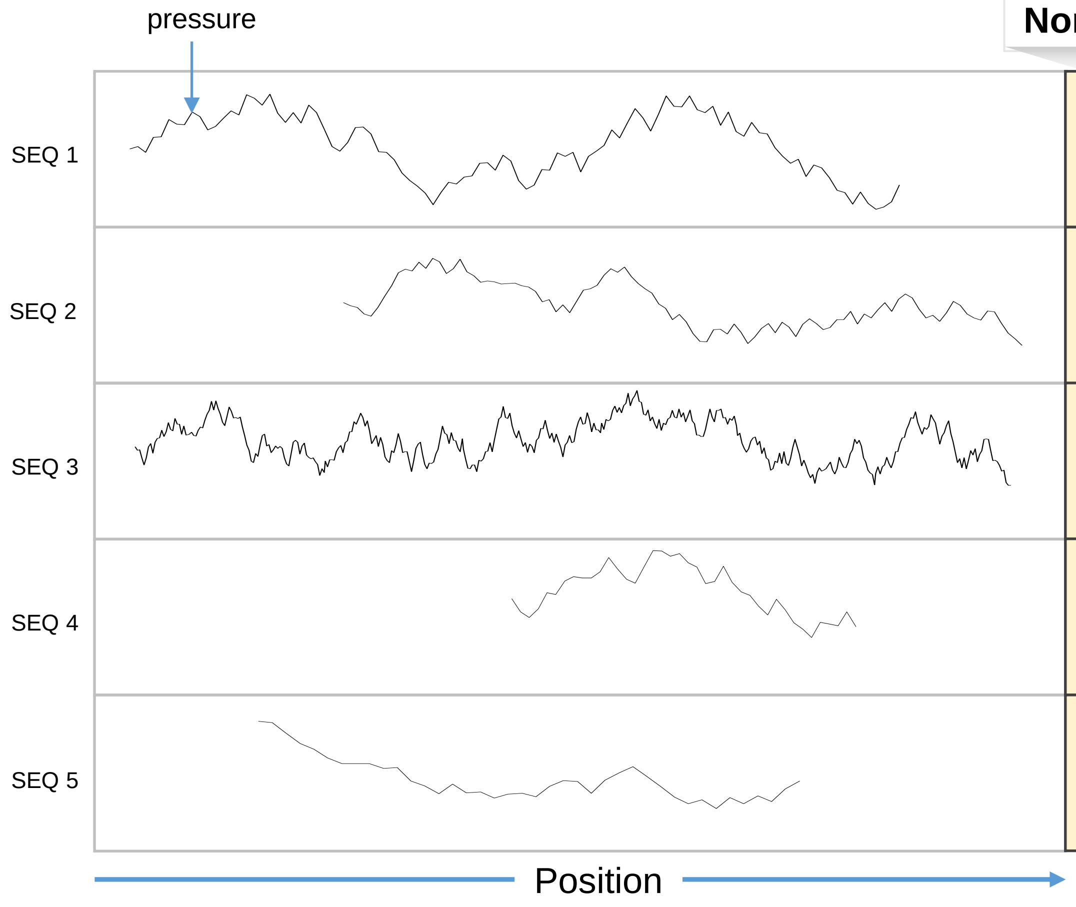
Time-Series Characteristics



Time-Series Characteristics



Time-Series Characteristics



Non-sequential attributes

seq_id: 1
sensor: Sensor 1
country: Italy
city: Asiago
province: Veneto

seq_id: 2
sensor: Sensor 2
country: Italy
city: Merate
province: Lombardy

seq_id: 3
sensor: Sensor 3
country: USA
city: Cambridge
province: MA

seq_id: 4
sensor: Sensor 4
country: USA
city: Delaware
province: OH

seq_id: 5
sensor: Sensor 5
country: Canada
city: Toronto
province: Ontario

Query Types

The Types of Queries

Simple

Selection-Projection-Transformation

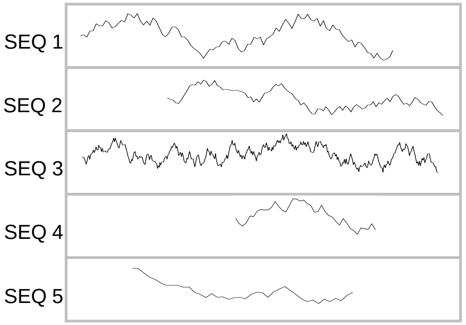
Complex

Analytical/Mining Queries

Query Types

Simple

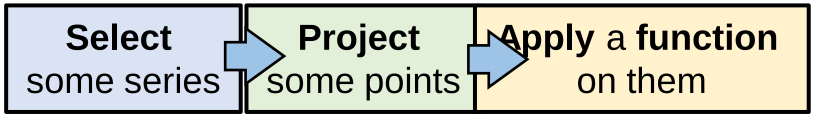
Selection-Projection-Transformation



Query Types

Simple

Selection-Projection-Transformation



Select some series

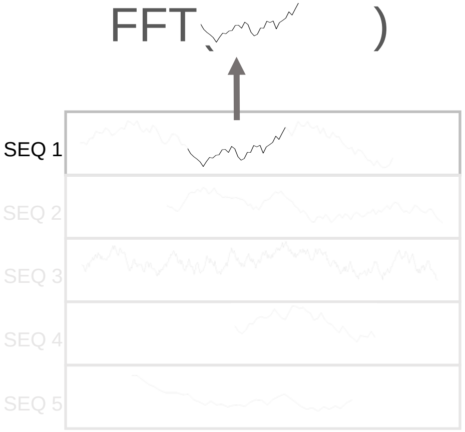
Project some points

Apply a **function** on them

Query Type 1: Find all points of a subset of data series
*e.g., Bring me the **whole history** of "pressure" for "Sensor 1"*

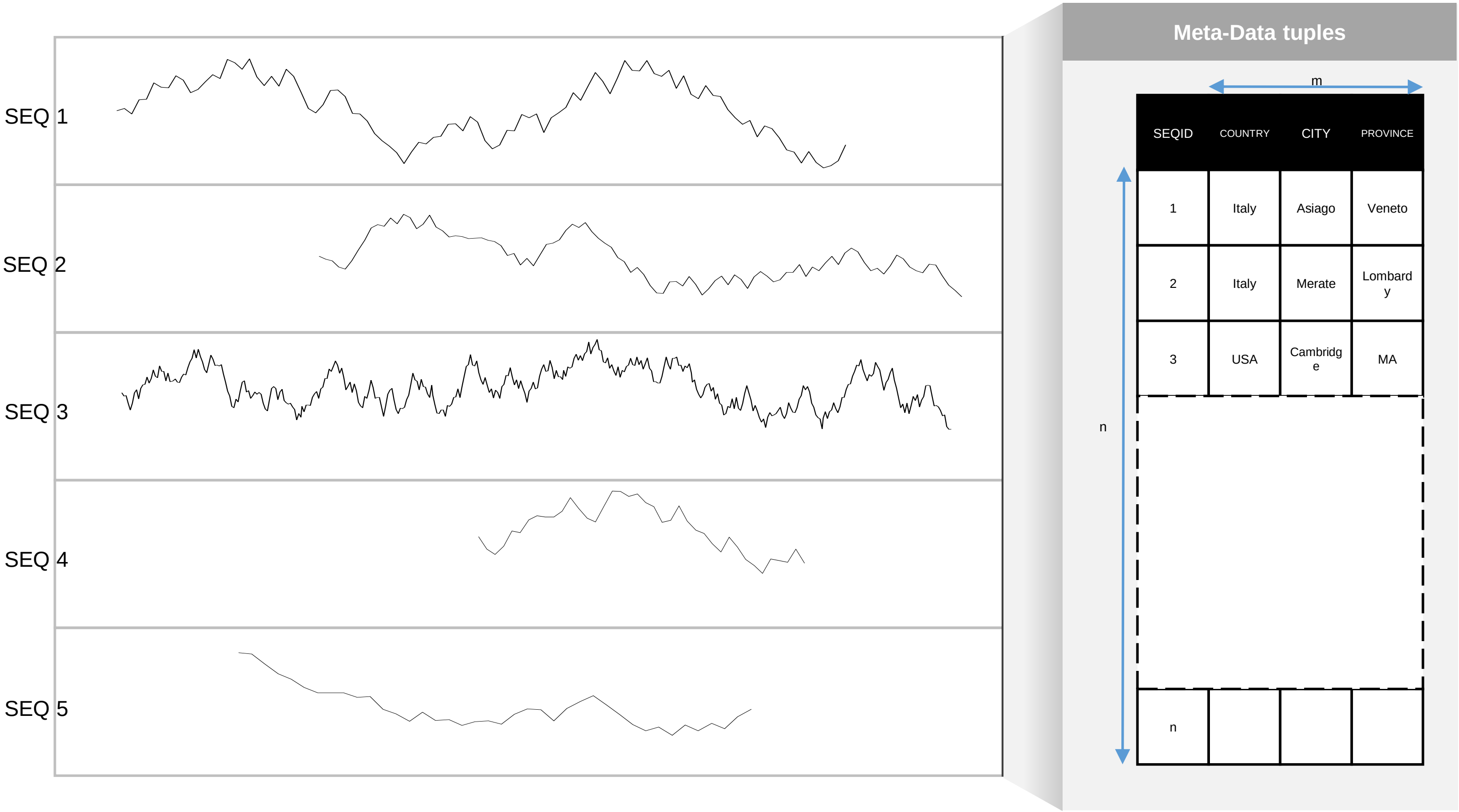
Query Type 2: Look at the points at a subset of the positions
*e.g., Compute the **average** pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points **based on a value**
*e.g., Bring me **all pressure** values above a **threshold***



Storage

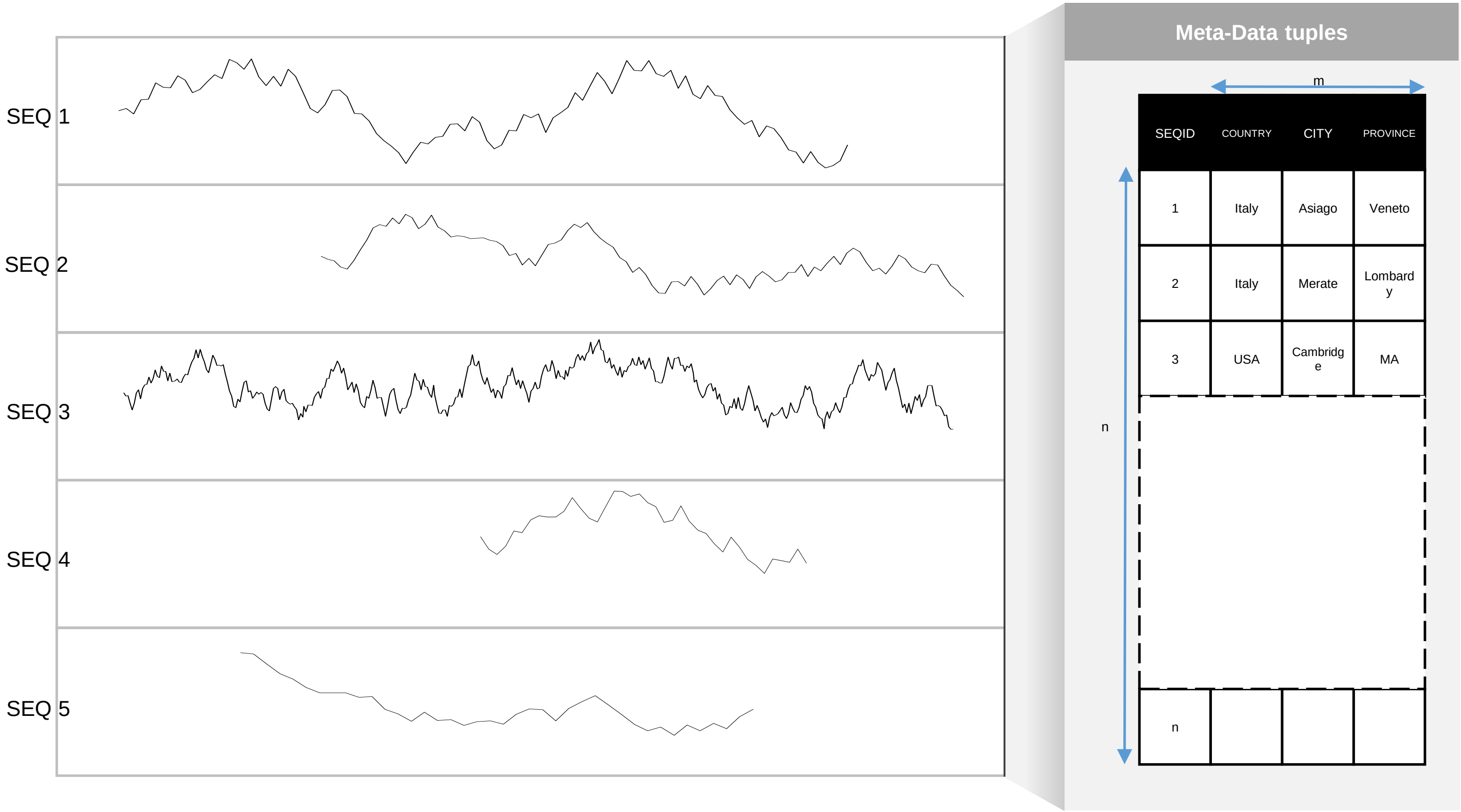
Storing meta-data



Meta-Data tuples

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2	Italy	Merate	Lombardy
3	USA	Cambridge	MA
n			

Storage



SEQ 1

SEQ 2

SEQ 3

SEQ 4

SEQ 5

Meta-Data tuples

m

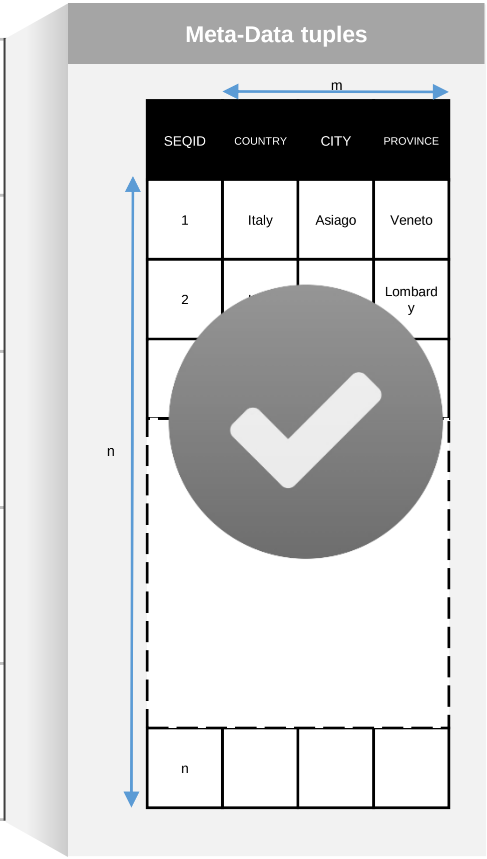
n

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2	Italy	Merate	Lombardy
3	USA	Cambridge	MA
n			

Storage

Storing meta-data

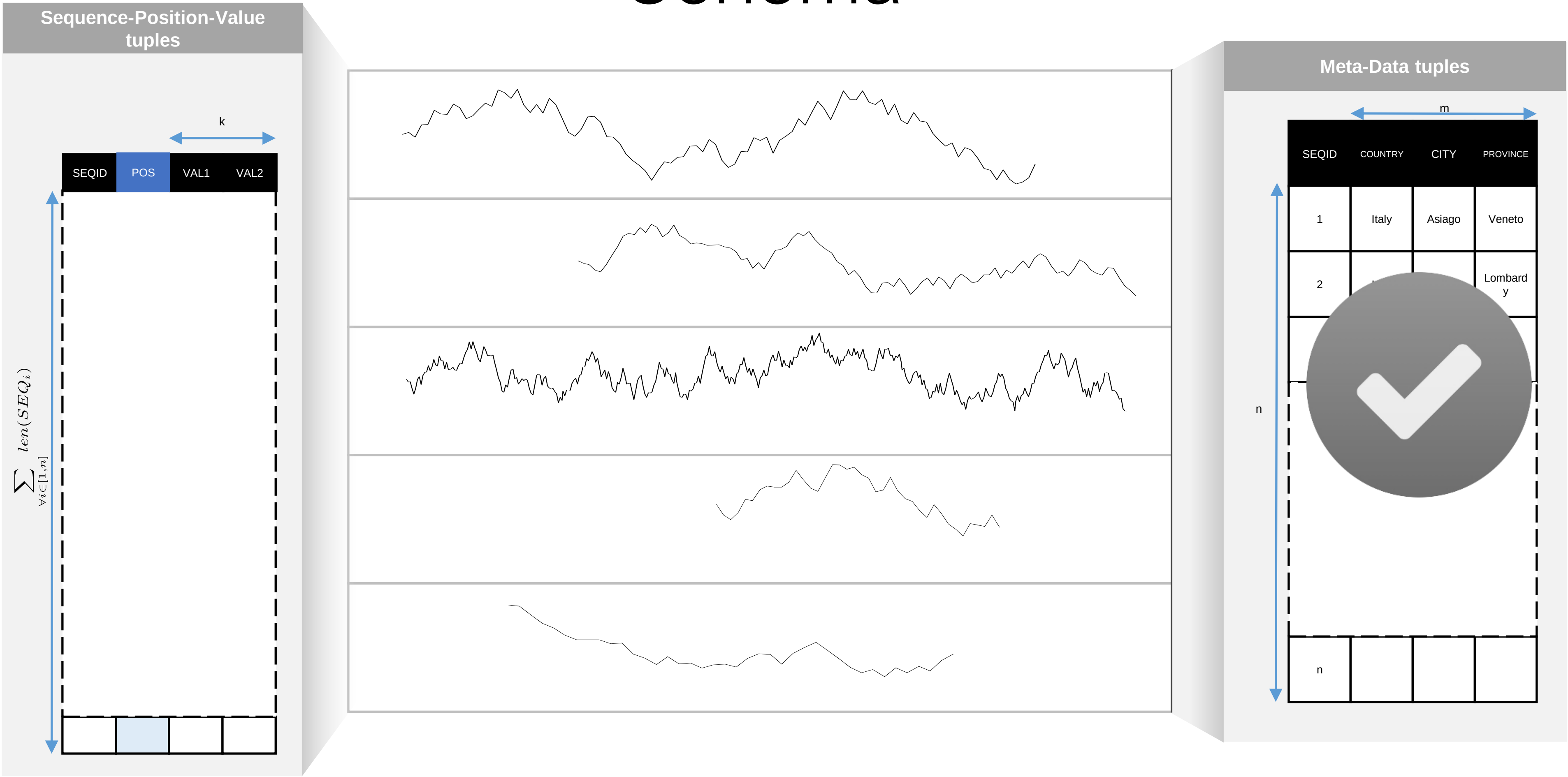
SEQ 1
SEQ 2
SEQ 3
SEQ 4
SEQ 5



Meta-Data tuples

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombard y
n			

Schema



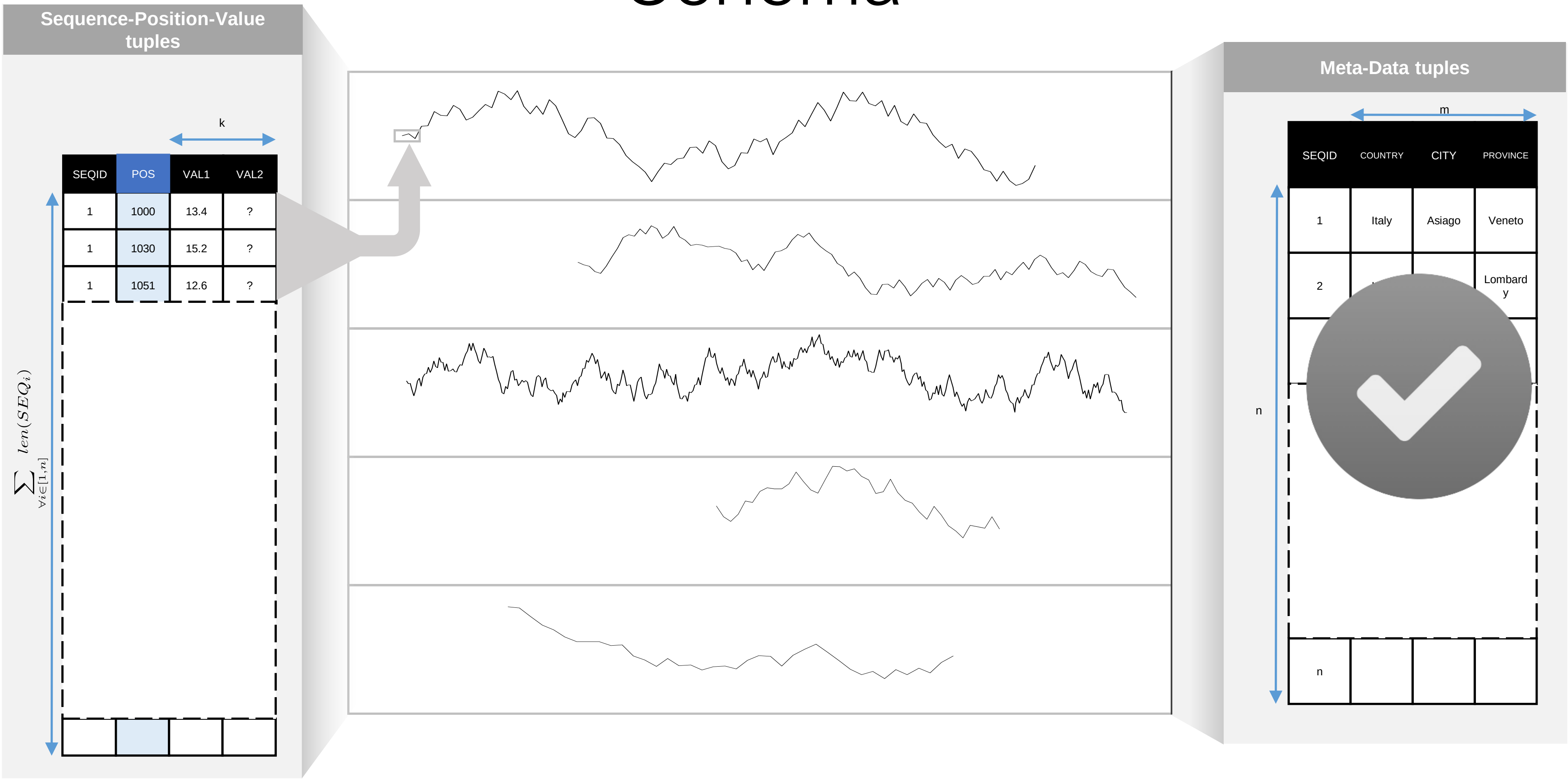
Sequence-Position-Value tuples

SEQID	POS	VAL1	VAL2

Meta-Data tuples

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombardy
n			

Schema



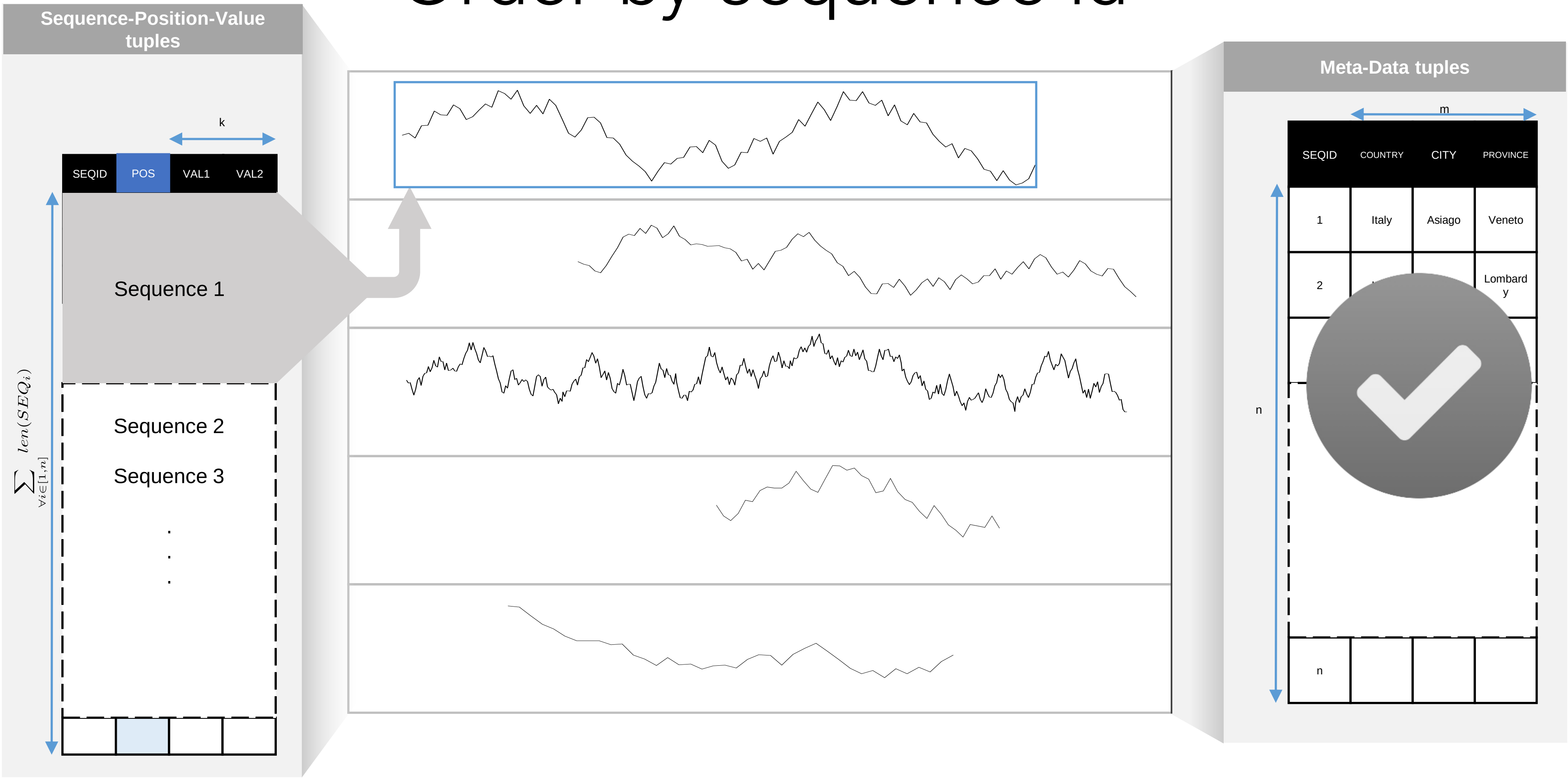
Sequence-Position-Value tuples

SEQID	POS	VAL1	VAL2
1	1000	13.4	?
1	1030	15.2	?
1	1051	12.6	?

Meta-Data tuples

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombard y
n			

Order by sequence id

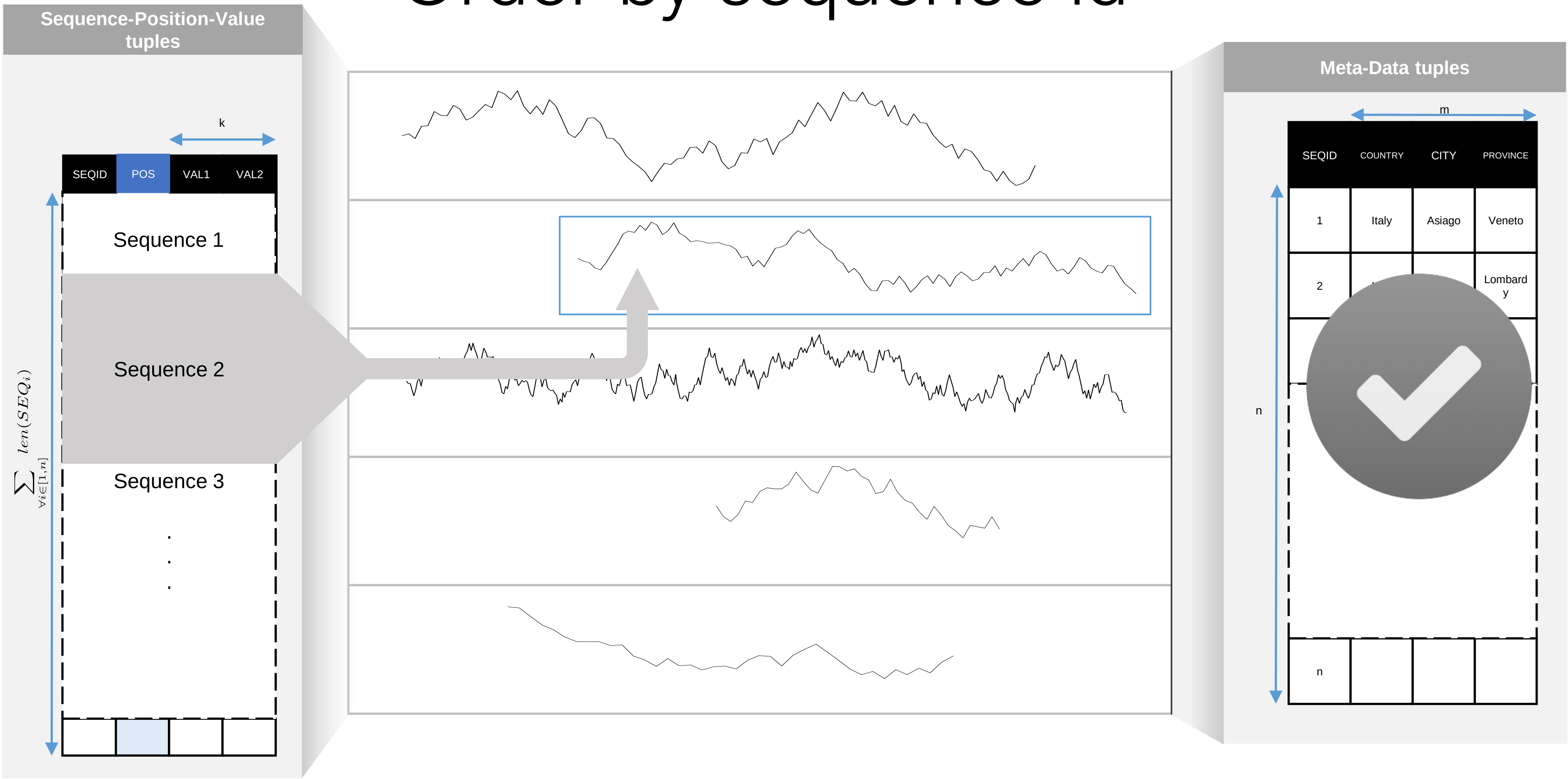


Sequence-Position-Value tuples
k
$\sum_{\forall i \in [1,n]} len(SEQ_i)$
Meta-Data tuples
m
n

SEQID	POS	VAL1	VAL2
Sequence 1			
Sequence 2			
Sequence 3			
. . .			

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombardy
n			

Order by sequence id



Sequence-Position-Value tuples

k

SEQID	POS	VAL1	VAL2

Sequence 1

Sequence 2

Sequence 3

. . .

$\sum_{\forall i\in[1,n]} len(SEQ_i)$

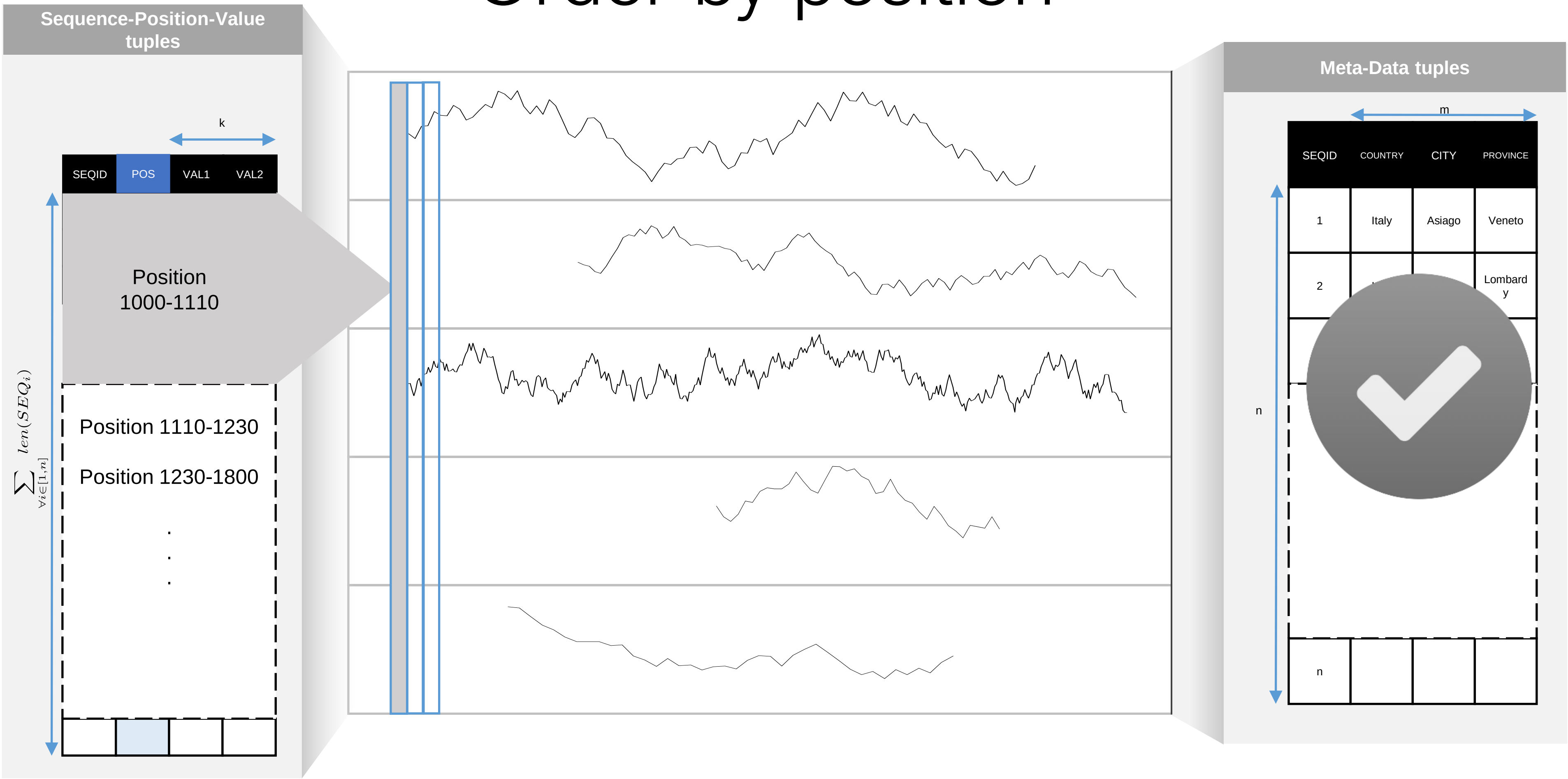
Meta-Data tuples

m

n

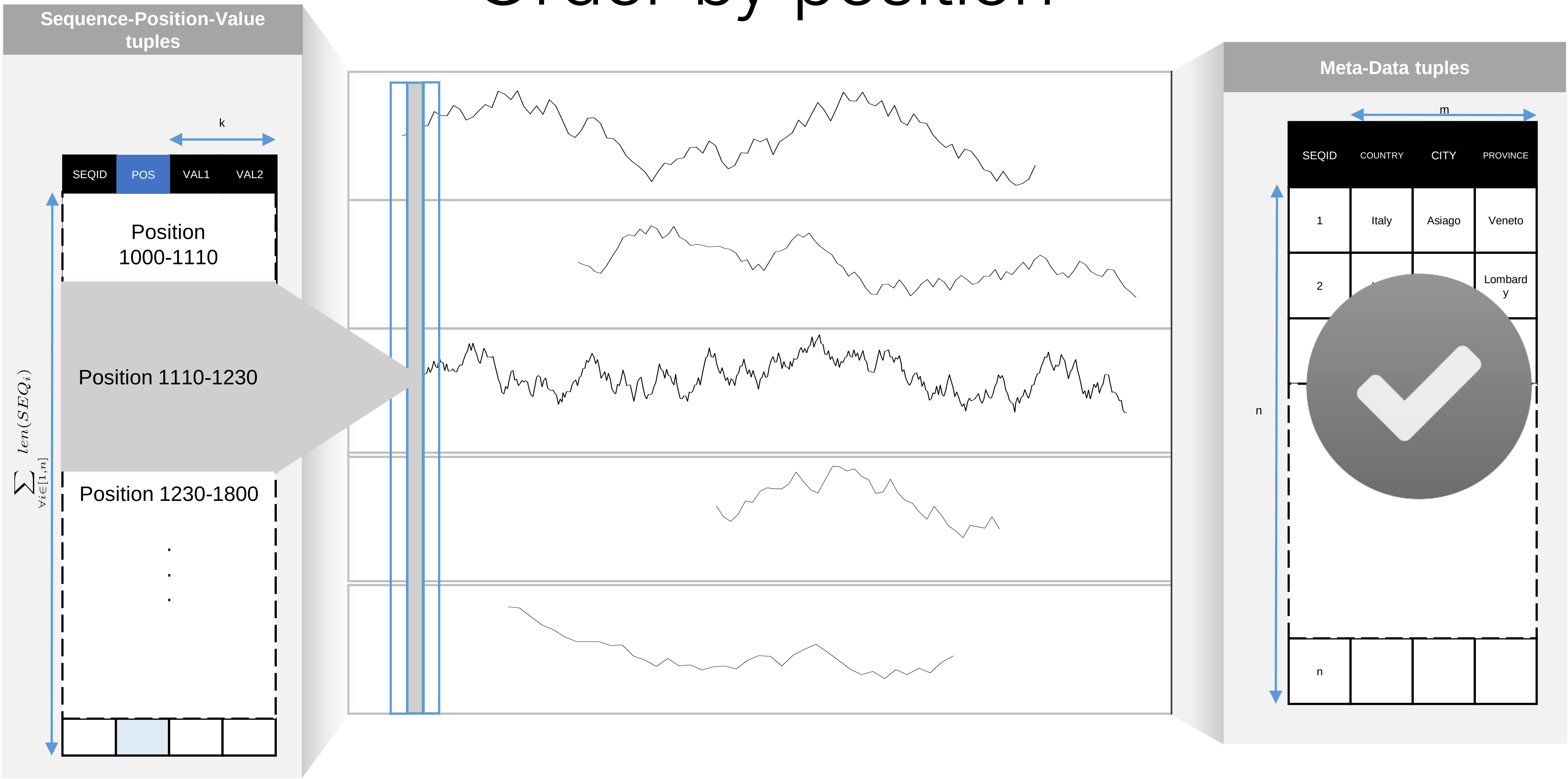
SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombard y
n			

Order by position



SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombardy
n			

Order by position



Sequence-Position-Value tuples

k

SEQID	POS	VAL1	VAL2

Position 1000-1110

Position 1110-1230

Position 1230-1800

.
.
.

$\sum_{\forall i \in [1,n]} len(SEQ_i)$

Meta-Data tuples

m

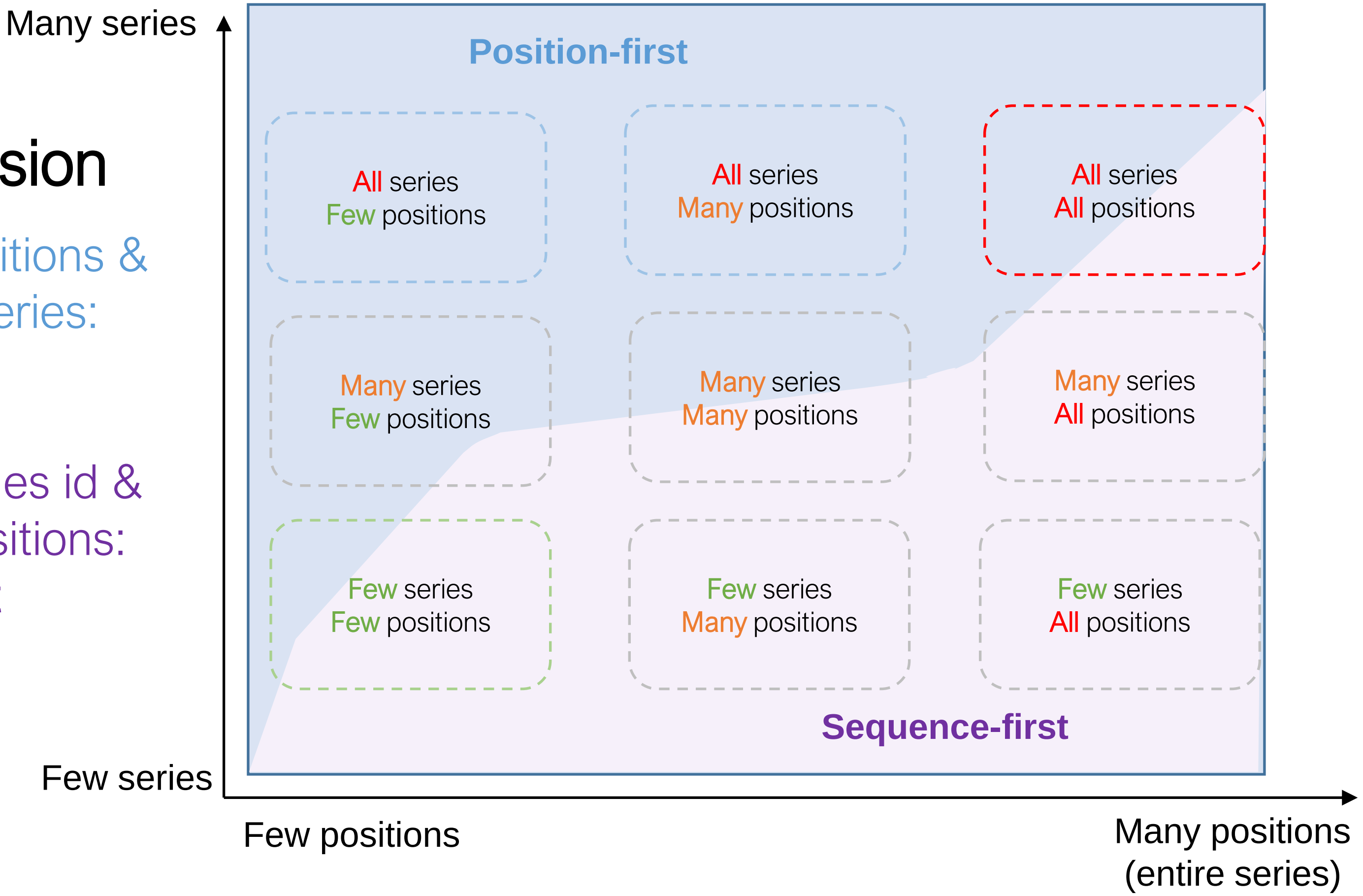
n

SEQID	COUNTRY	CITY	PROVINCE
1	Italy	Asiago	Veneto
2			Lombard y
n			

Simple Conclusion

Heavy filtering on positions & Accessing lots of series: **position-first**

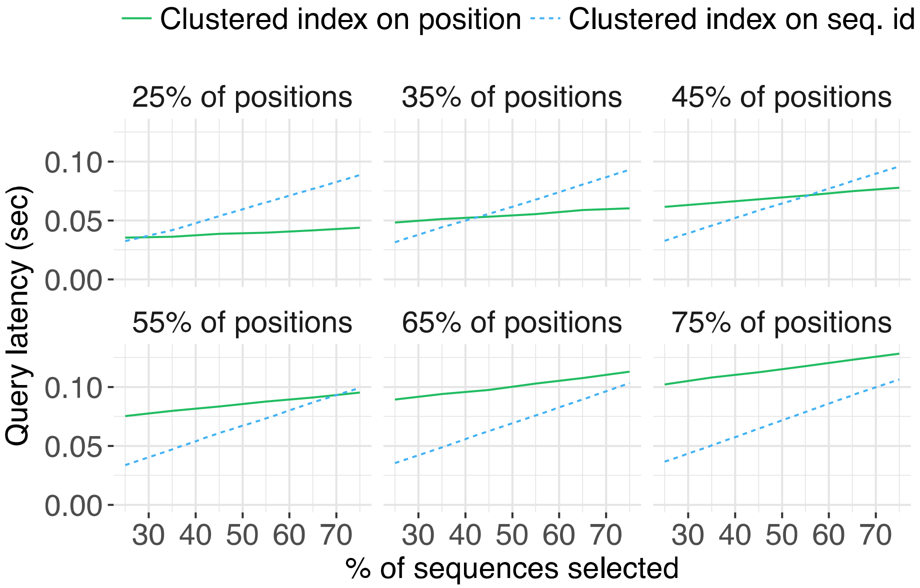
Heavy filtering on series id & accessing lots of positions: **sequence-first**



Simple Conclusion

Heavy filtering on positions & Accessing lots of series: **position-first**

Heavy filtering on series id & accessing lots of positions: **sequence-first**



*DBMS-X

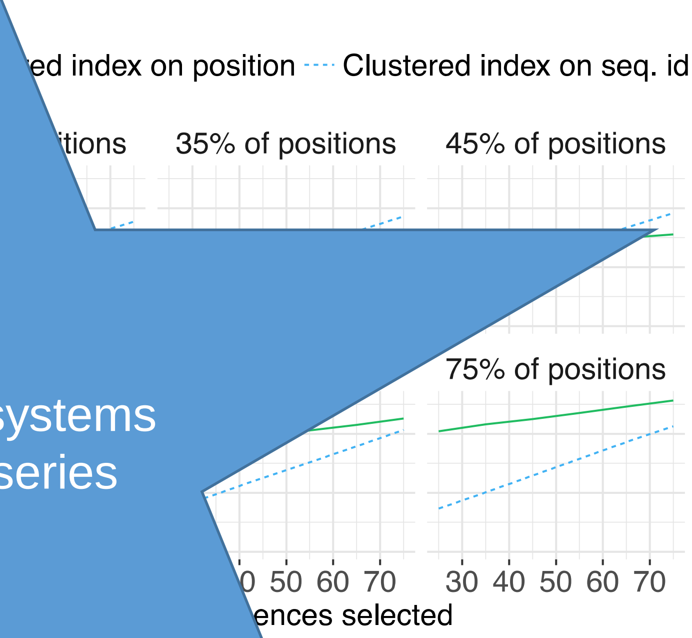
Simple Conclusion

Heavy filtering on positions & Accessing

position

Heavy filtering on series id

accessing lots of positions: **sequence-first**



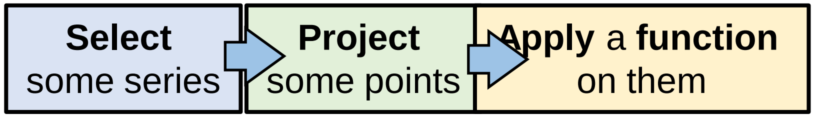
Most existing systems sort data by series

*DBMS-X

Query Types

Simple

Selection-Projection-Transformation



Select some series

Project some points

Apply a **function** on them

Query Type 1: Find all points of a subset of data series
*e.g., Bring me the **whole history** of "pressure" for "Sensor 1"*

Query Type 2: Look at the points at a subset of the positions
*e.g., Compute the **average** pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points **based on a value**
*e.g., Bring me **all pressure** values above a **threshold***

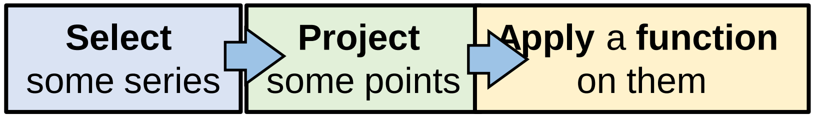
Query Types

Classic **1/n-dimensional indexes & layouts** for **point** and **range queries:**

Point get: Get seq id = 1
Range: Get positions 10 - 100

Simple

Selection-Projection-Transformation


Select some series
Project some points
Apply a **function** on them

Query Type 1: Find all points of a subset of data series
*e.g., Bring me the **whole history** of "pressure" for "Sensor 1"*

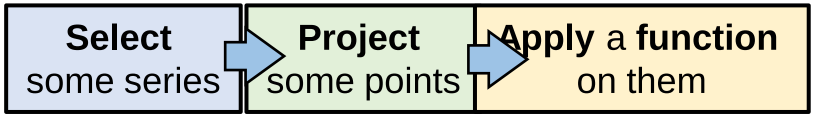
Query Type 2: Look at the points at a subset of the positions
*e.g., Compute the **average** pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points **based on a value**
*e.g., Bring me **all pressure** values above a **threshold***

Query Types

Simple

Selection-Projection-Transformation



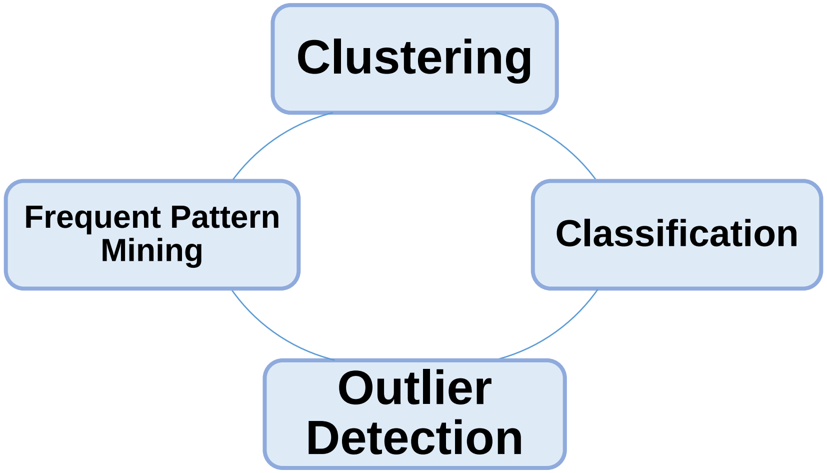
Query Type 1: Find all points of a subset of data series
*e.g., Bring me the **whole history** of "pressure" for "Sensor 1"*

Query Type 2: Look at the points at a subset of the positions
*e.g., Compute the **average** pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points **based on a value**
*e.g., Bring me **all pressure** values above a **threshold***

Complex

Analytical/Mining Queries



Query Types

Simple

Selection-Projection-Transformation

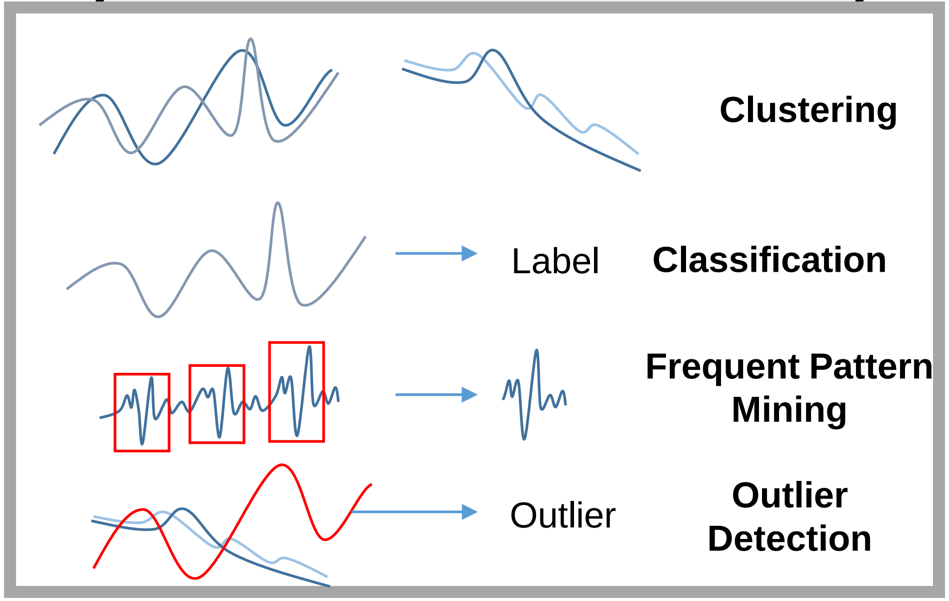
Select some series → **Project** some points → **Apply** a **function** on them

Query Type 1: Find all points of a subset of data series
*e.g., Bring me the **whole history** of "pressure" for "Sensor 1"*

Query Type 2: Look at the points at a subset of the positions
*e.g., Compute the **average** pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points based on a value
*e.g., Bring me **all pressure** values above a **threshold***

Complex



Clustering

Classification

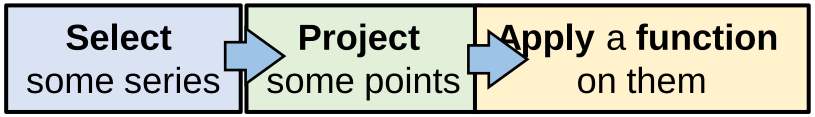
Frequent Pattern Mining

Outlier Detection

Query Types

Simple

Selection-Projection-Transformation



Select some series → **Project** some points → **Apply** a **function** on them

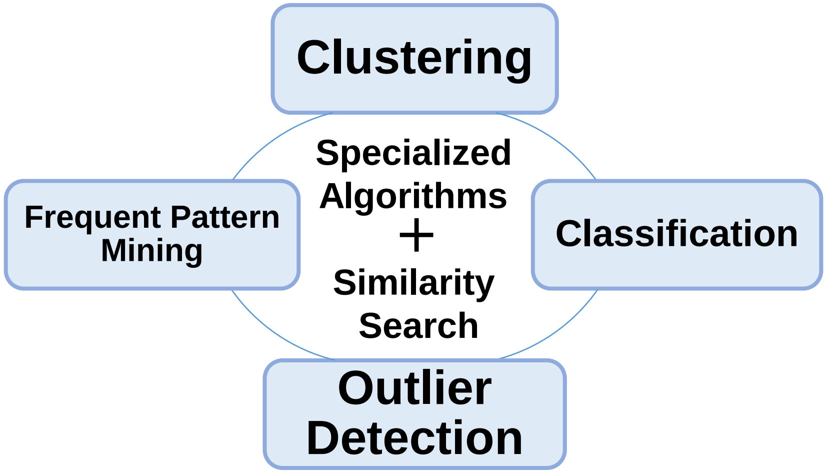
Query Type 1: Find all points of a subset of data series
*e.g., Bring me the **whole history** of "pressure" for "Sensor 1"*

Query Type 2: Look at the points at a subset of the positions
*e.g., Compute the **average** pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points based on a value
*e.g., Bring me **all pressure** values above a **threshold***

Complex

Analytical/Mining Queries



Clustering

Frequent Pattern Mining

Specialized Algorithms + Similarity Search

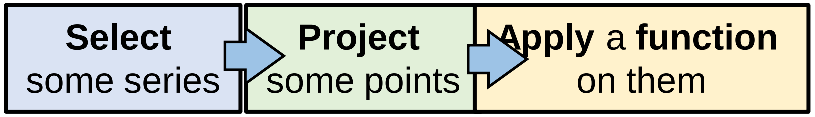
Classification

Outlier Detection

Query Types

Simple

Selection-Projection-Transformation



Select some series

Project some points

Apply a **function** on them

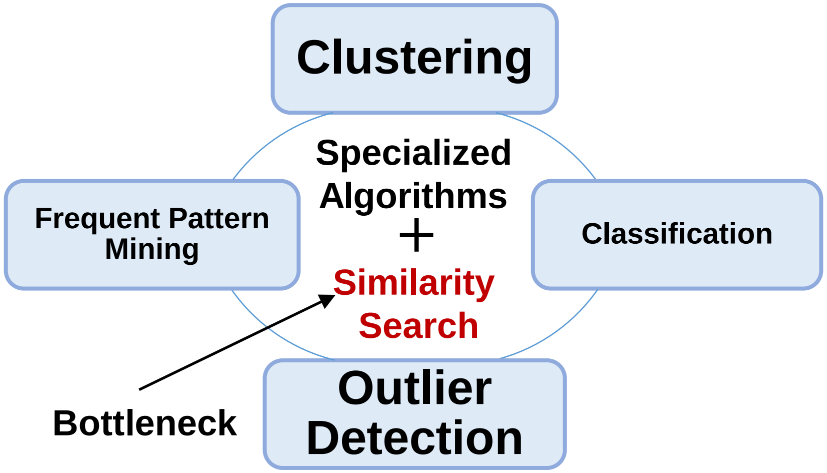
Query Type 1: Find all points of a subset of data series
e.g., Bring me the ***whole history*** *of "pressure" for "Sensor 1"*

Query Type 2: Look at the points at a subset of the positions
e.g., Compute the ***average*** *pressure for all sensors for the range of positions that cover the 2nd to the 12th of March.*

Query Type 3: Look at a subset of points **based on a value**
e.g., Bring me ***all pressure*** *values above a* ***threshold***

Complex

Analytical/Mining Queries



Clustering

Frequent Pattern Mining

Specialized Algorithms
+
Similarity Search

Classification

Bottleneck

Outlier Detection

Time-Series Management Systems

a few more details on the popular systems:

- InfluxDB
- TimescaleDB

InfluxDB

- Storage Engine:
  - **Log Structured Merge Tree: LSM-Tree** variant that expects data to arrive ordered by time and partitions them by distinct sequence. It then stores each series contiguously.

- Schema:
  - Tags and fields. Tags are used to describe meta-data and fields are used to store quantities that change over time.

- Queries
  - It supports group by (only on tags), join (on timestamps and fields), selections, projections, and aggregations.
  - It also supports continuous queries

TimescaleDB

- **Storage:** Uses PostgreSQL as the backend.
  - It partitions time-series into multiple tables, forming a single virtual entity called a **hypertable**.
  - It allows for the **compression** of data, something that Postgres does not do by default.

- **Schema:** Tables are **normal Postgres tables**, where one has to specify a time column in order to create a hypertable.

- **Queries: Full SQL support**, with the addition of custom time-series functions.
  - **Custom time-series operators:** first, last, histogram, interpolation, time bucketing, gap filling, etc.
  - It also supports **continuous queries**

Challenges and Open Problems

Massive Data Series Collections

Publications

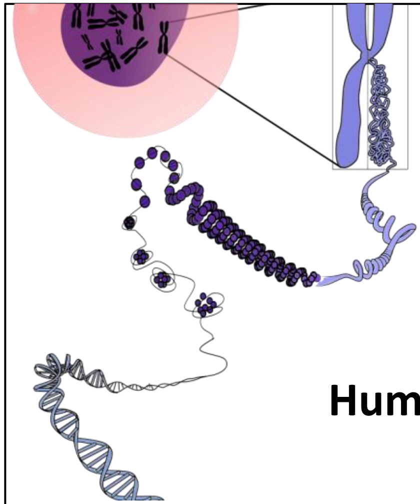
SIGREC'19

NASA's Solar Observatory

1.5 TB per day

Large Synoptic Survey Telescope (2019)

~30 TB per night



Human Genome project

130 TB

passenger aircrafts

20 TB per hour

data center and services monitoring

2B data series

4M points/sec

facebook

The Road Ahead

Publications

ICDE'18

HPCS'17

SIGREC'15

"enable practitioners and non-expert users to easily and efficiently manage and analyze massive data series collections"

Publications

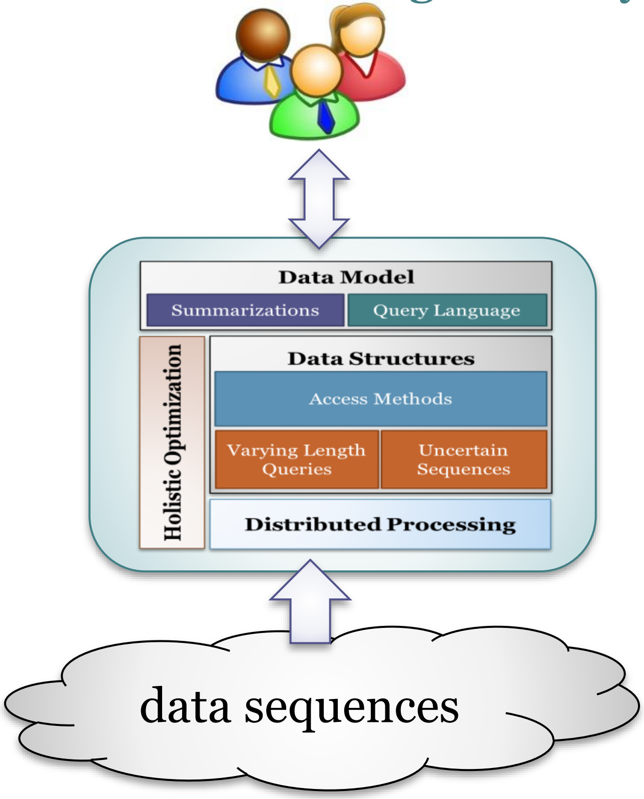
ICDE’18

HPCS’17

SIGREC’15

The Road Ahead

- Big Sequence Management System
  - general purpose data series management system

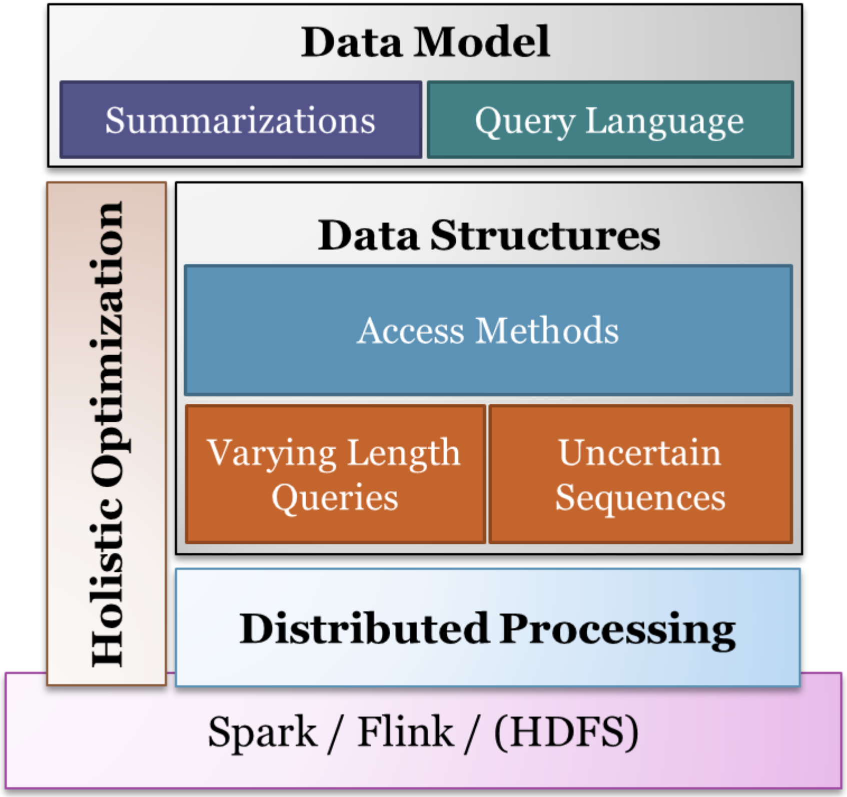


Publications

- ICDE'18
- HPCS'17
- SIGREC'15

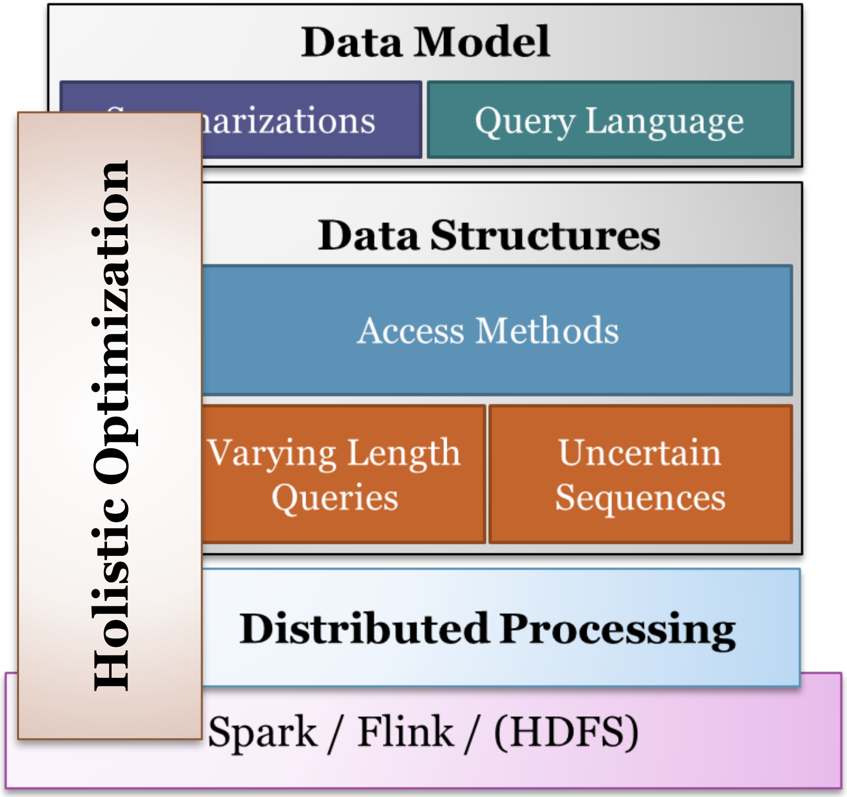
The Road Ahead

- Big Sequence Management System



The Road Ahead

- Big Sequence Management System



Data Model

Summarizations

Query Language

Data Structures

Access Methods

Varying Length Queries

Uncertain Sequences

Distributed Processing

Spark / Flink / (HDFS)

Holistic Optimization

Publications

- ICDE'18
- HPCS'17
- SIGREC'15
- PVLDB'19

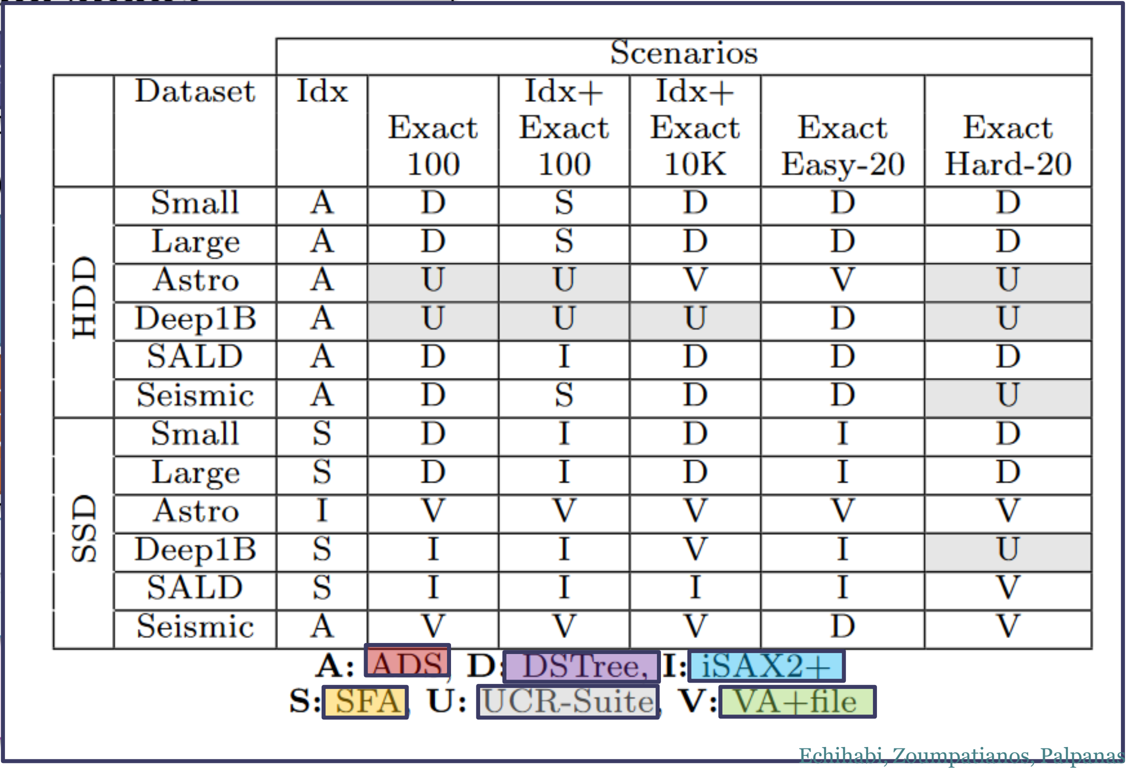
The Road Ahead

- Big Sequence Management System

Publications

- ICDE'18
- HPCS'17
- SIGREC'15
- PVLDB'19

Data Model
...marization
D
Varying
Quer
Distr
Spark /
Holistic Optimization



		Scenarios					
	Dataset	Idx	Exact 100	Idx+ Exact 100	Idx+ Exact 10K	Exact Easy-20	Exact Hard-20
HDD	Small	A	D	S	D	D	D
	Large	A	D	S	D	D	D
	Astro	A	U	U	V	V	U
	Deep1B	A	U	U	U	D	U
	SALD	A	D	I	D	D	D
	Seismic	A	D	S	D	D	U
SSD	Small	S	D	I	D	I	D
	Large	S	D	I	D	I	D
	Astro	I	V	V	V	V	V
	Deep1B	S	I	I	V	I	U
	SALD	S	I	I	I	I	V
	Seismic	A	V	V	V	D	V

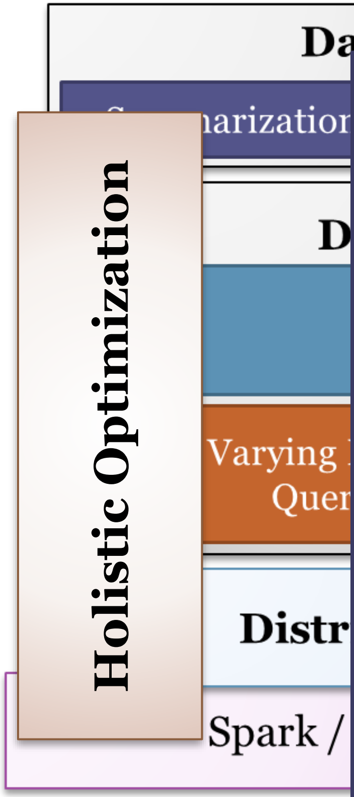
A: ADS, D: DSTree, I: iSAX2+, S: SFA, U: UCR-Suite, V: VA+file

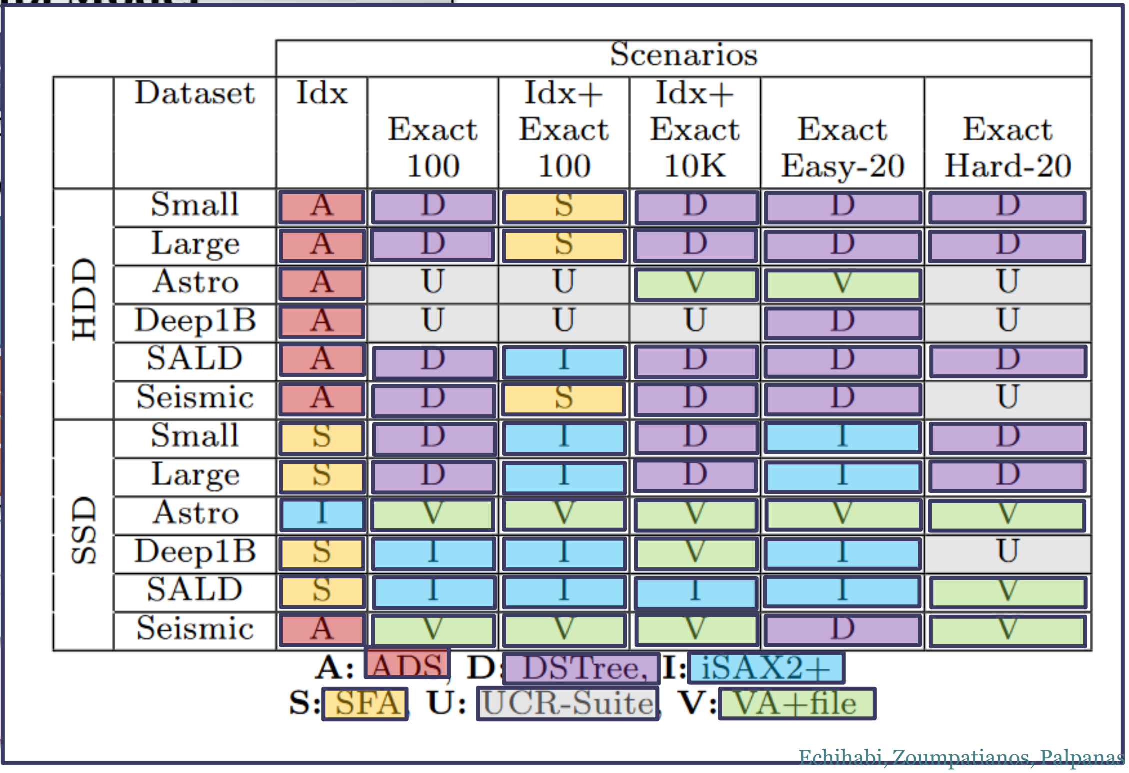
The Road Ahead

Publications

- ICDE'18
- HPCS'17
- SIGREC'15
- PVLDB'19

- Big Sequence Management System





		Scenarios					
	Dataset	Idx	Exact 100	Idx+ Exact 100	Idx+ Exact 10K	Exact Easy-20	Exact Hard-20
HDD	Small	A	D	S	D	D	D
	Large	A	D	S	D	D	D
	Astro	A	U	U	V	V	U
	Deep1B	A	U	U	U	D	U
	SALD	A	D	I	D	D	D
	Seismic	A	D	S	D	D	U
SSD	Small	S	D	I	D	I	D
	Large	S	D	I	D	I	D
	Astro	I	V	V	V	V	V
	Deep1B	S	I	I	V	I	U
	SALD	S	I	I	I	I	V
	Seismic	A	V	V	V	D	V

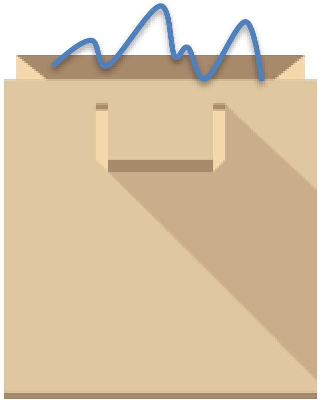
A: ADS, D: DSTree, I: iSAX2+,
S: SFA, U: UCR-Suite, V: VA+file

Benchmarking Data Series Indexes?

Previous Studies

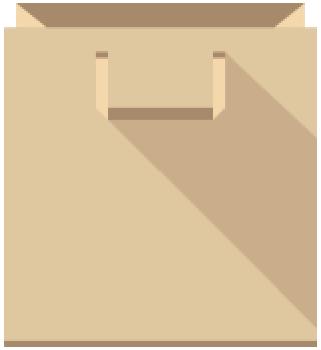
evaluate **performance** of **indexing methods** using **random queries**

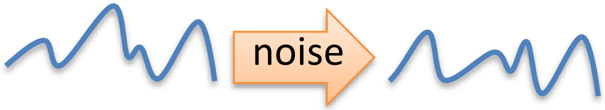
- chosen from the data (with/without noise)



Previous Studies

With or without noise





Problem with Random Queries

No control on their ***characteristics***

We **cannot properly evaluate** summarizations and indexes

We need queries that cover the entire range from easy to hard

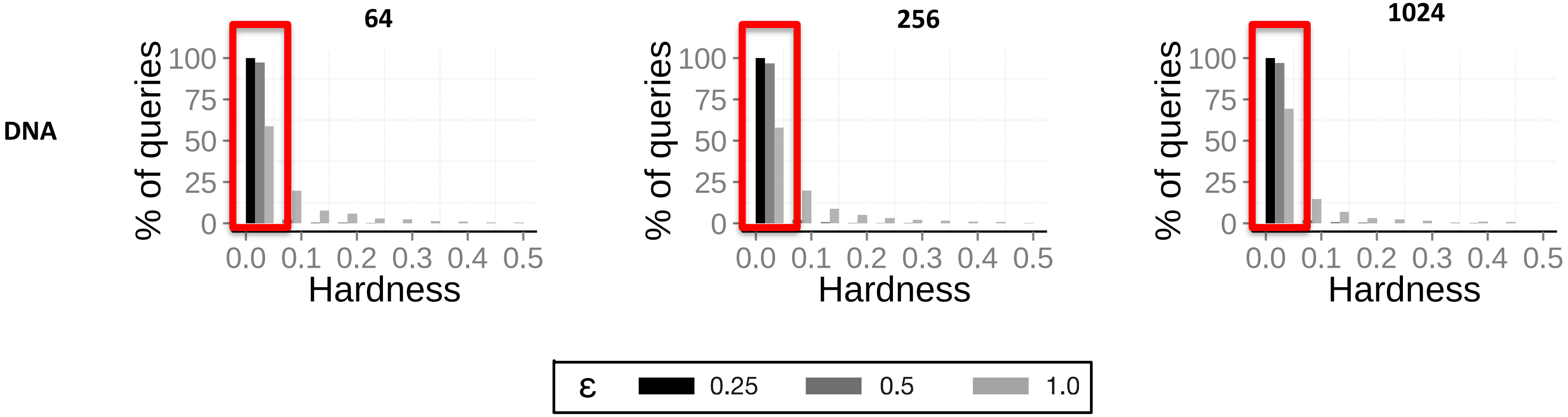
Publications

KDD '15

TKDE '18

Previous Workloads

Most previous workloads are ***skewed*** to ***easy*** queries



Echihabi, Zoumpatianos, Palpanas - ISCC 2020

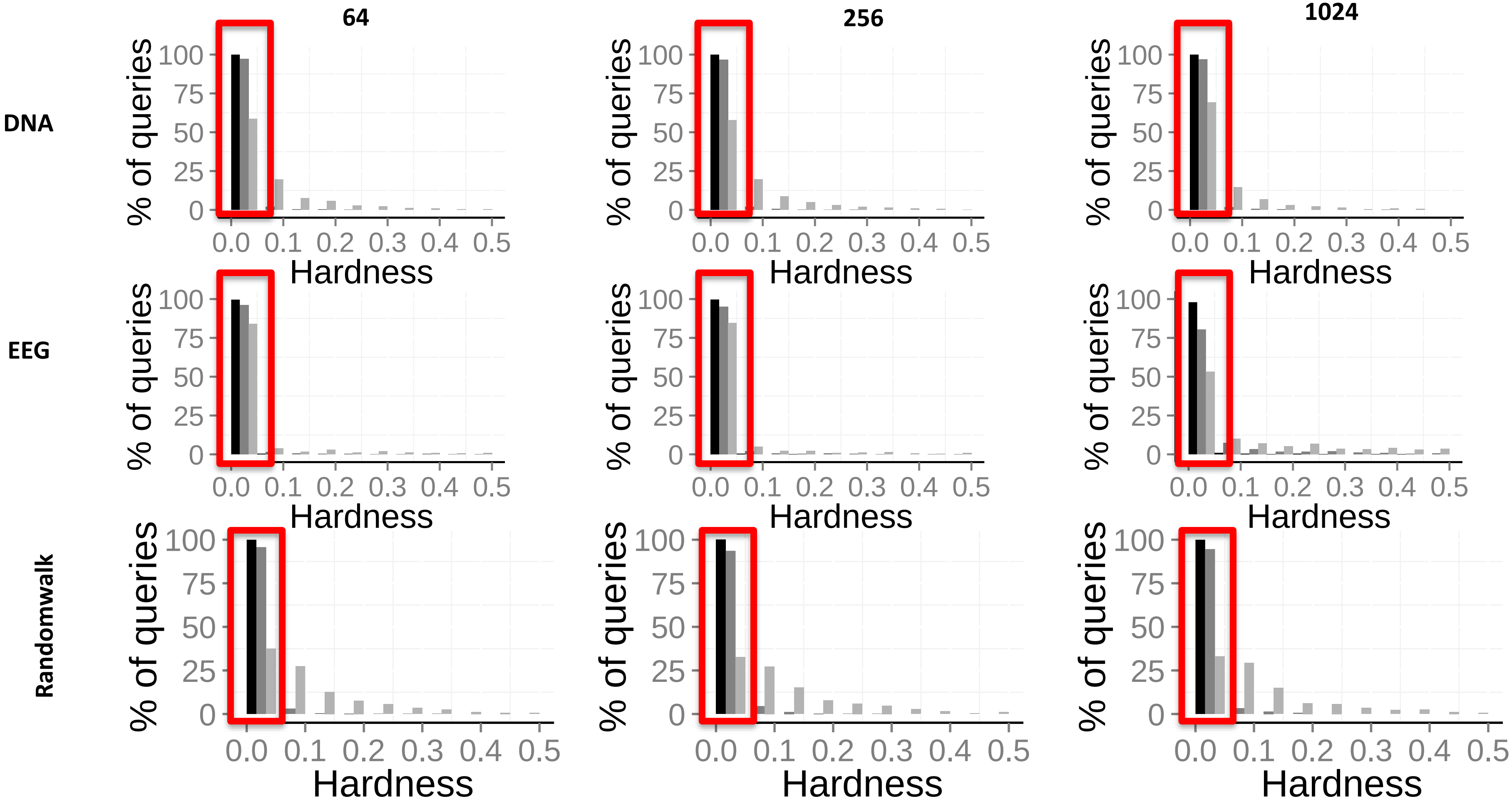
Publications

KDD ‘15

TKDE ‘18

Previous Workloads

Most previous workloads are ***skewed*** to ***easy*** queries



Echihabi, Zoumpatianos, Palpanas - ISCC 2020

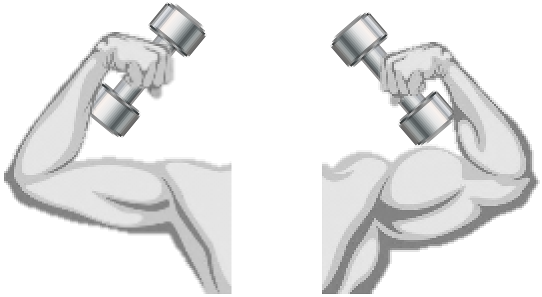
Publications

KDD '15

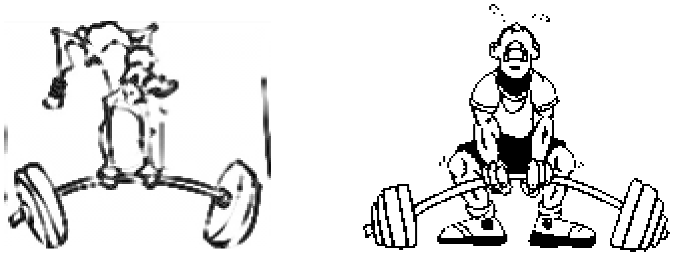
TKDE '18

Benchmark Workloads

If all queries are **easy**
all indexes look **good**

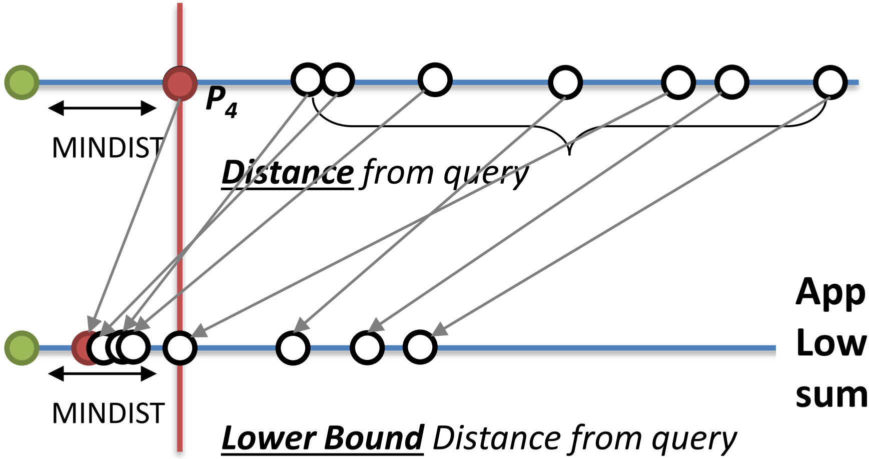


If all queries are **hard**
all indexes look **bad**



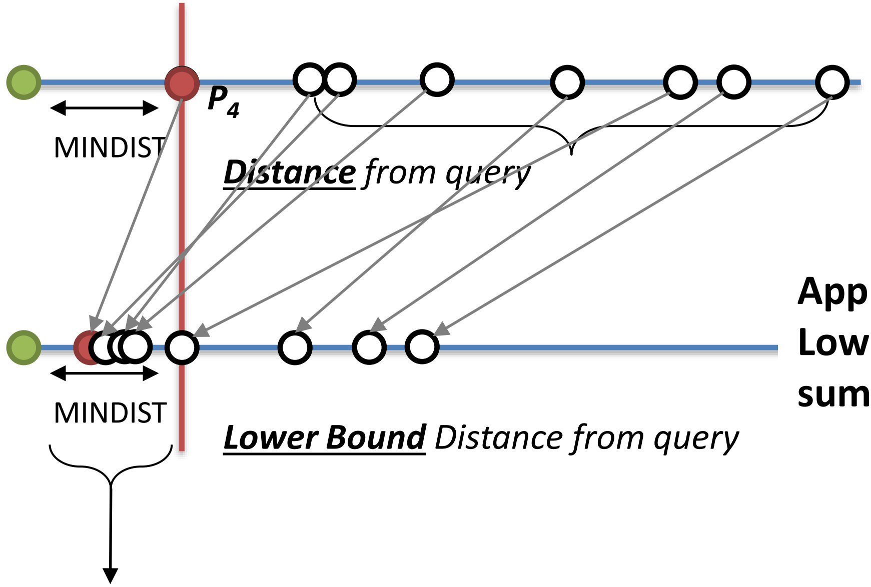
need **methods** for **generating** queries of **varying hardness**

Characterizing Queries



Approximating distances using **Lower Bounding functions** on **summarizations**.

Characterizing Queries

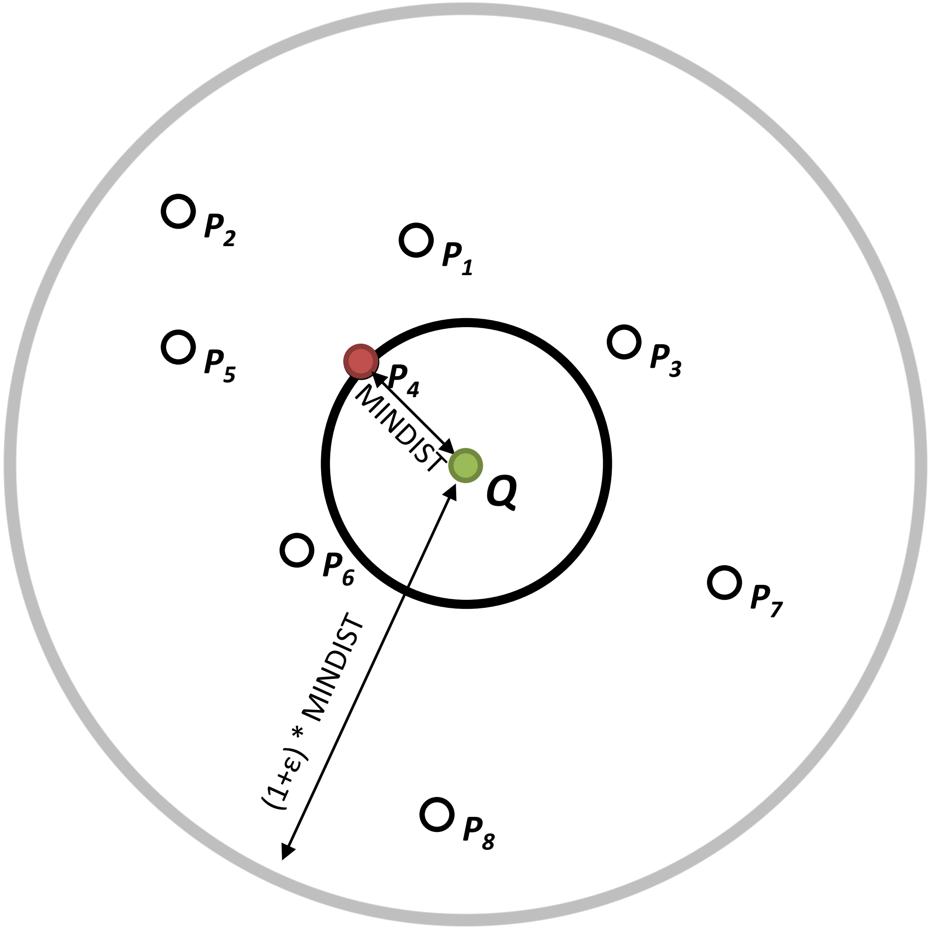


Approximating distances using **Lower Bounding functions** on **summarizations**.

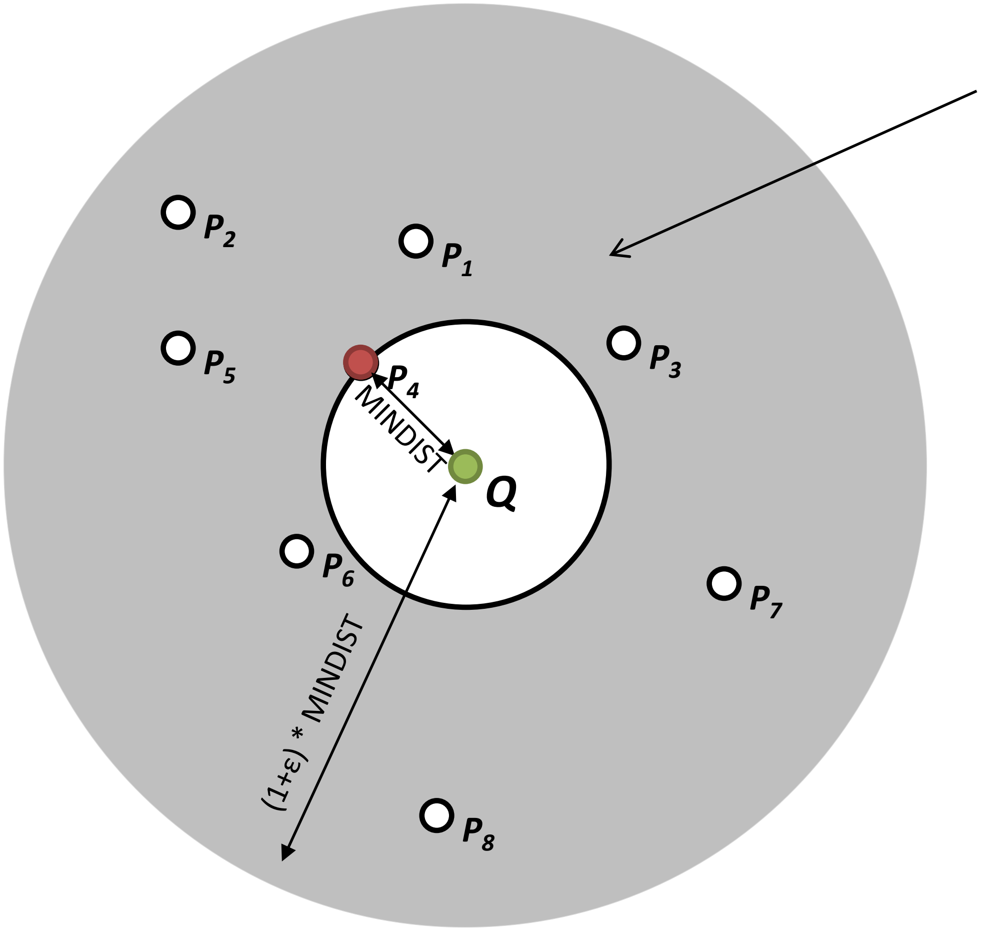
Points with **lower bounds** below **MINDIST cannot be pruned**

Must be **read from disk** in order **to dismiss false positives**

Hardness



Hardness



We define an ε-area

*(1+*ε) * MINDIST

Hardness

$$\frac{\text{\# of data –series in ε-area}}{\text{\# all data series}}$$

Echihabi, Zoumpatianos, Palpanas - ISCC 2020

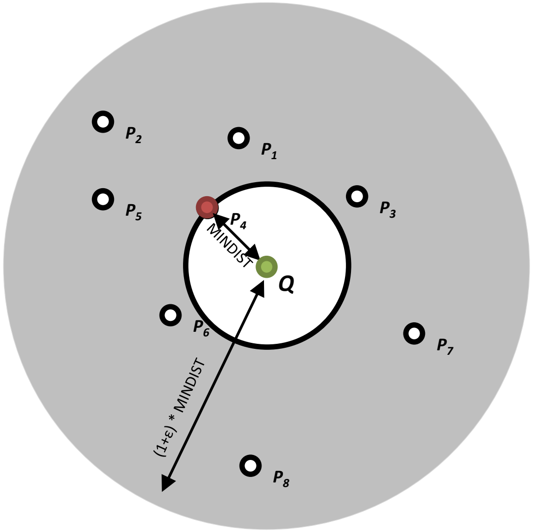
Hardness

Significance

Queries with **larger hardness** tend to have a **larger minimum effort**

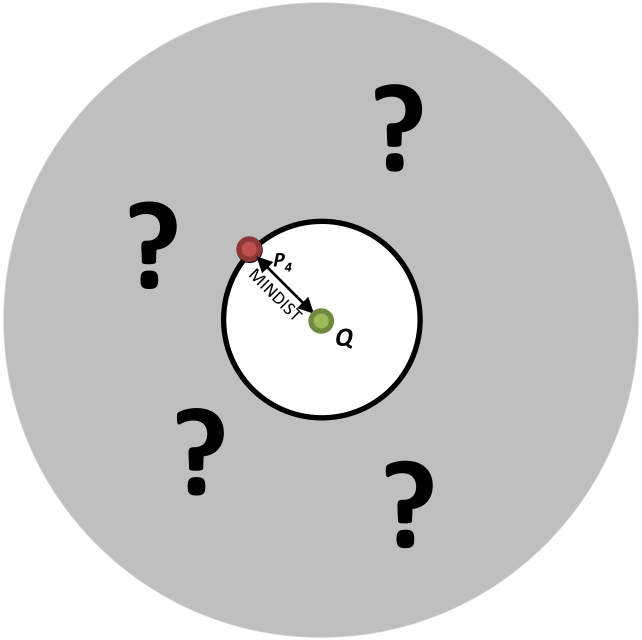
data series **close to the answer**

higher chance that their **lower bounding distance** will be **less than MINDIST**



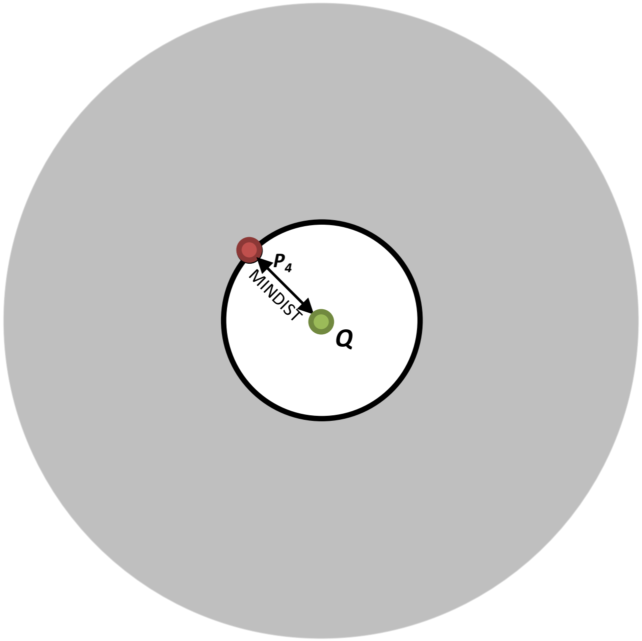
Workload Generation

Random queries have **random hardness**

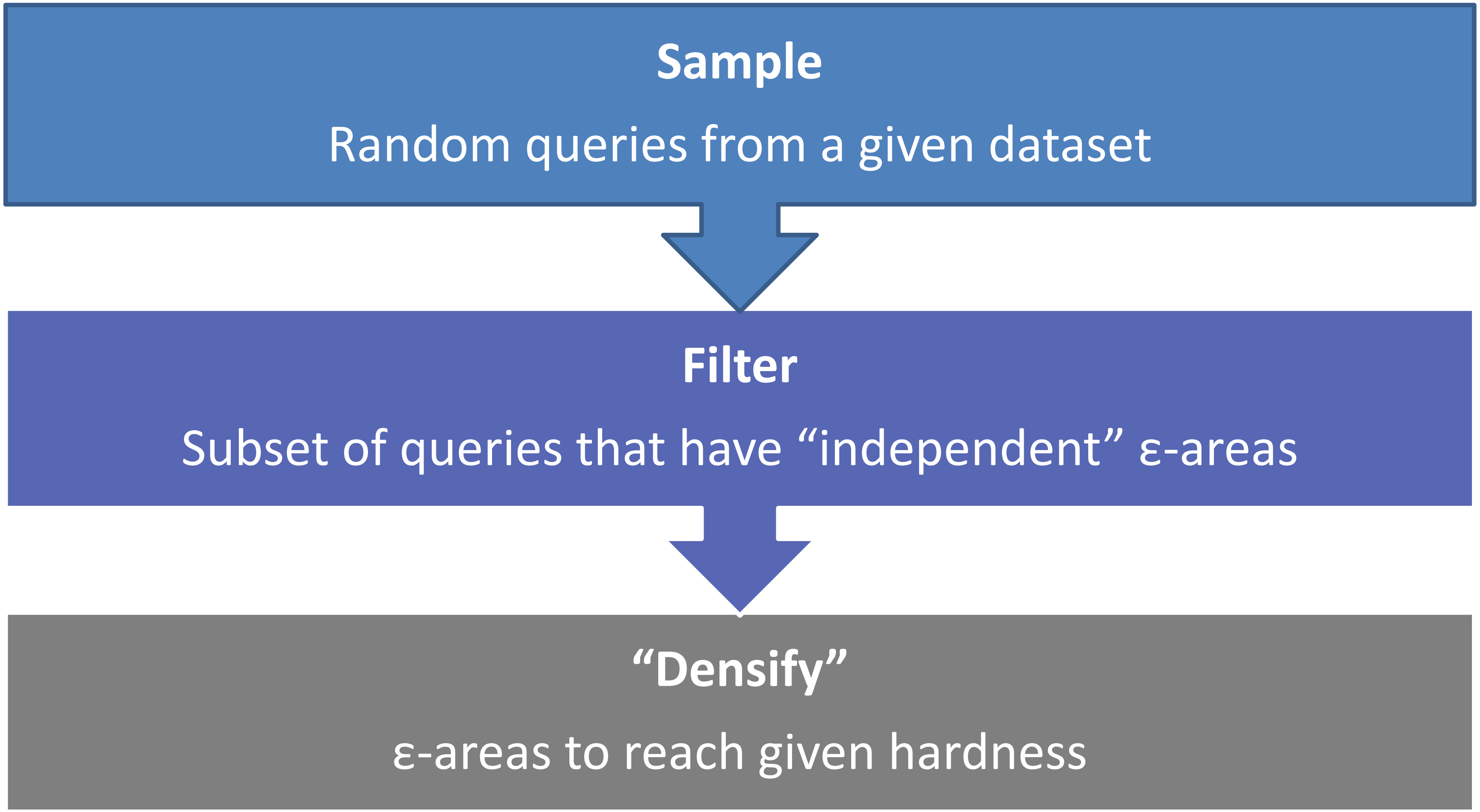


Workload Generation

Can we **generate** queries of **controlled hardness**?



3 Step Process



Sample

Random queries from a given dataset

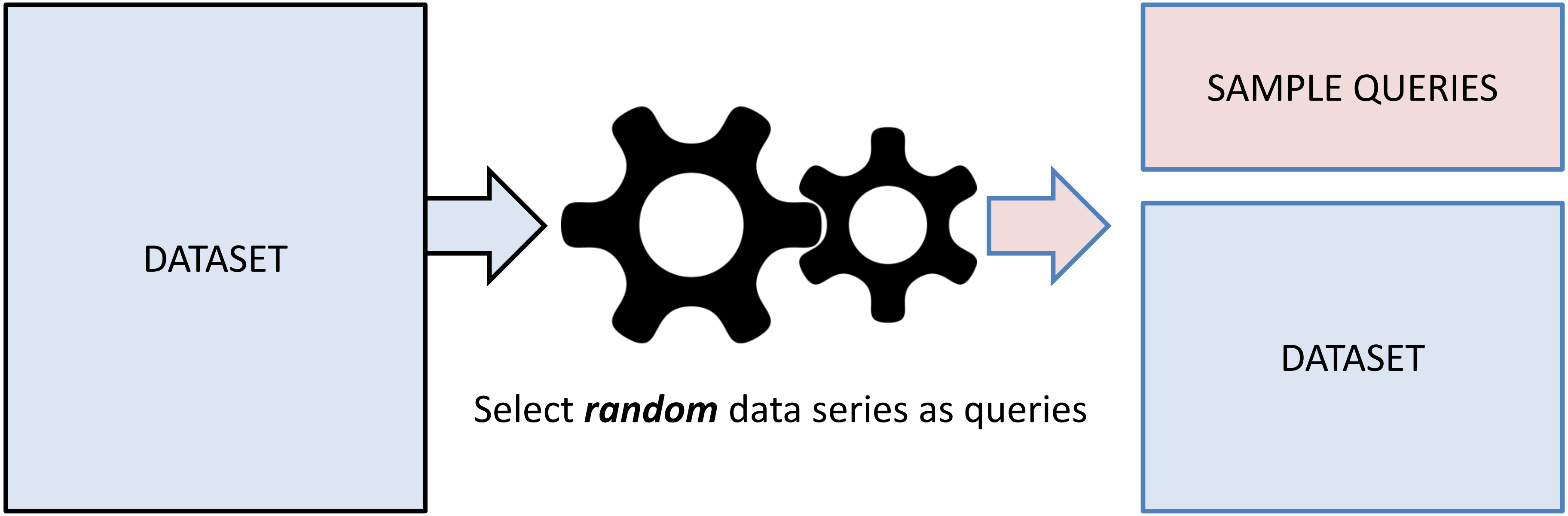
Filter

Subset of queries that have “independent” ε-areas

“Densify”

ε-areas to reach given hardness

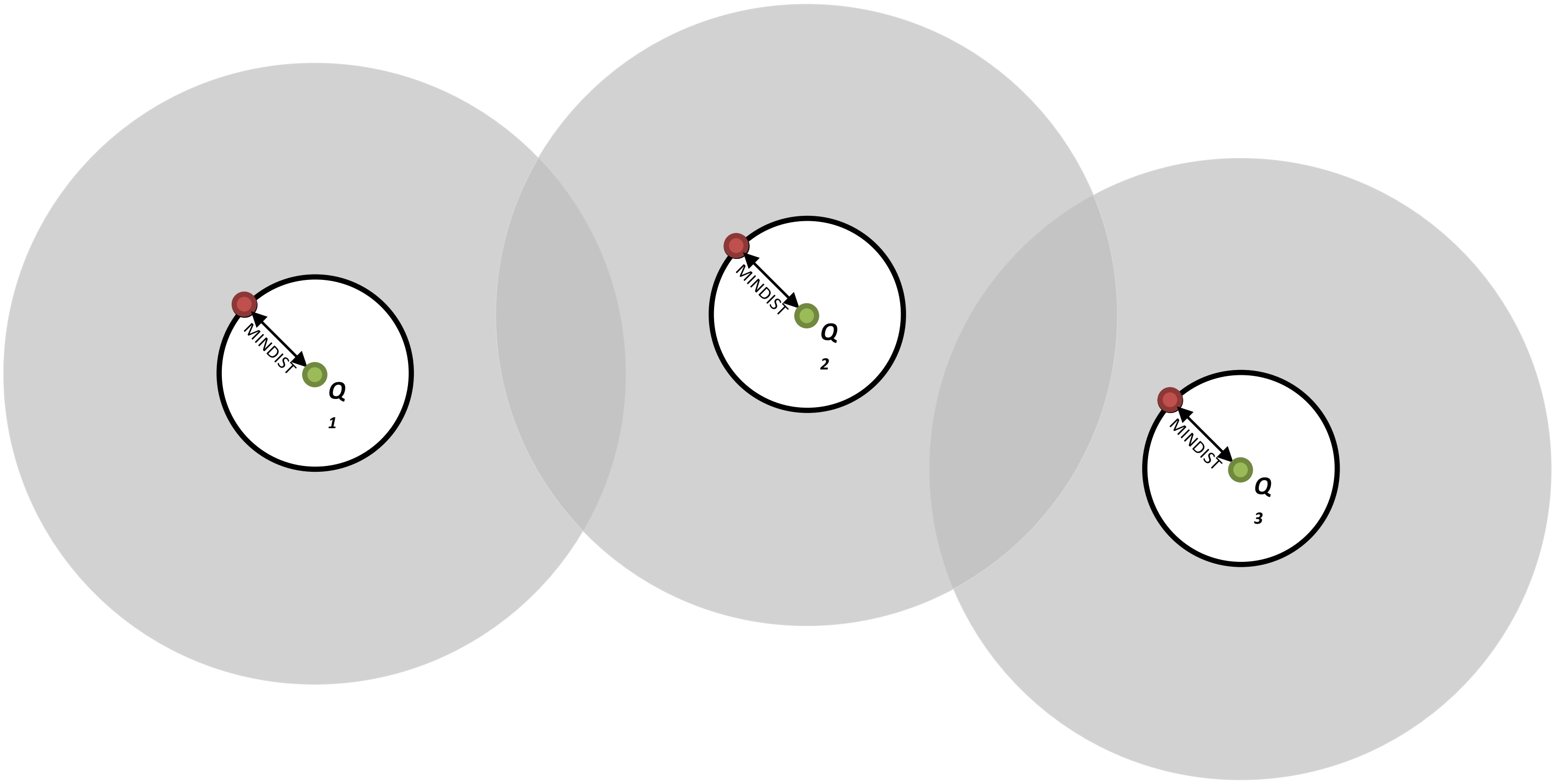
Step 1: Sampling



Select ***random*** data series as queries

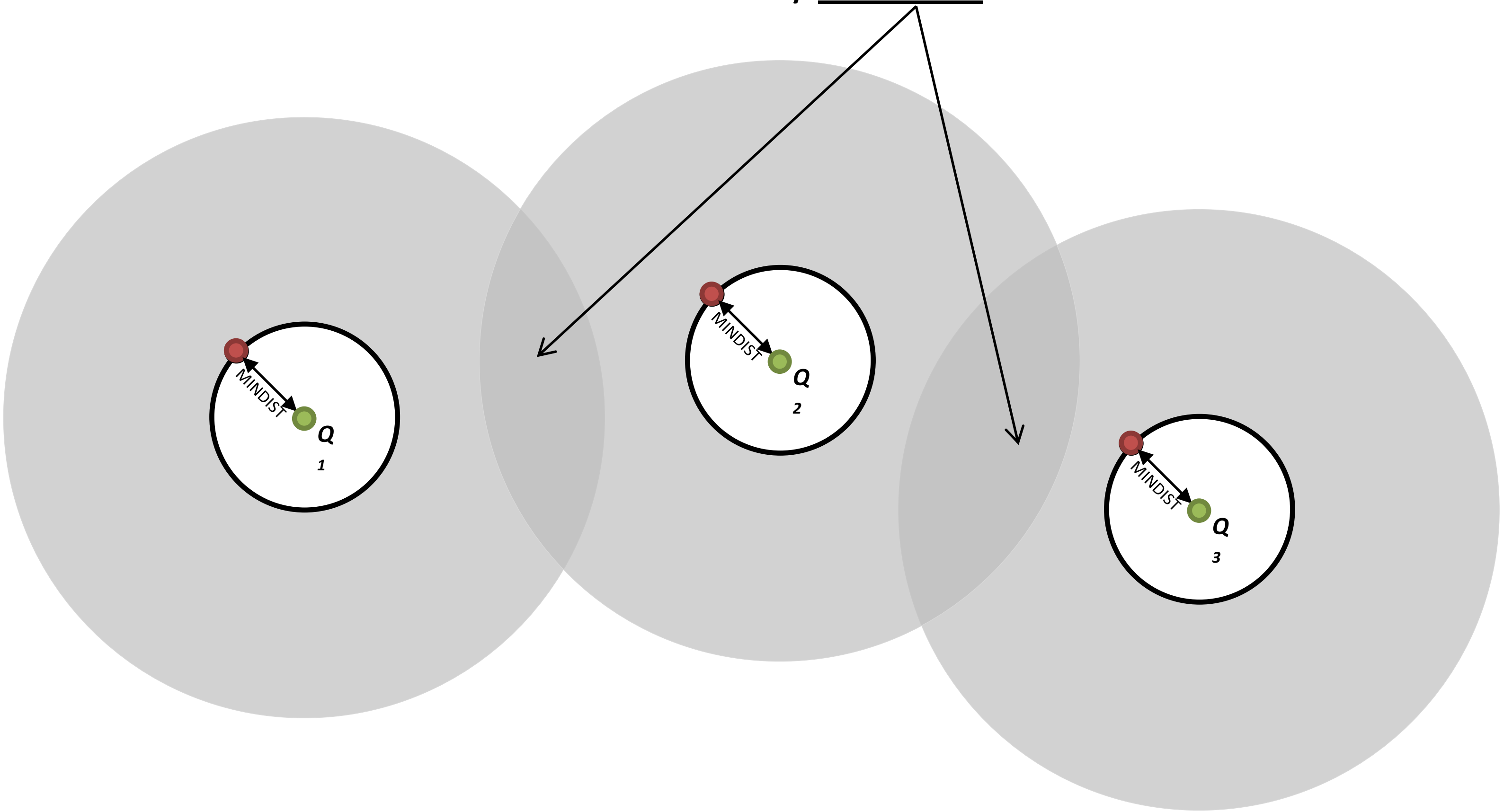
Step 2: Filtering-out "intersecting" queries

We need to **independently** control the ε-areas



Step 2: Filtering-out "intersecting" queries

The ε-areas of (Q_1, Q_2) and (Q_2, Q_3) cannot be **independently controlled** because they **intersect**

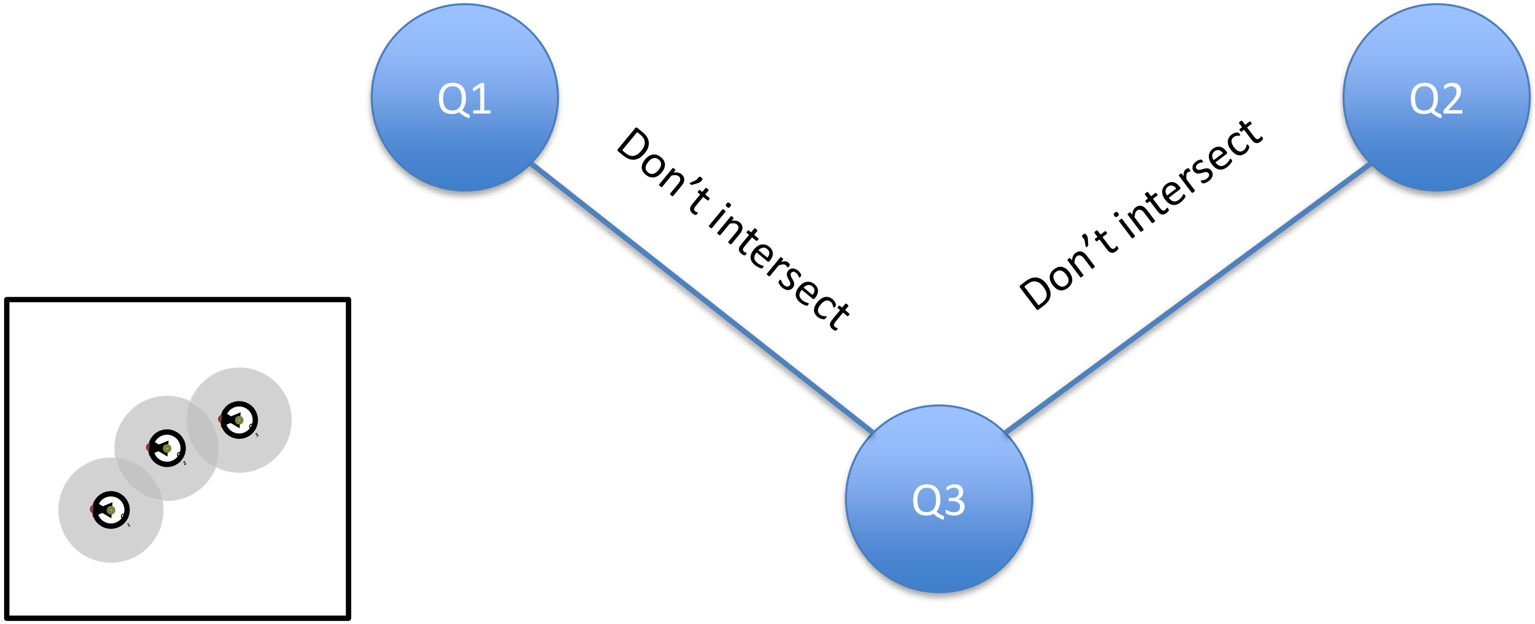


Step 2: Filtering-out "intersecting" queries

Can be formulated as a **graph problem**

1 node per query

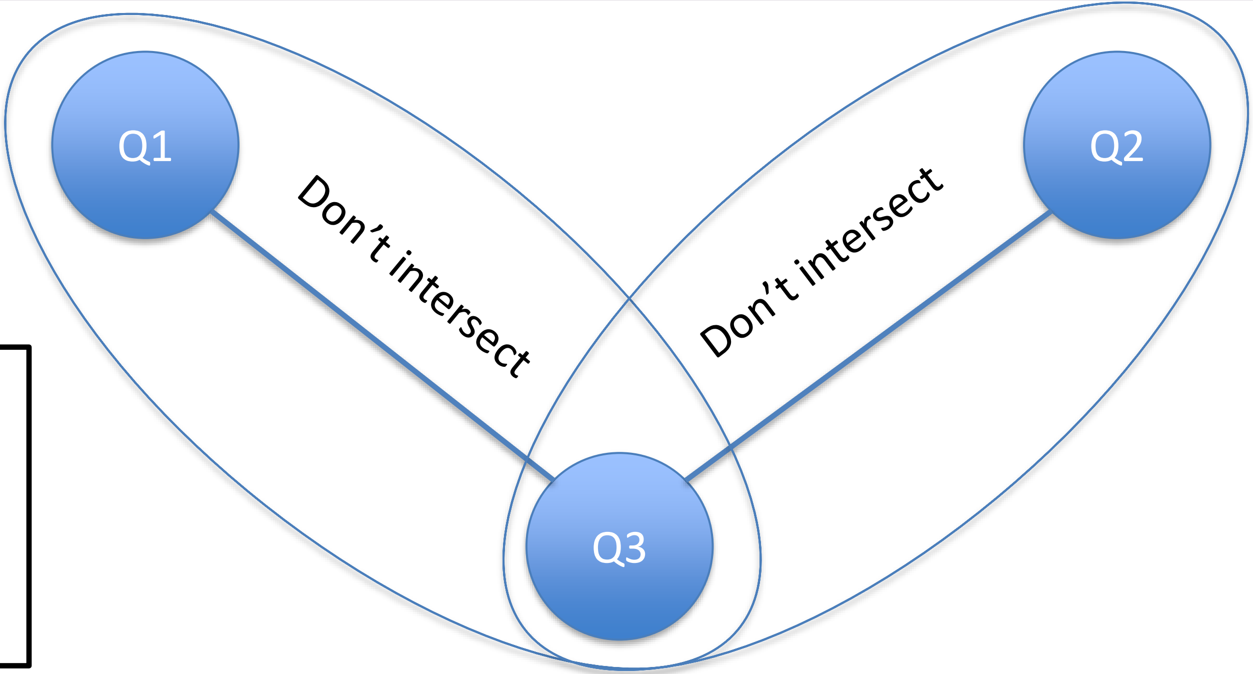
1 edge for each pair that doesn't intersect

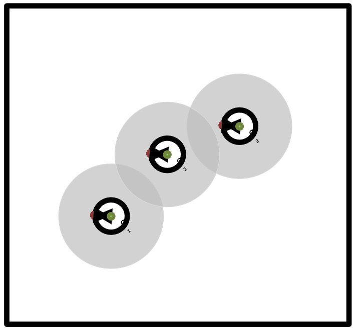


Step 2: Filtering-out "intersecting" queries

Solution

We need to find the **maximum** clique in the graph
(NP-Complete: we find a large enough clique using a heuristic)





Step 3: Densifying
Number of data series to add

1. Given a set of hardnesses as input

2. We decide the number of data series to add for each query by solving a linear system of equations:

$$\alpha_i = \frac{N_i + x_i}{N + \sum_{j=1}^{n} x_j}$$

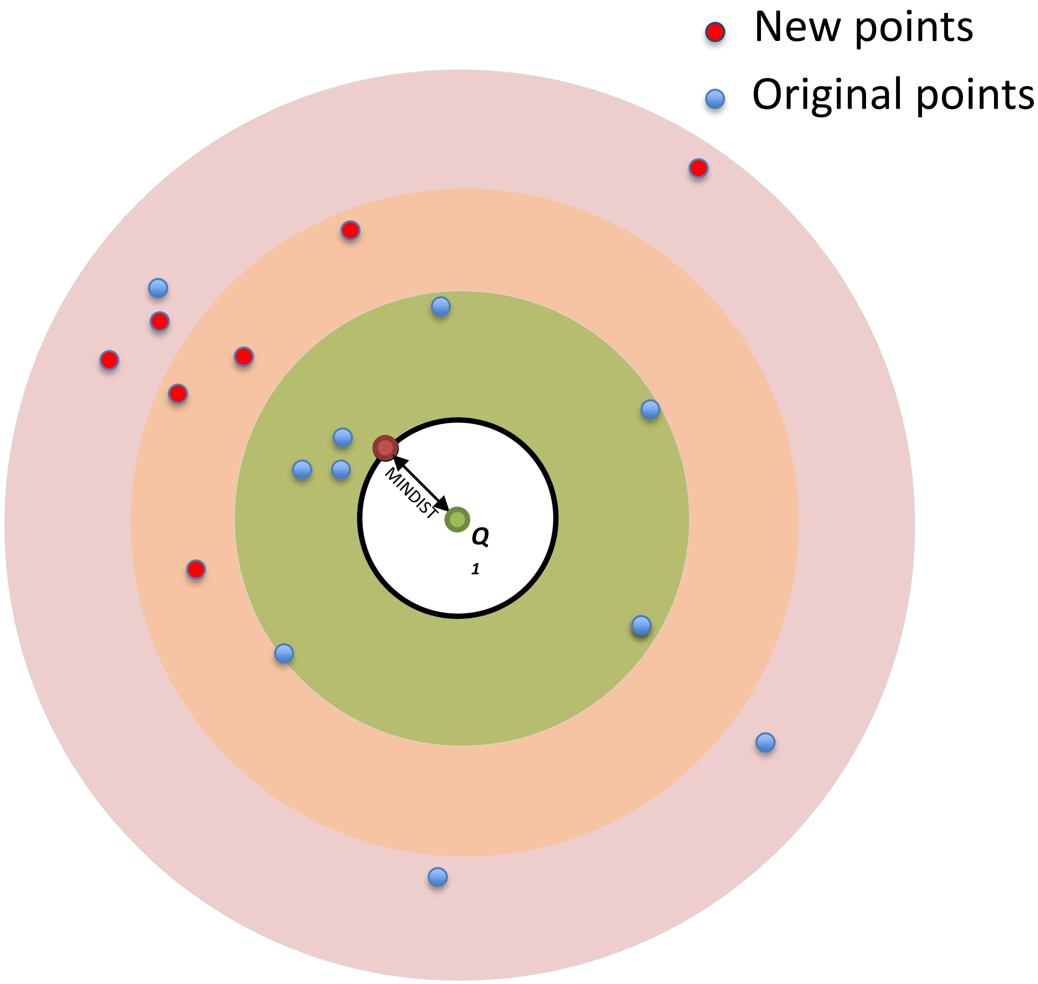
- α_i : hardness,
- X_i : number of data series to add
- N_i : number of data series already in e-area
- N : Total number of data series

Densification Method:

Equi-densification

Distribute points such that:
The **worse** a summarization
the more data it checks

Equal number of points in every “zone”



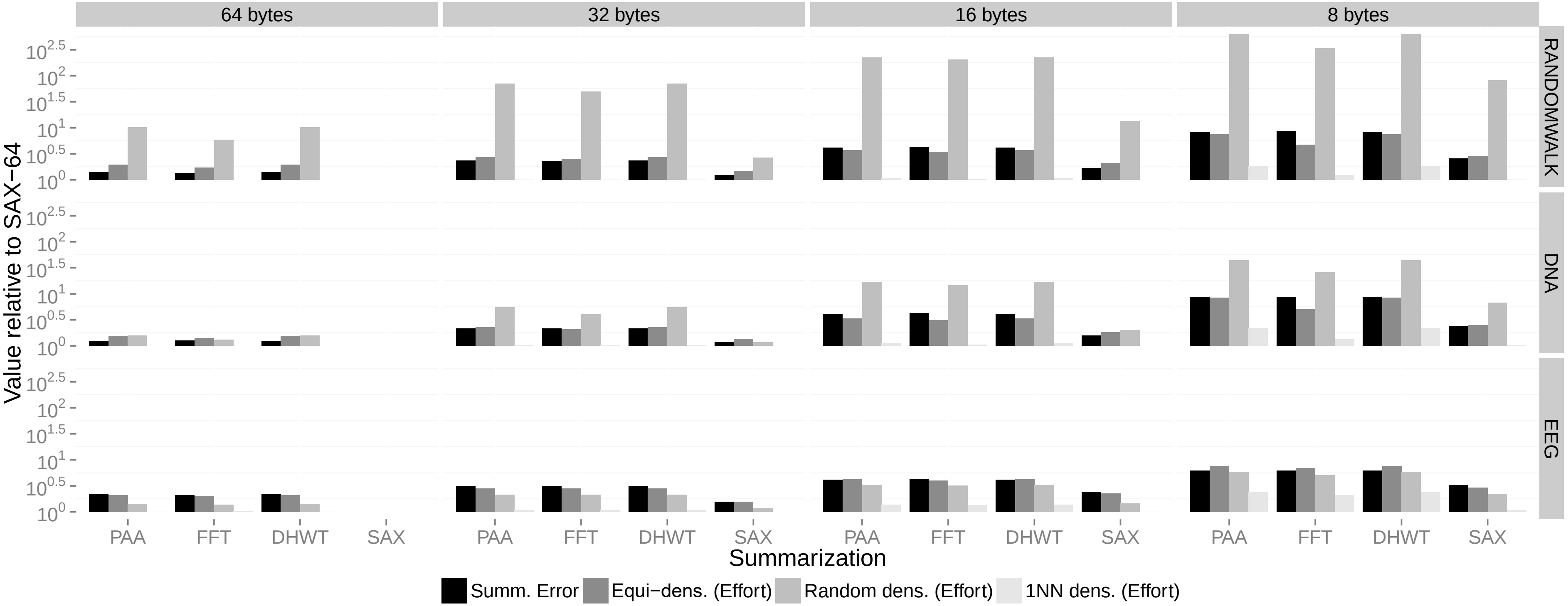
Experiments

Densification Methods

Using all datasets with size 256 (100 queries for each dens. method), we measured the:

- ***1-TLB: Summarization Error** (0: perfect bound, 1: worst possible bound)*
- ***Minimum Effort** for a set of summarizations using 8 – 64 bytes.*

Normalized over SAX-64



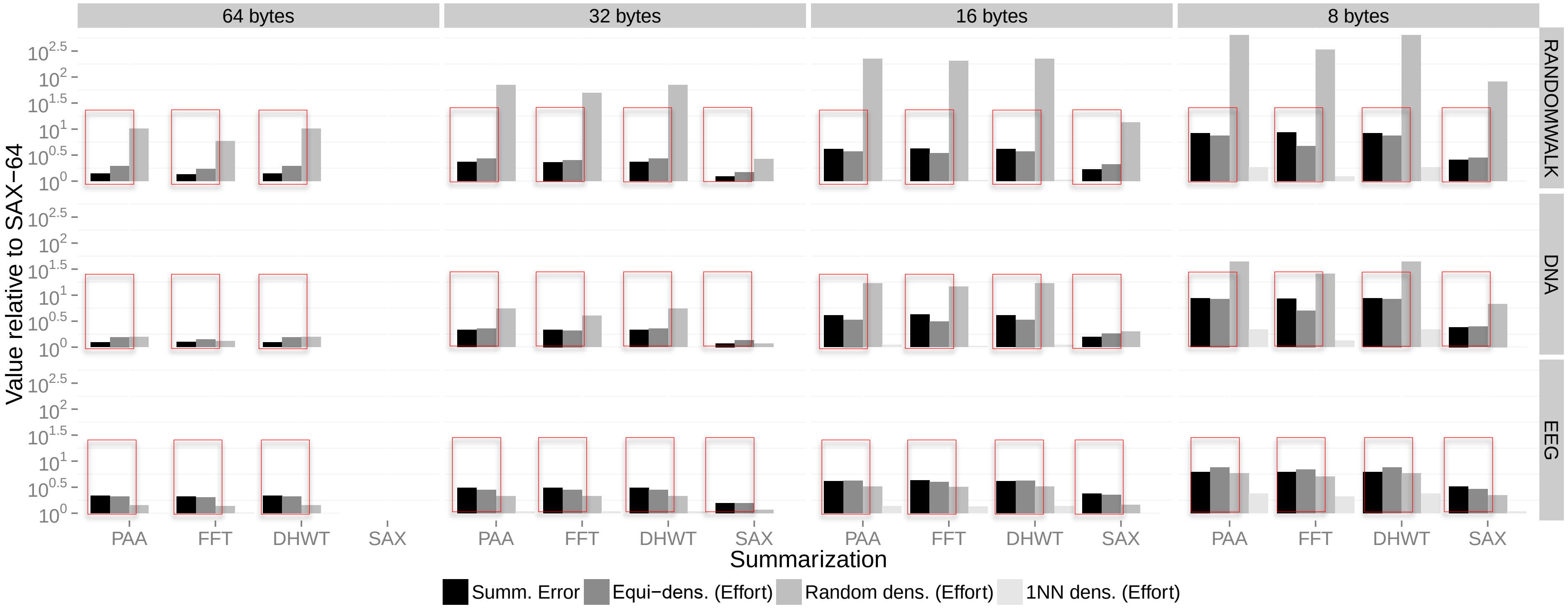
Echihabi, Zoumpatianos, Palpanas - ISCC 2020

Experiments
Densification Methods

For **equi-densification**

normalized Effort is closer to the **normalized Summarization Error**

The worse a summarization the bigger effort it does



Echihabi, Zoumpatianos, Palpanas - ISCC 2020

Summary

Pros:

Theoretical background
Methodology for characterizing
NN queries for data series indexes

Nearest neighbor query workload generator
Designed to stress-test data series indexes
at varying levels of difficulty

Cons:

Time complexity
Need new approach to scale to very large datasets

Interactive Analytics?

Interactive Analytics?

- data series analytics is computationally expensive
  - very high inherent complexity

- may not always be possible to remove delays
  - but could try to hide them!

Need for Interactive Analytics

Publications

BigVis'19

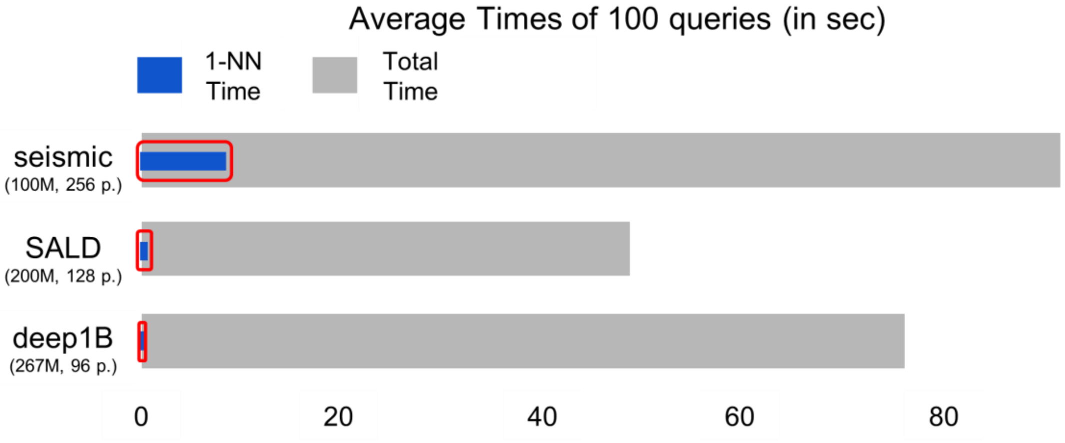
- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution

Need for Interactive Analytics

Publications

BigVis'19

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution

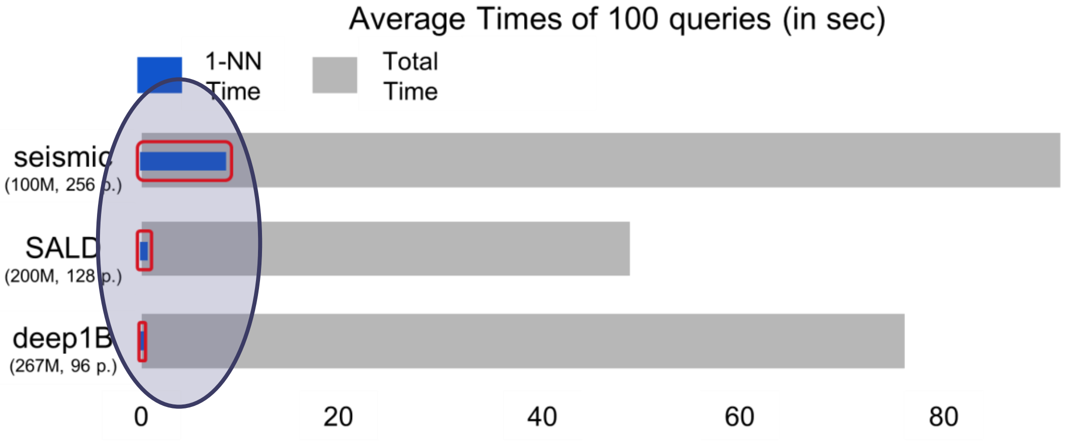


Need for Interactive Analytics

Publications

BigVis'19

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution

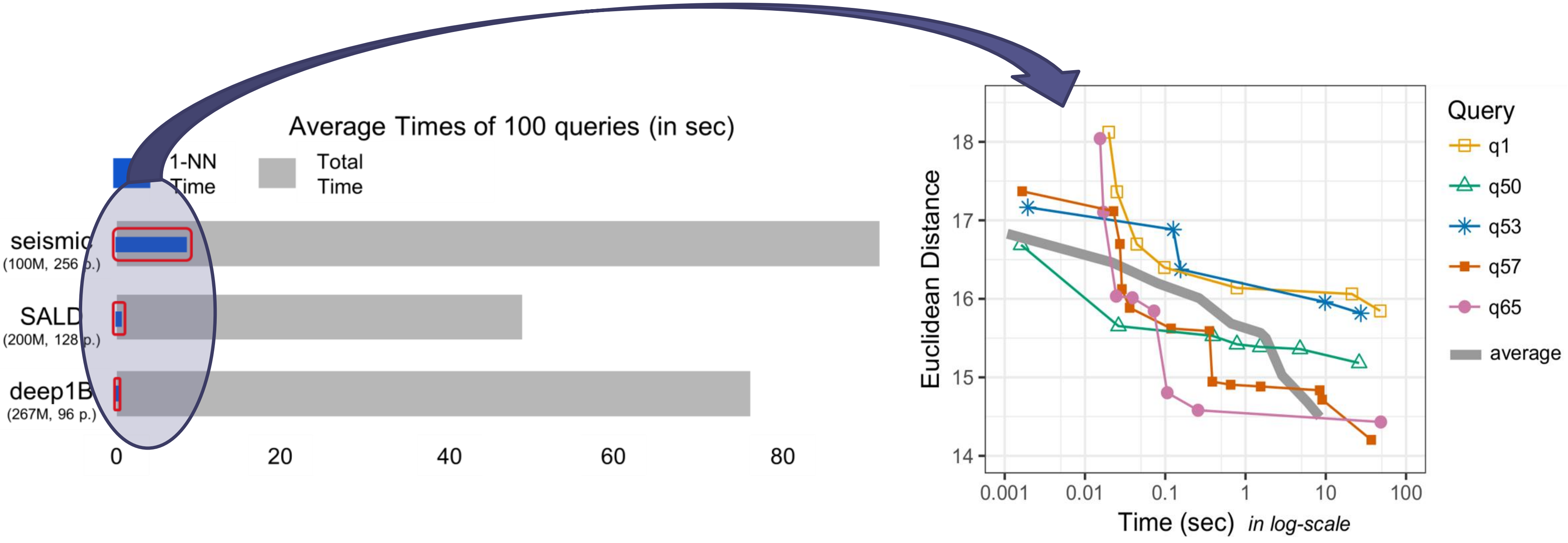


Need for Interactive Analytics

Publications

BigVis'19

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution



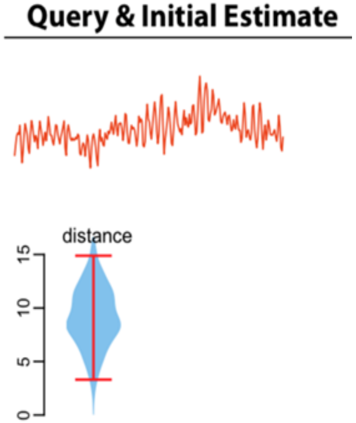
Need for Interactive Analytics

Publications

BigVis'19

SIGMOD'20

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way



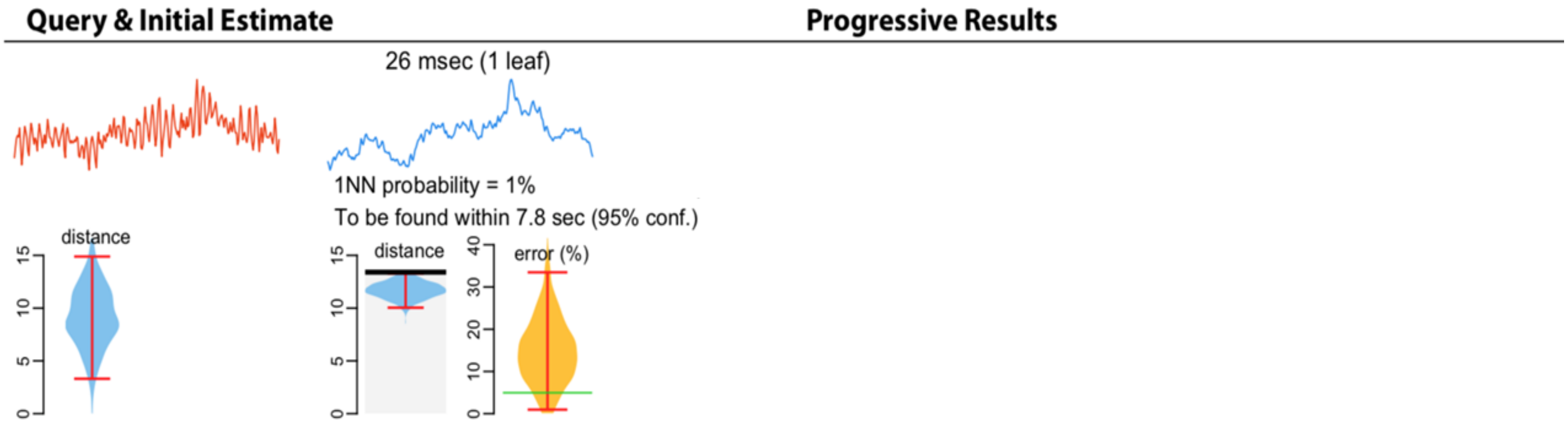
Need for Interactive Analytics

Publications

BigVis'19

SIGMOD'20

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way



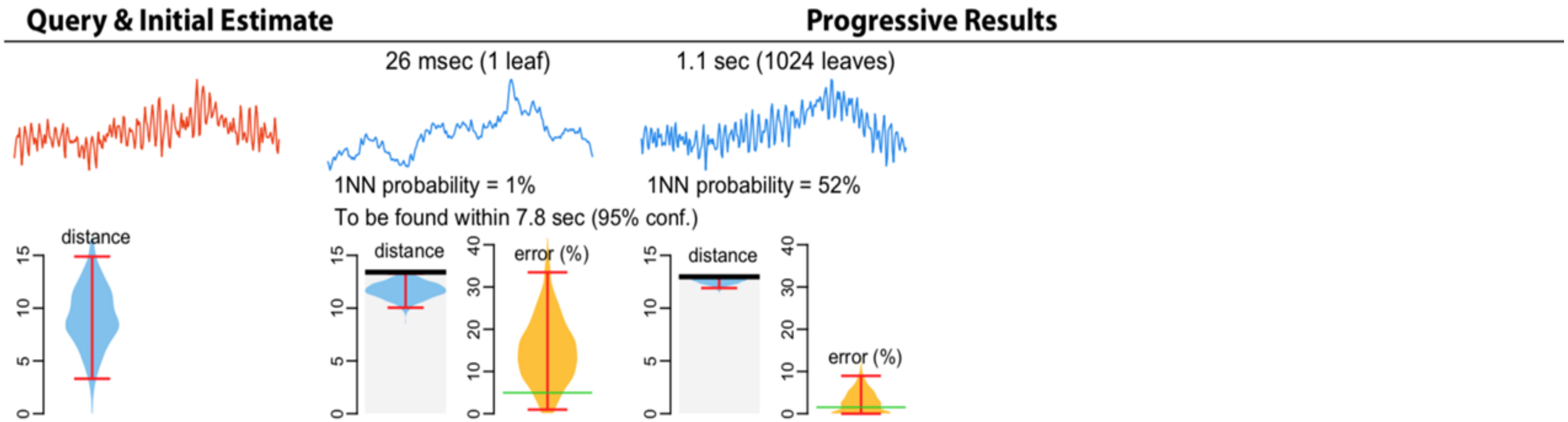
Need for Interactive Analytics

Publications

BigVis'19

SIGMOD'20

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way



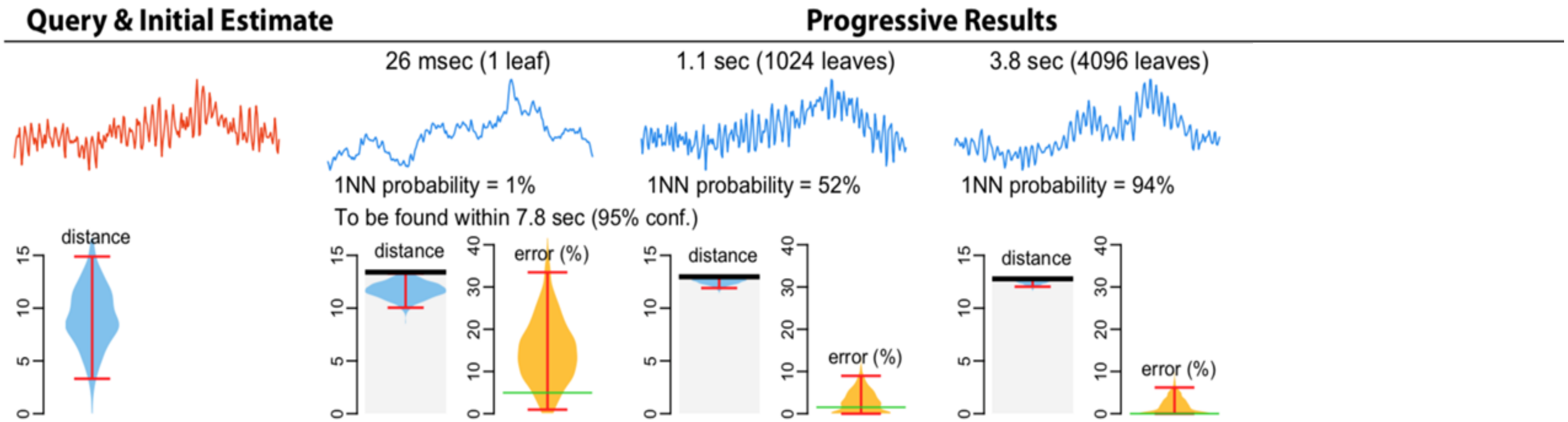
Need for Interactive Analytics

Publications

BigVis'19

SIGMOD'20

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way



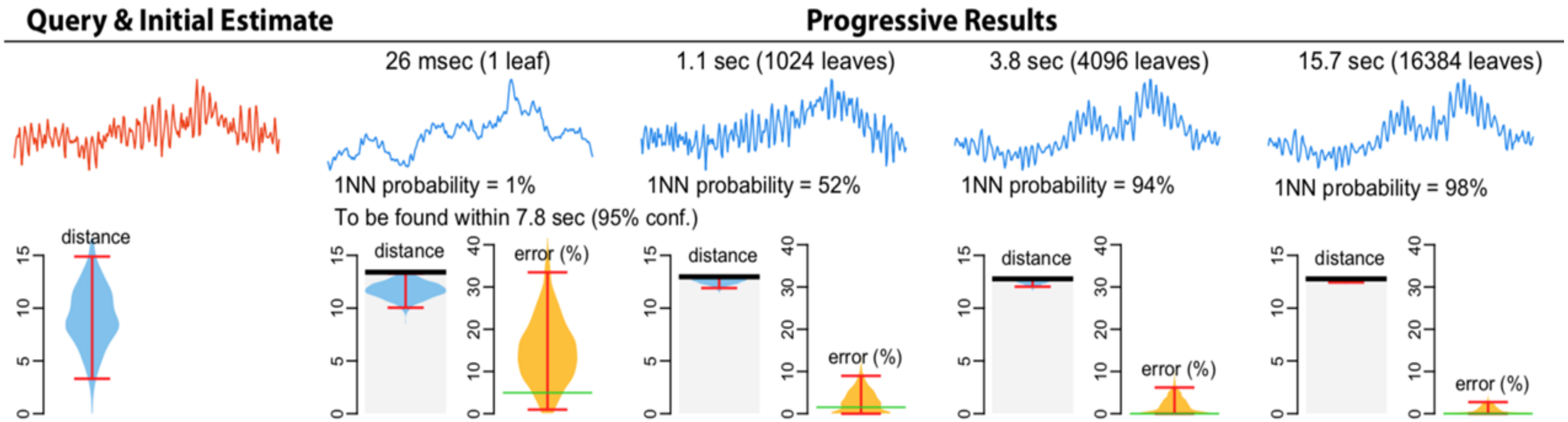
Need for Interactive Analytics

Publications

BigVis'19

SIGMOD'20

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way



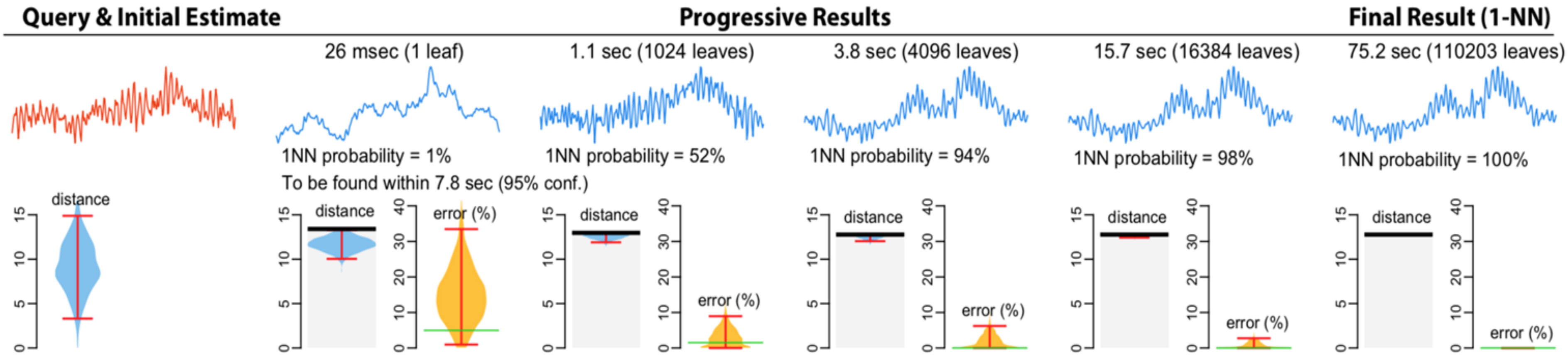
Need for Interactive Analytics

Publications

BigVis'19

SIGMOD'20

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way



Need for Interactive Analytics

Publications

BigVis'19

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way

Need for Interactive Analytics

Publications

BigVis'19

VIS'18

- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution
    - provide bounds on the errors (of the intermediate results) along the way

- several exciting research problems in intersection of visualization and data management
  - *frontend*: HCI/visualizations for querying/results display
  - *backend*: efficiently supporting these operations

Need for Parallelization/Distribution

Publications

HPCS'17

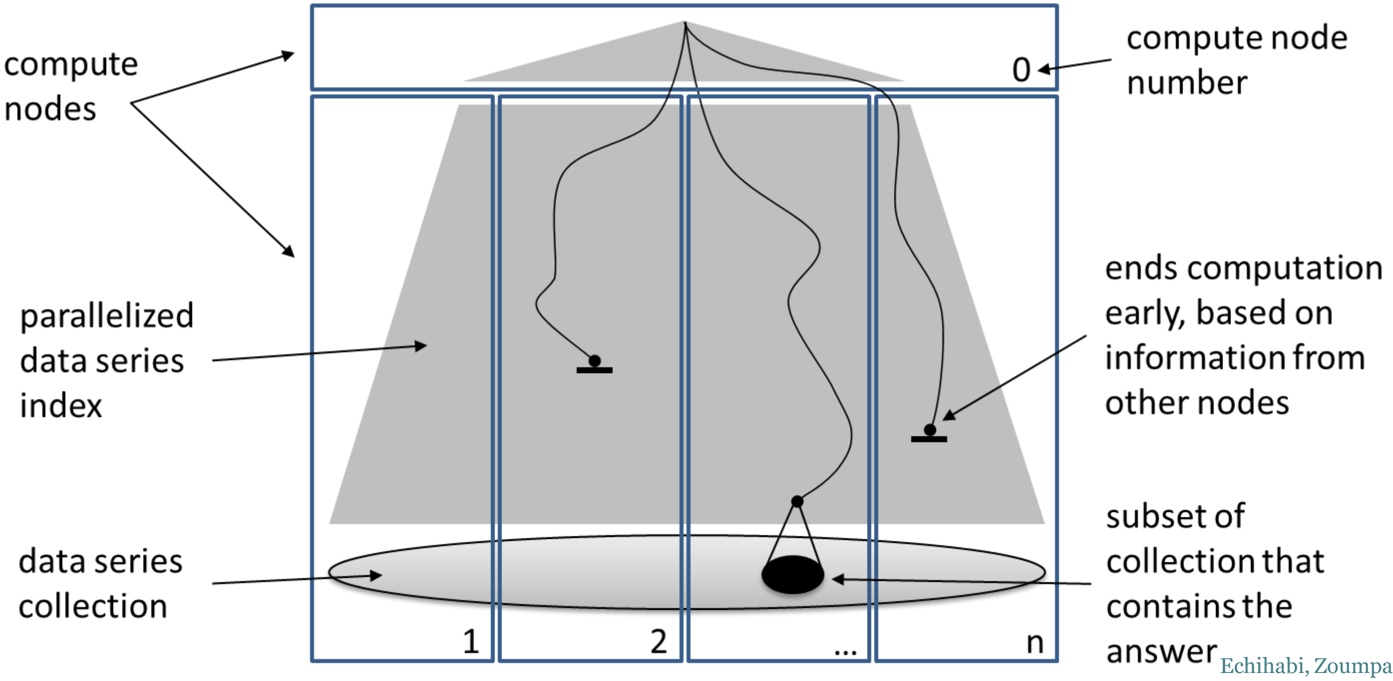
- take advantage of all modern hardware opportunities!
  - Single Instruction Multiple Data (SIMD)
    - natural for data series operations
  - multi-tier CPU caches
    - design data structures aligned to cache lines
  - multi-core and multi-socket architectures
    - use parallelism inside each computation server
  - Graphics Processing Units (GPUs)
    - propose massively parallel techniques for GPUs
  - new storage solutions: NVRAMs, FPGAs
    - develop algorithms that take these new characteristics/tradeoffs into account
  - compute clusters
    - distribute operation over many machines

Publications

HPCS'17

Need for Parallelization/Distribution

- further scale-up and scale-out possible!
  - techniques inherently parallelizable
    - across cores, across machines


compute nodes
parallelized data series index
data series collection
compute node number
0
ends computation early, based on information from other nodes
subset of collection that contains the answer
1
2
...
n

Need for Parallelization/Distribution

Publications

HPCS'17

- further scale-up and scale-out possible!
  - techniques inherently parallelizable
    - across cores, across machines
- more involved solutions required when optimizing for energy
  - reducing execution time is relatively easy
  - minimizing total work (energy) is more challenging

Data Series vs. high-d Vectors

- two sides of the same(?) coin
  - data series as multidimensional points
  - for a specific ordering of the dimensions

Data Series vs. high-d Vectors

- two sides of the same(?) coin
    - data series as multidimensional points
    - for a specific ordering of the dimensions

- several techniques for similarity search in high-d vectors
    - using LSH (SRS), space quantization (IMI), k-NN graphs (HNSW)

Data Series vs. high-d Vectors

- two sides of the same(?) coin
  - data series as multidimensional points
  - for a specific ordering of the dimensions
- several techniques for similarity search in high-d vectors
  - using LSH (SRS), space quantization (IMI), k-NN graphs (HNSW)
- how do these high-d vector techniques compare to data series techniques?
  - currently conducting extensive experimental comparison

Data Series vs. high-d Vectors

Publications

PVLDB'20

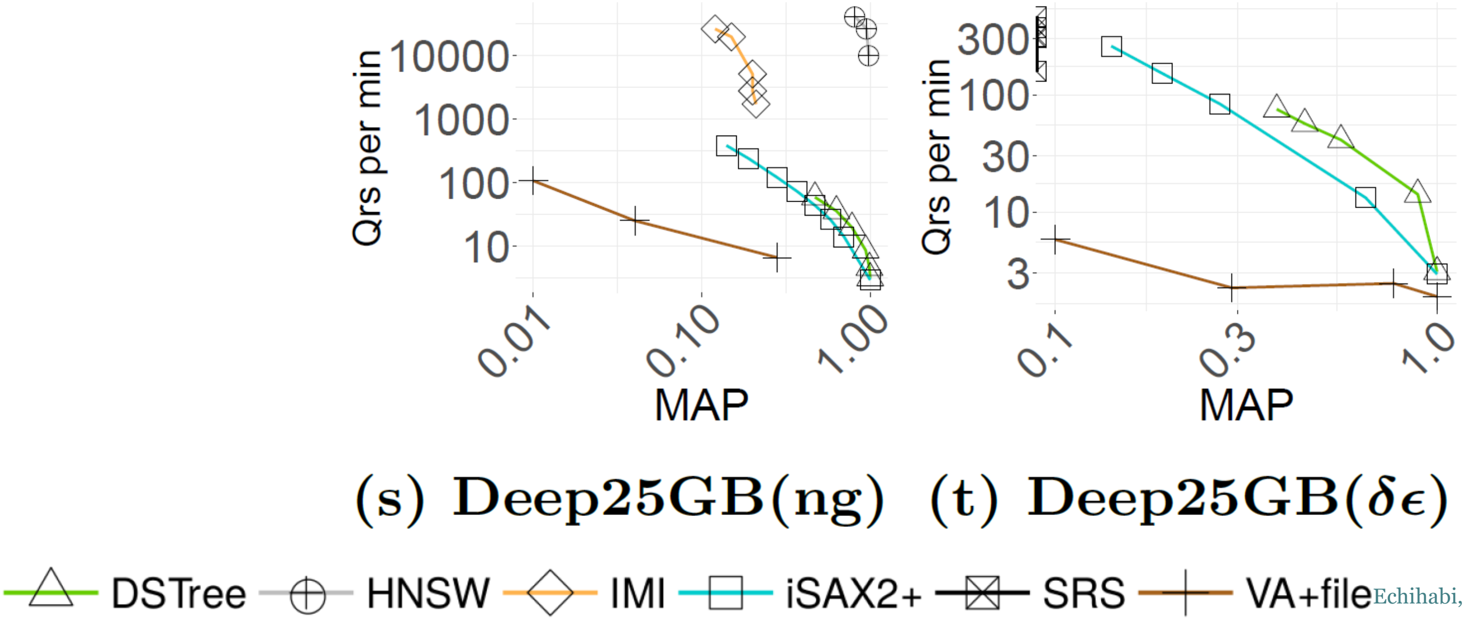
- **data series techniques** are the **overall winners**, even on **general high-d vector** data

Data Series vs. high-d Vectors

Publications

PVLDB‘20

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk



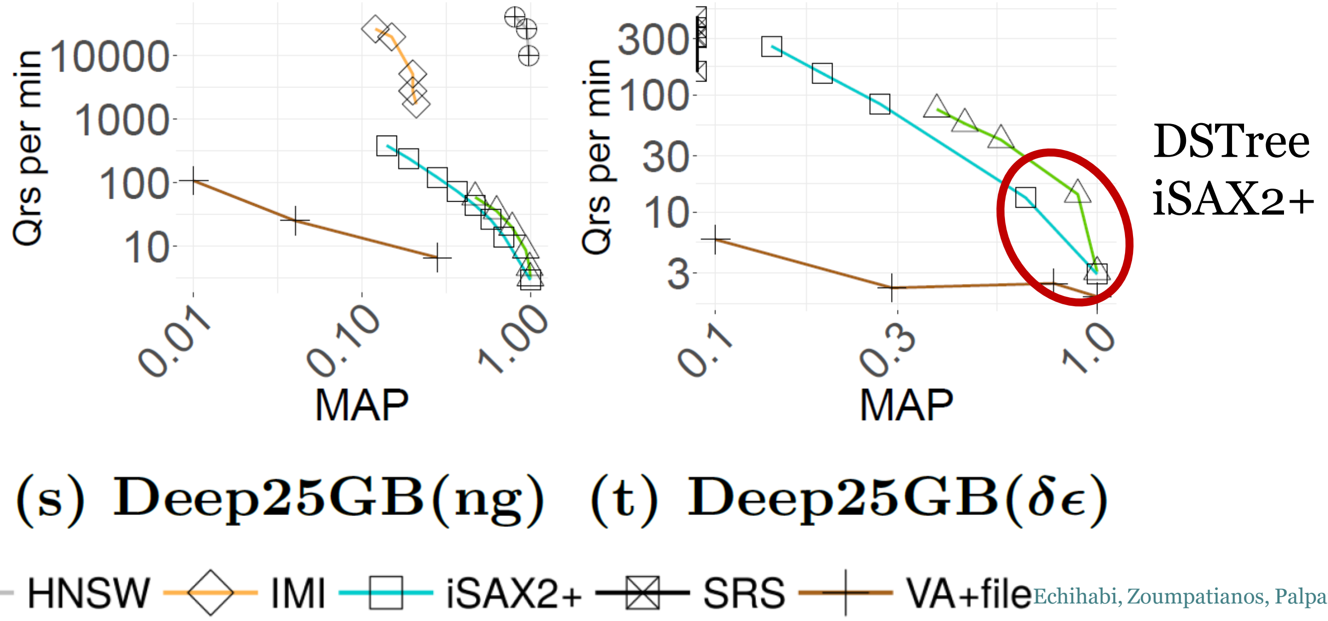
(s) Deep25GB(ng) (t) Deep25GB($\delta\epsilon$)

Publications

PVLDB'20

Data Series vs. high-d Vectors

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk



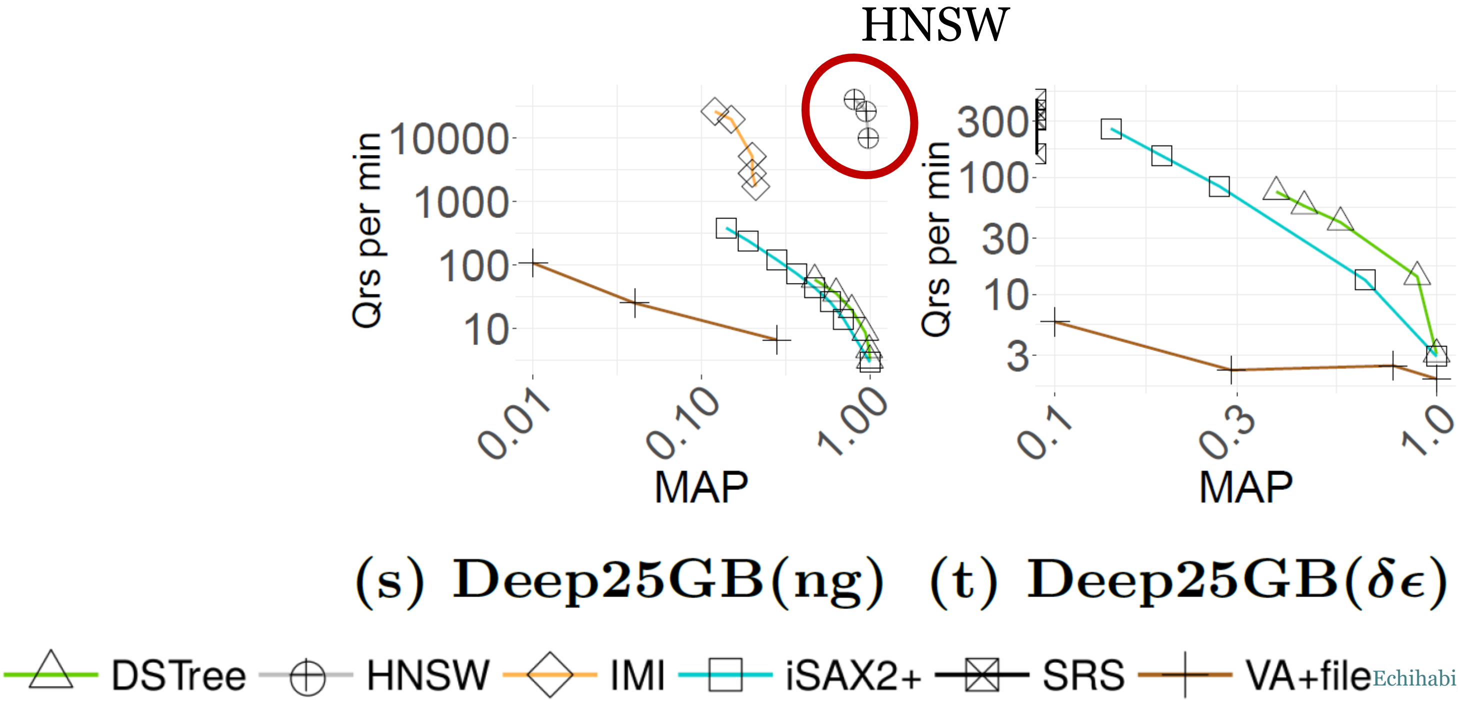
(s) Deep25GB(ng) (t) Deep25GB(δϵ)

Data Series vs. high-d Vectors

Publications

PVLDB'20

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk



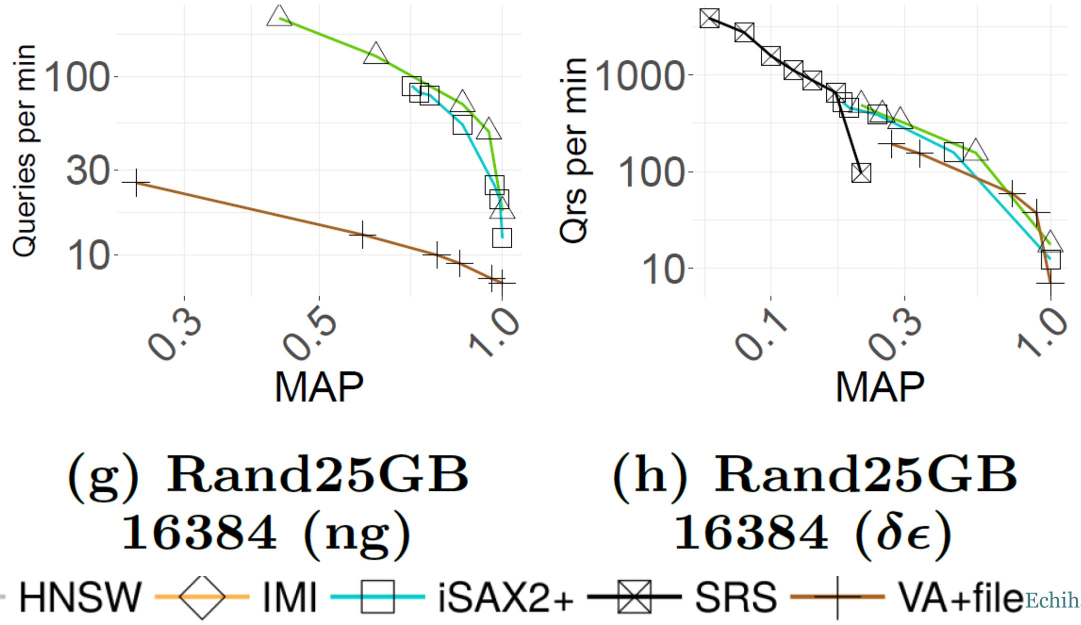
(s) Deep25GB(ng) (t) Deep25GB(δϵ)

Publications

PVLDB'20

Data Series vs. high-d Vectors

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk
  - perform the best for long vectors, in-memory and on-disk



(g) Rand25GB 16384 (ng)

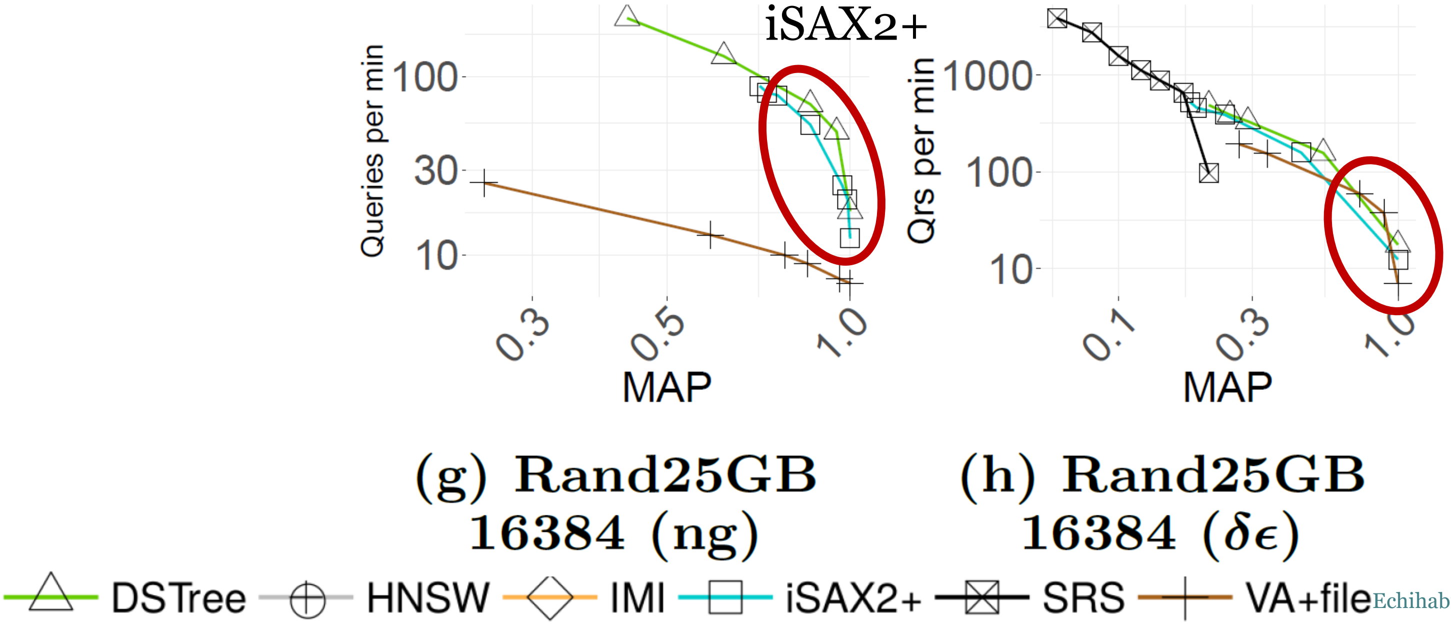
(h) Rand25GB 16384 (δε)

Data Series vs. high-d Vectors

Publications

PVLDB'20

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk
  - perform the best for long vectors, in-memory and on-disk



DSTree
iSAX2+

DSTree
iSAX2+
VA+file

(g) Rand25GB 16384 (ng)

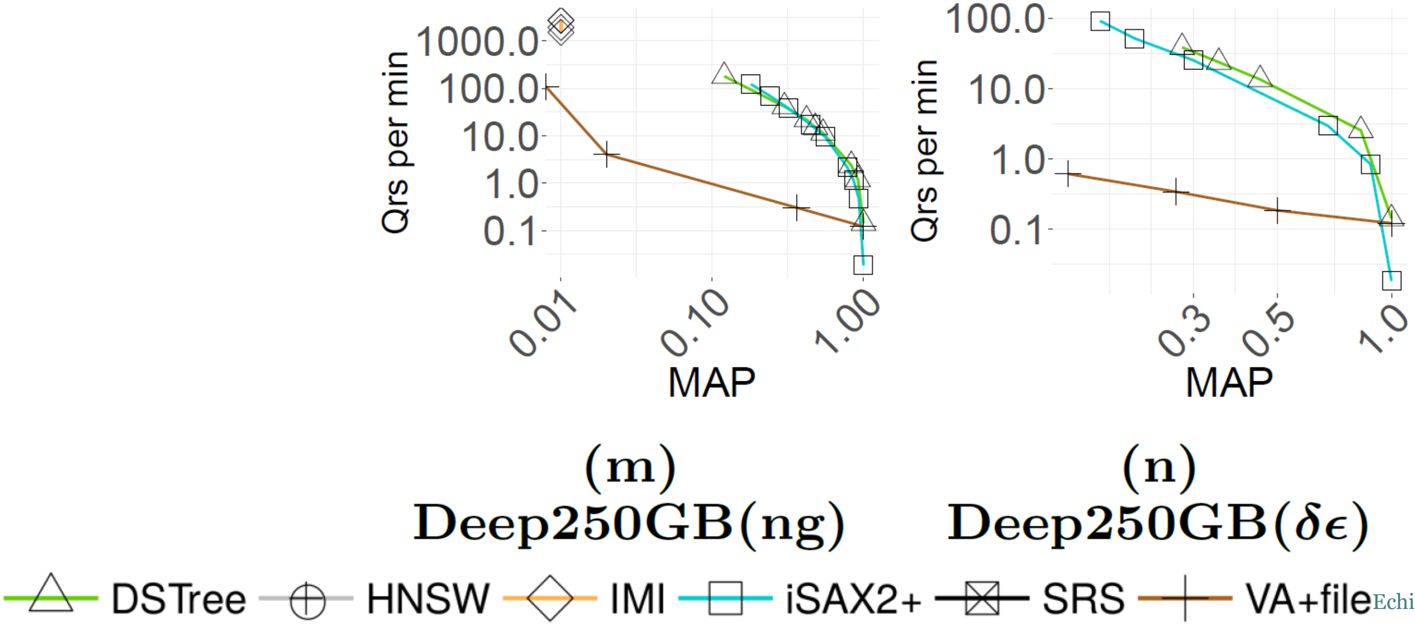
(h) Rand25GB 16384 ($\delta\epsilon$)

Publications

PVLDB'20

Data Series vs. high-d Vectors

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk
  - perform the best for long vectors, in-memory and on-disk
  - perform the best for disk-resident vectors



(m) Deep250GB(ng)

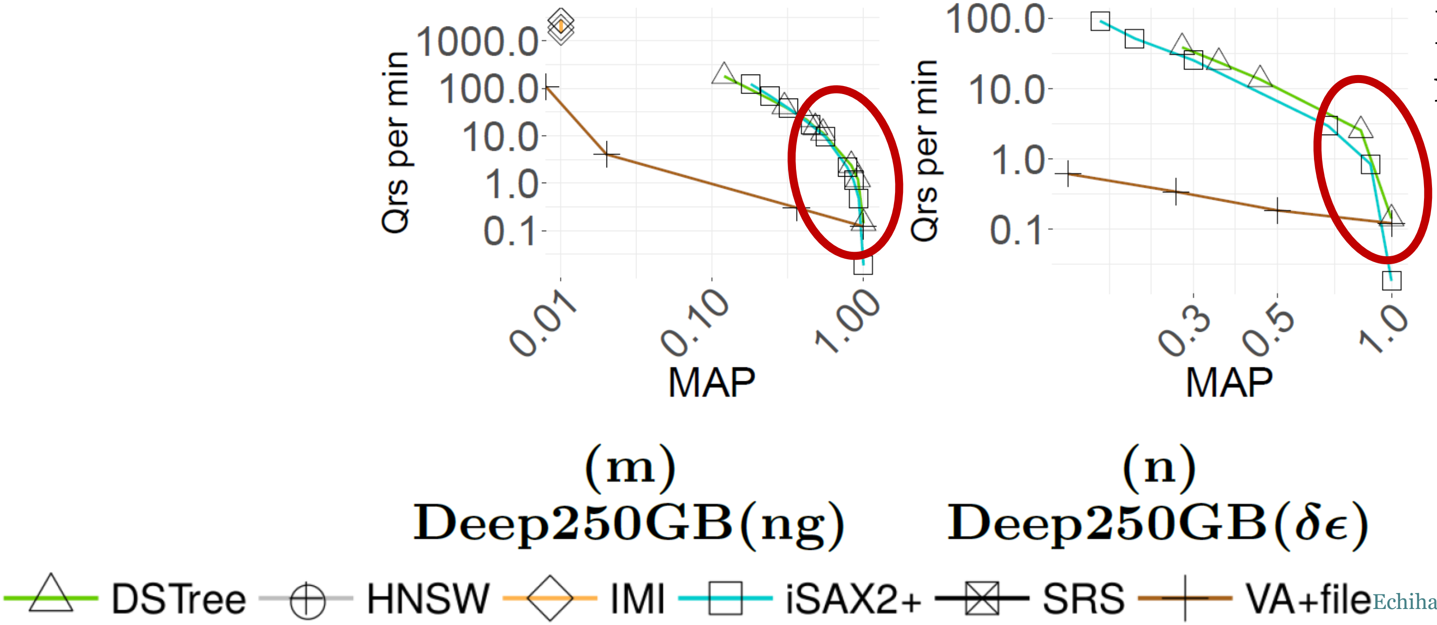
(n) Deep250GB(δϵ)

Data Series vs. high-d Vectors

Publications

PVLDB'20

- **data series techniques** are the **overall winners**, even on **general high-d vector** data
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk
  - perform the best for long vectors, in-memory and on-disk
  - perform the best for disk-resident vectors



DSTree
iSAX2+

(m) Deep250GB(ng)

(n) Deep250GB(δε)

Data Series vs. high-d Vectors

- **data series techniques** are the **overall winners**, even on **general high-d vector** data

- several new applications (and challenges) for data series similarity search techniques!

Connections to Deep Learning

- data series indexing for deep embeddings

Connections to Deep Learning

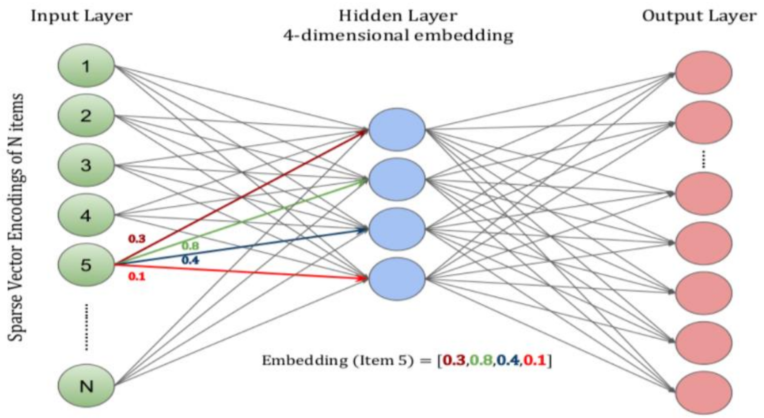
- data series indexing for deep embeddings

sequences
text
images
video
graphs
…

Connections to Deep Learning

- data series indexing for deep embeddings

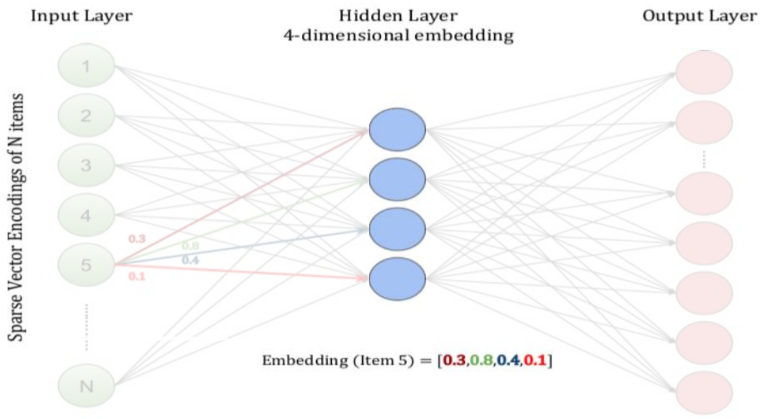
sequences
text
images
video
graphs
…



Connections to Deep Learning

- data series indexing for deep embeddings

sequences
text
images
video
graphs
…



deep embeddings
high-d vectors learned using a DNN

Connections to Deep Learning

- data series indexing for deep embeddings
  - deep embeddings are high-d vectors
  - data series techniques provide effective/scalable similarity search

Connections to Deep Learning

- data series indexing for deep embeddings
  - deep embeddings are high-d vectors
  - data series techniques provide effective/scalable similarity search
- deep learning for summarizing data series
  - eg, autoencoders can learn efficient data series summaries

Connections to Deep Learning

- data series indexing for deep embeddings
  - deep embeddings are high-d vectors
  - data series techniques provide effective/scalable similarity search
- deep learning for summarizing data series
  - eg, autoencoders can learn efficient data series summaries
- deep learning for designing index data structures
  - learn an index for similarity search

Connections to Deep Learning

- data series indexing for deep embeddings
    - deep embeddings are high-d vectors
    - data series techniques provide effective/scalable similarity search
- deep learning for summarizing data series
    - eg, autoencoders can learn efficient data series summaries
- deep learning for designing index data structures
    - learn an index for similarity search
- deep learning for query optimization
    - search space is vast
    - learn optimization function

Conclusions

- data series is a very common data type
    - across several different domains and applications

Conclusions

- data series is a very common data type
  - across several different domains and applications
- complex data series analytics are challenging
  - have very high complexity
  - efficiency comes from data series management/indexing techniques

Conclusions

- data series is a very common data type
  - across several different domains and applications
- complex data series analytics are challenging
  - have very high complexity
  - efficiency comes from data series management/indexing techniques
- need for Sequence Management System
  - optimize operations based on data/hardware characteristics
  - transparent to user

Conclusions

- data series is a very common data type
    - across several different domains and applications
- complex data series analytics are challenging
    - have very high complexity
    - efficiency comes from data series management/indexing techniques
- need for Sequence Management System
    - optimize operations based on data/hardware characteristics
    - transparent to user
- several exciting research opportunities

thank you!

google: **Karima Echihabi**
Themis Palpanas
Kostas Zoumpatianos

visit: http://**nestordb.com**

References

- N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In SIGMOD, pages 322–331, 1990.
- Keogh, E., Chu, S., Hart, D. & Pazzani, M. (2001). An Online Algorithm for Segmenting Time Series. In *Proceedings of IEEE International Conference on Data Mining*. pp 289-296.
- T. Palpanas, M. Vlachos, E. Keogh, D. Gunopulos, W. Truppel (2004). Online Amnesic Approximation of Streaming Time Series. In *ICDE* . Boston, MA, USA, March 2004.
- Douglas, D. H. & Peucker, T. K.(1973). Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature. *Canadian Cartographer*, Vol. 10, No. 2, December. pp. 112-122.
- Duda, R. O. and Hart, P. E. 1973. Pattern Classification and Scene Analysis. Wiley, New York.
- Ge, X. & Smyth P. (2001). Segmental Semi-Markov Models for Endpoint Detection in Plasma Etching. To appear in *IEEE Transactions on Semiconductor Engineering*.
- Heckbert, P. S. & Garland, M. (1997). Survey of polygonal surface simplification algorithms, Multiresolution Surface Modeling Course. *Proceedings of the 24th International Conference on Computer Graphics and Interactive Techniques*.
- Eamonn J. Keogh, Shruti Kasetty: On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. Data Min. Knowl. Discov. 7(4): 349-371 (2003)

References

- Hunter, J. & McIntosh, N. (1999). Knowledge-based event detection in complex time series data. *Artificial Intelligence in Medicine*. pp. 271-280. Springer.
- Ishijima, M., et al. (1983). Scan-Along Polygonal Approximation for Data Compression of Electrocardiograms. *IEEE Transactions on Biomedical Engineering*. BME-30(11):723-729.
- Koski, A., Juhola, M. & Meriste, M. (1995). Syntactic Recognition of ECG Signals By Attributed Finite Automata. *Pattern Recognition*, 28 (12), pp. 1927-1940.
- Keogh, E. & Pazzani, M. (1999). Relevance feedback retrieval of time series data. *Proceedings of the 22th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval.*
- Keogh, E., & Pazzani, M. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. *Proceedings of the 4th International Conference of Knowledge Discovery and Data Mining*. pp 239-241, AAAI Press.
- Keogh, E., & Smyth, P. (1997). A probabilistic approach to fast pattern matching in time series databases. *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*. pp 24-20.
- P. Ciaccia, M. Patella, and P. Zezula. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. Proceedings of VLDB'97, pp 426–435.
- Lavrenko, V., Schmill, M., Lawrie, D., Ogilvie, P., Jensen, D., & Allan, J. (2000). Mining of Concurent Text and Time Series. *Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining*. pp. 37-44.

References

- McKee, J.J., Evans, N.E., & Owens, F.J. (1994). Efficient implementation of the Fan/SAPA-2 algorithm using fixed point arithmetic. *Automedica*. Vol. 16, pp 109-117.
- Pavlidis, T. (1976). Waveform segmentation through functional approximation. *IEEE Transactions on Computers*.
- Qu, Y., Wang, C. & Wang, S. (1998). Supporting fast search in time series for movement patterns in multiples scales. *Proceedings of the 7th International Conference on Information and Knowledge Management*.
- Ramer, U. (1972). An iterative procedure for the polygonal approximation of planar curves. *Computer Graphics and Image Processing*. 1: pp. 244-256.
- Shatkay, H. (1995). Approximate Queries and Representations for Large Data Sequences. *Technical Report cs-95-03*, Department of Computer Science, Brown University.
- Shatkay, H., & Zdonik, S. (1996). Approximate queries and representations for large data sequences. *Proceedings of the 12th IEEE International Conference on Data Engineering*. pp 546-553.
- Vullings, H.J.L.M., Verhaegen, M.H.G. & Verbruggen H.B. (1997). ECG Segmentation Using Time-Warping. *Proceedings of the 2nd International Symposium on Intelligent Data Analysis*.
- P. Ciaccia and M. Patella. PAC Nearest Neighbor Queries: Approximate and Controlled Search in HighDimensional and Metric Spaces. In ICDE, pages 244– 255, 2000.
- H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. El Abbadi. Vector Approximation Based Indexing for Non-uniform High Dimensional Data Sets. In CIKM, pp 202–209, 2000.

References

- Wang, C. & Wang, S. (2000). Supporting content-based searches on time Series via approximation. *Proceedings of the 12th International Conference on Scientific and Statistical Database Management*.
- Jessica Lin, Eamonn J. Keogh, Li Wei, Stefano Lonardi: Experiencing SAX: a novel symbolic representation of time series. Data Min. Knowl. Discov. 15(2): 107-144 (2007)
- Jin Shieh, Eamonn J. Keogh: iSAX: indexing and mining terabyte sized time series. KDD 2008: 623-631
- Themis Palpanas, Michail Vlachos, Eamonn J. Keogh, Dimitrios Gunopulos: Streaming Time Series Summarization Using User-Defined Amnesic Functions. IEEE Trans. Knowl. Data Eng. 20(7): 992-1006 (2008)
- Alessandro Camerra, Themis Palpanas, Jin Shieh, Eamonn J. Keogh: iSAX 2.0: Indexing and Mining One Billion Time Series. ICDM 2010: 58-67
- S. Kashyap and P. Karras. Scalable kNN search on vertically stored time series. In KDD, pages 1334–1342 (2011)
- P. Schafer and M. Hogvist. Sfa: A symbolic fourier approximation and index for similarity search in high dimensional datasets. EDBT Conference 2012: 516–527
- T. Rakthanmanon, B. J. L. Campana, A. Mueen, G. E. A. P. A. Batista, M. B. Westover, Q. Zhu, J. Zakaria, and E. J. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In KDD, pages 262–270. ACM, 2012.
- Y. Wang, P. Wang, J. Pei, W. Wang, and S. Huang. A data-adaptive and dynamic segmentation index for whole matching on time series. PVLDB, 6(10):793–804, 2013.
- Alessandro Camerra, Jin Shieh, Themis Palpanas, Thanawin Rakthanmanon, Eamonn J. Keogh: Beyond one billion time series: indexing and mining very large time series collections with iSAX2+. Knowl. Inf. Syst. 39(1): 123-151 (2014)
- Y. Sun, W. Wang, J. Qin, Y. Zhang, and X. Lin. SRS: Solving c-approximate Nearest Neighbor Queries in High Dimensional Euclidean Space with a Tiny Index. PVLDB, 8(1):1–12, 2014
- Kostas Zoumpatianos, Stratos Idreos, Themis Palpanas: Indexing for interactive exploration of big data series. SIGMOD Conference 2014: 1555-1566

References

- Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov. Approximate nearest neighbor algorithm based on navigable small world graphs. Information Systems, 45:61 – 68, 2014.
- Kostas Zoumpatianos, Stratos Idreos, Themis Palpanas: RINSE: Interactive Data Series Exploration with ADS+. Proc. VLDB Endow. 8(12): 1912-1915 (2015)
- Kostas Zoumpatianos, Yin Lou, Themis Palpanas, Johannes Gehrke: Query Workloads for Data Series Indexes. KDD 2015: 1603-1612
- Q. Huang, J. Feng, Y. Zhang, Q. Fang, and W. Ng. Query-aware Locality-sensitive Hashing for Approximate Nearest Neighbor Search. PVLDB, 9(1):1–12, 2015
- Kostas Zoumpatianos, Yin Lou, Ioana Ileana, Themis Palpanas, Johannes Gehrke: Generating data series query workloads. VLDB J. 27(6): 823-846 (2018)
- Themis Palpanas: Big Sequence Management: A glimpse of the Past, the Present, and the Future. SOFSEM 2016: 63-80
- Kostas Zoumpatianos, Stratos Idreos, Themis Palpanas: ADS: the adaptive data series index. VLDB J. 25(6): 843-866 (2016)
- Djamel Edine Yagoubi, Reza Akbarinia, Florent Masseglia, Themis Palpanas: DPiSAX: Massively Distributed Partitioned iSAX. ICDM 2017: 1135-1140
- J. Wang, T. Zhang, j. song, N. Sebe, and H. T. Shen. A survey on learning to hash. TPAMI, 40(4): 769-790 (2018).
- Haridimos Kondylakis, Niv Dayan, Kostas Zoumpatianos, Themis Palpanas: Coconut: A Scalable Bottom-Up Approach for Building Data Series Indexes. Proc. VLDB Endow. 11(6): 677-690 (2018)
- Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, Anastasia Bezerianos: Comparing Similarity Perception in Time Series Visualizations. IEEE Trans. Vis. Comput. Graph. 25(1): 523-533 (2019)

References

- A. Mueen, Y. Zhu, M. Yeh, K. Kamgar, K. Viswanathan, C. Gupta, and E. Keogh. The Fastest Similarity Search Algorithm for Time Series Subsequences under Euclidean Distance, August 2017. http://www.cs.unm.edu/~mueen/ FastestSimilaritySearch.html.
- M. Norouzi and D. J. Fleet. Cartesian K-Means. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '13, pages 3017–3024, 2013
- M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In VISAPP International Conference on Computer Vision Theory and Applications, pages 331–340, 2009
- A. Arora, S. Sinha, P. Kumar, and A. Bhattacharya. HD-index: Pushing the Scalability-accuracy Boundary for Approximate kNN Search in High-dimensional Spaces. PVLDB, 11(8):906–919, 2018
- Y. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. CoRR, abs/1603.09320, 2016
- T. Ge, K. He, Q. Ke, and J. Sun. Optimized Product Quantization. IEEE Trans. Pattern Anal. Mach. Intell., 36(4):744–755, Apr. 2014
- A. Babenko and V. Lempitsky. The Inverted MultiIndex. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(6):1247–1260, June 2015.
- C. Fu, C. Xiang, C. Wang, and D. Cai. Fast Approximate Nearest Neighbor Search with the Navigating Spreading-out Graph. PVLDB, 12(5):461–474, 2019.

References

- Kostas Zoumpatianos, Themis Palpanas: Data Series Management: Fulfilling the Need for Big Sequence Analytics. ICDE 2018: 1677-1678
- Michele Linardi, Themis Palpanas: ULISSE: ULtra Compact Index for Variable-Length Similarity Search in Data Series. ICDE 2018: 1356-1359
- Djamel Edine Yagoubi, Reza Akbarinia, Florent Masseglia, Themis Palpanas: Massively Distributed Time Series Indexing and Querying. IEEE Trans. Knowl. Data Eng. 32(1): 108-120 (2020)
- Botao Peng, Panagiota Fatourou, Themis Palpanas: ParIS: The Next Destination for Fast Data Series Indexing and Query Answering. BigData 2018: 791-800
- Anna Gogolou, Theophanis Tsandilas, Karima Echihabi, Anastasia Bezerianos, Themis Palpanas: Data Series Progressive Similarity Search with Probabilistic Quality Guarantees. SIGMOD Conference 2020: 1857-1873
- Cagatay Turkay, Nicola Pezzotti, Carsten Binnig, Hendrik Strobelt, Barbara Hammer, Daniel A. Keim, Jean-Daniel Fekete, Themis Palpanas, Yunhai Wang, Florin Rusu: Progressive Data Science: Potential and Challenges. CoRR abs/1812.08032 (2018)
- Michele Linardi, Themis Palpanas: Scalable, Variable-Length Similarity Search in Data Series: The ULISSE Approach. Proc. VLDB Endow. 11(13): 2236-2248 (2018)

References

- Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas, Houda Benbrahim: The Lernaean Hydra of Data Series Similarity Search: An Experimental Evaluation of the State of the Art. Proc. VLDB Endow. 12(2): 112-127 (2018)
- Haridimos Kondylakis, Niv Dayan, Kostas Zoumpatianos, Themis Palpanas: Coconut Palm: Static and Streaming Data Series Exploration Now in your Palm. SIGMOD Conference 2019: 1941-1944
- Themis Palpanas, Volker Beckmann: Report on the First and Second Interdisciplinary Time Series Analysis Workshop (ITISA). SIGMOD Rec. 48(3): 36-40 (2019)
- Oleksandra Levchenko, Boyan Kolev, Djamel Edine Yagoubi, Dennis E. Shasha, Themis Palpanas, Patrick Valduriez, Reza Akbarinia, Florent Masseglia: Distributed Algorithms to Find Similar Time Series. ECML/PKDD (3) 2019: 781-785
- Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas, Houda Benbrahim: Return of the Lernaean Hydra: Experimental Evaluation of Data Series Approximate Similarity Search. Proc. VLDB Endow. 13(3): 403-420 (2019)
- Themis Palpanas. Evolution of a Data Series Index - The iSAX Family of Data Series Indexes. CCIS, 1197 (2020)

References

- Botao Peng, Panagiota Fatourou, Themis Palpanas: MESSI: In-Memory Data Series Indexing. ICDE 2020: 337-348
- Haridimos Kondylakis, Niv Dayan, Kostas Zoumpatianos, Themis Palpanas: Coconut: sortable summarizations for scalable indexes over static and streaming data series. VLDB J. 28(6): 847-869 (2019)
- Botao Peng, Panagiota Fatourou, Themis Palpanas. Paris+: Data series indexing on multi-core architectures. TKDE, 2020
- Michele Linardi, Themis Palpanas. Scalable Data Series Subsequence Matching with ULISSE. VLDBJ 2020
- Anna Gogolou, Theophanis Tsandilas, Karima Echihabi, Anastasia Bezerianos, Themis Palpanas: Data Series Progressive Similarity Search with Probabilistic Quality Guarantees. SIGMOD Conference 2020: 1857-1873
- Karima Echihabi, Kostas Zoumpatianos, and Themis Palpanas. Scalable Machine Learning on High-Dimensional Vectors: From Data Series to Deep Network Embeddings. In WIMS, 2020
- Botao Peng, Panagiota Fatourou, Themis Palpanas. SING: Sequence Indexing Using GPUs. ICDE, 2021

References

- InfluxDB: https://www.influxdata.com/
- Timescale: https://www.timescale.com
- Beringei: https://github.com/facebookarchive/beringei
- Druid: https://druid.apache.org
- Prometheus: https://Prometheus.io
- CrateDB: https://crate.io
- IoTDb: https://iotdb.apache.org
- OpenTSDB: http://opentsdb.net/
- QuasarDB: https://www.quasardb.net/
- Timestream: https://aws.amazon.com/timestream/