High-Dimensional Similarity Search for Scalable Data Science

Karima Echihabi

Mohammed VI Polytechnic University

Kostas Zoumpatianos

Snowflake Computing

work done while at Harvard University & University of Paris

Themis Palpanas

University of Paris & French University Institute (IUF)

International Conference on Data Engineering (ICDE), April 2021

Questions This Tutorial Answers

- how important is high-dimensional data nowadays?

- what types of analyses are performed on high-d data?
- how can we speed up such an analysis?

- what are the different kinds of similarity search?
- what are the state-of-the-art high-d similarity search methods?
- how do methods designed for data series compare to those designed for general high-d vector similarity search?

- what are the open research problems in this area?
- what are the connections to deep learning?

Acknowledgements

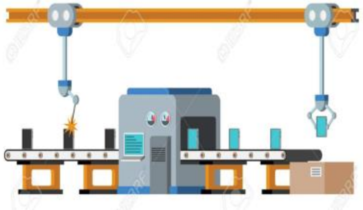
- thanks for slides to
  - Michail Vlachos
  - Eamonn Keogh
  - Panagiotis Papapetrou
  - George Kollios
  - Dimitrios Gunopulos
  - Christos Faloutsos
  - Panos Karras
  - Peng Wang
  - Liang Zhang
  - Reza Akbarinia
  - Marco Patella
  - Wei Wang
  - Yury Malkov
  - Matthijs Douze
  - Cong Fu
  - Arnab Bhattacharya
  - Qiang Huang
  - Artem Babenko
  - David Lowe

Introduction, Motivation

High-d data is everywhere

Finance

Paleontology



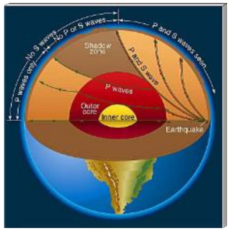
Manufacturing

Aviation

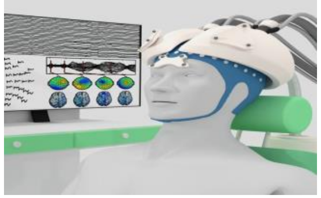
Agriculture

Astronomy

Criminology

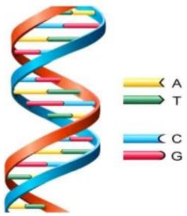


Seismology



Neuroscience

Medicine



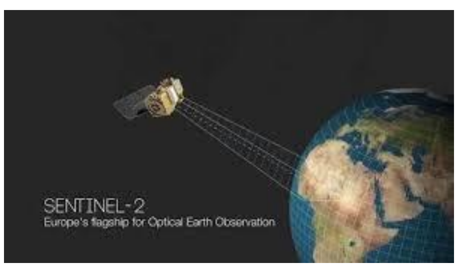
Biology

High-d collections are massive

≈ 500 ZB per year

≈ 130 TB

> 40 PB per day



> 5 TB per day

> 500 TB per day

1 PB = 1 thousand TB

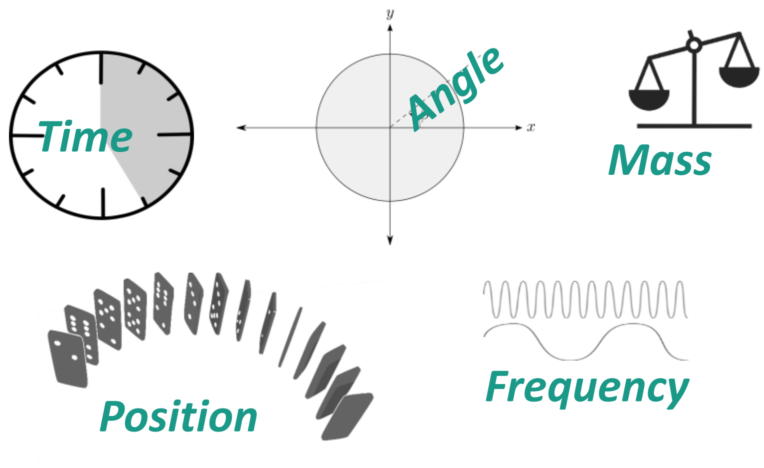
1 ZB = 1 billion TB

Popular High-d data

Popular High-d data

Data series

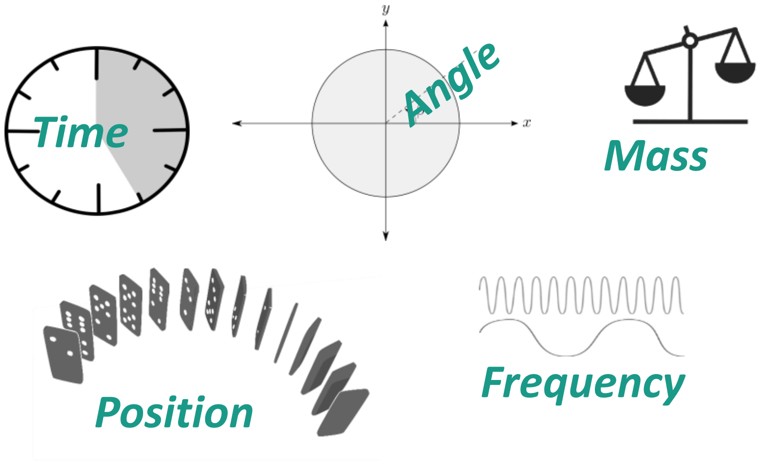
A collection of points ordered over a dimension



Popular High-d data

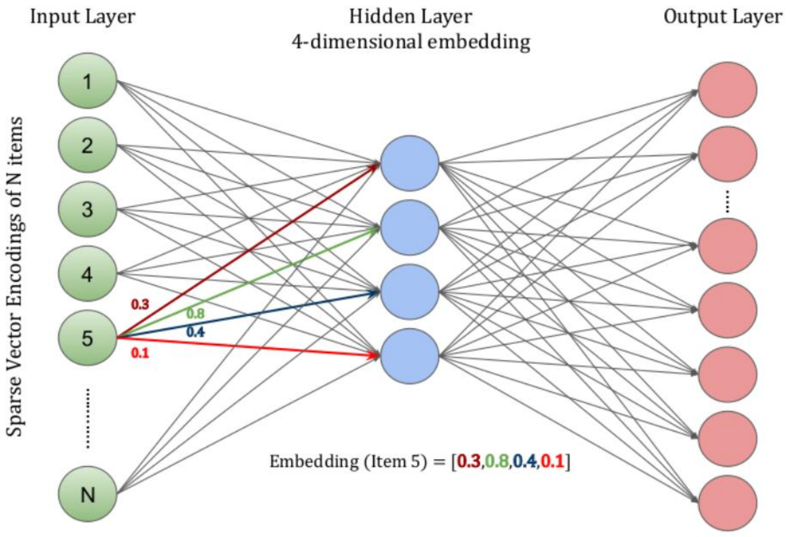
Data series

A collection of points ordered over a dimension



Deep Embeddings

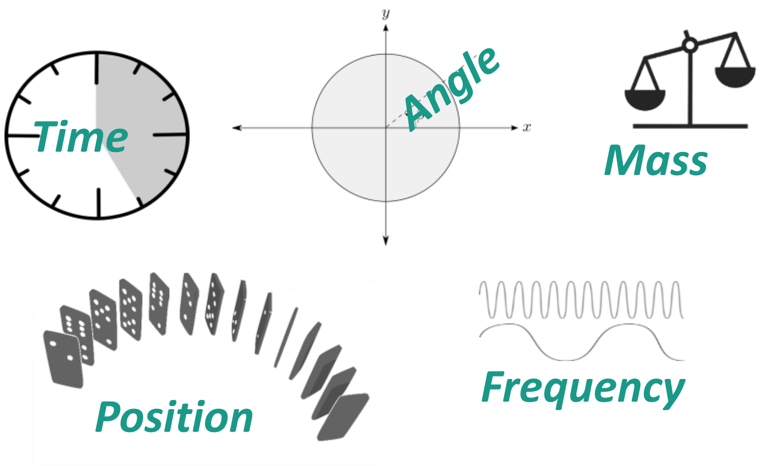
A low-d vector learned from data using a DNN



Popular High-d data

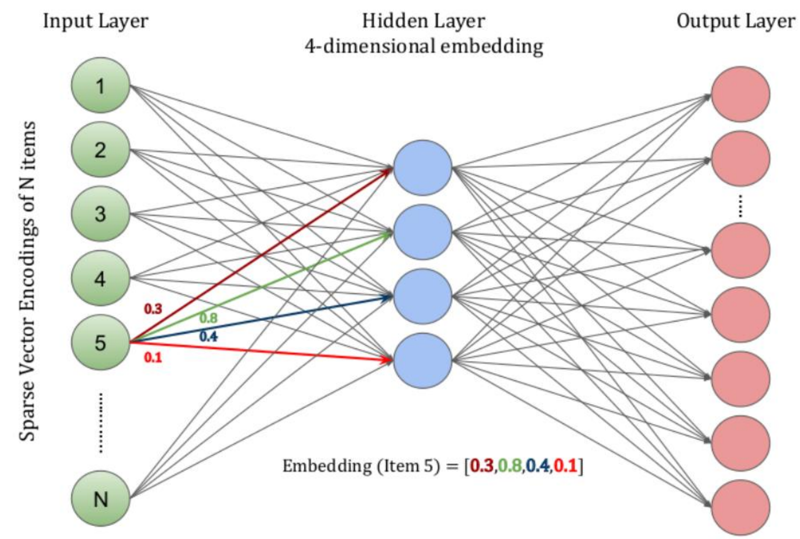
Data series

A collection of points ordered over a dimension



Deep Embeddings

A low-d vector learned from data using a DNN

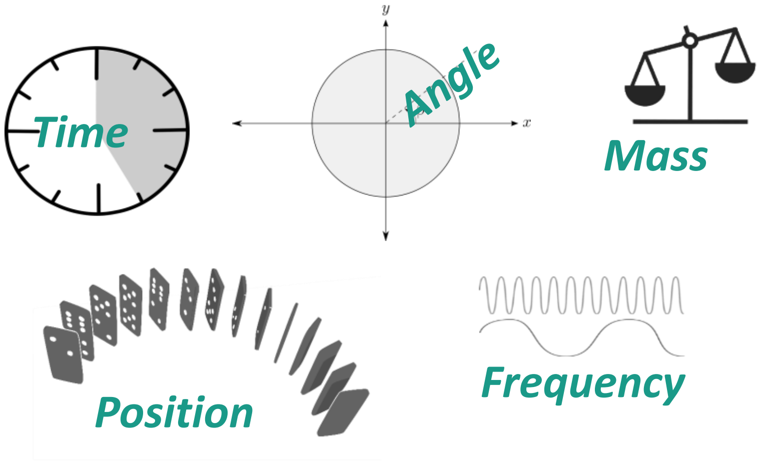


embedded
text, images, video, graphs, etc.

Popular High-d data

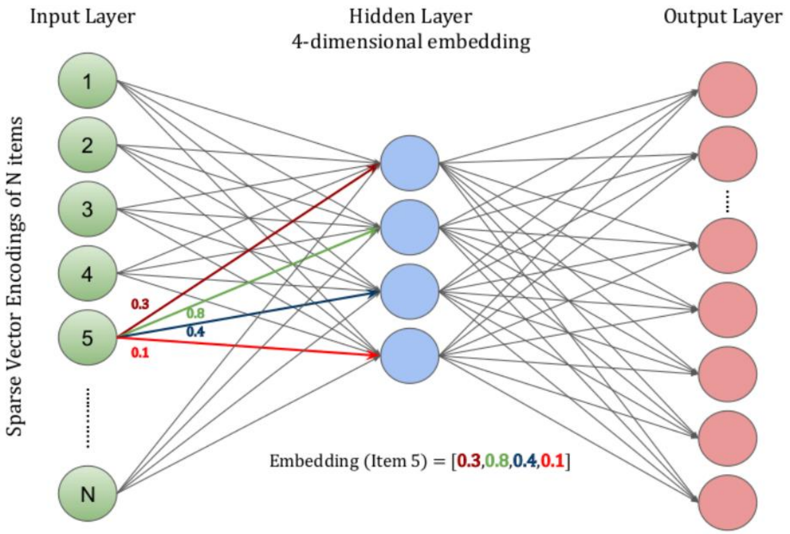
Data series

A collection of points ordered over a dimension



Deep Embeddings

A low-d vector learned from data using a DNN

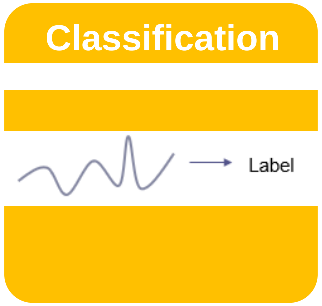


embedded
text, images, video, graphs, etc.

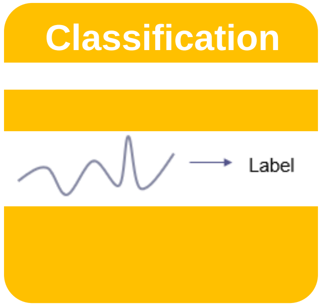
High-d data -> High-d vector

Echihabi, Zoumpatianos, Palpanas - ICDE 2021

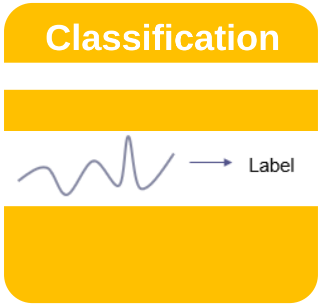
Extracting value requires analytics



Extracting value requires analytics



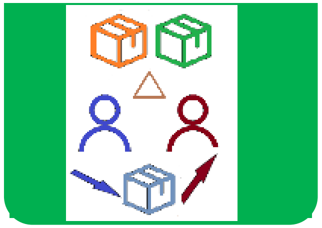
Extracting value requires analytics



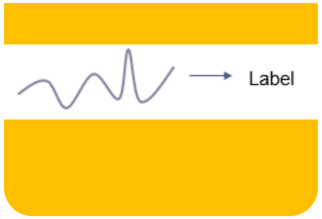
Extracting value requires analytics

Clustering

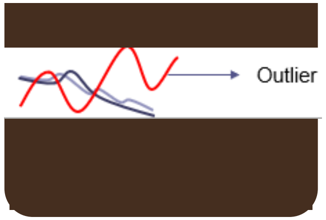
Recommendation



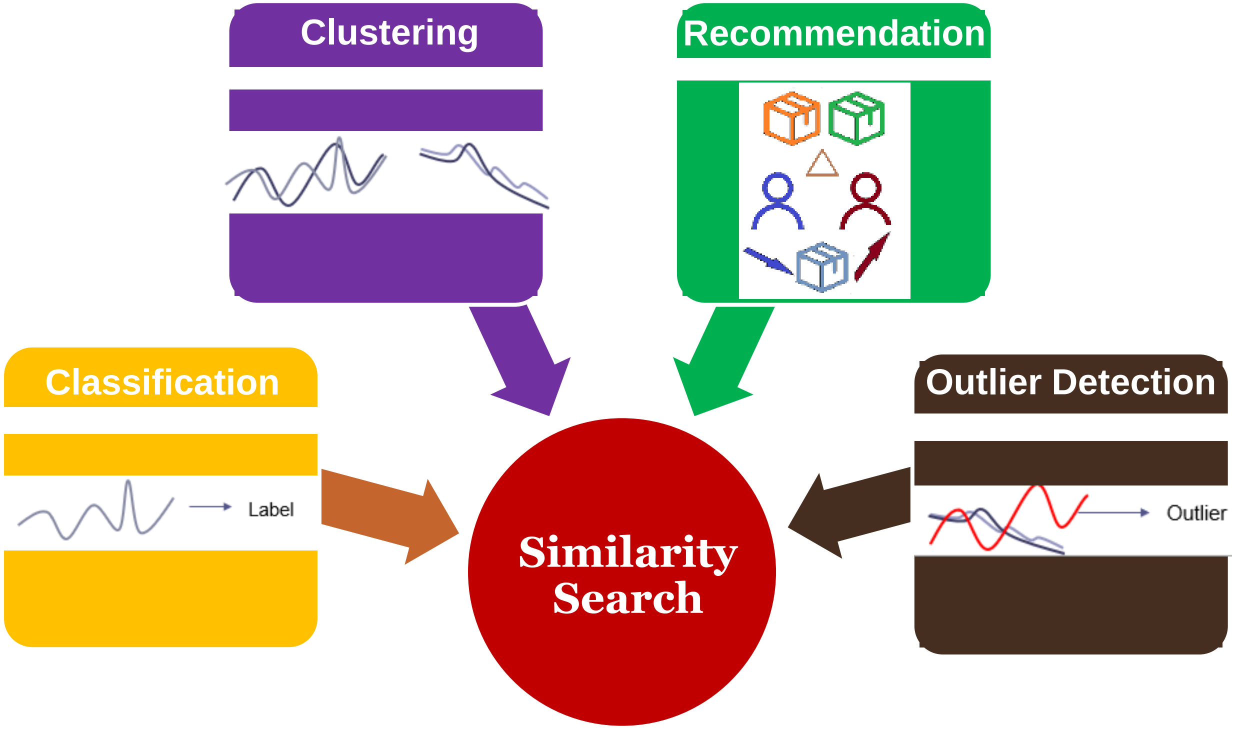
Classification



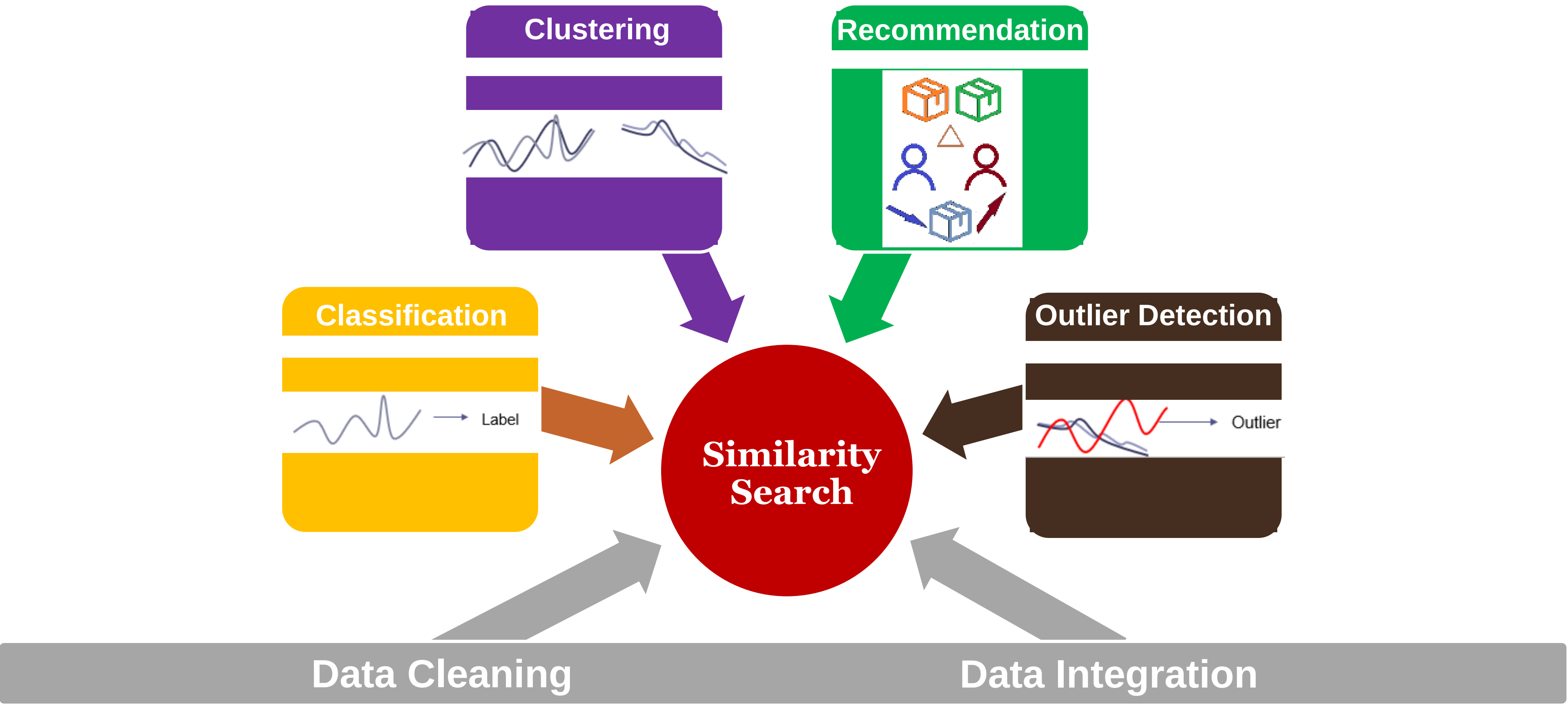
Outlier Detection



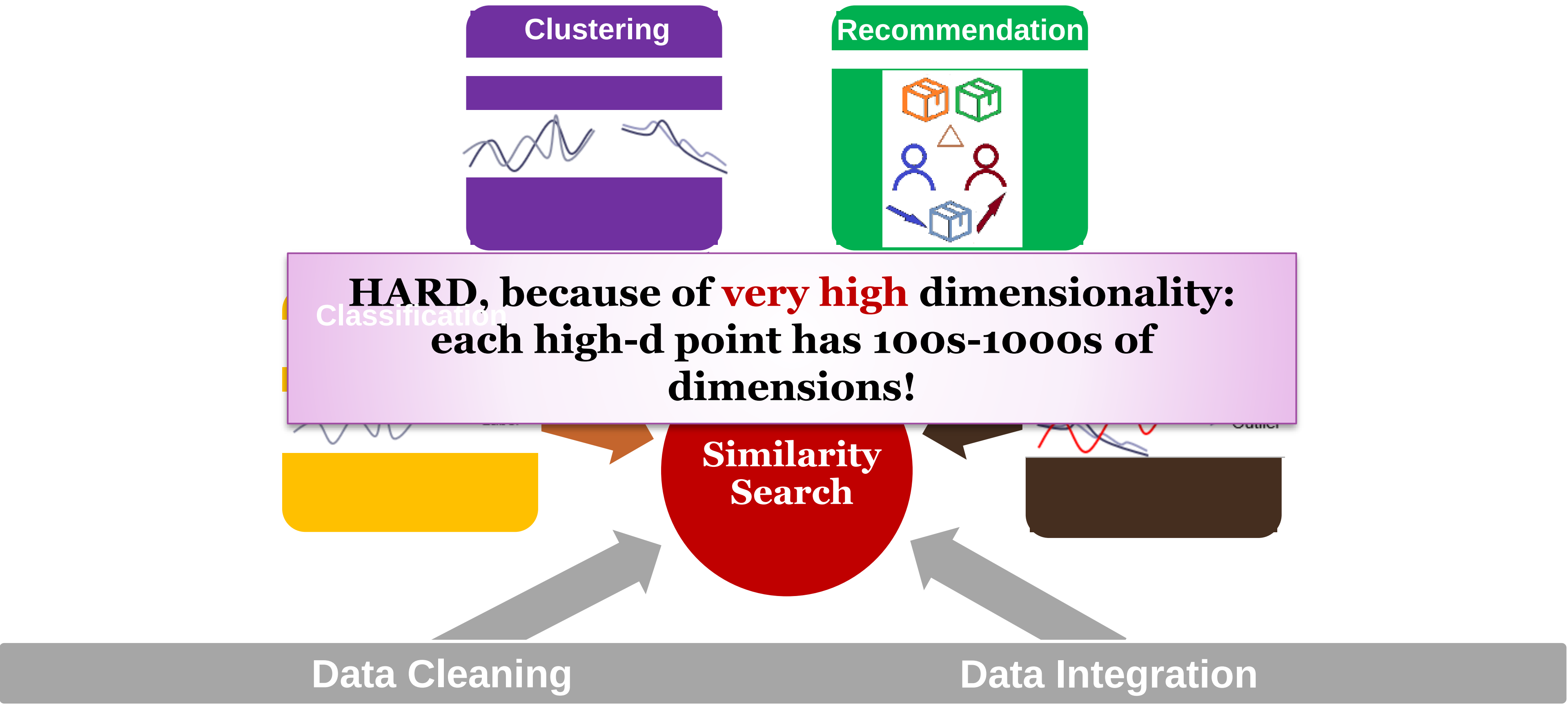
Extracting value requires analytics



Extracting value requires analytics

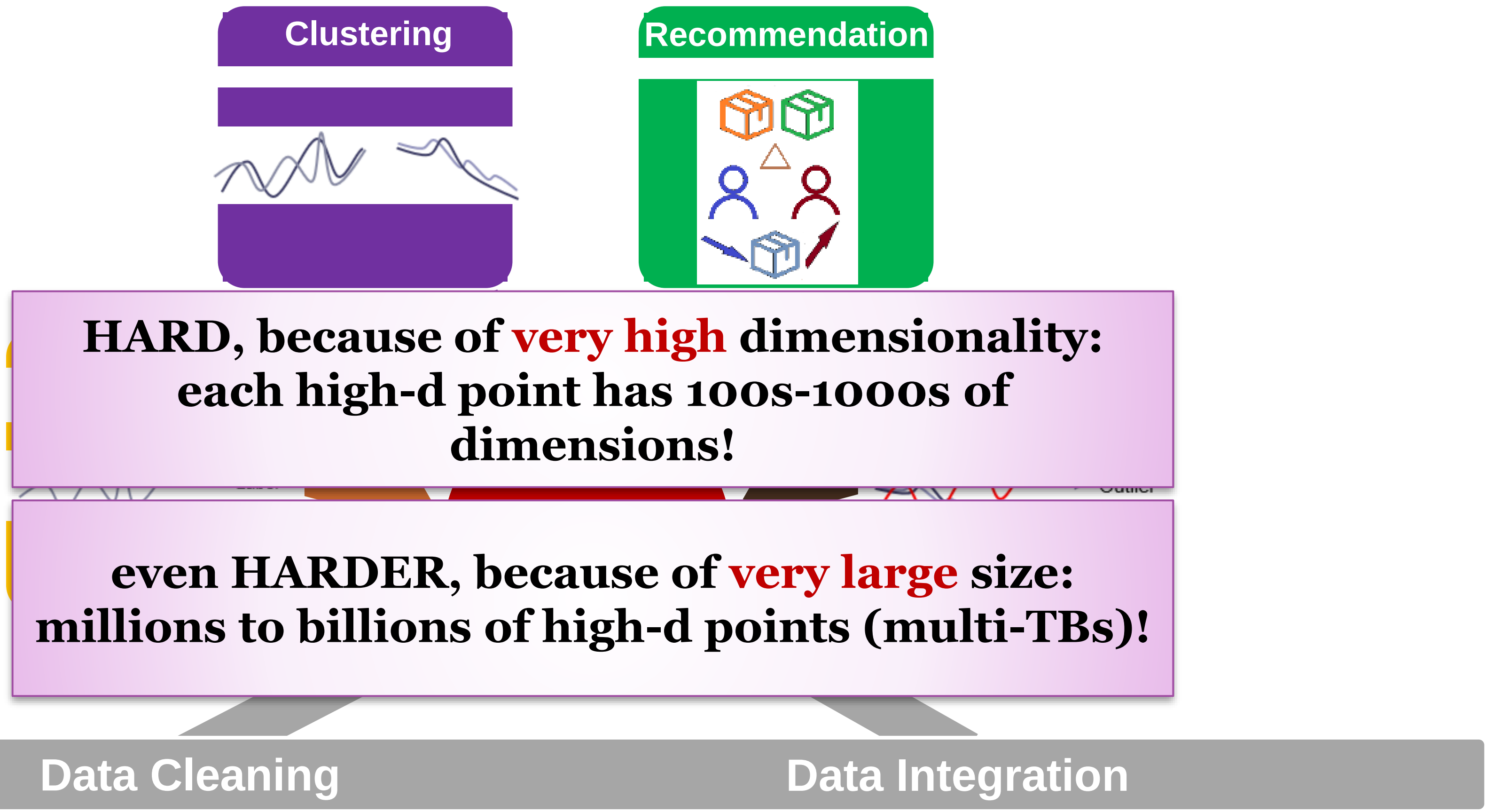


Extracting value requires analytics



HARD, because of very high dimensionality: each high-d point has 100s-1000s of dimensions!

Extracting value requires analytics



HARD, because of very high dimensionality: each high-d point has 100s-1000s of dimensions!

even HARDER, because of very large size: millions to billions of high-d points (multi-TBs)!

High-d Similarity Search

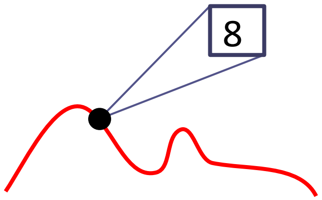
High-d Similarity Search Problem Variations

Problem Variations

Series

Problem Variations

Series

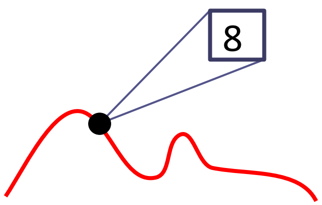


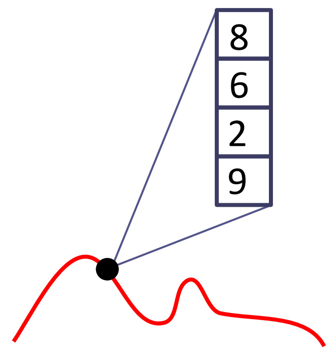
Univariate

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

Problem Variations

Series





Univariate

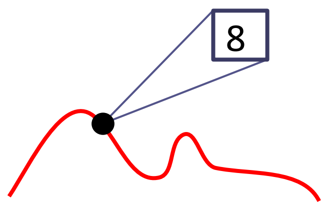
each point represents one value (e.g., temperature)

Multivariate

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

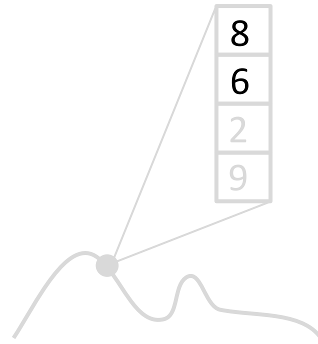
Problem Variations

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

Data Series Distance Measures

Publications

Ding-PVLDB'08

Paparrizos-SIGMOD'20

- similarity search is based on measuring distance between sequences
- dozens of distance measures have been proposed
  - lock-step
    - Minkowski, Manhattan, Euclidean, Maximum, DISSIM, ...
  - sliding
    - Normalized Cross-Correlation, SBD, ...
  - elastic
    - DTW, LCSS, MSM, EDR, ERP, Swale, ...
  - kernel-based
    - KDTW, GAK, SINK, ...
  - embedding
    - GRAIL, RWS, SPIRAL, ...

Problem Variations

Data Series Distance Measures

Publications

Ding-PVLDB'08

Paparrizos-SIGMOD'20

- similarity search is based on measuring distance between sequences
- dozens of distance measures have been proposed
  - lock-step
    - Minkowski, Manhattan, Euclidean, Maximum, DISSIM, …
  - sliding
    - Normalized Cross-Correlation, SBD, …
  - elastic
    - DTW, LCSS, MSM, EDR, ERP, Swale, …
  - kernel-based
    - KDTW, GAK, SINK, …
  - embedding
    - GRAIL, RWS, SPIRAL, …

Problem Variations

High-d Vectors Distance Measures

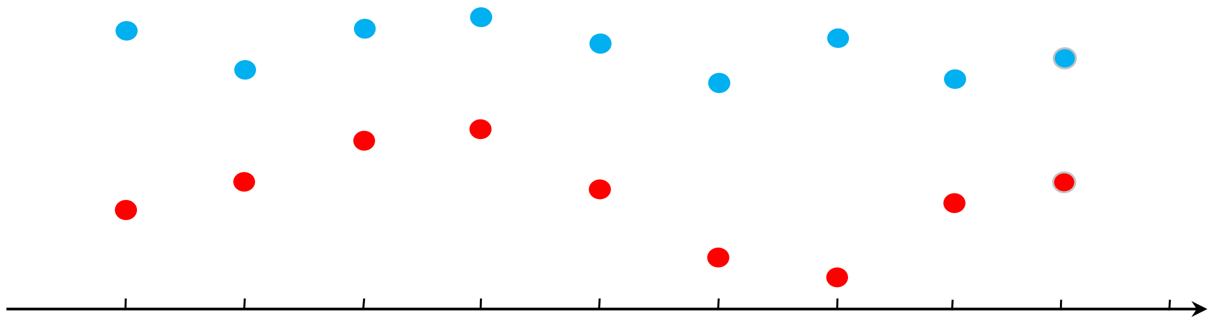
- similarity search is based on measuring distance between vectors
- A variety of distance measures have been proposed
  - L_p distances ($0<p\leq 2, \infty$), (Euclidean for $p = 2$)
  - Cosine distance
  - Correlation
  - Hamming distance
  - …

Problem Variations

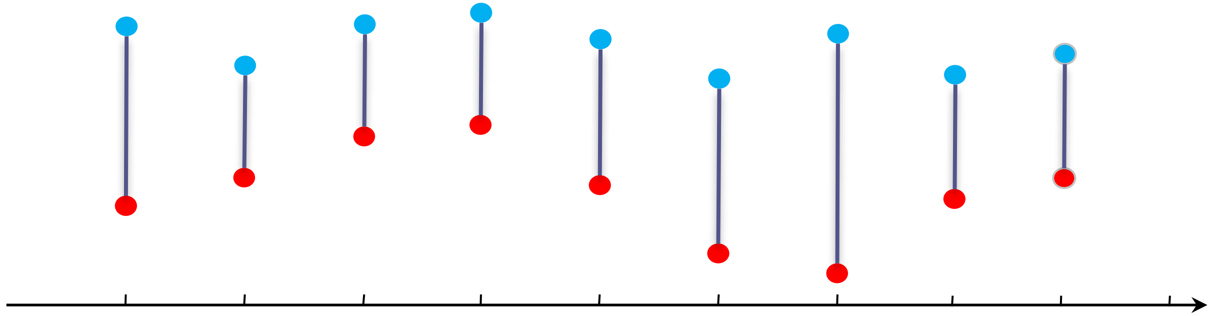
High-d Vectors Distance Measures

- similarity search is based on measuring distance between vectors
- A variety of distance measures have been proposed
  - L_p distances ($0<p\leq 2, \infty$), (Euclidean for $p = 2$)
  - Cosine distance
  - Correlation
  - Hamming distance
  - …

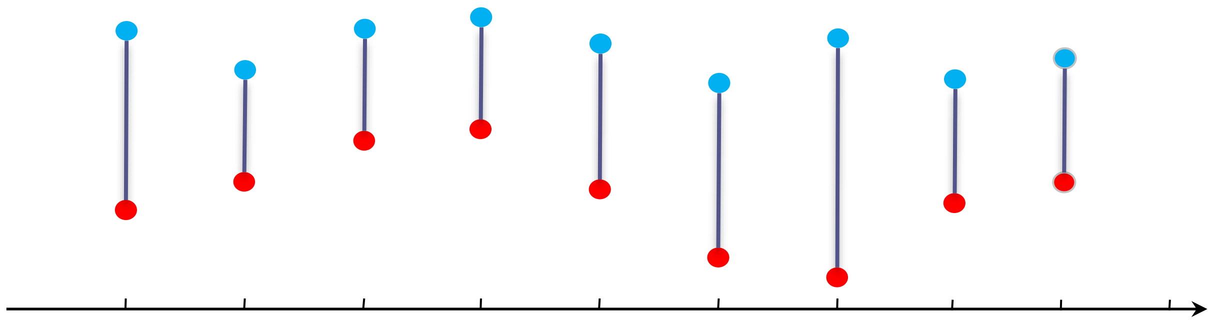
Euclidean Distance



Euclidean Distance



Euclidean Distance



- Euclidean distance
  - pair-wise point distance

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

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

Publications

Keogh-DMKD'03

- smart implementation of distance function

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

Publications

Keogh-DMKD'03

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

Publications

Keogh-DMKD'03

- 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

Publications

Keogh-DMKD'03

Similarity Matching
Fast Euclidean Distance

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

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

Publications

Keogh-DMKD'03

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

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):

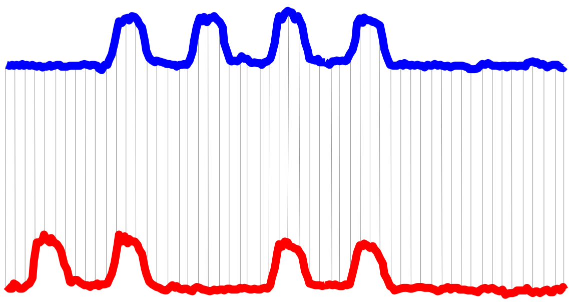
$$PC = \frac{1}{n-1} \sum_{i=1}^{n} x_i \cdot y_i \quad \text{and} \quad ED^2 = 2n(n-1) - 2\sum_{i=1}^{n} x_i y_i$$

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.

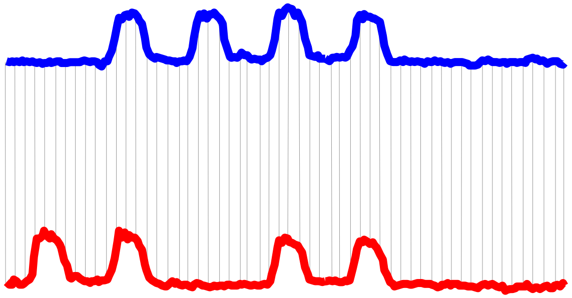
Distance Measures: LCSS against Euclidean, DTW

- Euclidean
  - rigid

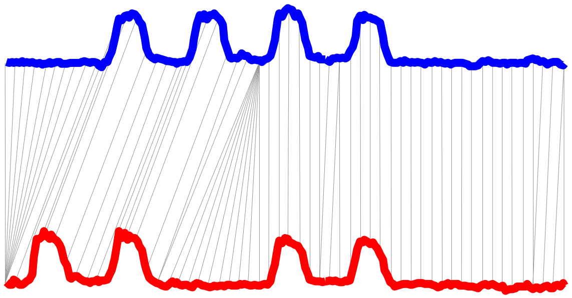


Distance Measures: LCSS against Euclidean, DTW

- Euclidean
  - rigid

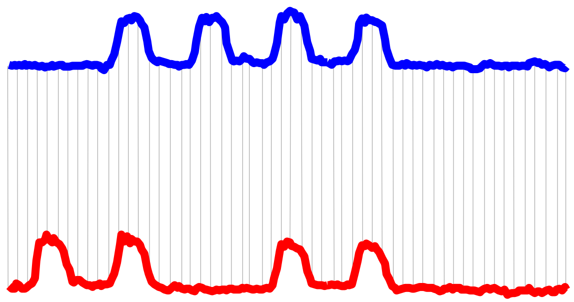


- Dynamic Time Warping (DTW)
  - allows local scaling

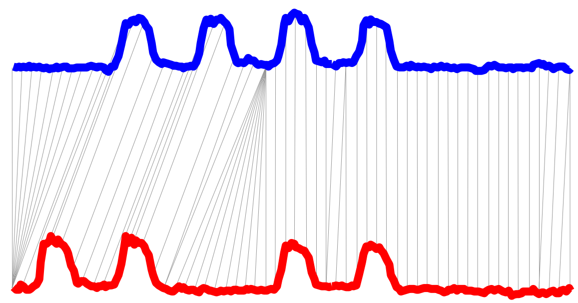


Distance Measures: LCSS against Euclidean, DTW

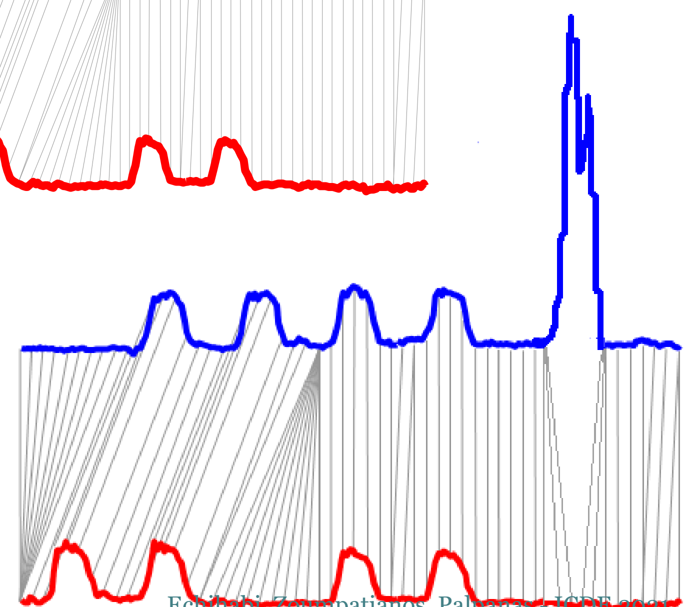
- Euclidean
  - rigid



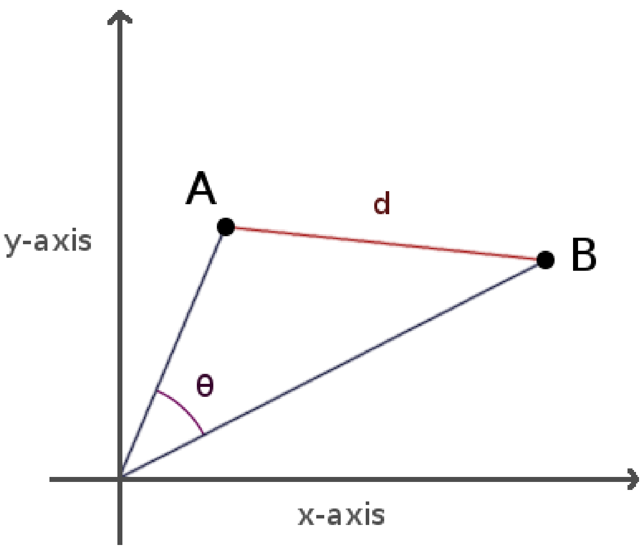
- Dynamic Time Warping (DTW)
  - allows local scaling



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



Distance Measures: Cosine Distance

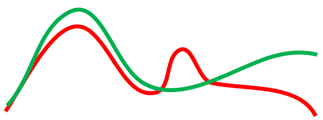


$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

- Cosine distance = 1 - cosine similarity

Problem Variations

Queries



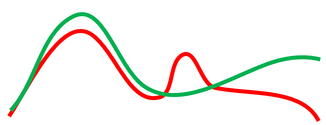
Whole matching

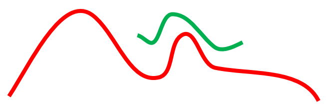
Entire query

Entire candidate

Problem Variations

Queries





<u>Whole matching</u>

Entire query

Entire candidate

<u>Subsequence matching</u>

Entire query

A subsequence of a candidate

Problem Variations

Queries

Nearest Neighbor (1NN)
k-Nearest Neighbor (kNN)
Farthest Neighbor
epsilon-Range

and more…

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

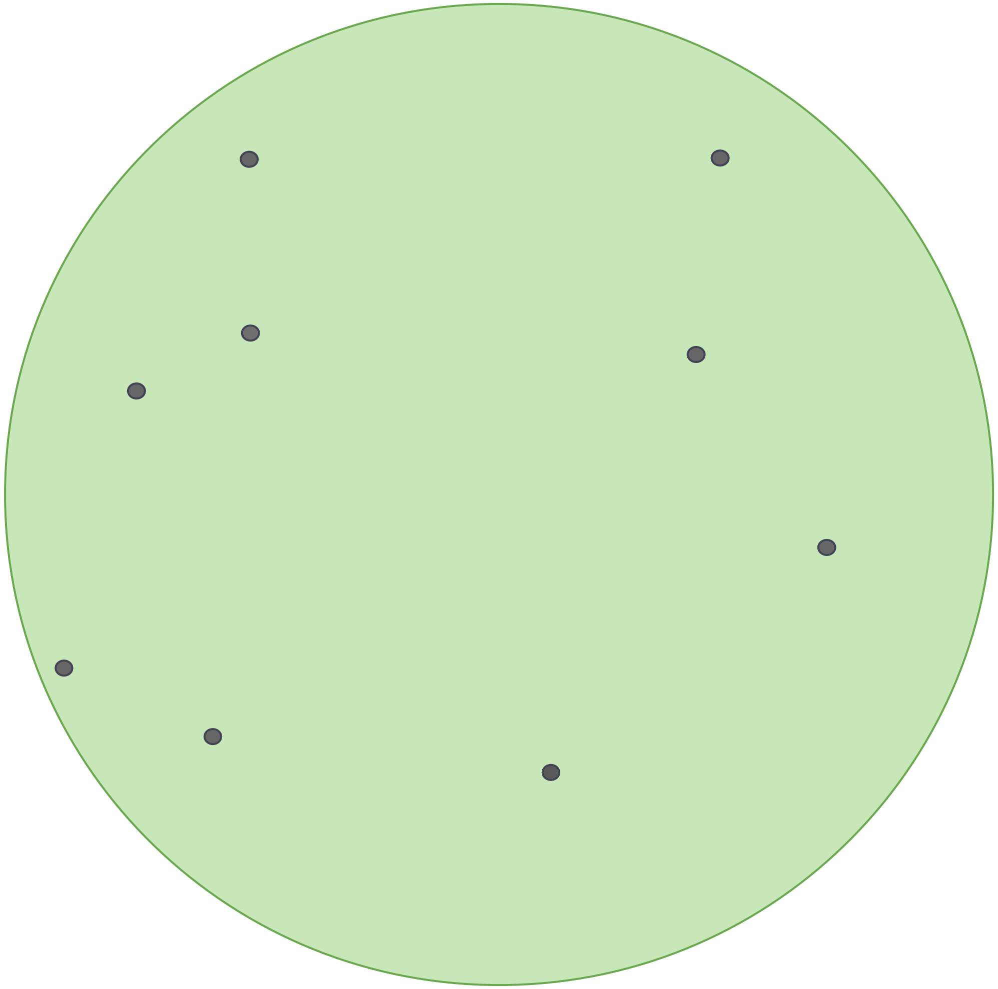
Queries

- Nearest Neighbor (1NN)
- k-Nearest Neighbor (kNN)
- Farthest Neighbor
- epsilon-Range
- And more…

Nearest Neighbor (NN) Queries…

Publications

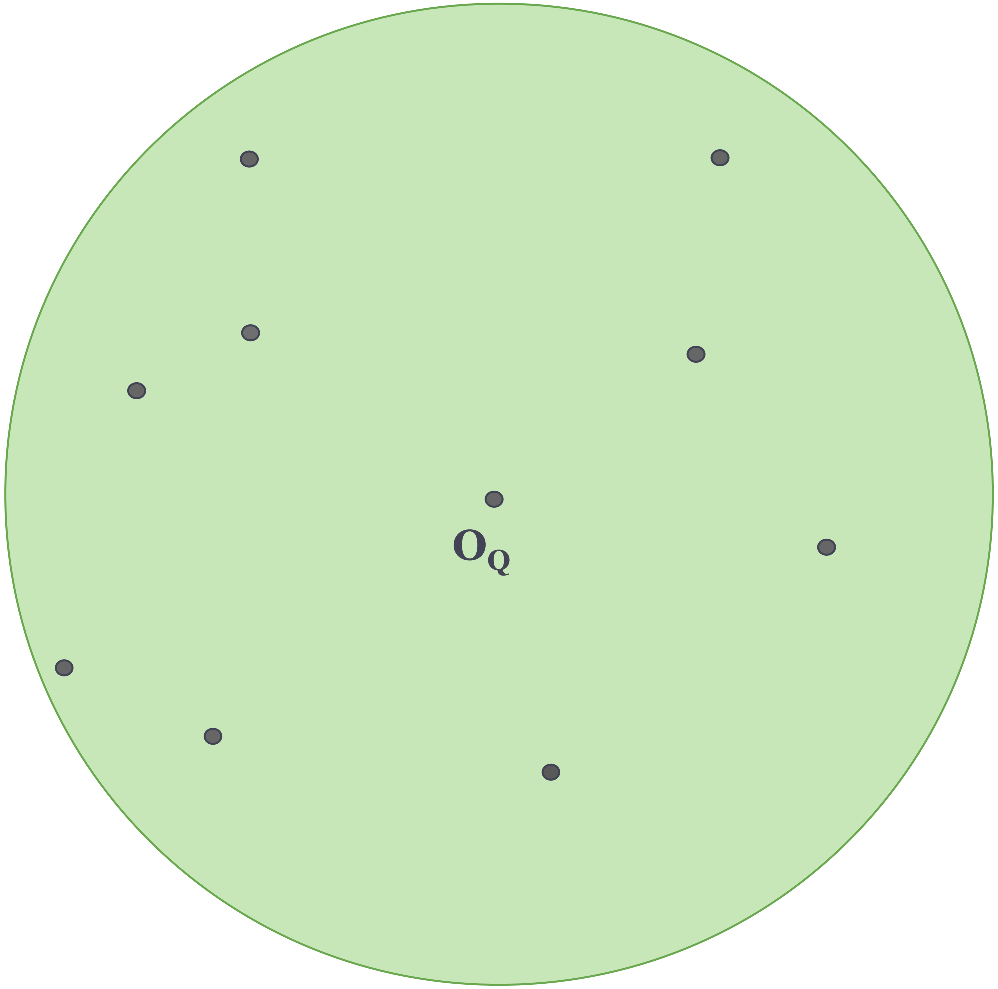
Echihabi et al. PVLDB'19



Nearest Neighbor (NN) Queries…

Publications

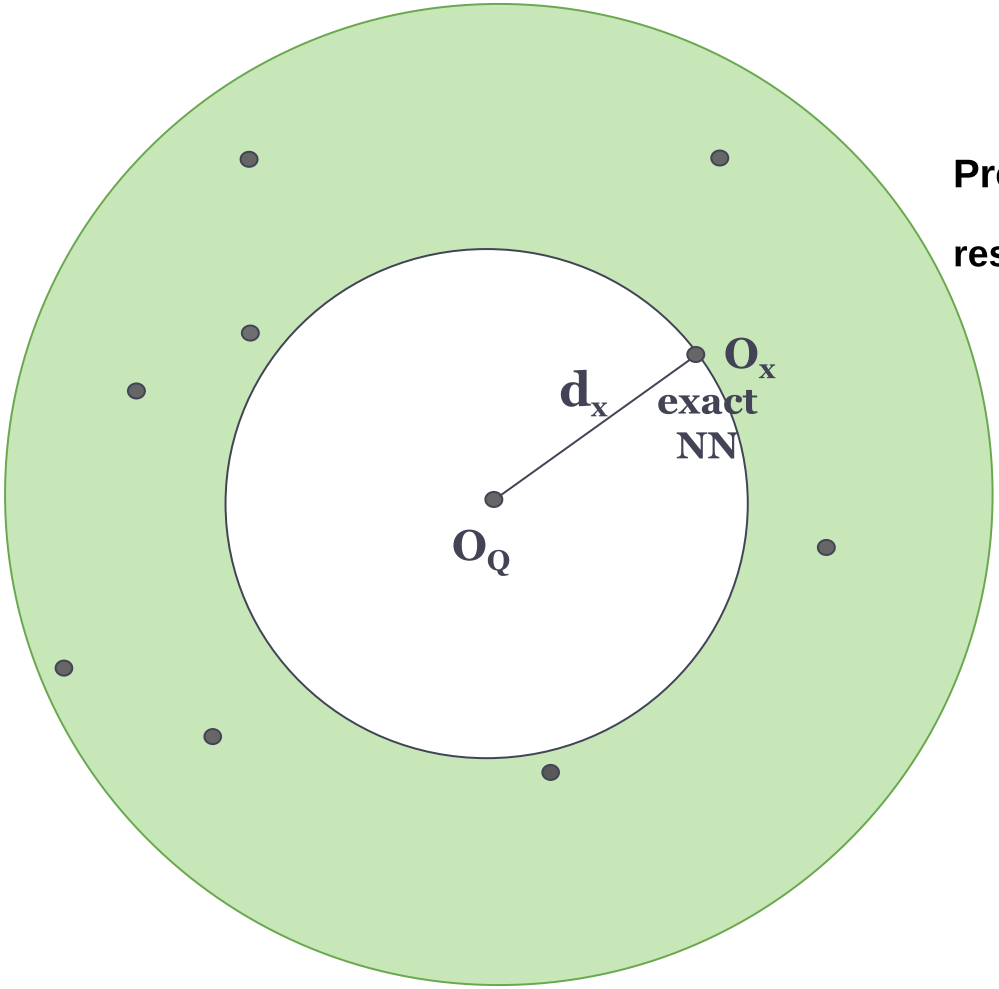
Echihabi et al. PVLDB‘19



Nearest Neighbor (NN) Queries…

Publications

Echihabi et al. PVLDB‘19



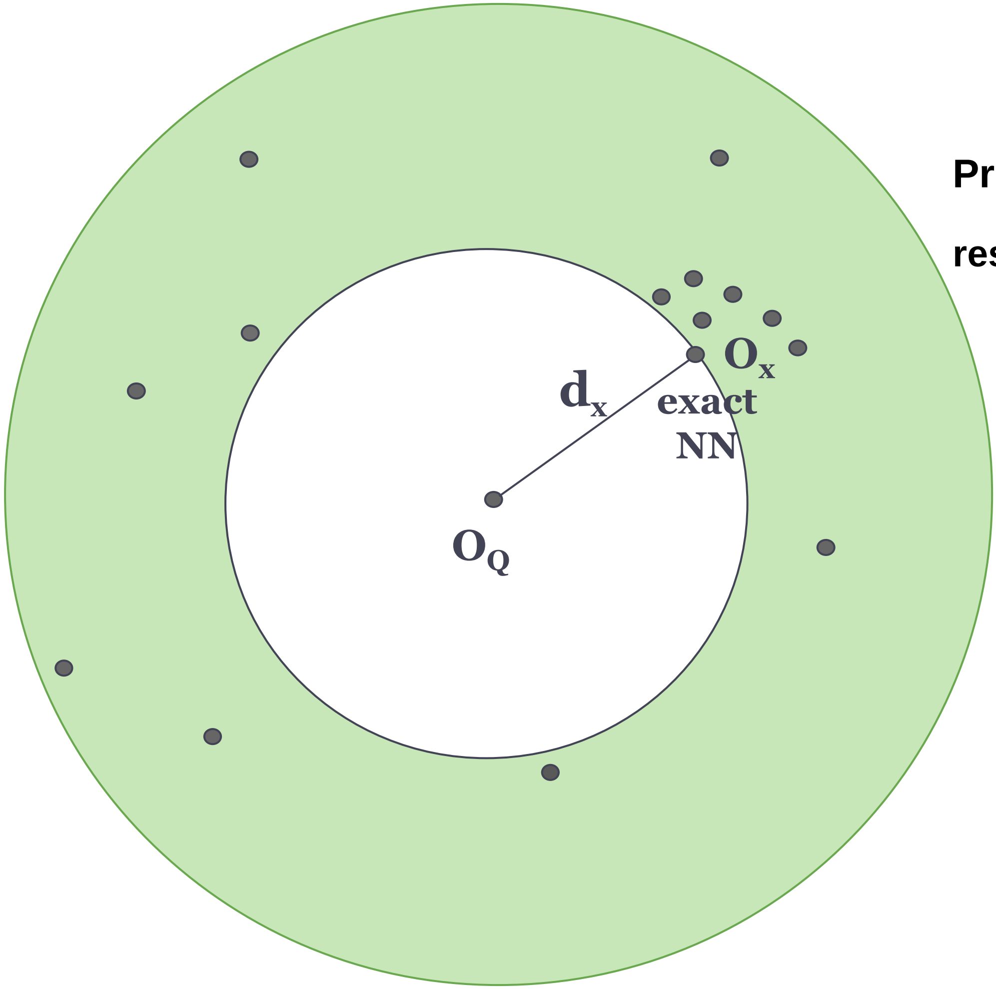
Prob($d_x = \min\{d_i\}$) = 1

result is exact NN

Nearest Neighbor (NN) Queries…

Publications

Echihabi et al. PVLDB‘19



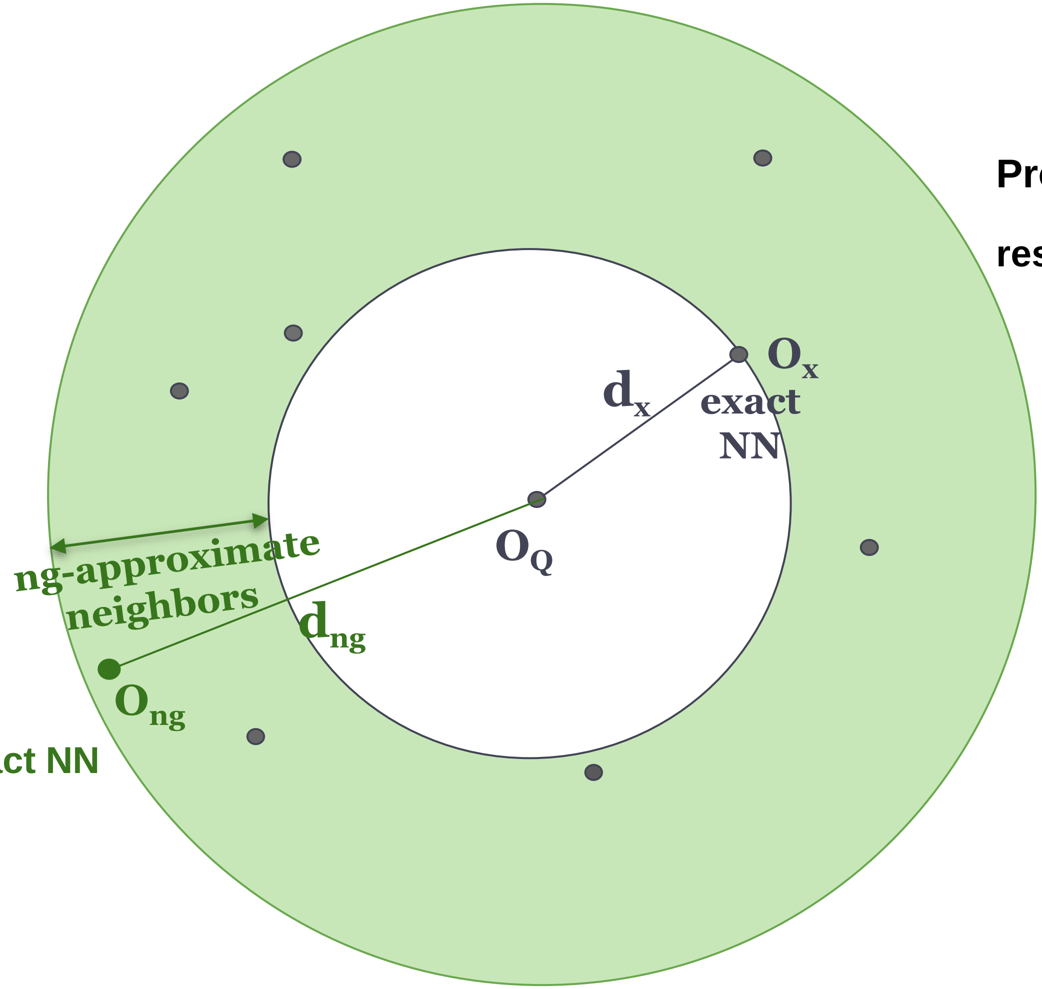
Prob($d_x = \min\{d_i\}$) = 1

result is exact NN

Nearest Neighbor (NN) Queries…

Publications

Echihabi et al. PVLDB'19



Prob($d_x = \min\{d_i\}$) = 1

result is exact NN

Prob(d_{ng} <>= ?) = ?

result within ? of exact NN

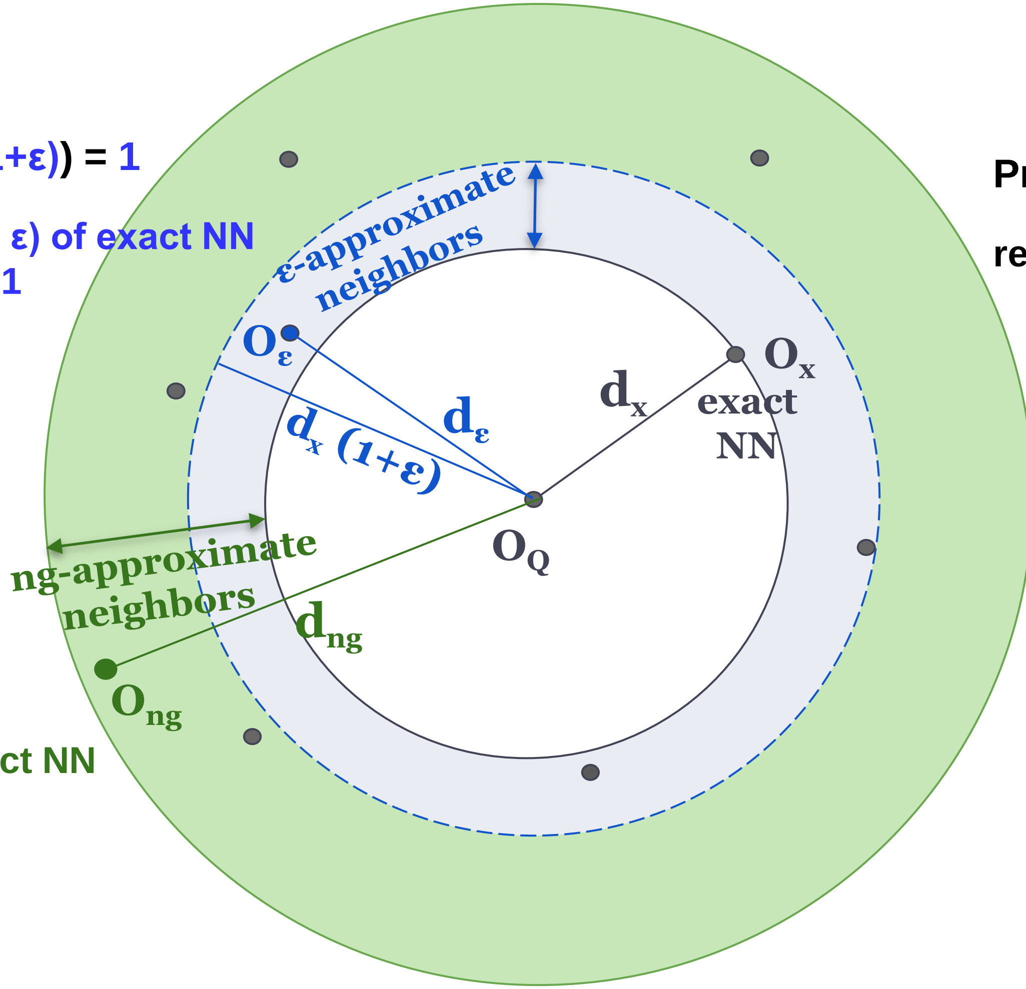
Nearest Neighbor (NN) Queries…

Publications

Echihabi et al. PVLDB‘19

Prob(d_ε <= d_x (1+ε)) = 1

result within (1+ ε) of exact NN
with probability 1



Prob($d_x = \min\{d_i\}$) = 1

result is exact NN

Prob(d_{ng} <>= ?) = ?

result within ? of exact NN

Nearest Neighbor (NN) Queries…

Publications

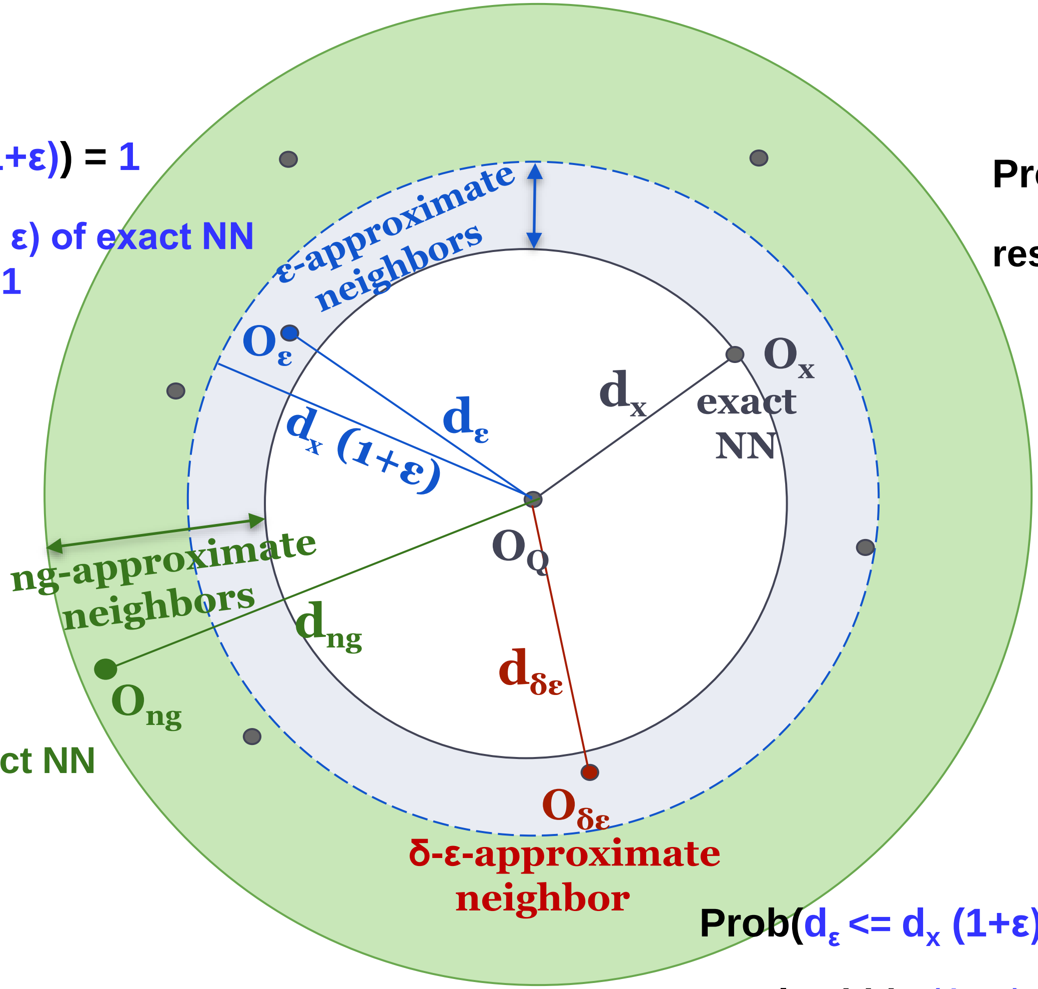
Echihabi et al. PVLDB'19

Prob(d_ε <= d_x (1+ε)) = 1

result within (1+ ε) of exact NN
with probability 1

Prob($d_x = \min\{d_i\}$) = 1

result is exact NN



Prob(d_{ng} <>= ?) = ?

result within ? of exact NN

Prob(d_ε <= d_x (1+ε)) >= δ

result within (1+ ε) of exact NN
with probability at least δ

Nearest Neighbor (NN) Queries…

Publications

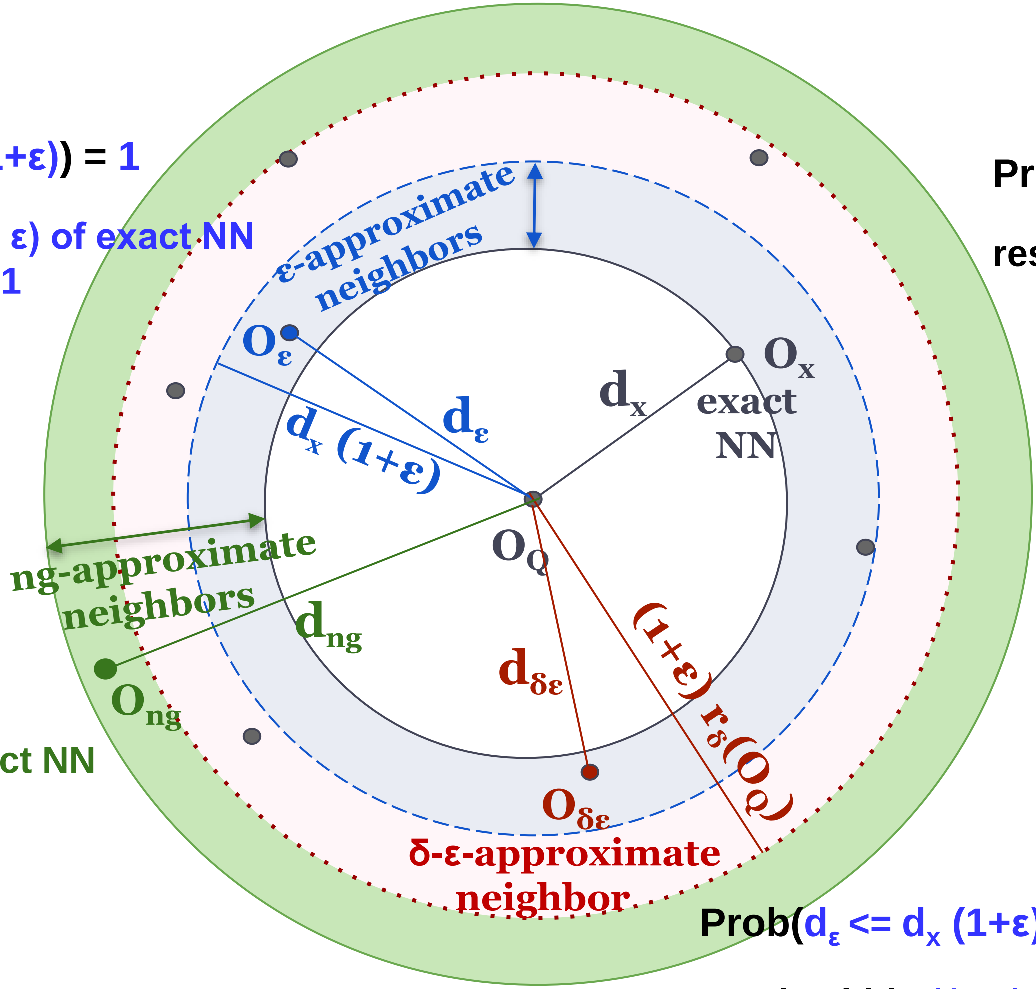
Echihabi et al. PVLDB'19

Prob($d_\varepsilon <= d_x (1+\varepsilon)$) = 1

result within (1+ ε) of exact NN with probability 1

Prob($d_x = \min\{d_i\}$) = 1

result is exact NN



Prob(d_{ng} <>= ?) = ?

result within ? of exact NN

Prob($d_\varepsilon <= d_x (1+\varepsilon)$) >= δ

result within (1+ ε) of exact NN with probability at least δ

Meaningfulness of NN queries in high-d spaces

Publications

Beyer et al. ICDT'99

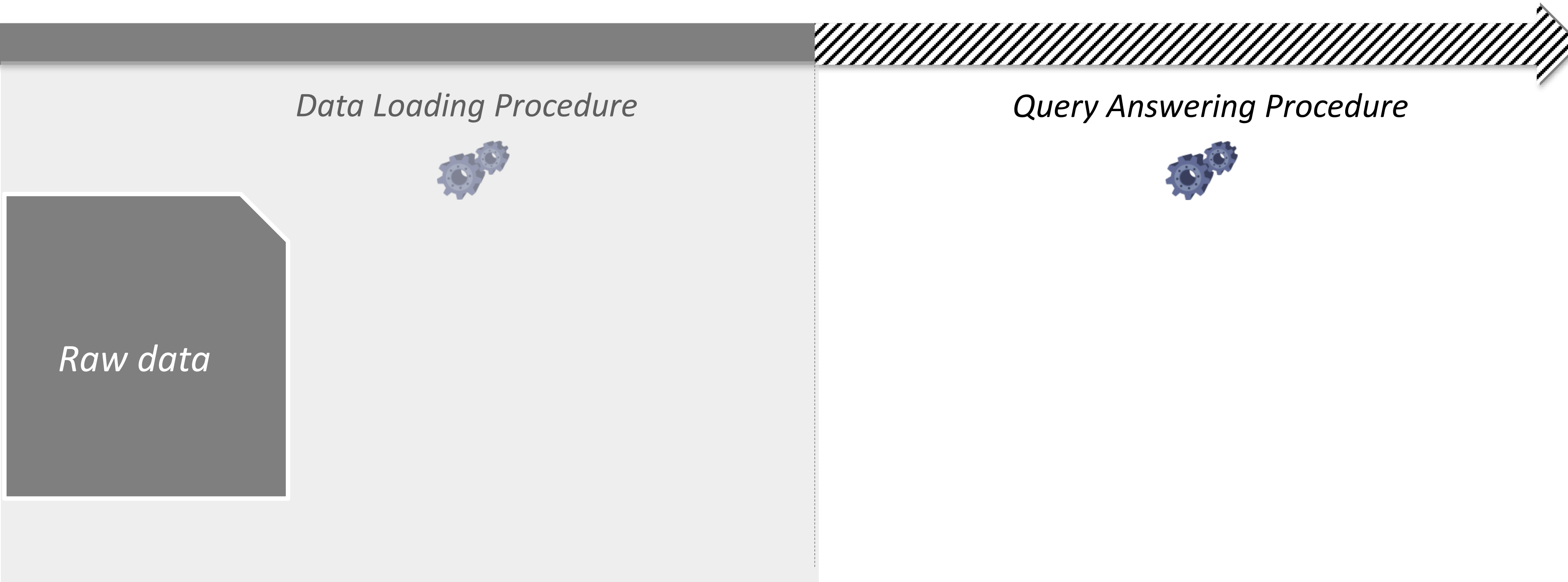
Aggarwal et al. ICDT'01

He et al. ICML'12

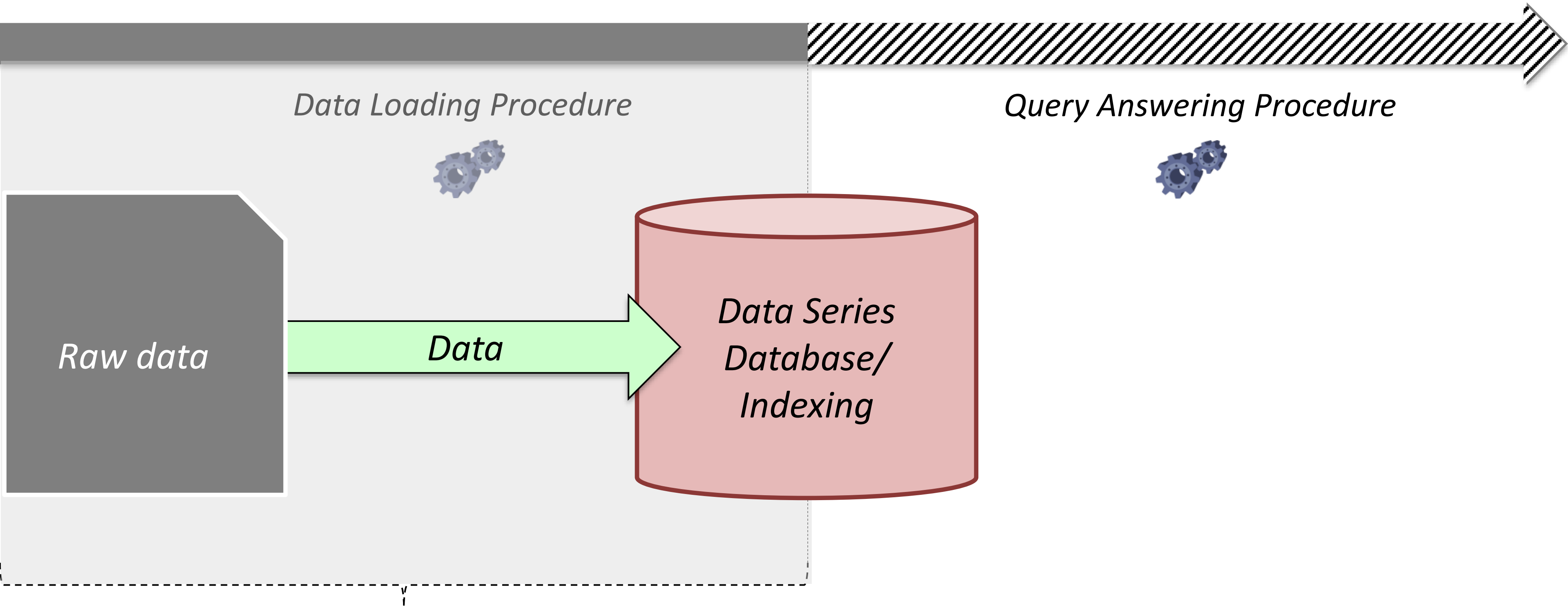
- Some studies have argued that NN search is not meaningful for a number of high dimensional datasets due to the concentration of distances.
  - However, these conclusions were based on over-restrictive assumptions such as:
    - data being identical and independently distributed (i.i.d.) in each dimension
    - dimensionality being the only factor determining meaningfulness
    - an asymptotic analysis of dimensionality growing to infinity
- Other studies have shown that high-dimensional NN search is meaningful for:
  - non-i.i.d data
  - data with low intrinsic dimensionality
  - for a variety of real world datasets

High-d Similarity Search Process

Similarity Search Process

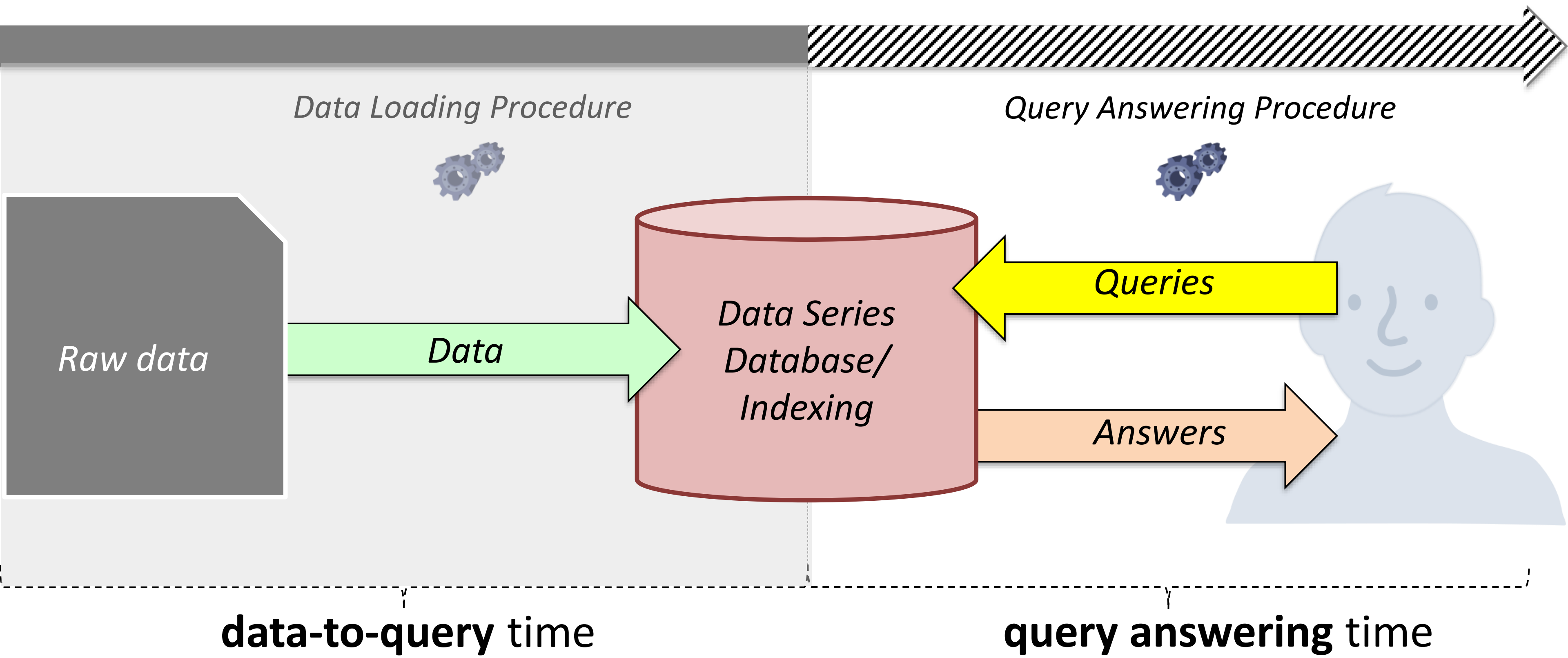


Similarity Search Process

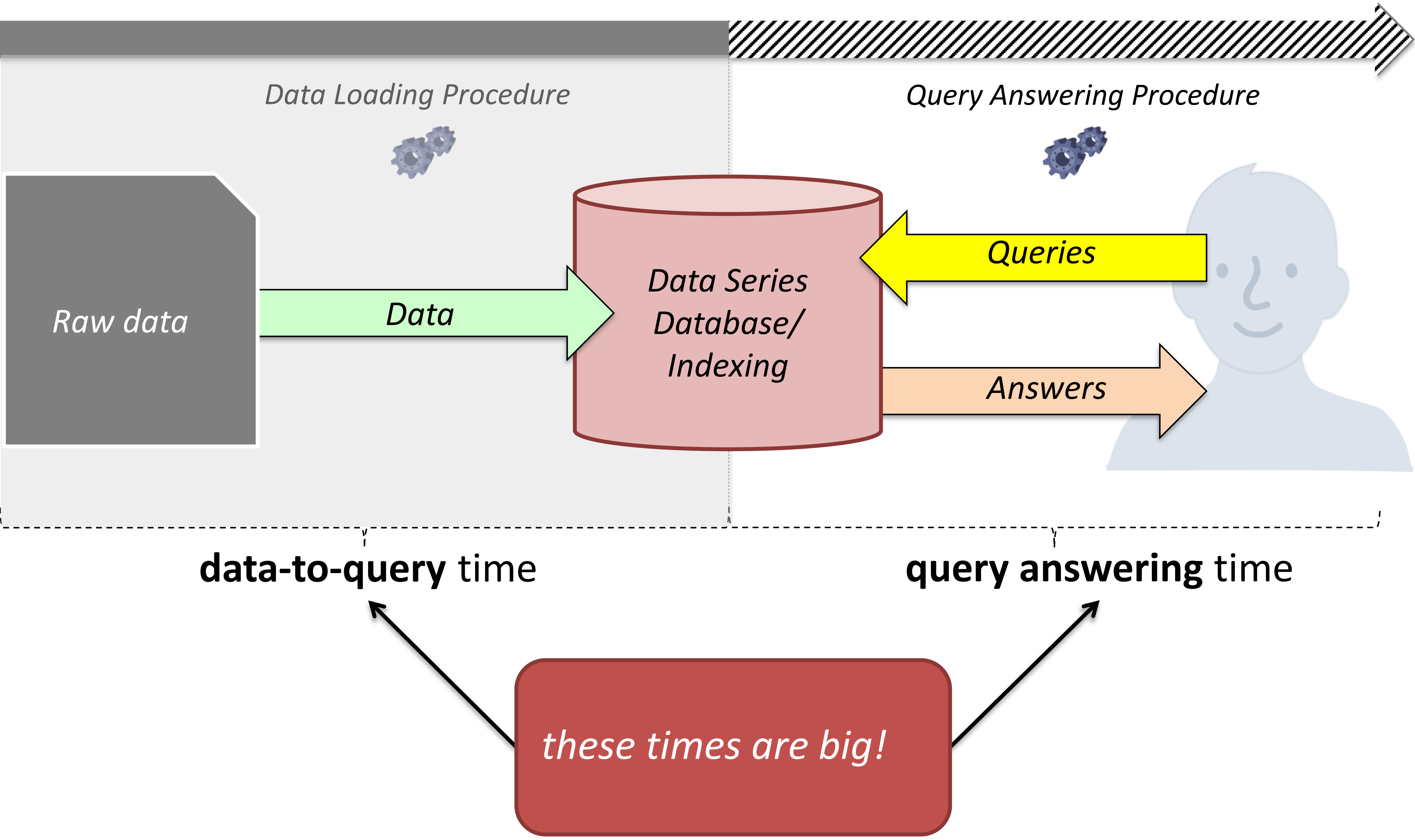


data-to-query time

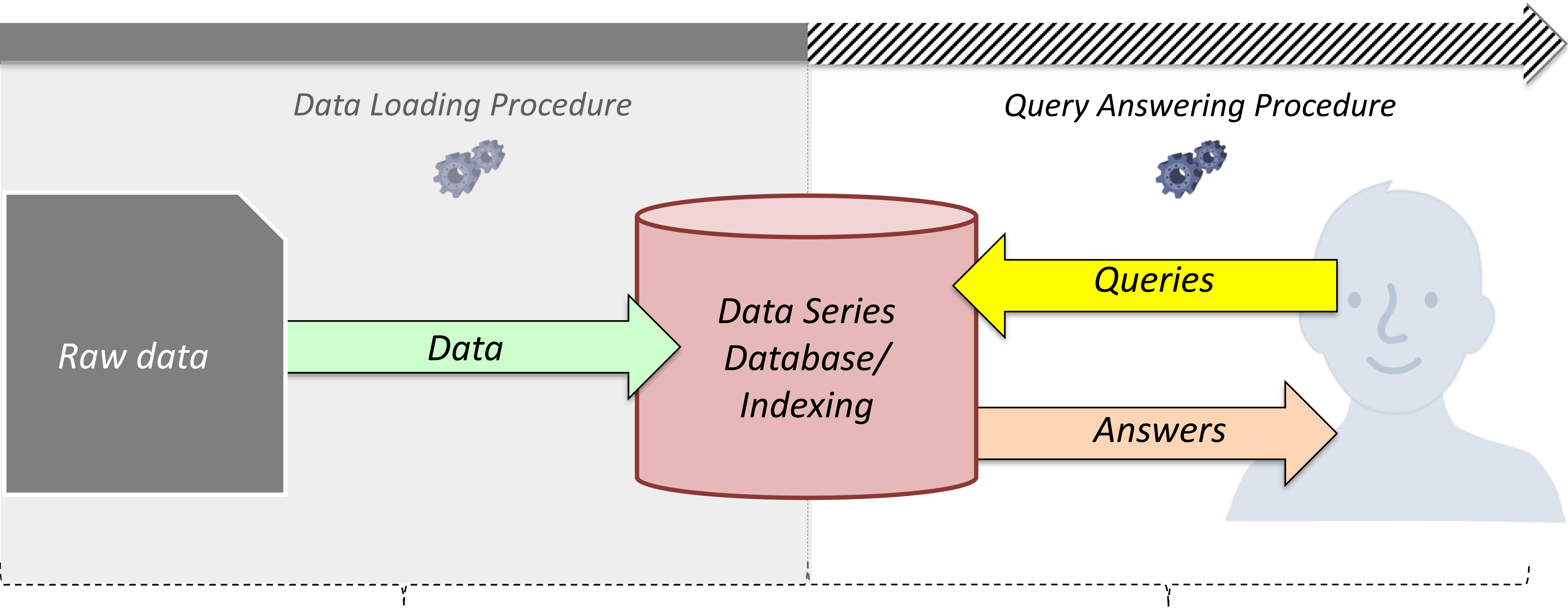
Similarity Search Process



Similarity Search Process



Similarity Search Process



data-to-query time

query answering time

we need solutions
for both problems!

Questions?

Data Series Similarity Search

Outline

- Pre-processing Tasks
- Classes of Methods
- State-of-the-art Techniques
- New extensions

Data Series Similarity Search Pre-processing Tasks

Pre-Processing
z-Normalization

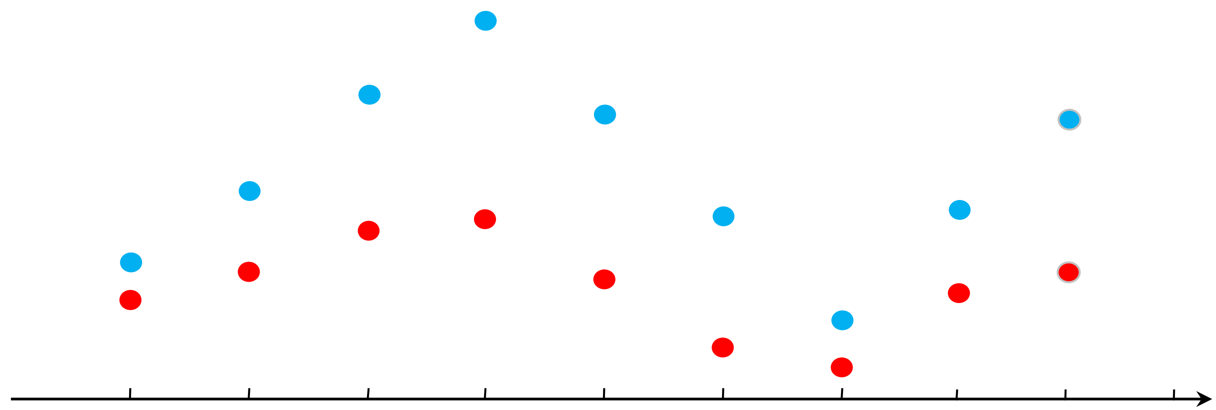
- data series encode trends
- usually interested in identifying similar trends

Pre-Processing
z-Normalization

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

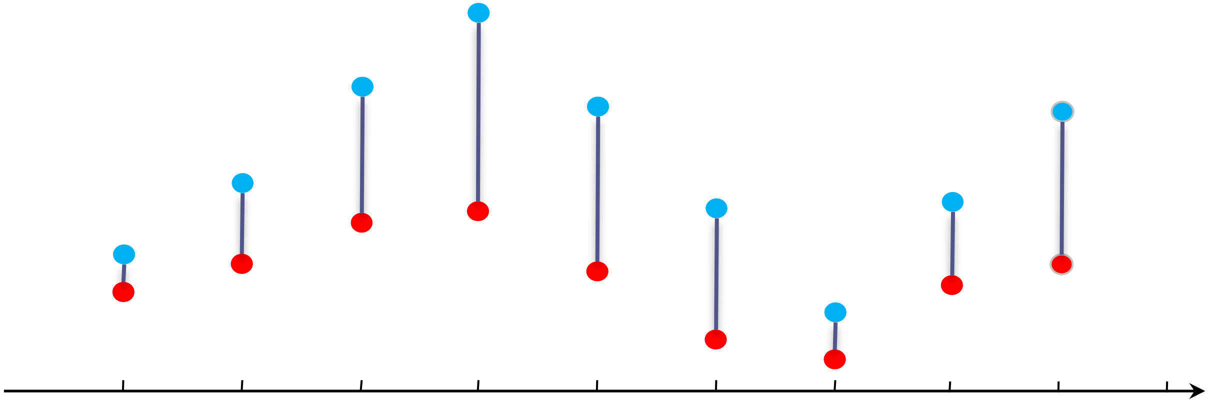
- but absolute values may mask this similarity

Pre-Processing
z-Normalization



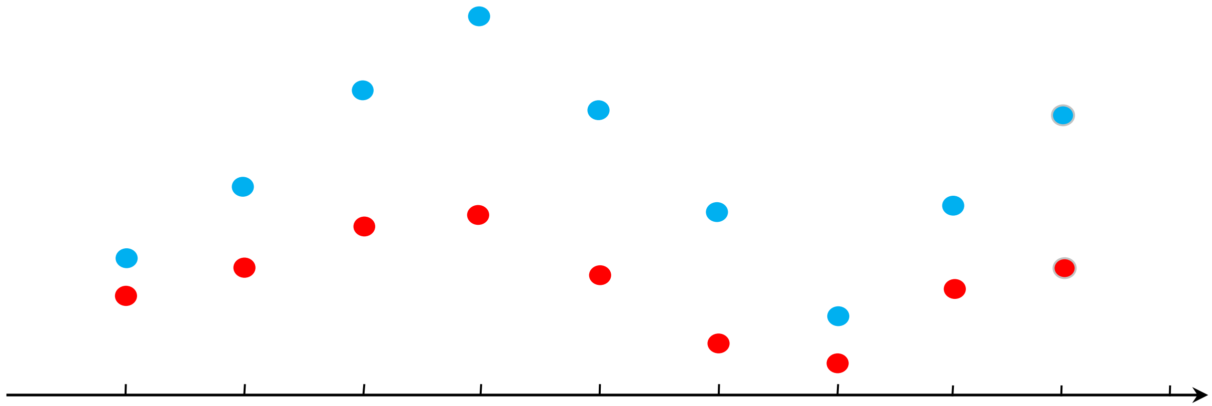
- two data series with similar trends

Pre-Processing z-Normalization



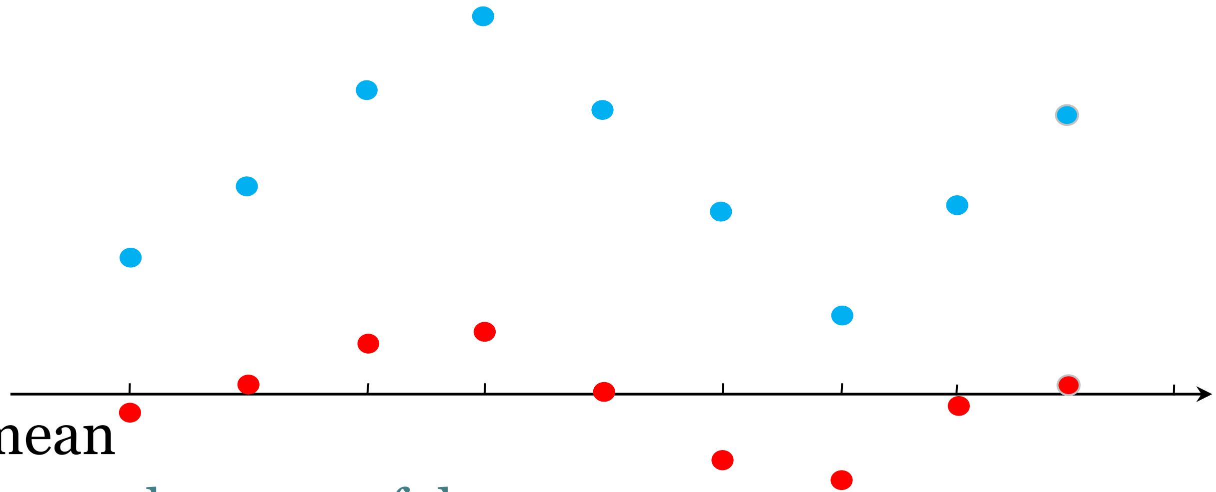
- two data series with similar trends
- but large distance…

Pre-Processing
z-Normalization



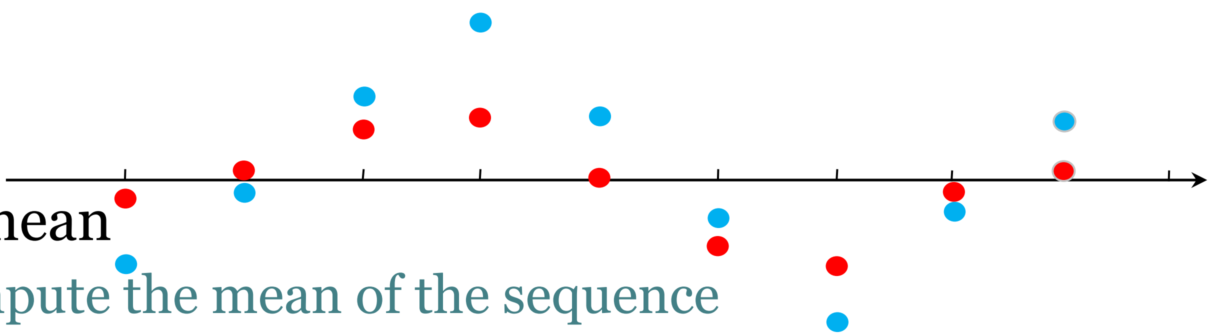
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing
z-Normalization



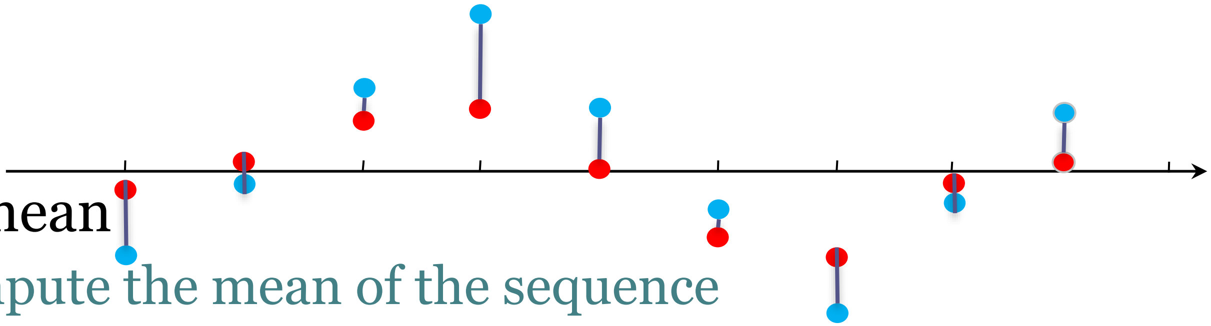
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing
z-Normalization



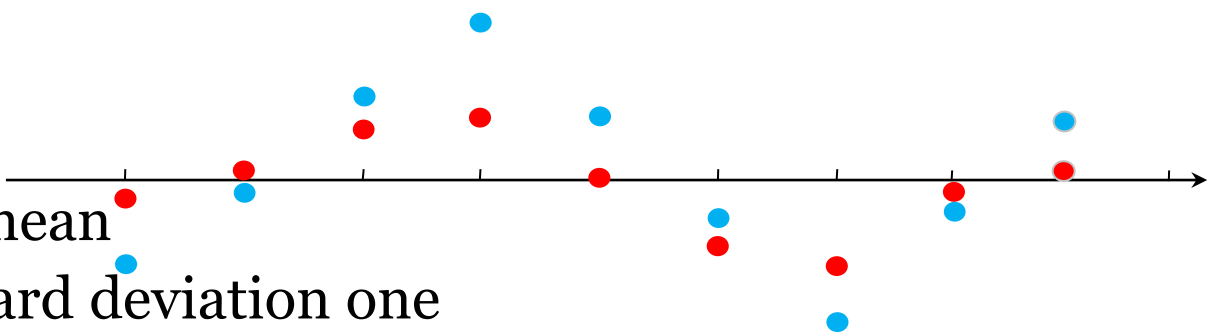
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing z-Normalization



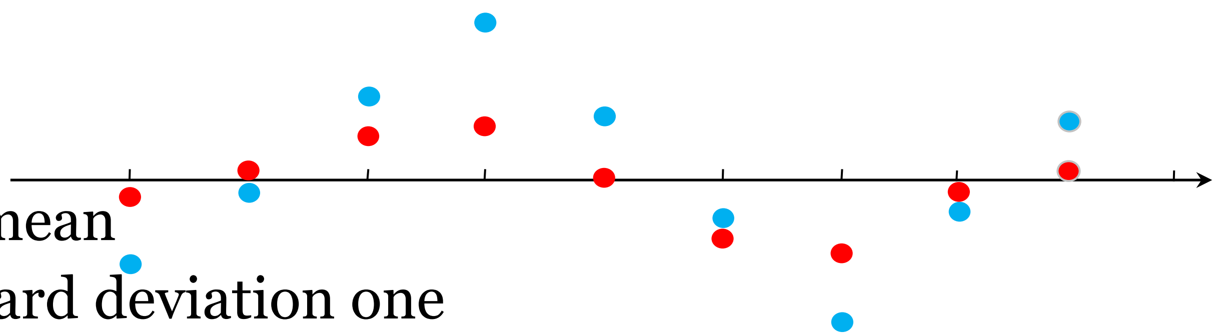
- zero mean
  - compute the mean of the sequence
  - subtract the mean from every value of the sequence

Pre-Processing
z-Normalization



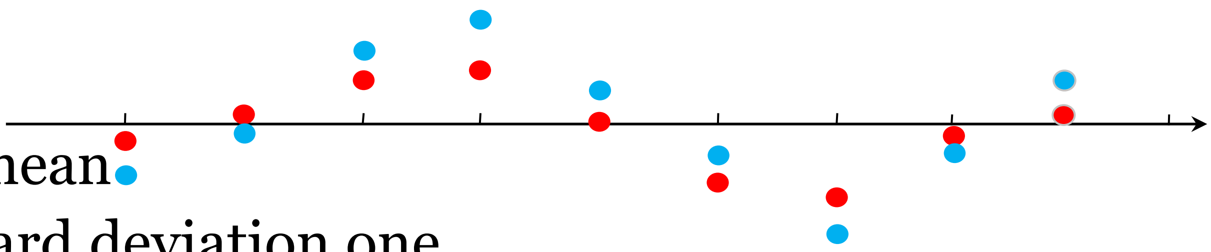
- zero mean
- standard deviation one
  - compute the standard deviation of the sequence
  - divide every value of the sequence by the stddev

Pre-Processing
z-Normalization



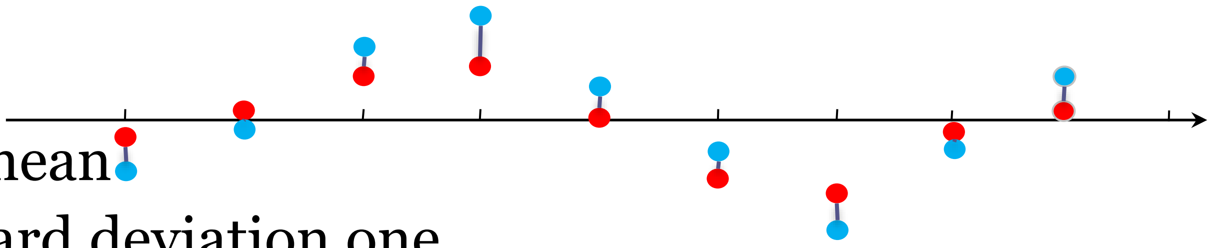
- zero mean
- standard deviation one
  - compute the standard deviation of the sequence
  - divide every value of the sequence by the stddev

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

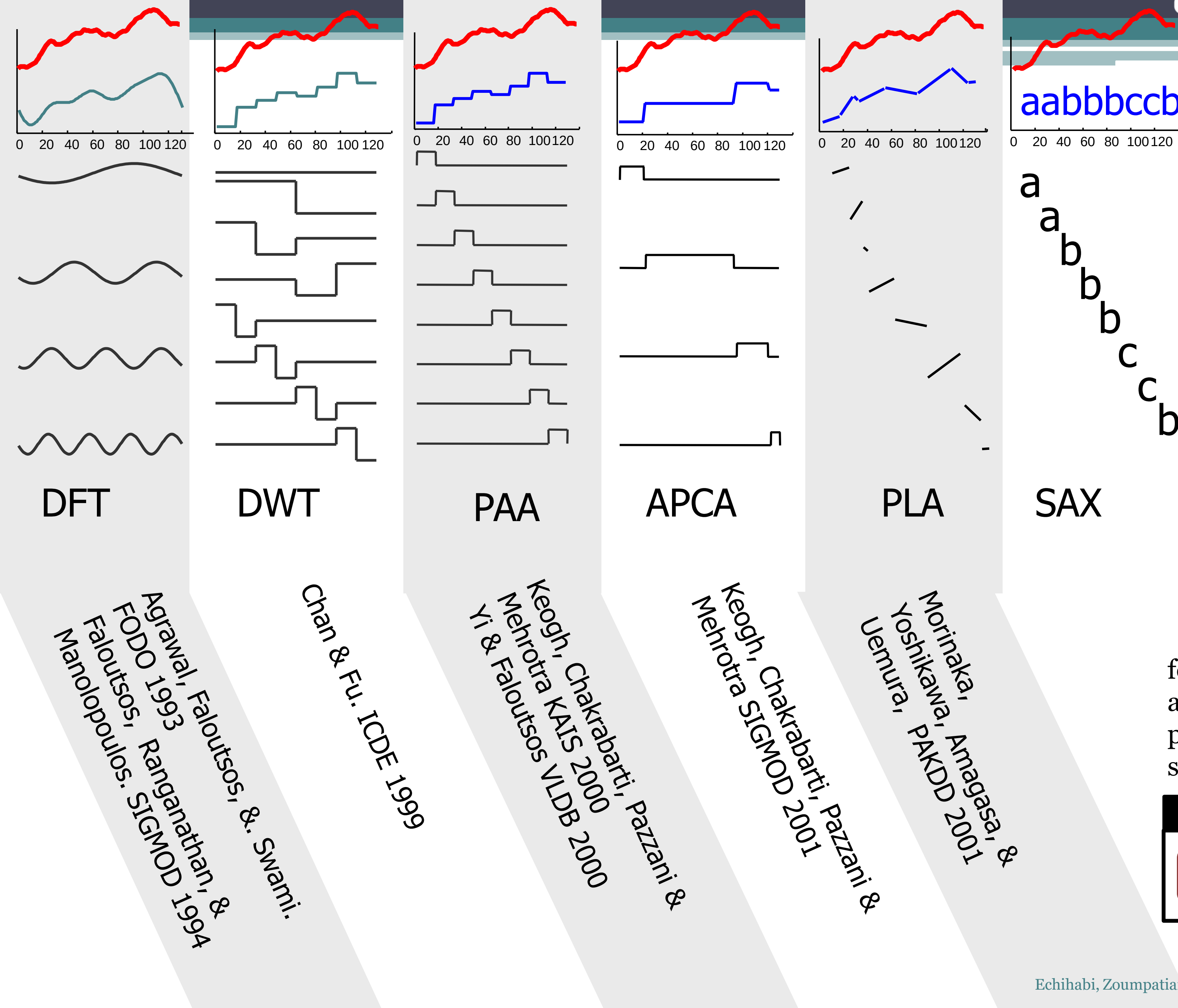
Pre-Processing
z-Normalization

- 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



for a complete and detailed presentation, see tutorial:

Publications

Keogh - KDD'04

Comparison of Representations

- which representation is the best?

Comparison of Representations

- which representation is the best?

- depends on data characteristics
  - periodic, smooth, spiky, ...

Publications

Palpanas et al. ICDE'04

Palpanas et al. TKDE'08

Shieh et al. KDD'08

Comparison of Representations

- which representation is the best?
- depends on data characteristics
  - periodic, smooth, spiky, …
- overall (averaged over many diverse datasets, using same memory budget), when measuring reconstruction error (RMSE)
  - no big differences among methods
  - DFT, PAA, DWT (Haar), iSAX slightly better
- should also take into account other factors
  - visualization, indexable, …

Data Series Similarity Search Common Framework

GEMINI Framework

Publications

Faloutsos-SIGMOD'94

- 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

GEMINI Framework

Publications

Faloutsos-SIGMOD'94

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

GEMINI: contractiveness

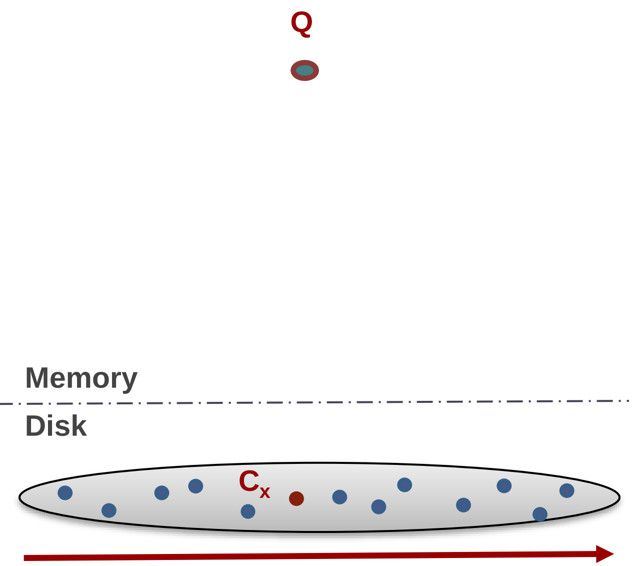
- GEMINI works when:

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

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

Data Series Similarity Search Classes of Methods

Similarity Matching
Serial Scan



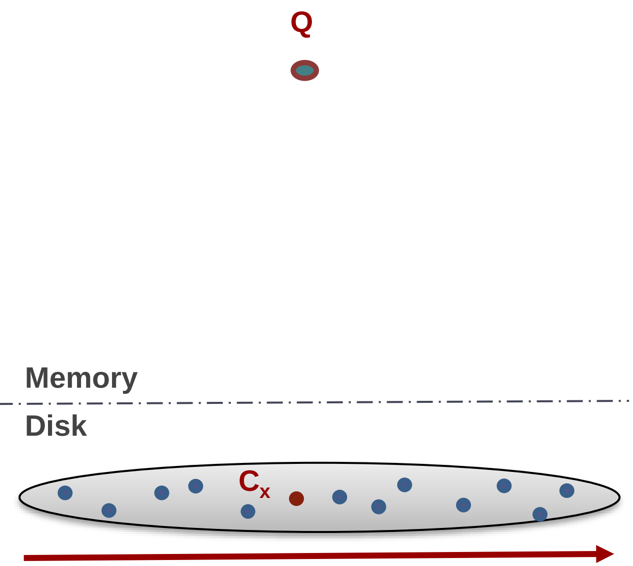
Q is compared to each raw candidate in the dataset before returning the answer C_x

(a) Serial scan

Answering a similarity search query using different access paths

Similarity Matching
Serial Scan

bsf = +∞

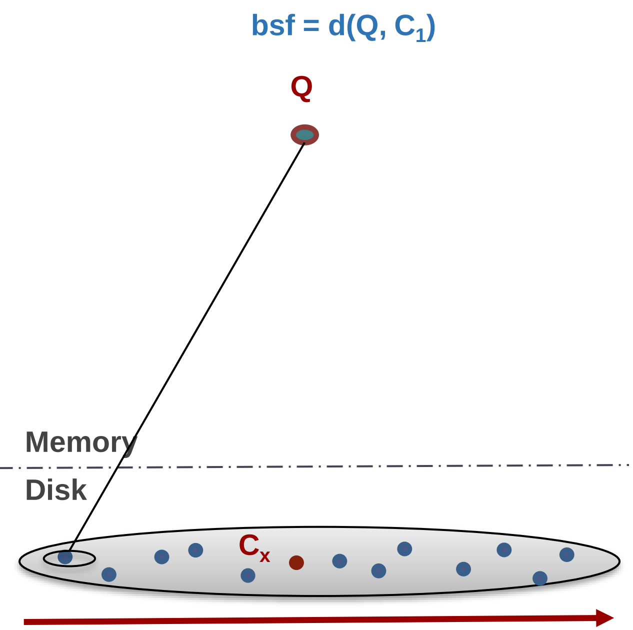


Q is compared to each raw candidate in the dataset before returning the answer C_x

(a) Serial scan

Answering a similarity search query using different access paths

Similarity Matching
Serial Scan

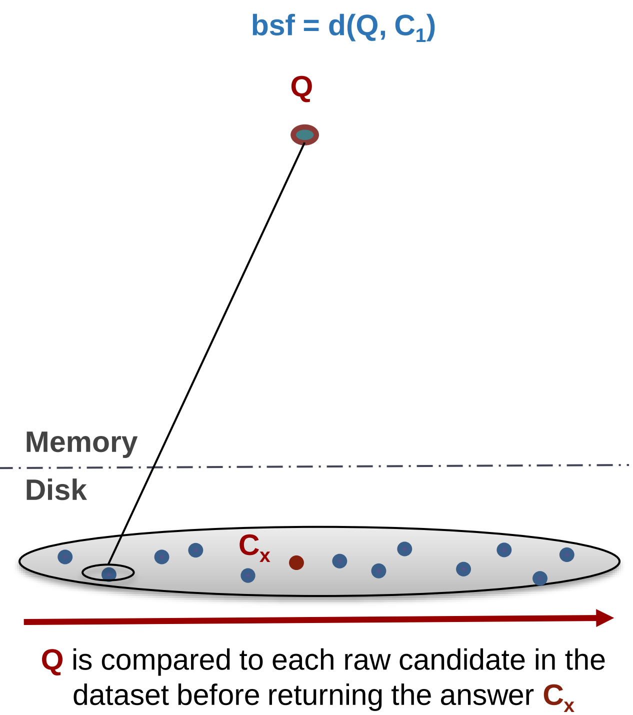


Q is compared to each raw candidate in the dataset before returning the answer **C_x**

(a) Serial scan

Answering a similarity search query using different access paths

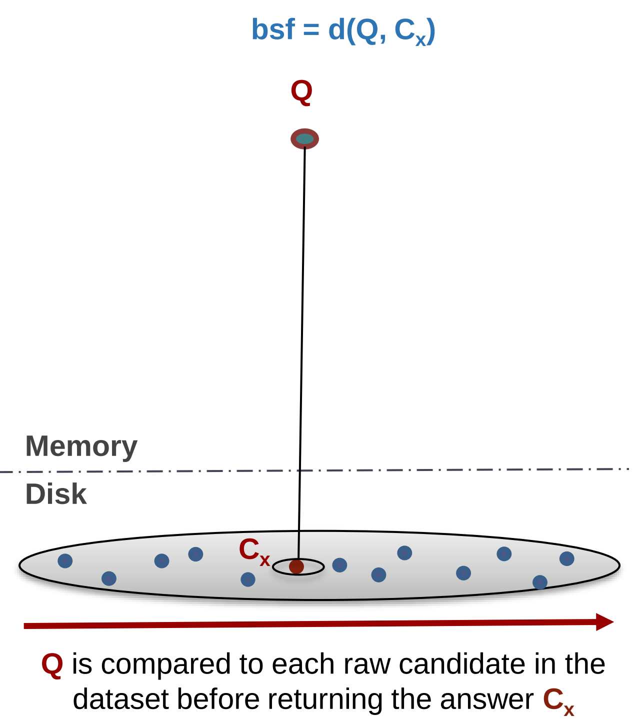
Similarity Matching
Serial Scan



(a) Serial scan

Answering a similarity search query using different access paths

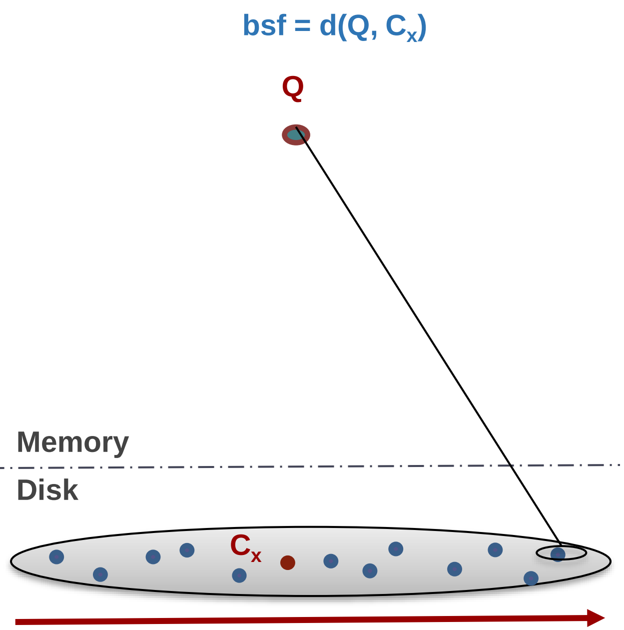
Similarity Matching
Serial Scan



(a) Serial scan

Answering a similarity search query using different access paths

Similarity Matching
Serial Scan

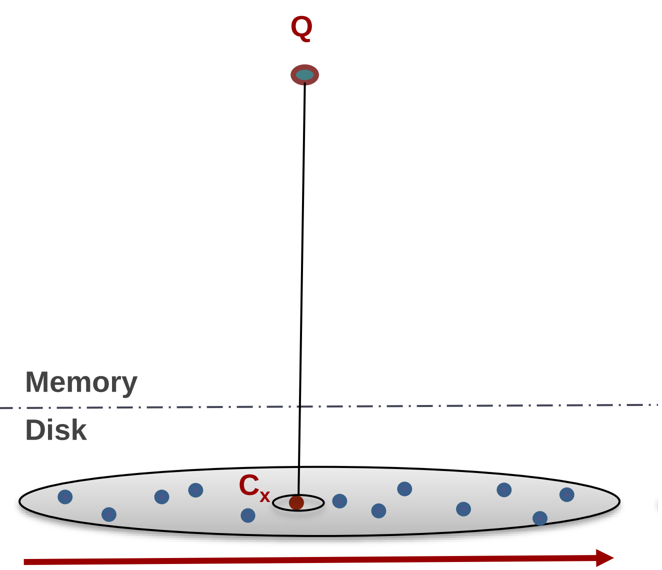


Q is compared to each raw candidate in the dataset before returning the answer **C_x**

(a) Serial scan

Answering a similarity search query using different access paths

Similarity Matching
Serial Scan

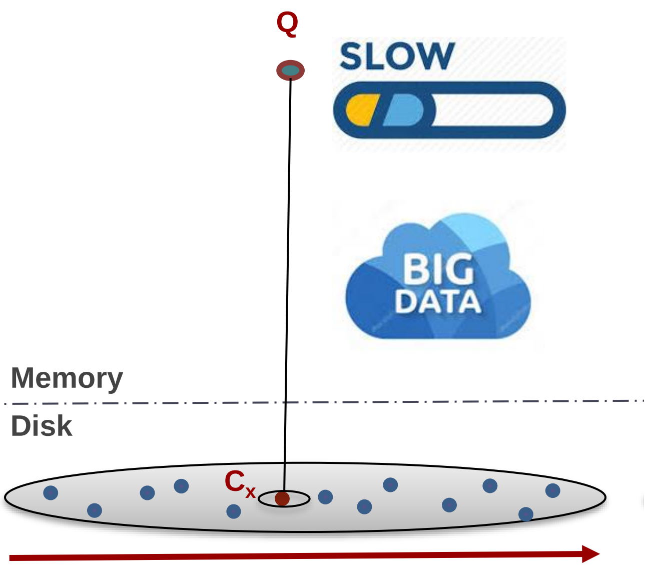


Q is compared to each raw candidate in the dataset before returning the answer C_x

(a) Serial scan

Answering a similarity search query using different access paths

Similarity Matching
Serial Scan

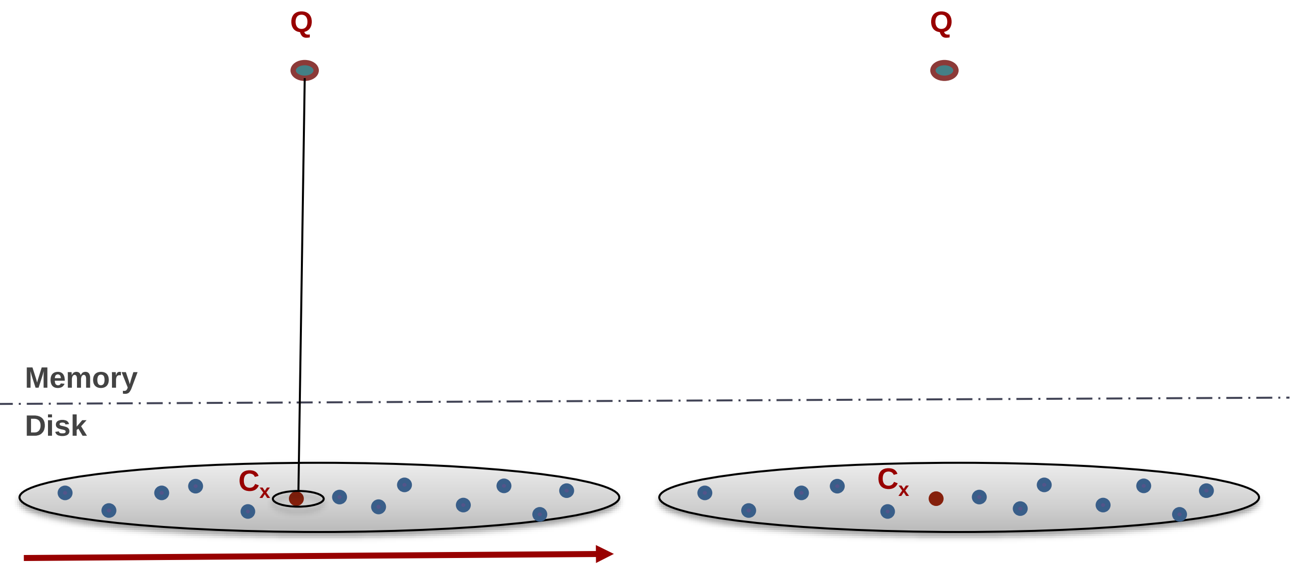


Q is compared to each raw candidate in the dataset before returning the answer C_x

(a) Serial scan

Answering a similarity search query using different access paths

Indexes vs. Scans



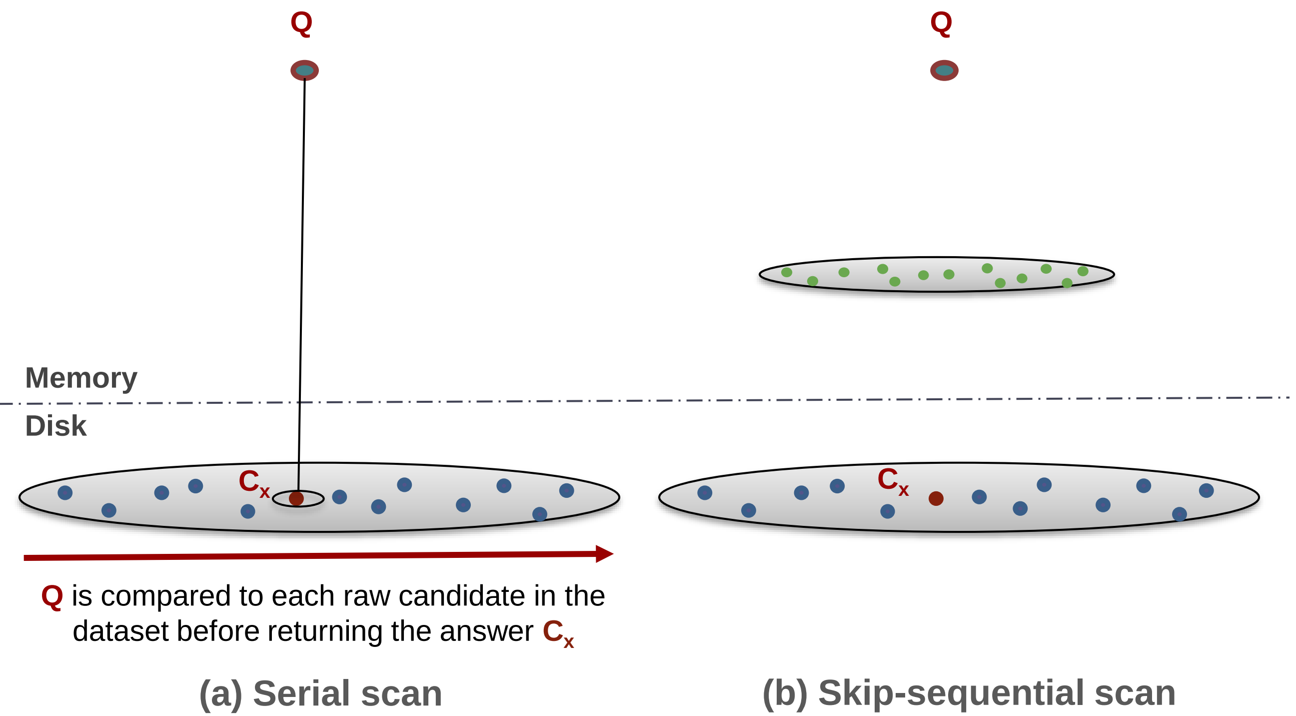
Q is compared to each raw candidate in the dataset before returning the answer **C_x**

(a) Serial scan

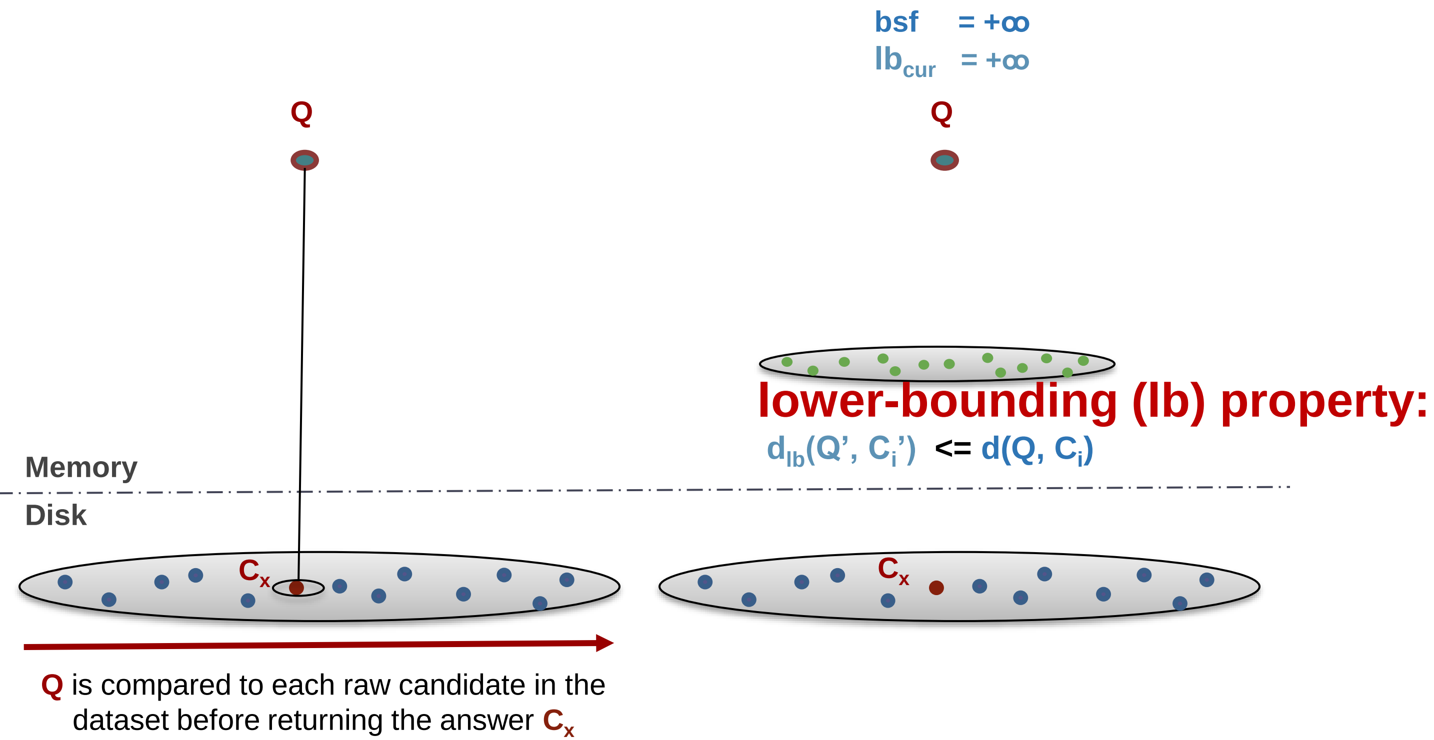
(b) Skip-sequential scan

<u>Answering a similarity search query using different access paths</u>

Indexes vs. Scans



Answering a similarity search query using different access paths



(a) Serial scan

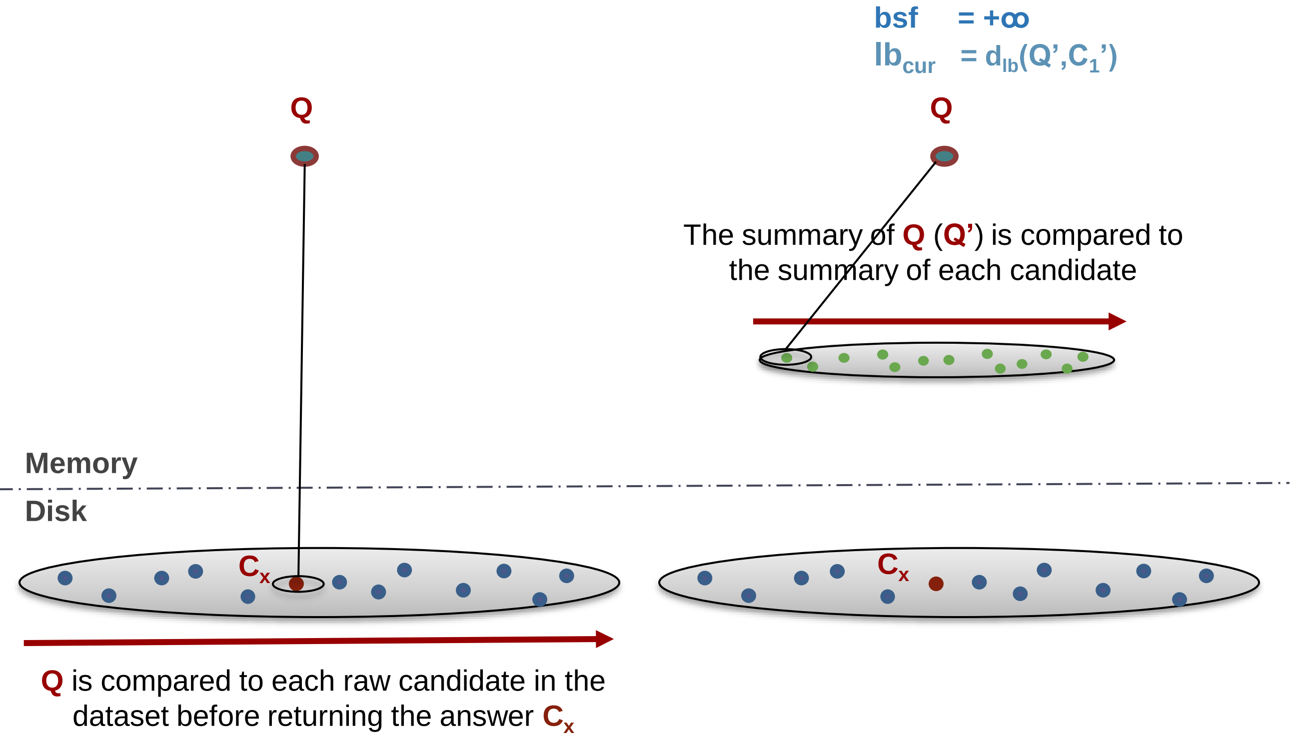
(b) Skip-sequential scan

Publications

Faloutsos-SIGMOD'94

Answering a similarity search query using different access paths

Indexes vs. Scans

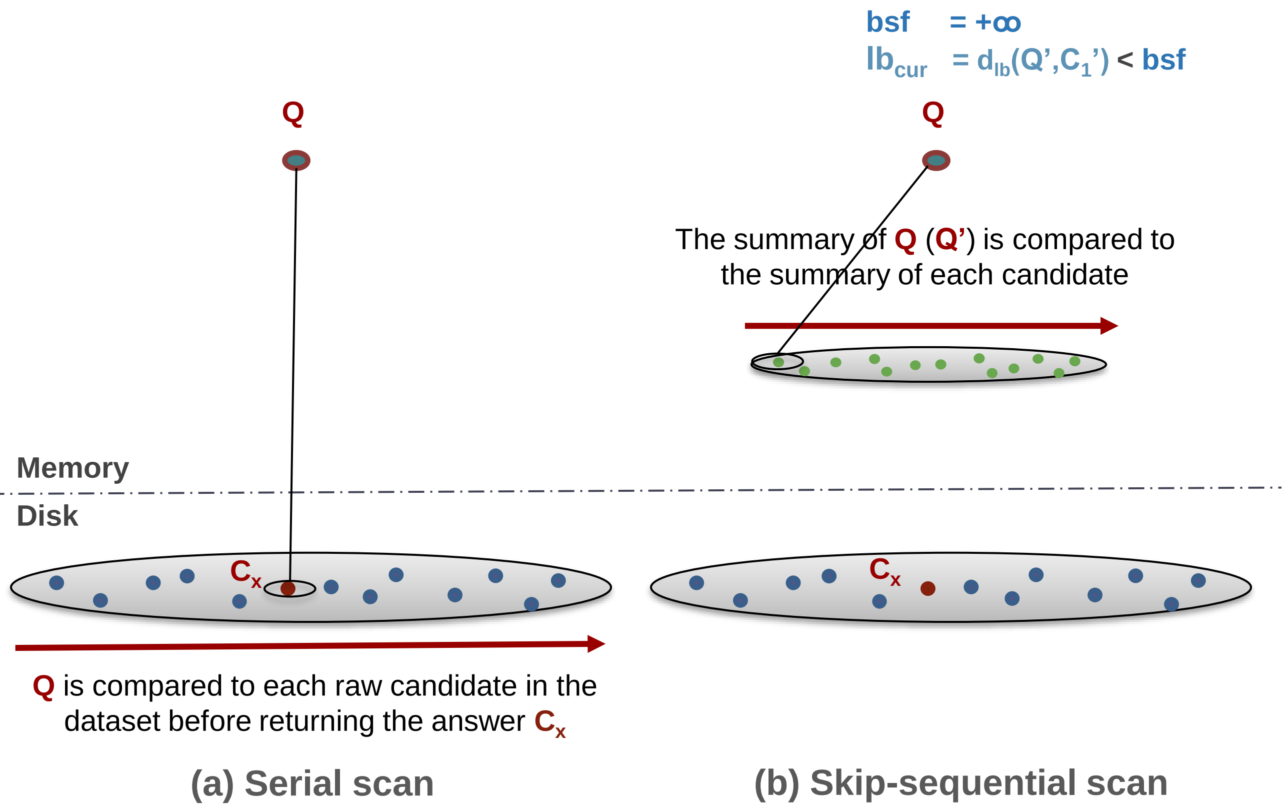


(a) Serial scan

(b) Skip-sequential scan

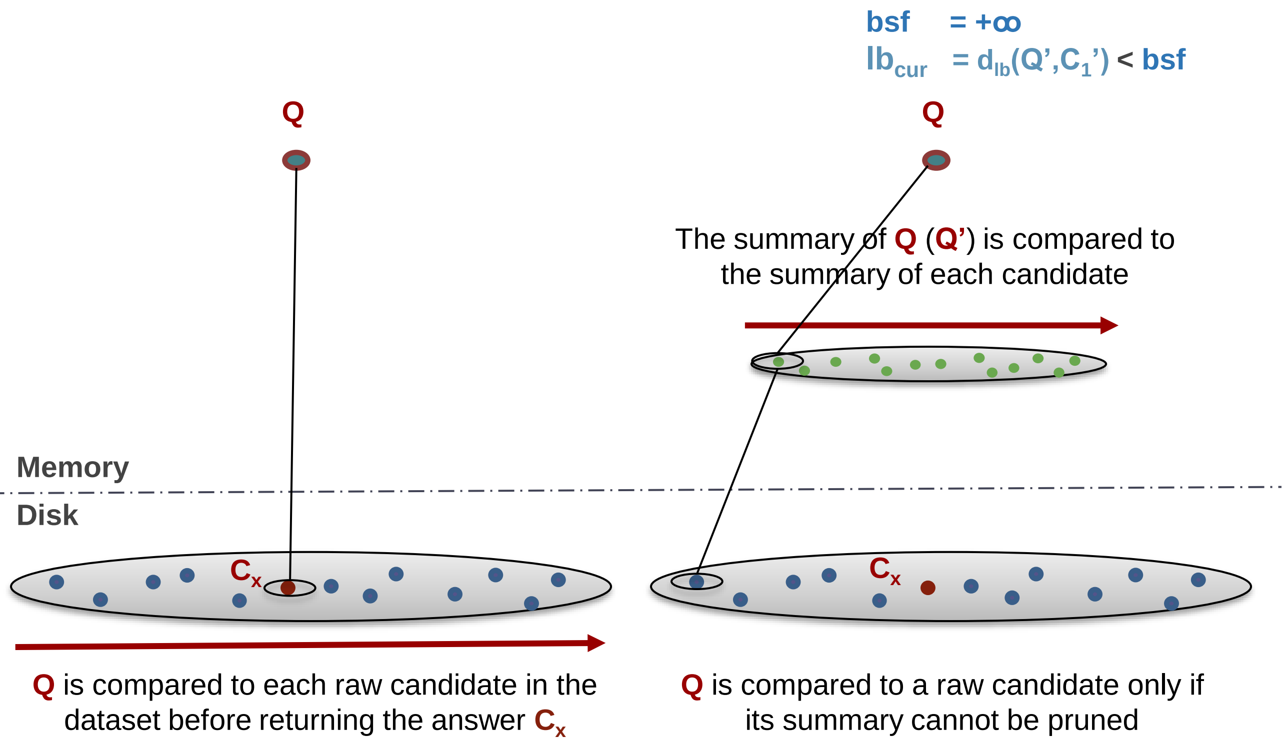
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

Indexes vs. Scans

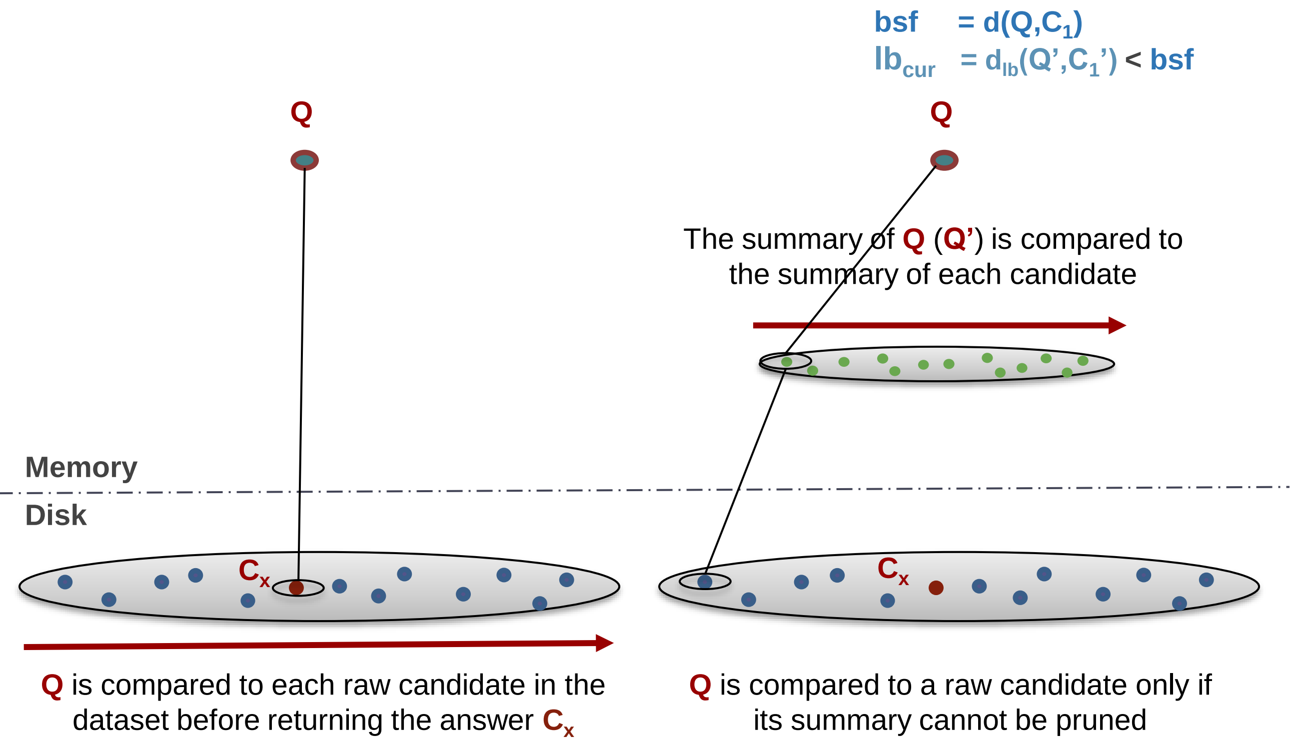


(a) Serial scan

(b) Skip-sequential scan

Answering a similarity search query using different access paths

Indexes vs. Scans

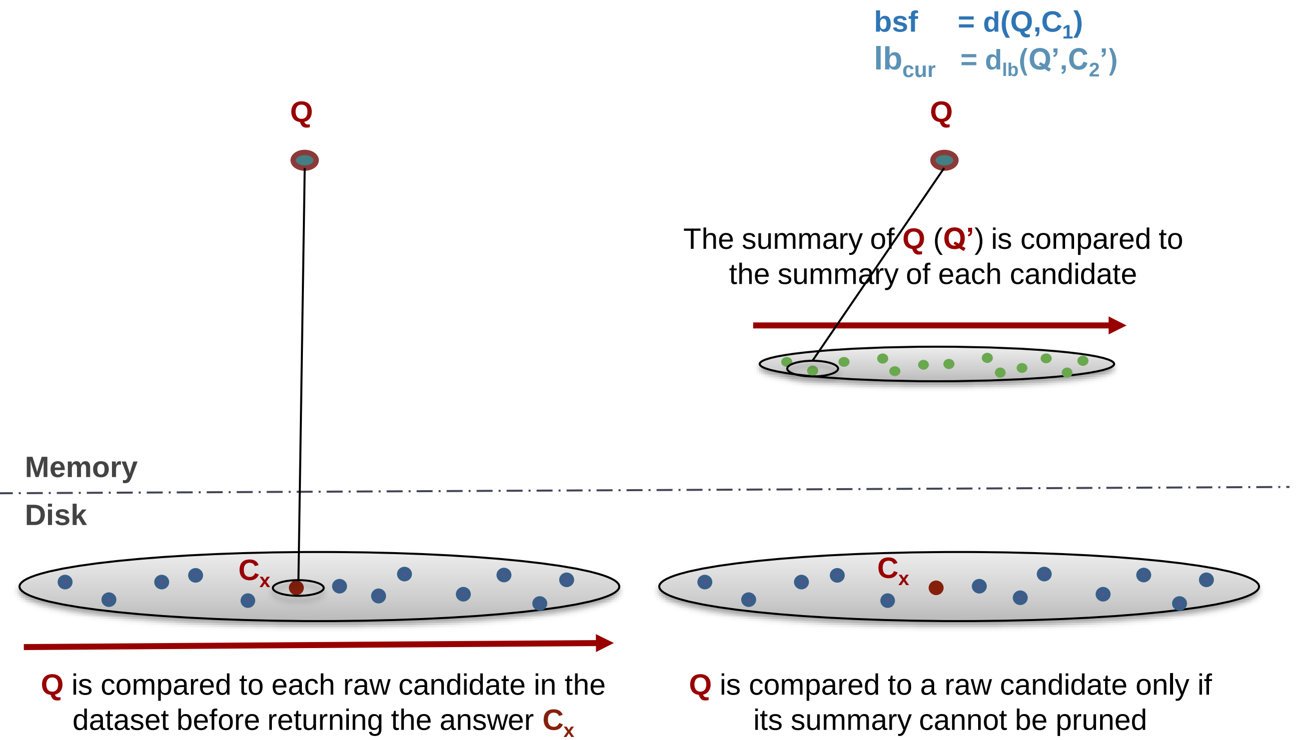


(a) Serial scan

(b) Skip-sequential scan

Answering a similarity search query using different access paths

Indexes vs. Scans

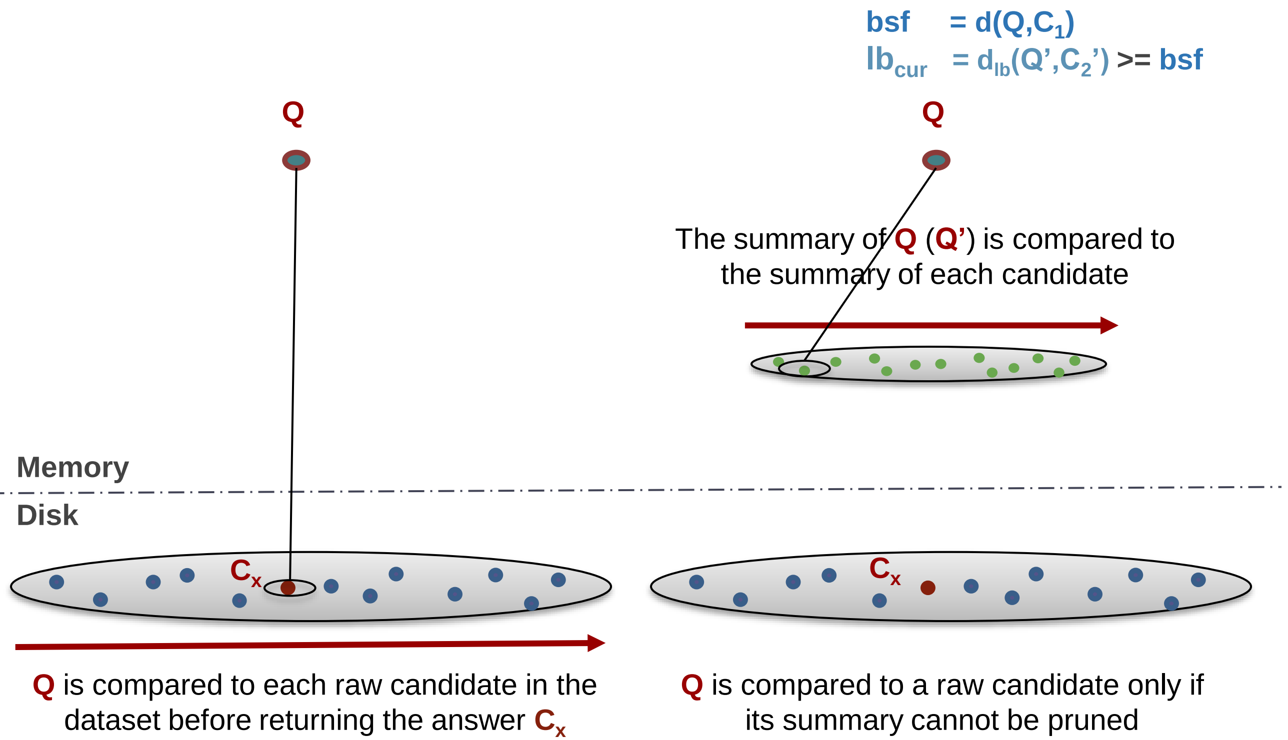


(a) Serial scan

(b) Skip-sequential scan

<u>**Answering a similarity search query using different access paths**</u>

Indexes vs. Scans

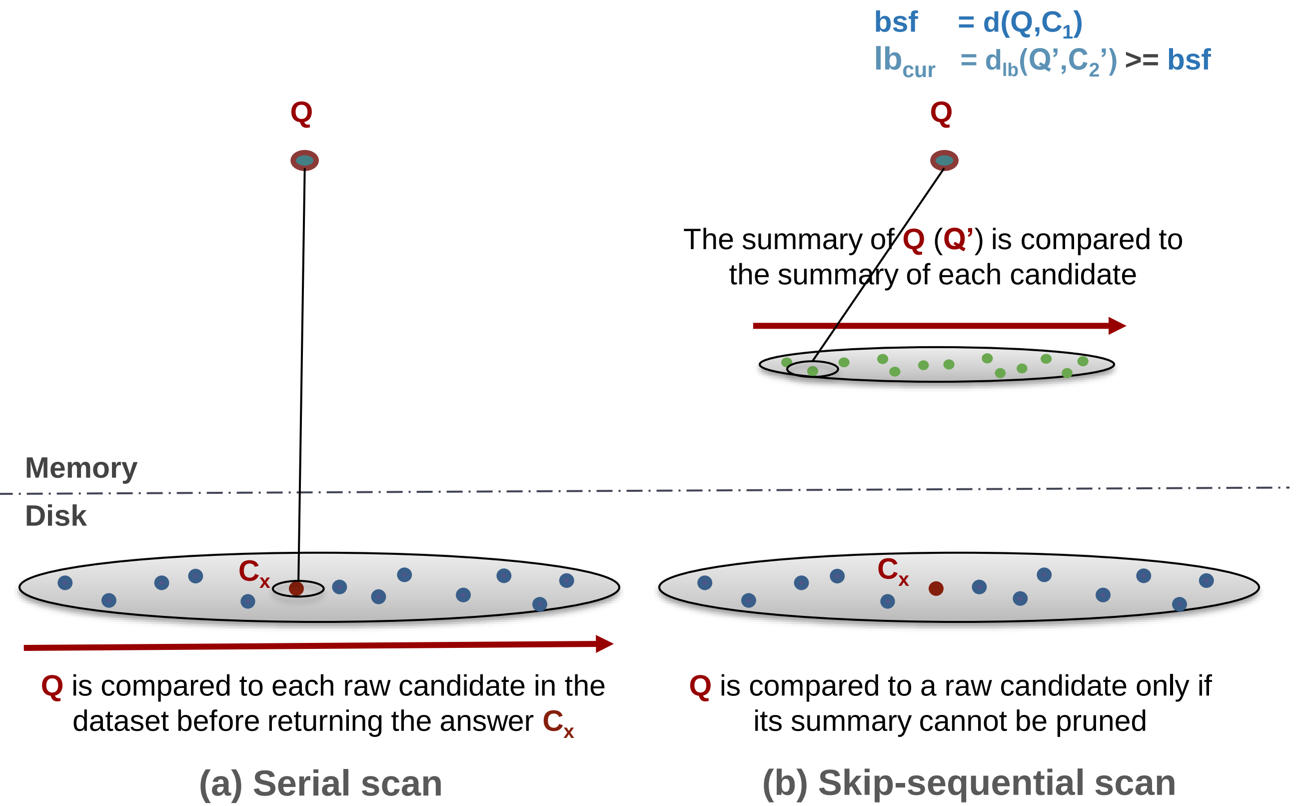


(a) Serial scan

(b) Skip-sequential scan

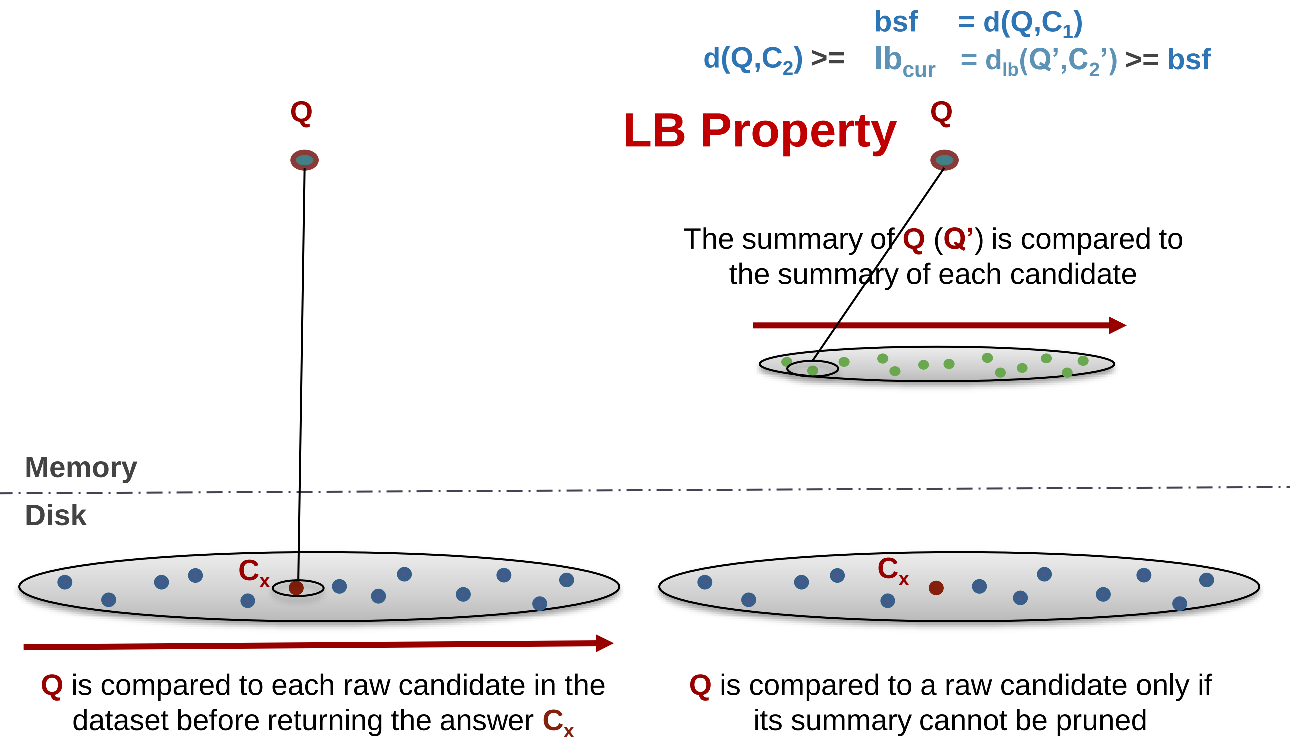
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

Indexes vs. Scans

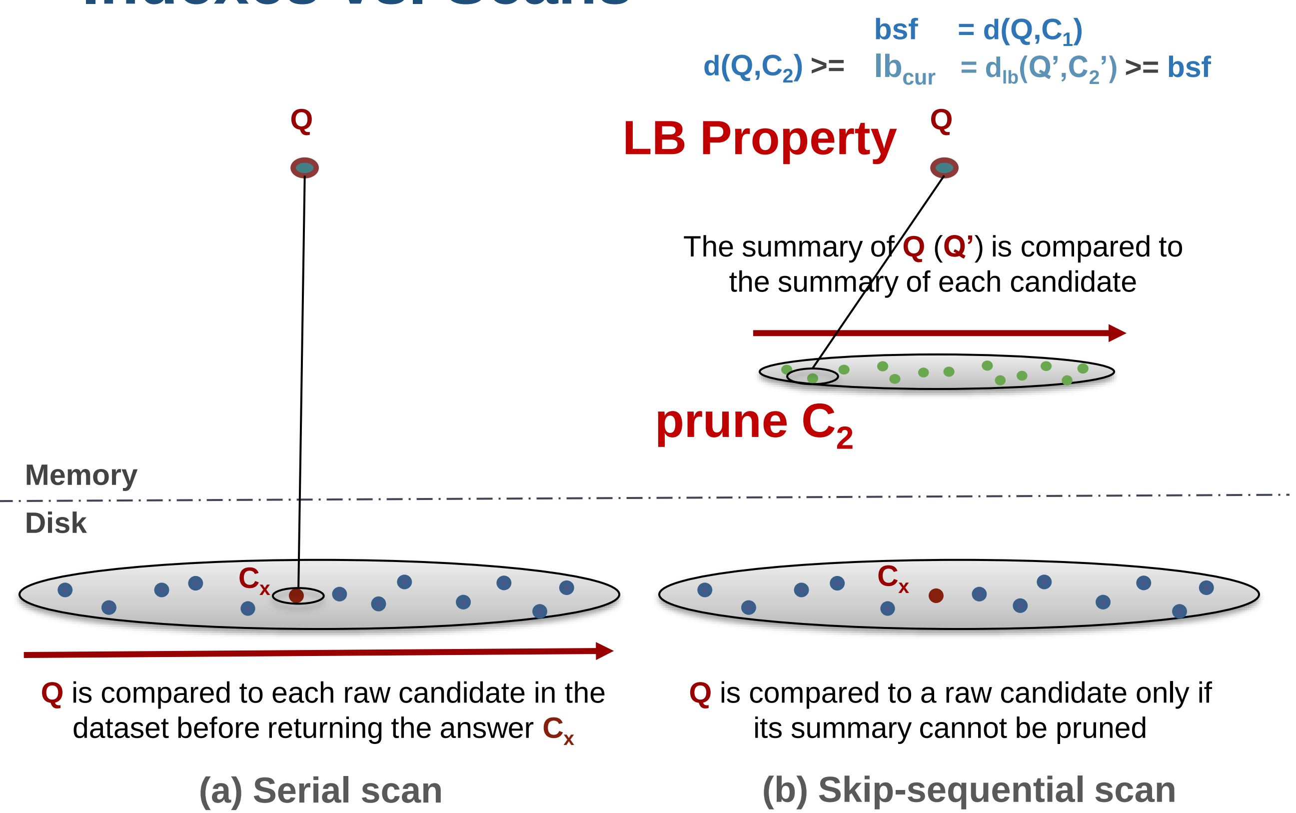


(a) Serial scan

(b) Skip-sequential scan

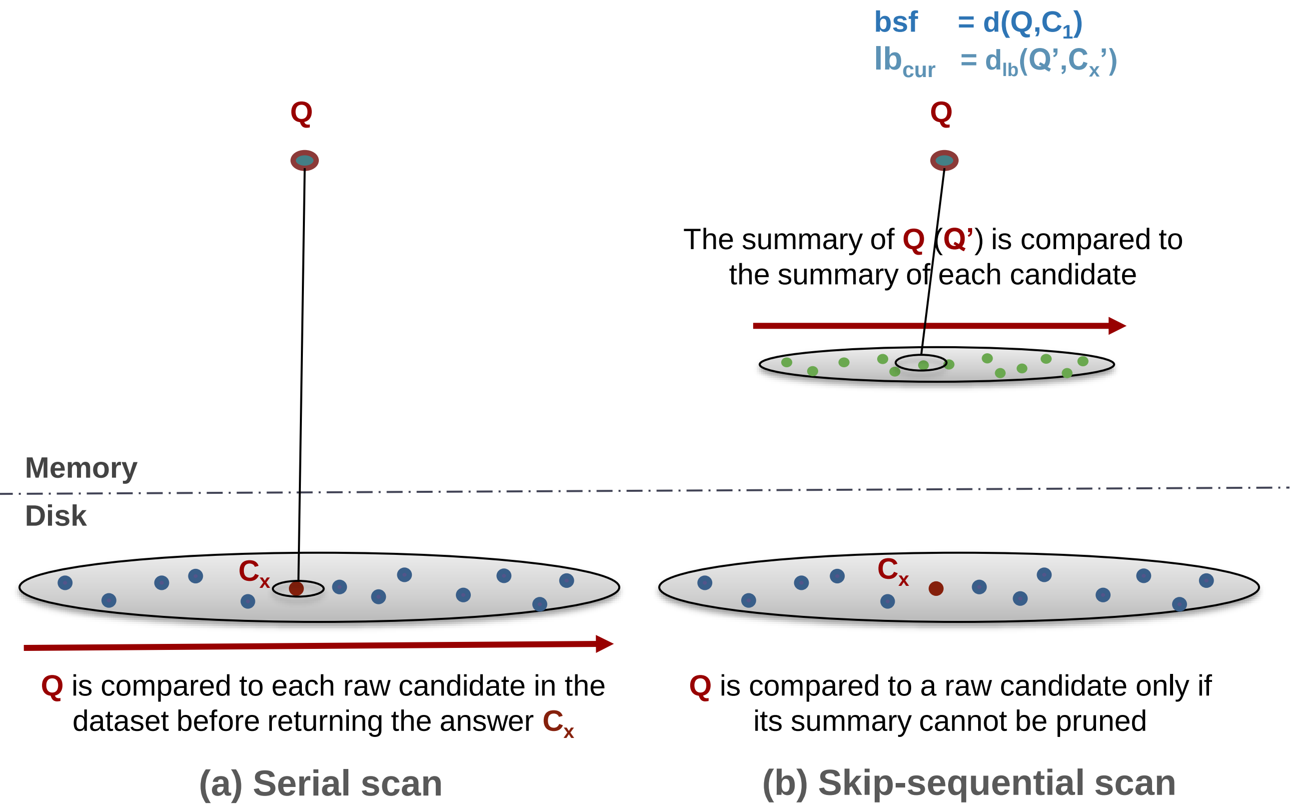
Answering a similarity search query using different access paths

Indexes vs. Scans



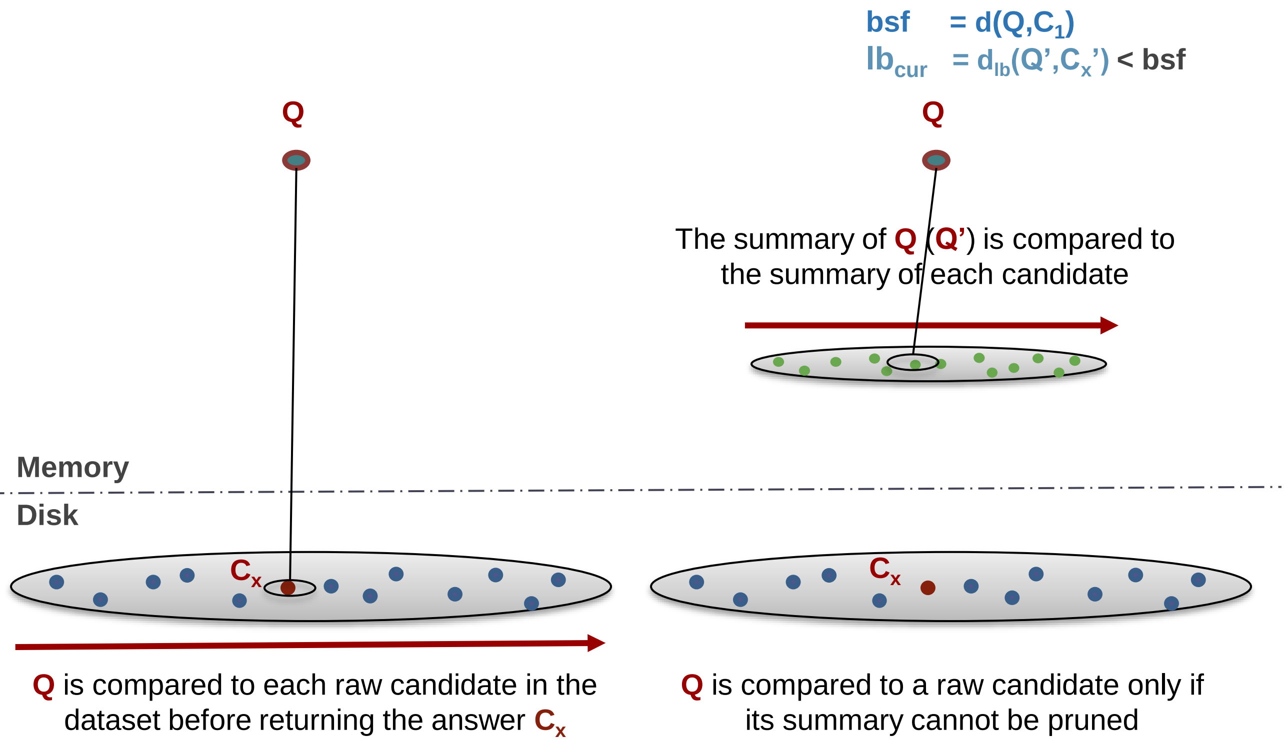
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

Indexes vs. Scans

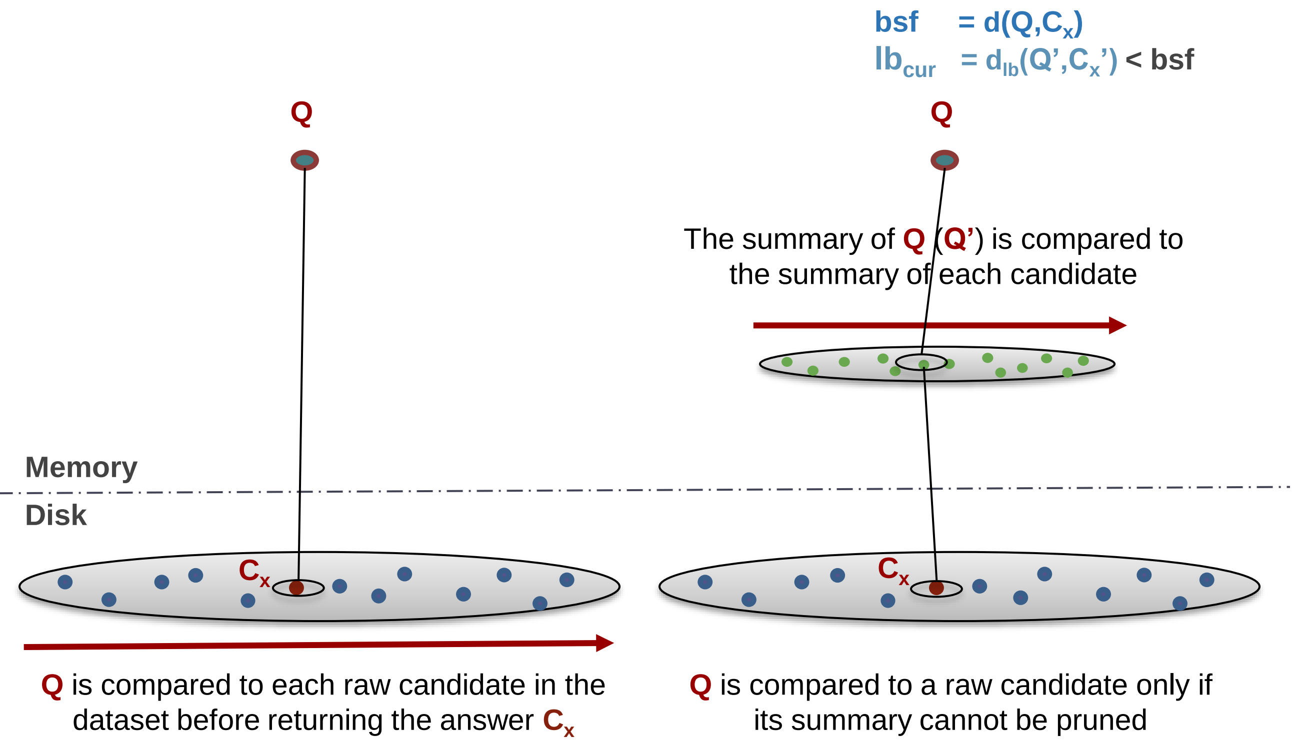


(a) Serial scan

(b) Skip-sequential scan

Answering a similarity search query using different access paths

Indexes vs. Scans



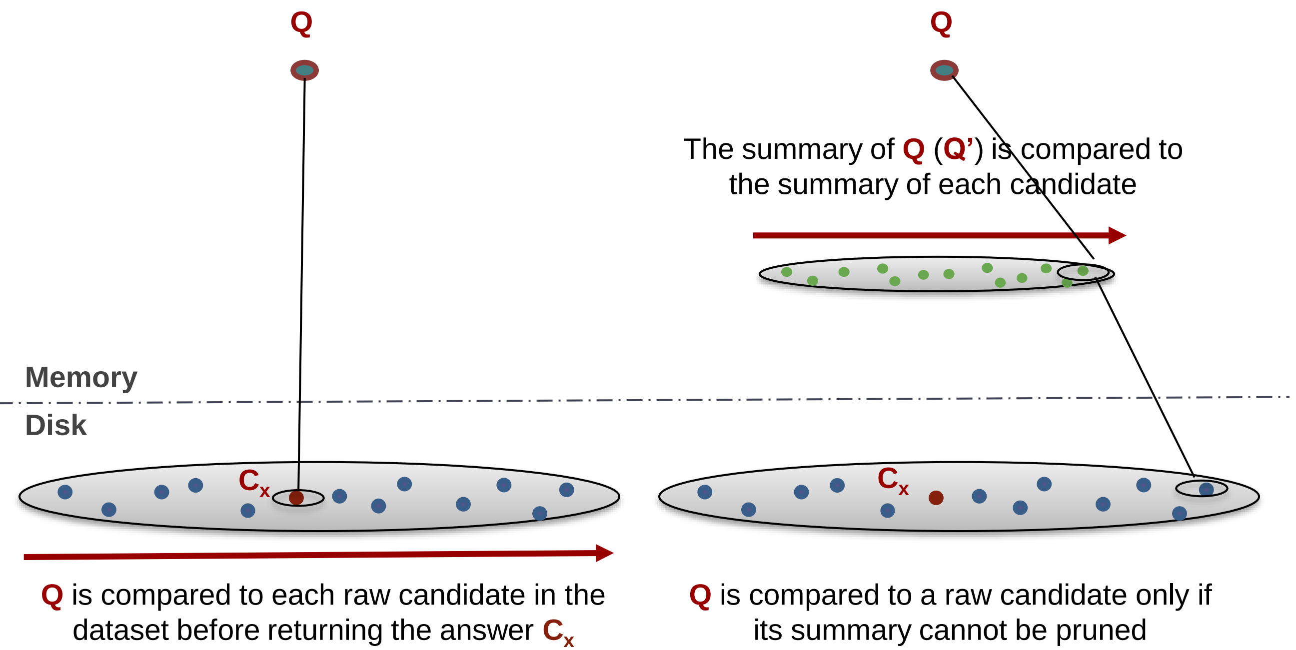
(a) Serial scan

(b) Skip-sequential scan

Answering a similarity search query using different access paths

Indexes vs. Scans

bsf $= d(Q, C_x)$
$lb_{cur} = d_{lb}(Q', C_n') < bsf$

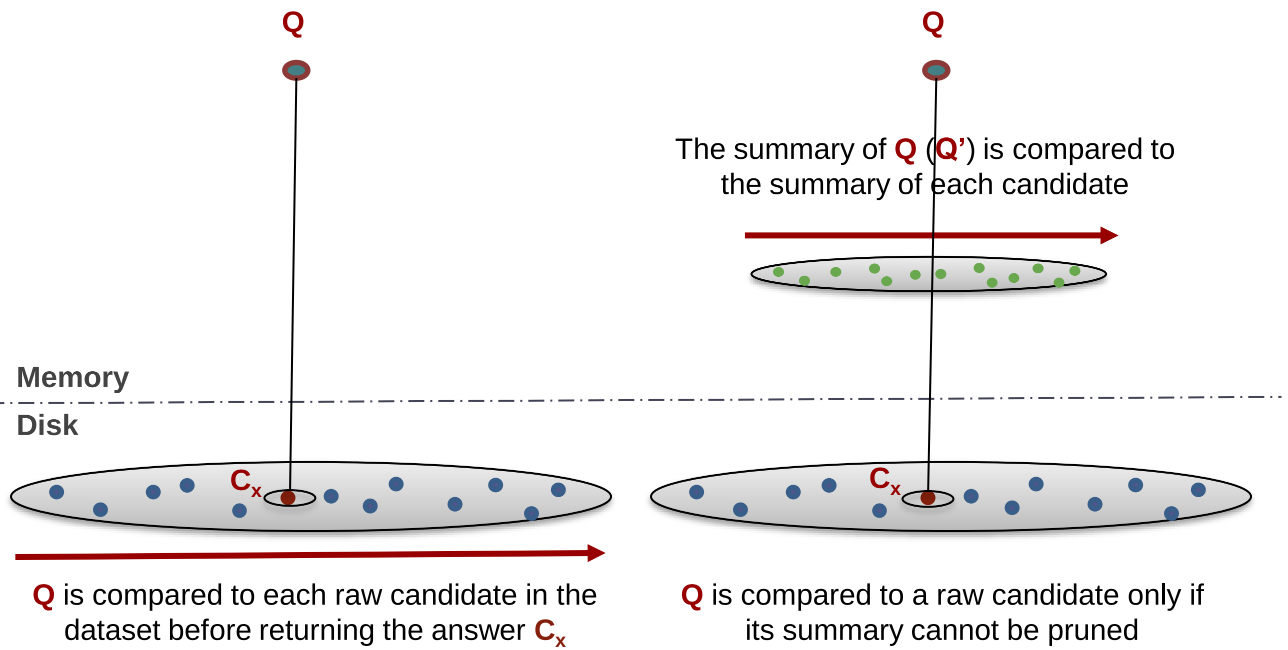


(a) Serial scan

(b) Skip-sequential scan

Answering a similarity search query using different access paths

Indexes vs. Scans

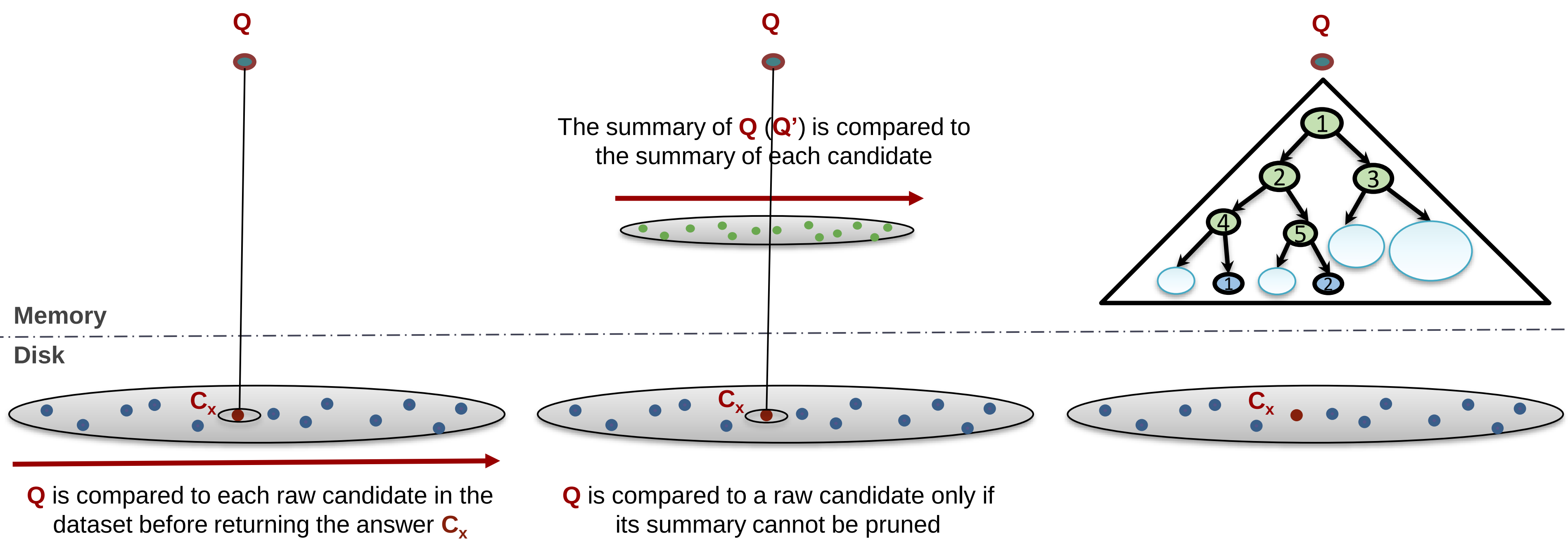


(a) Serial scan

(b) Skip-sequential scan

Answering a similarity search query using different access paths

Indexes vs. Scans



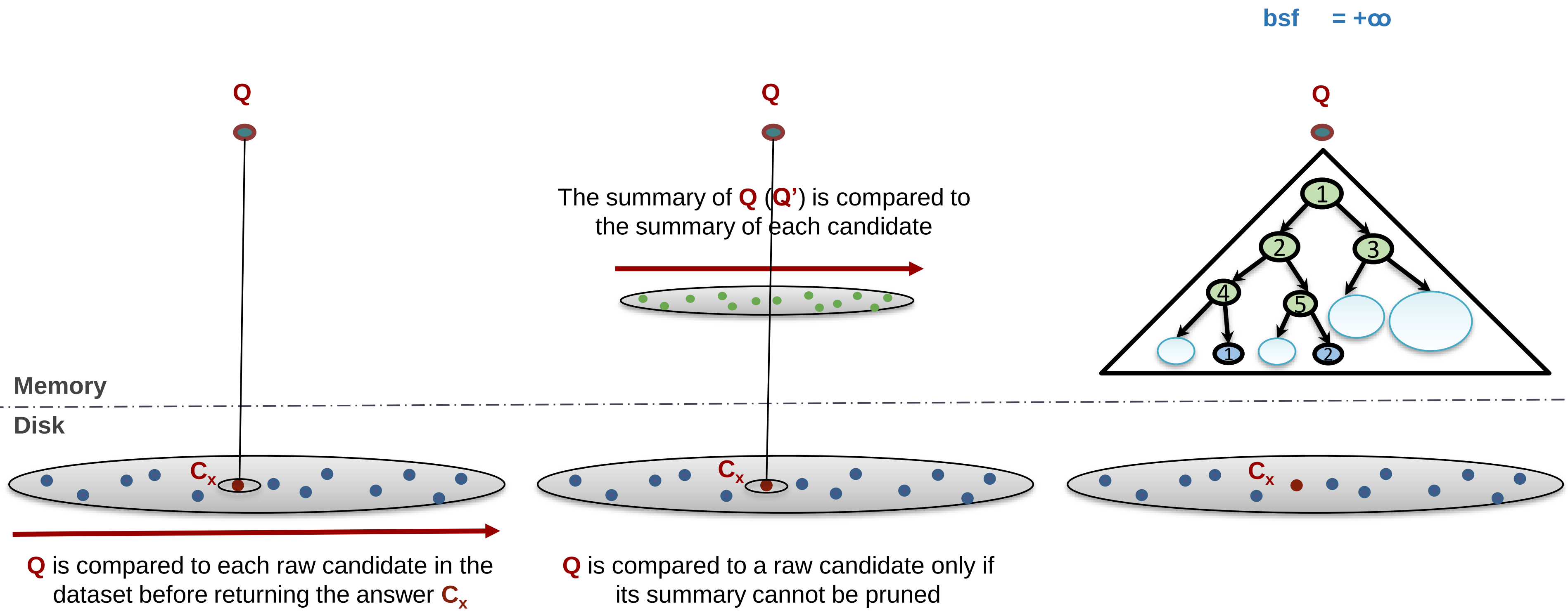
(a) Serial scan

(b) Skip-sequential scan

(c) Tree-based index

Answering a similarity search query using different access paths

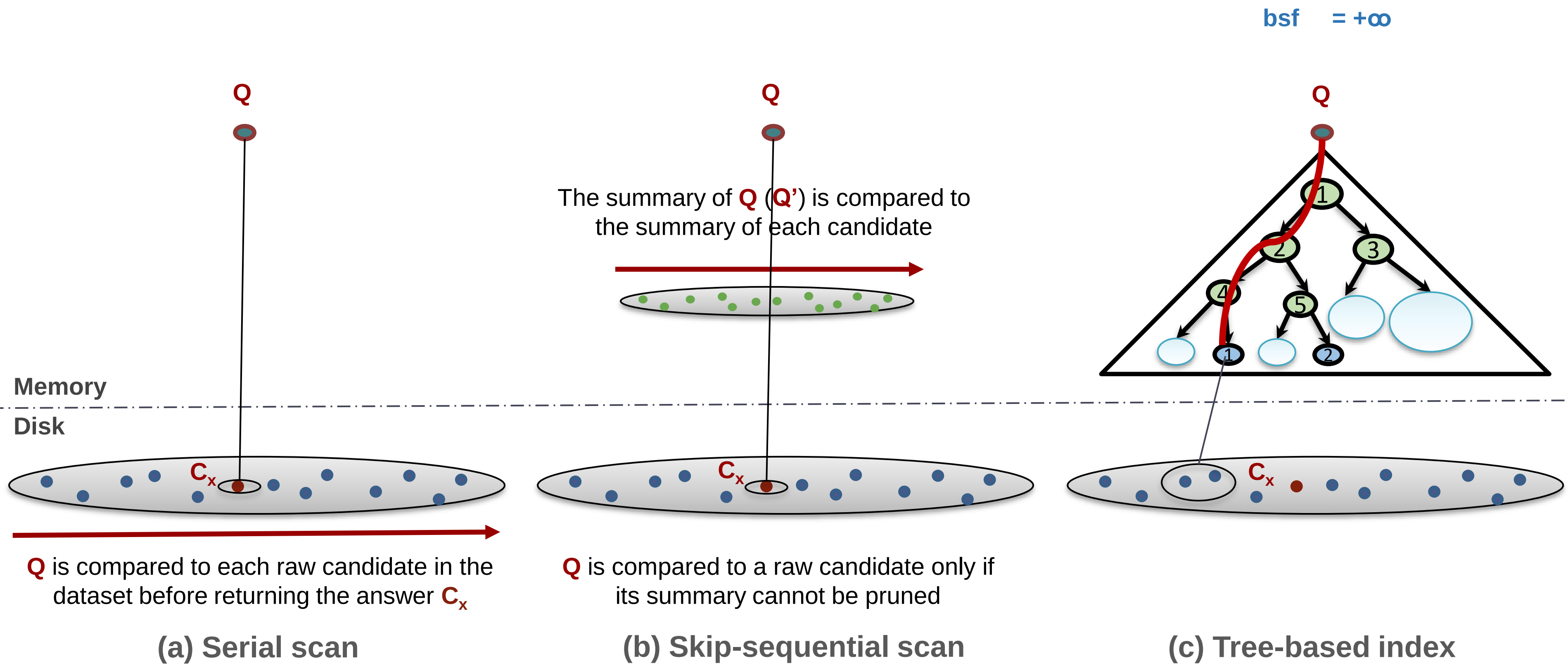
Indexes vs. Scans



(a) Serial scan (b) Skip-sequential scan (c) Tree-based index

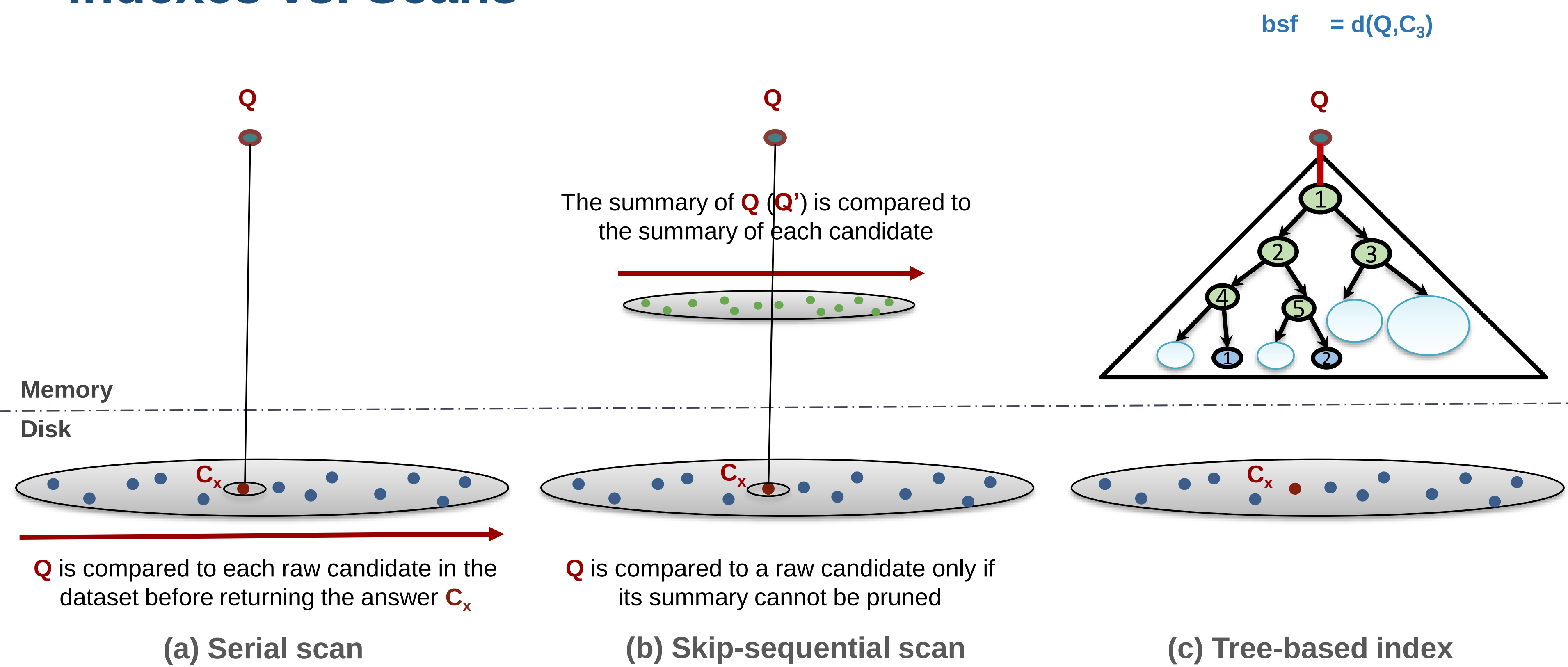
Answering a similarity search query using different access paths

Indexes vs. Scans



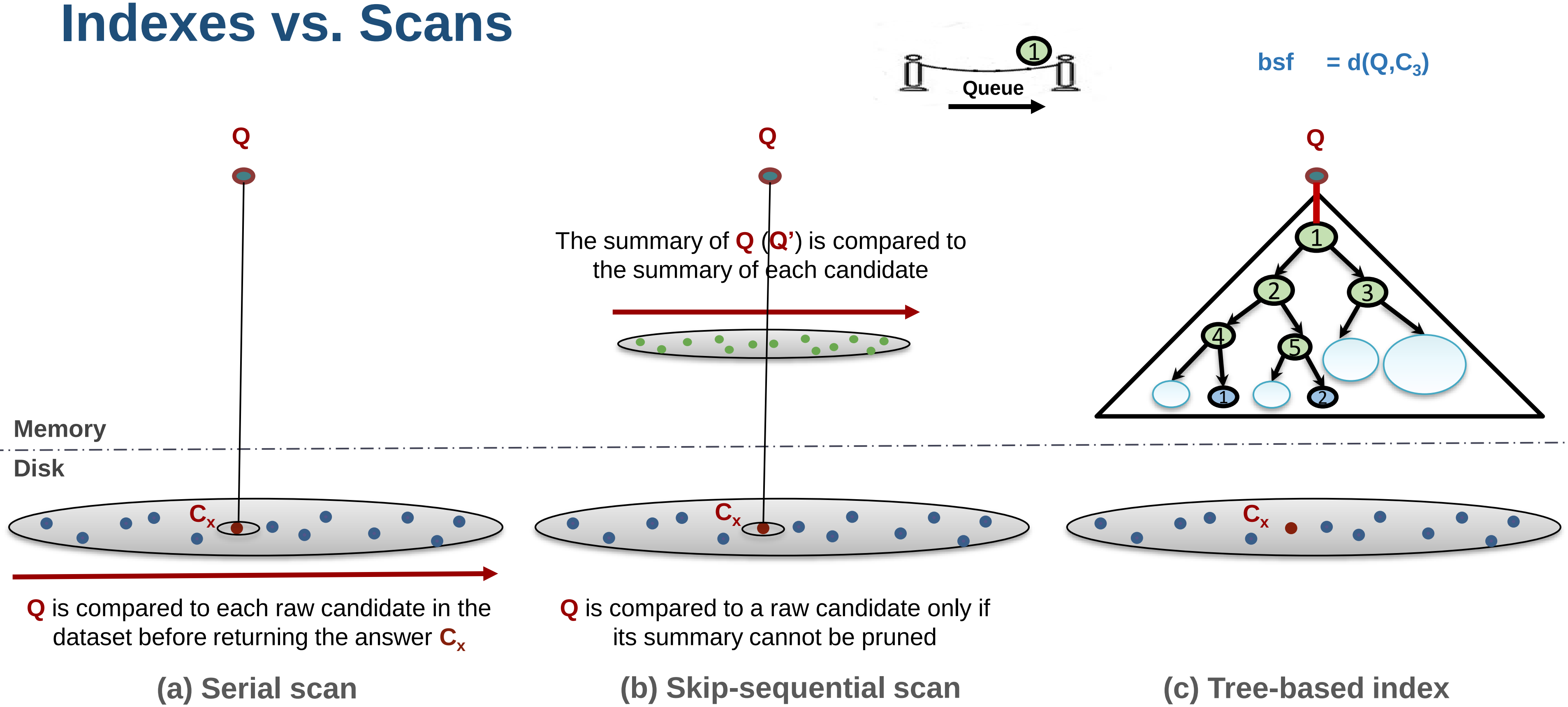
Answering a similarity search query using different access paths

Indexes vs. Scans



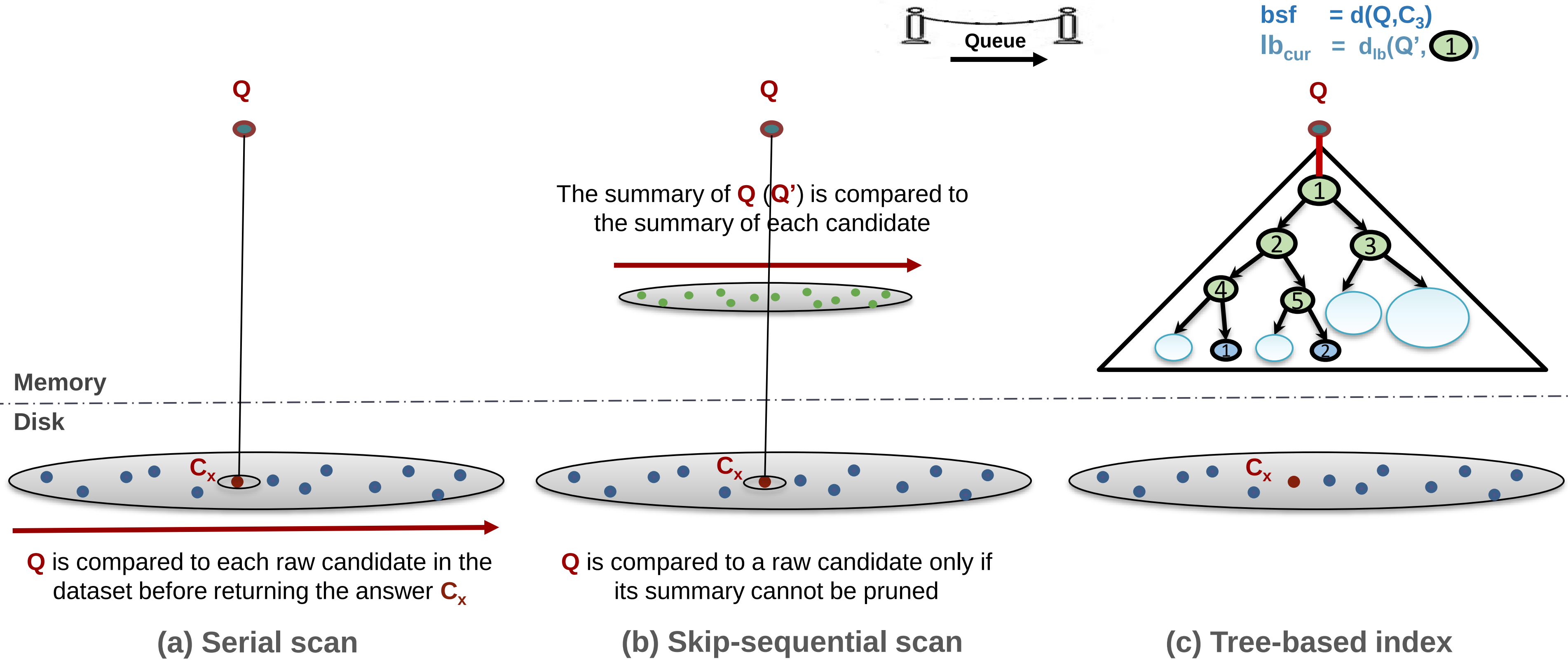
<u>**Answering a similarity search query using different access paths**</u>

Indexes vs. Scans



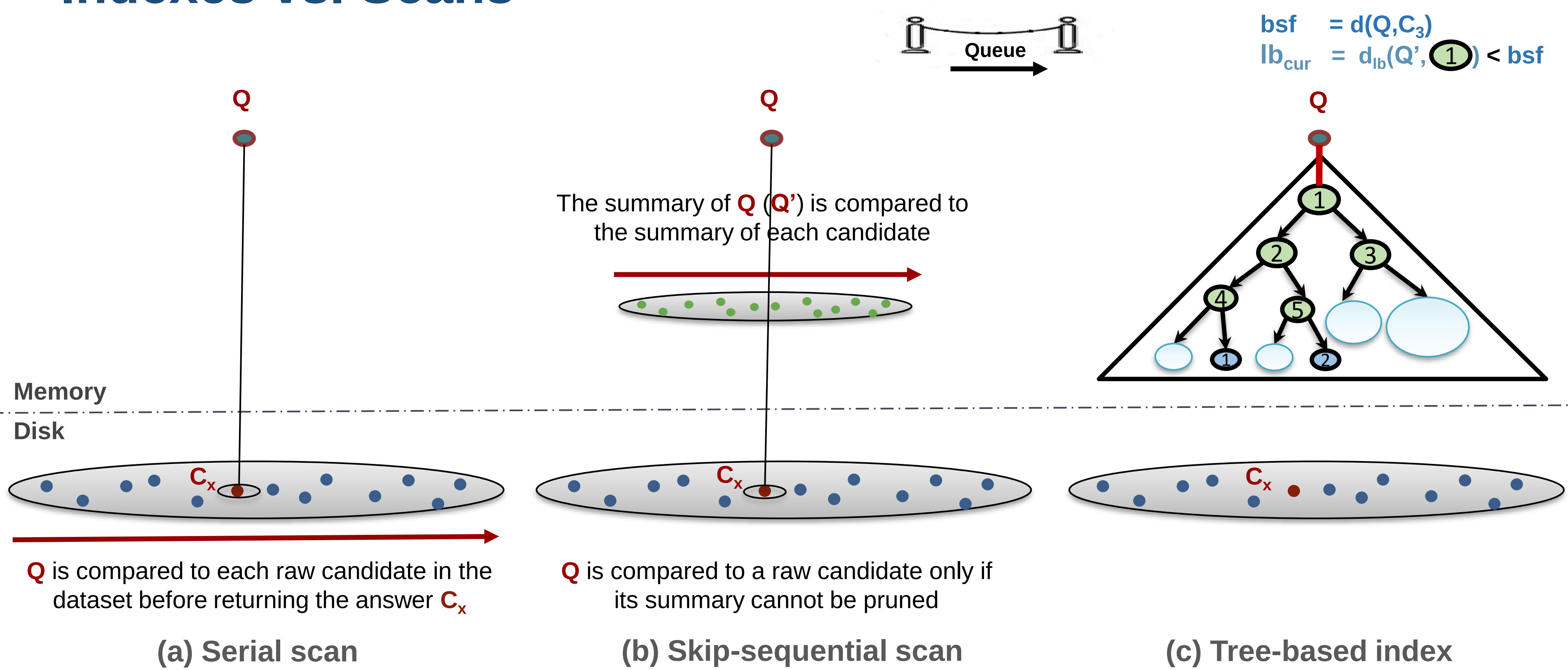
Answering a similarity search query using different access paths

Indexes vs. Scans



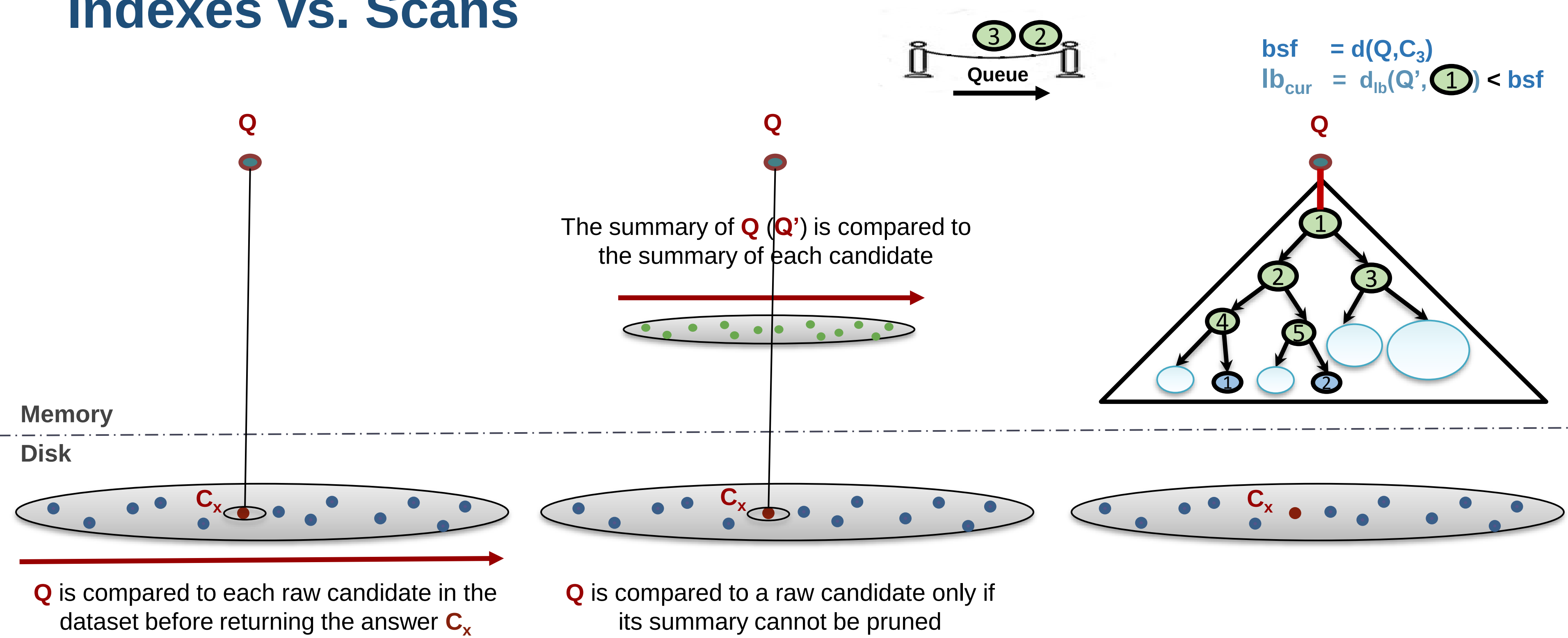
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

Indexes vs. Scans



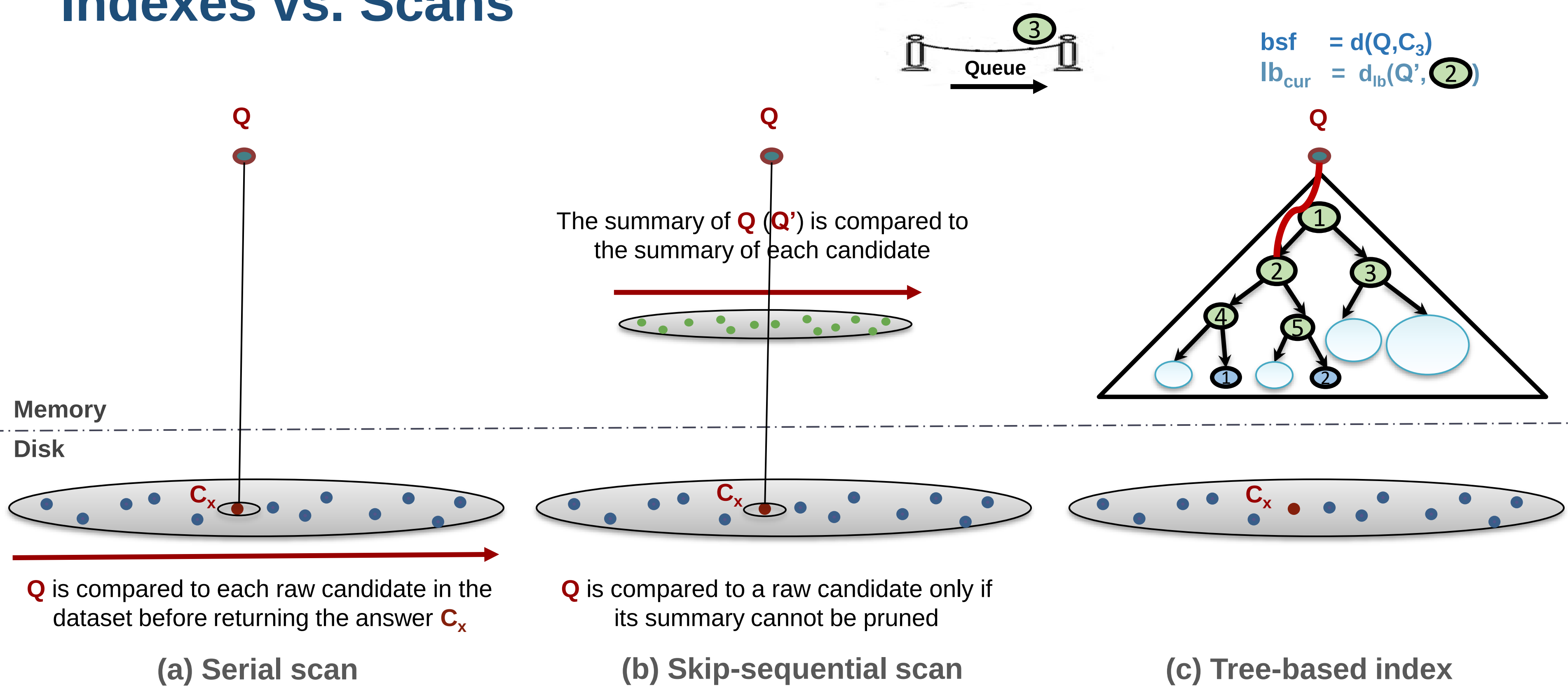
(a) Serial scan

(b) Skip-sequential scan

(c) Tree-based index

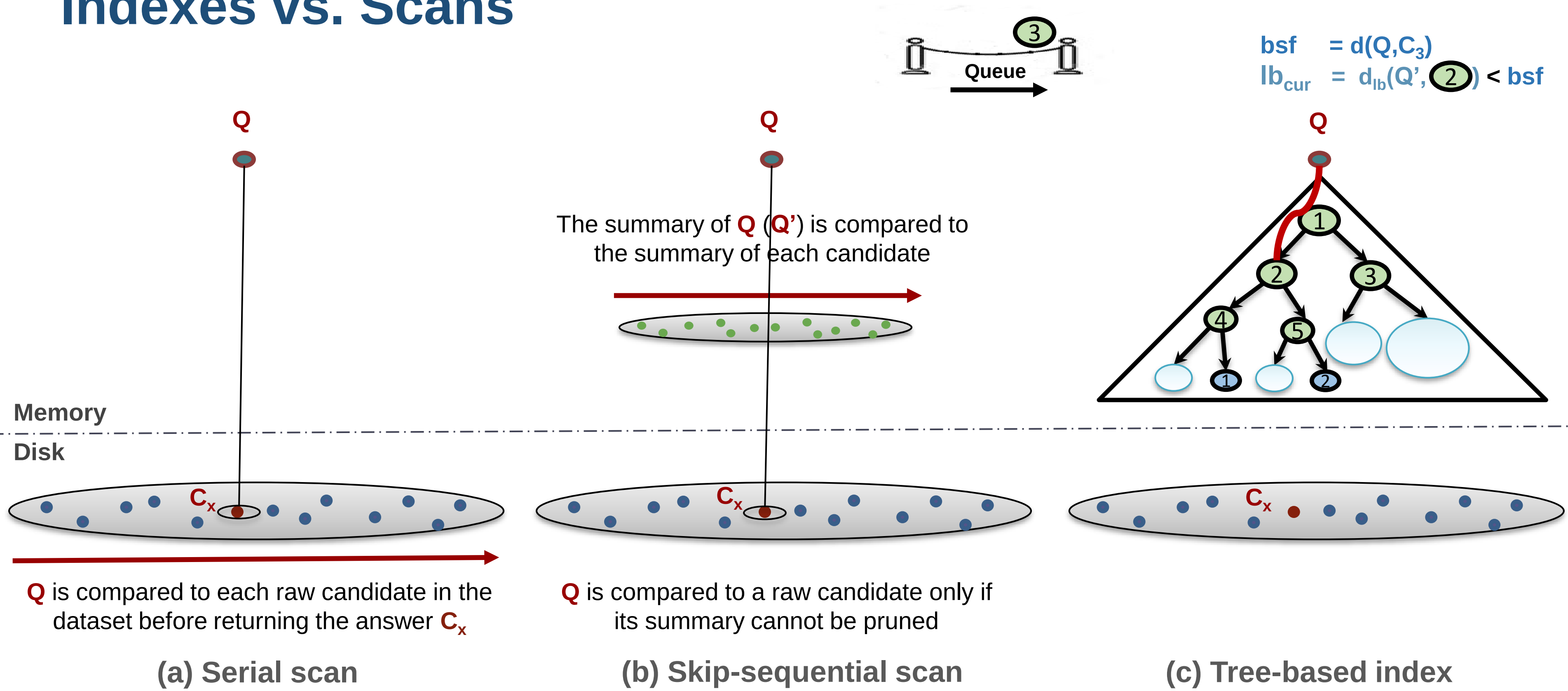
Answering a similarity search query using different access paths

Indexes vs. Scans



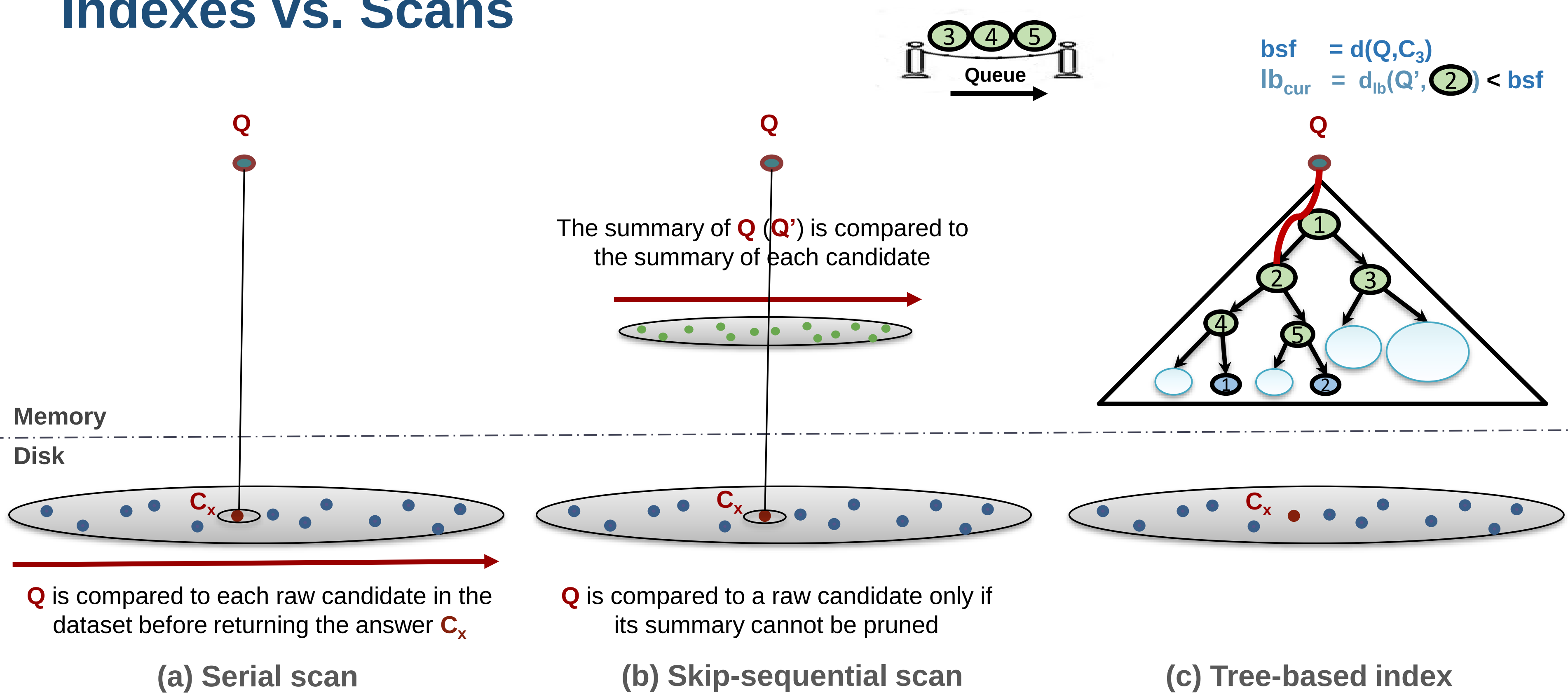
Answering a similarity search query using different access paths

Indexes vs. Scans



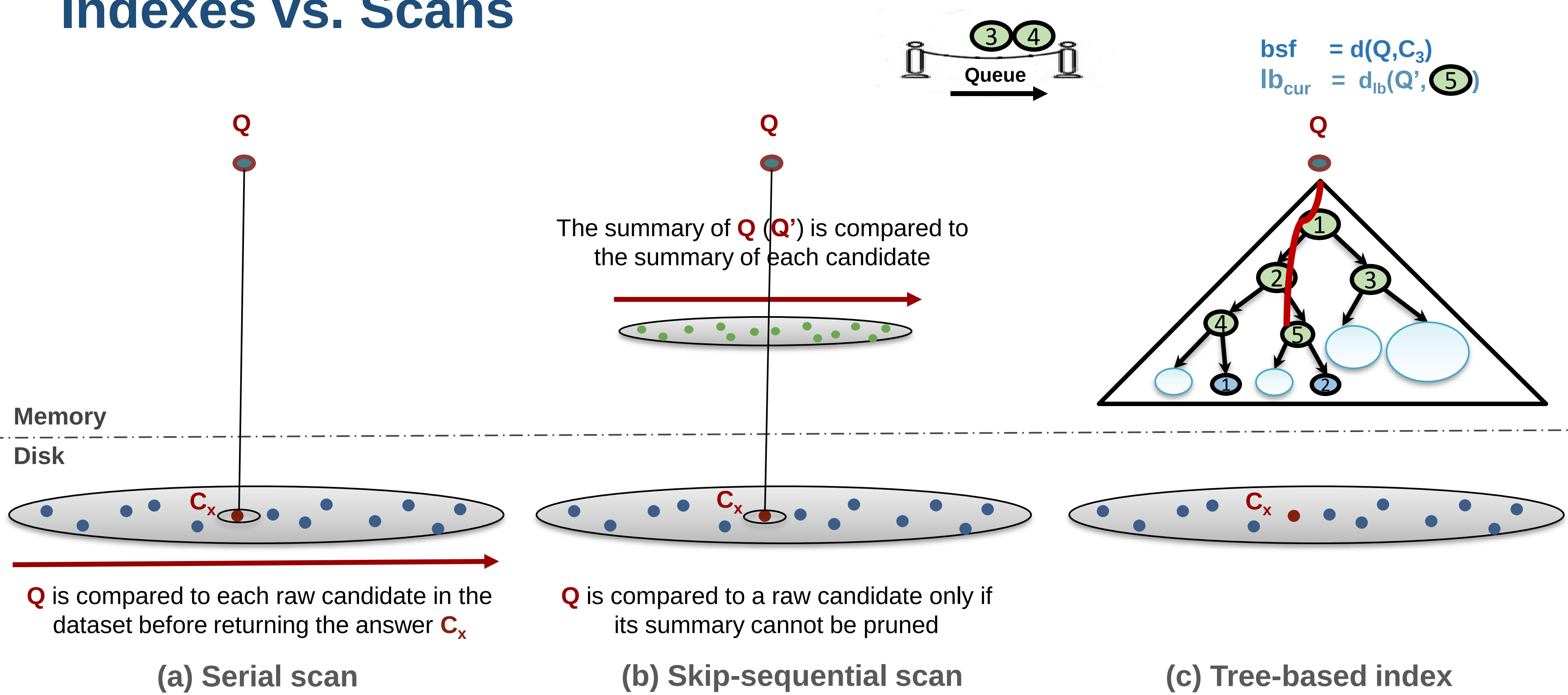
Answering a similarity search query using different access paths

Indexes vs. Scans



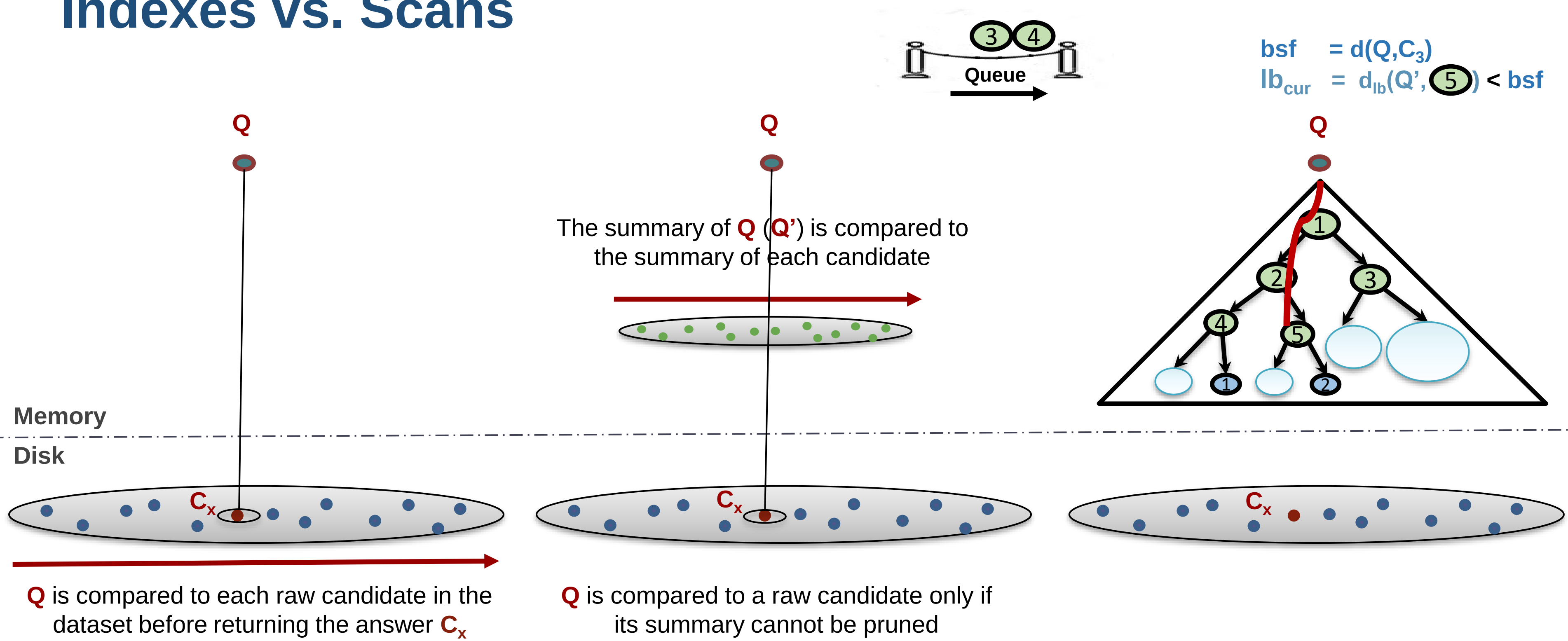
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

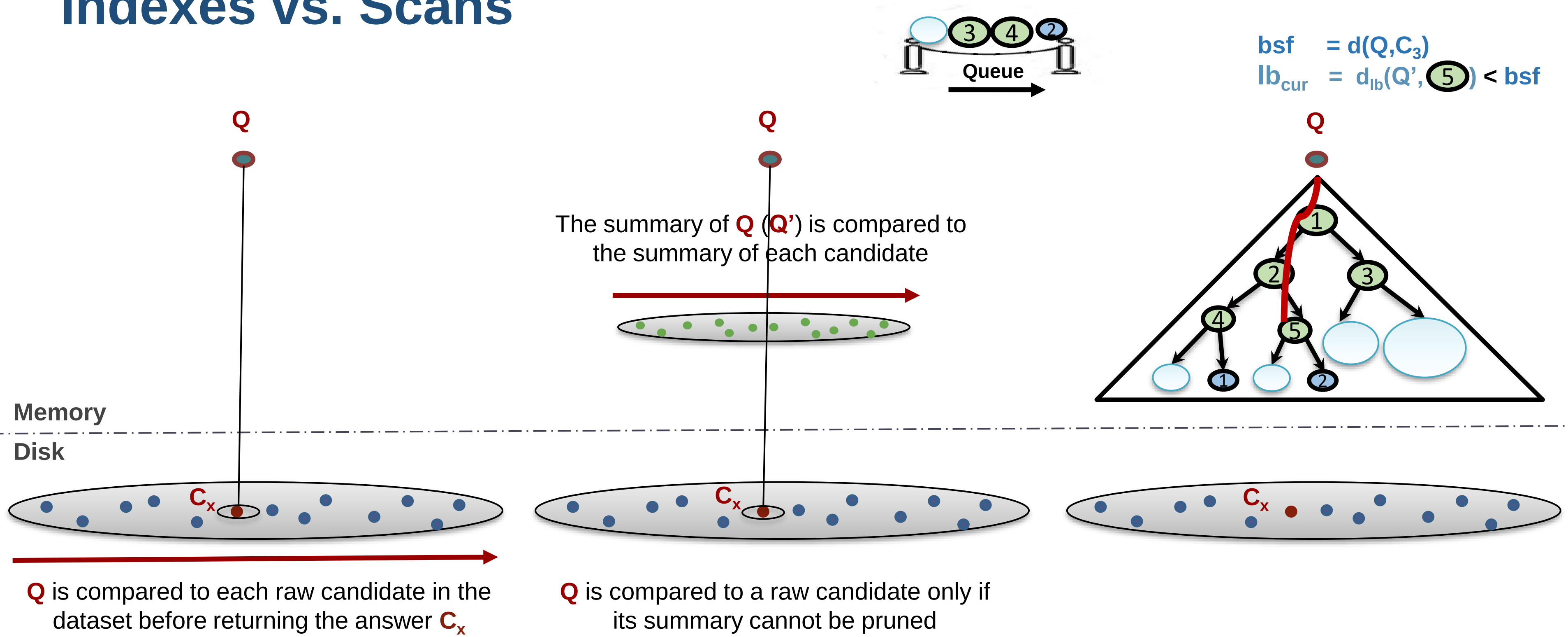
Indexes vs. Scans



(a) Serial scan (b) Skip-sequential scan (c) Tree-based index

Answering a similarity search query using different access paths

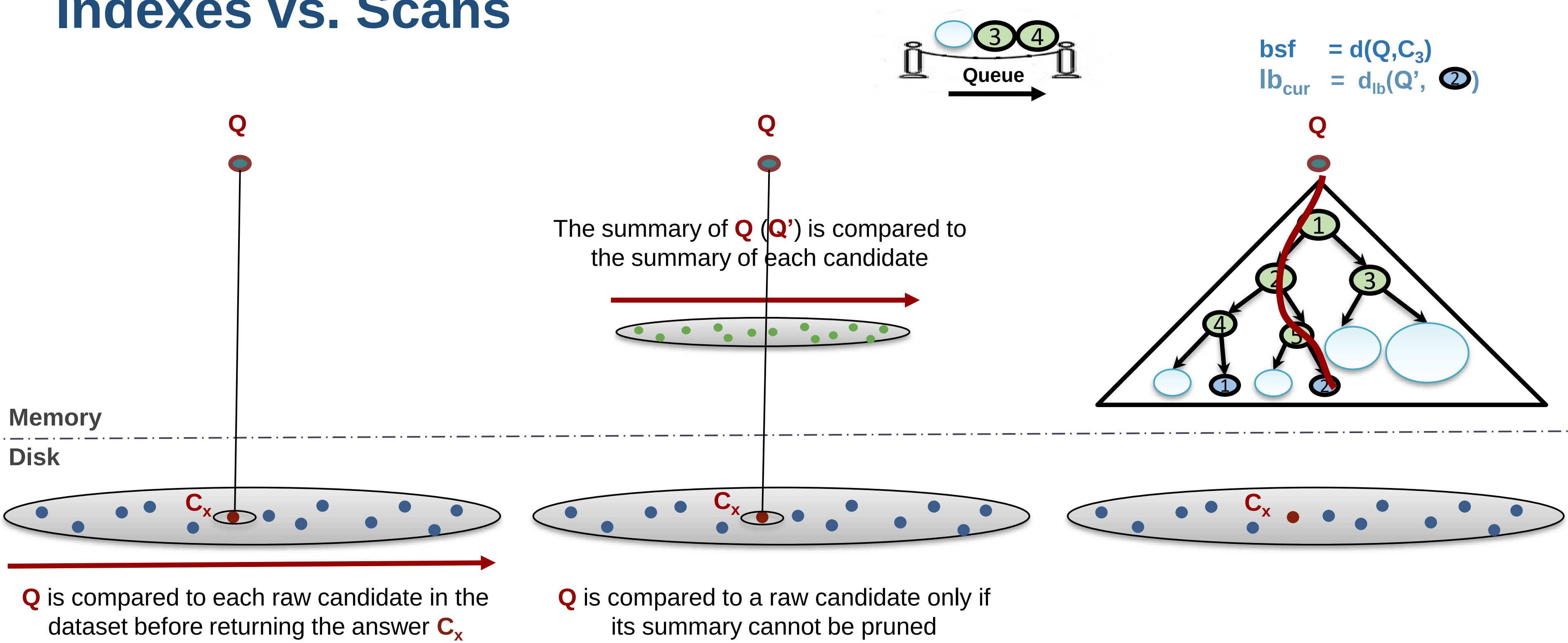
Indexes vs. Scans



(a) Serial scan (b) Skip-sequential scan (c) Tree-based index

Answering a similarity search query using different access paths

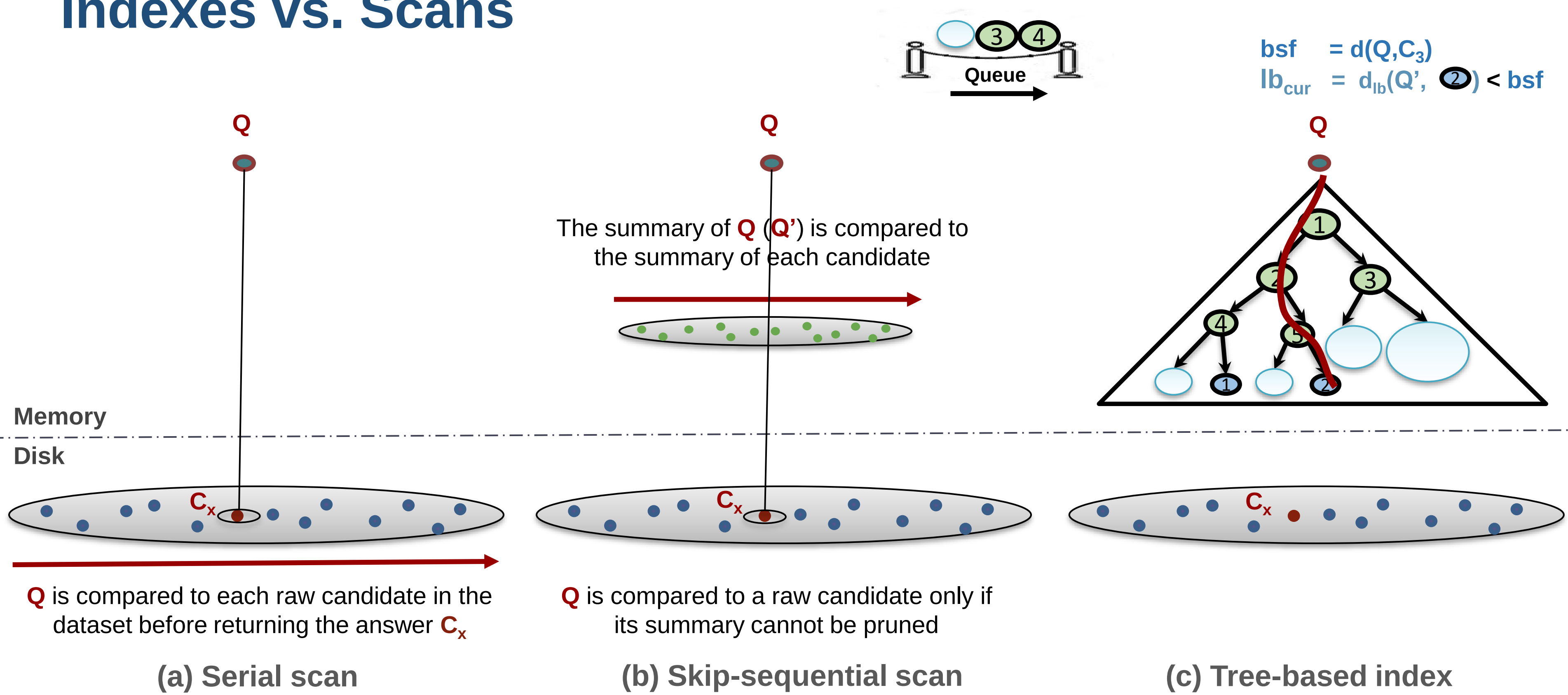
Indexes vs. Scans



(a) Serial scan (b) Skip-sequential scan (c) Tree-based index

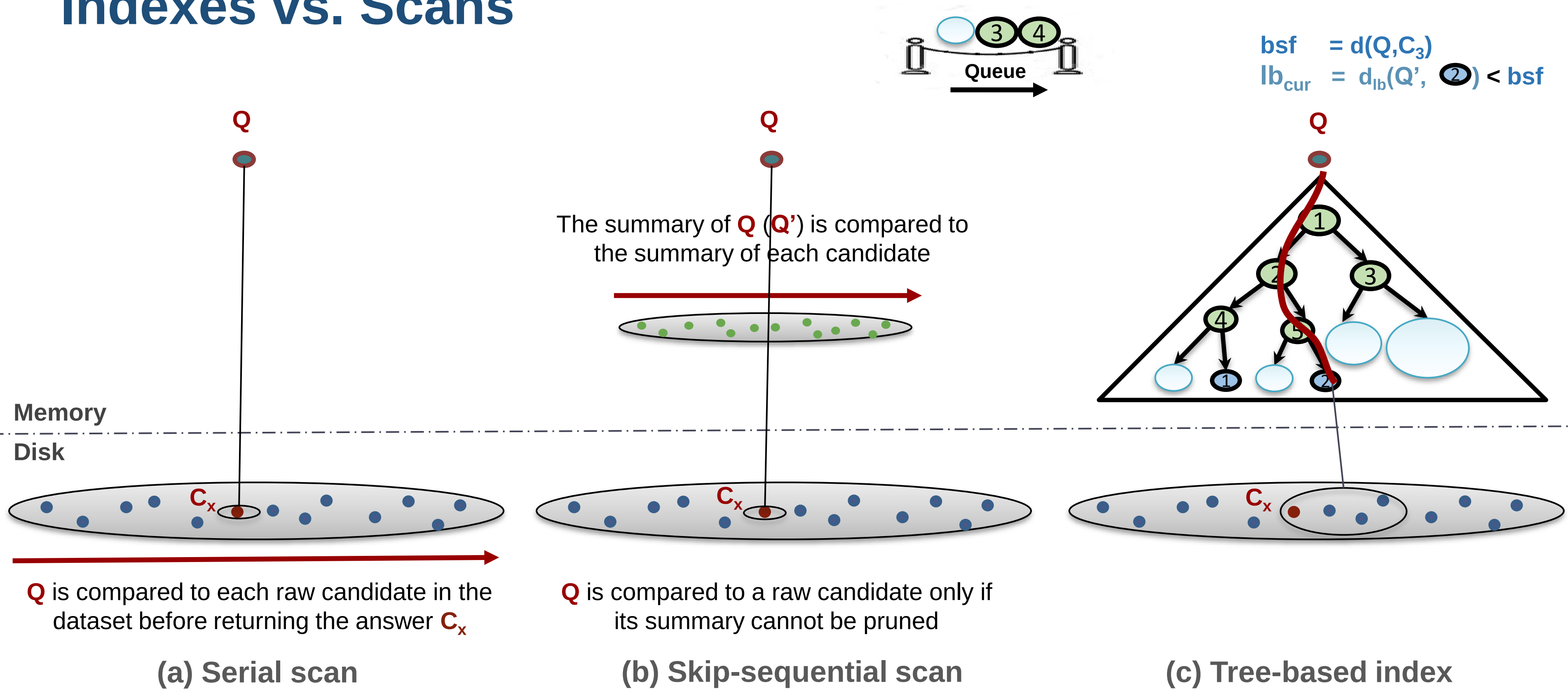
<u>Answering a similarity search query using different access paths</u>

Indexes vs. Scans



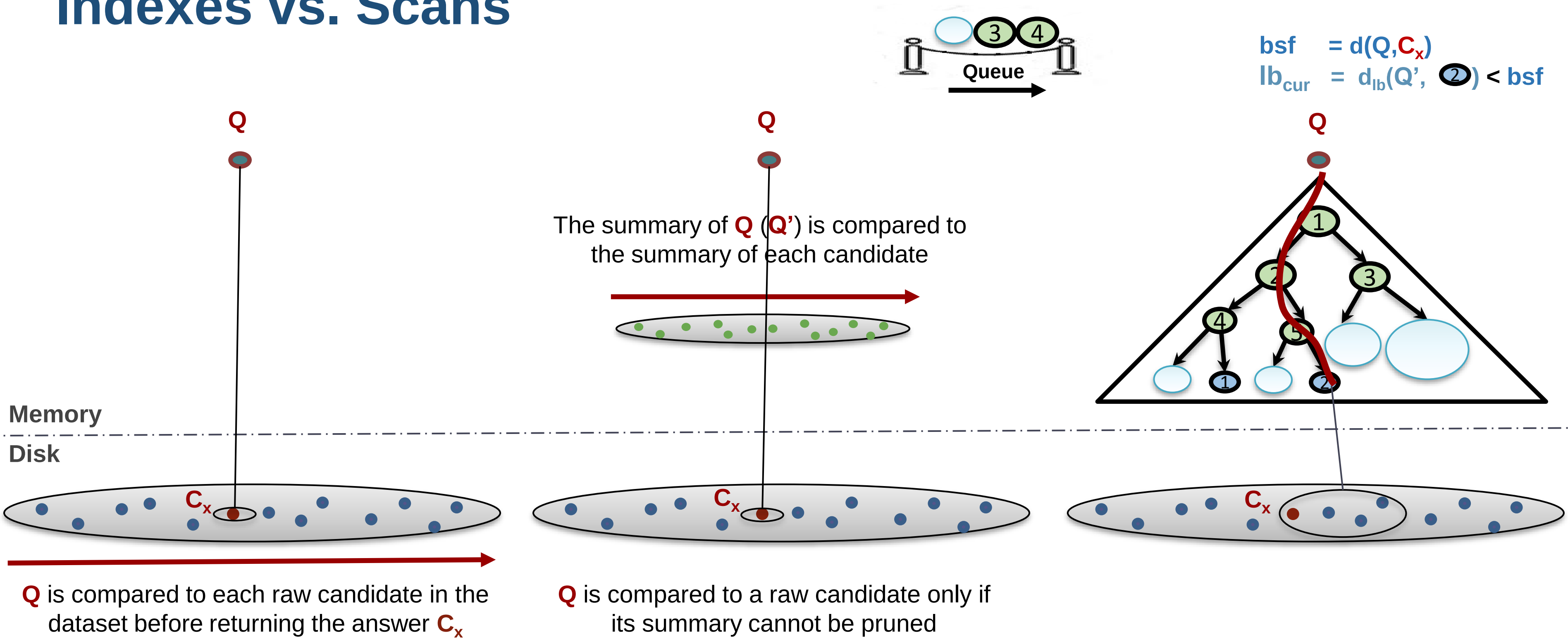
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

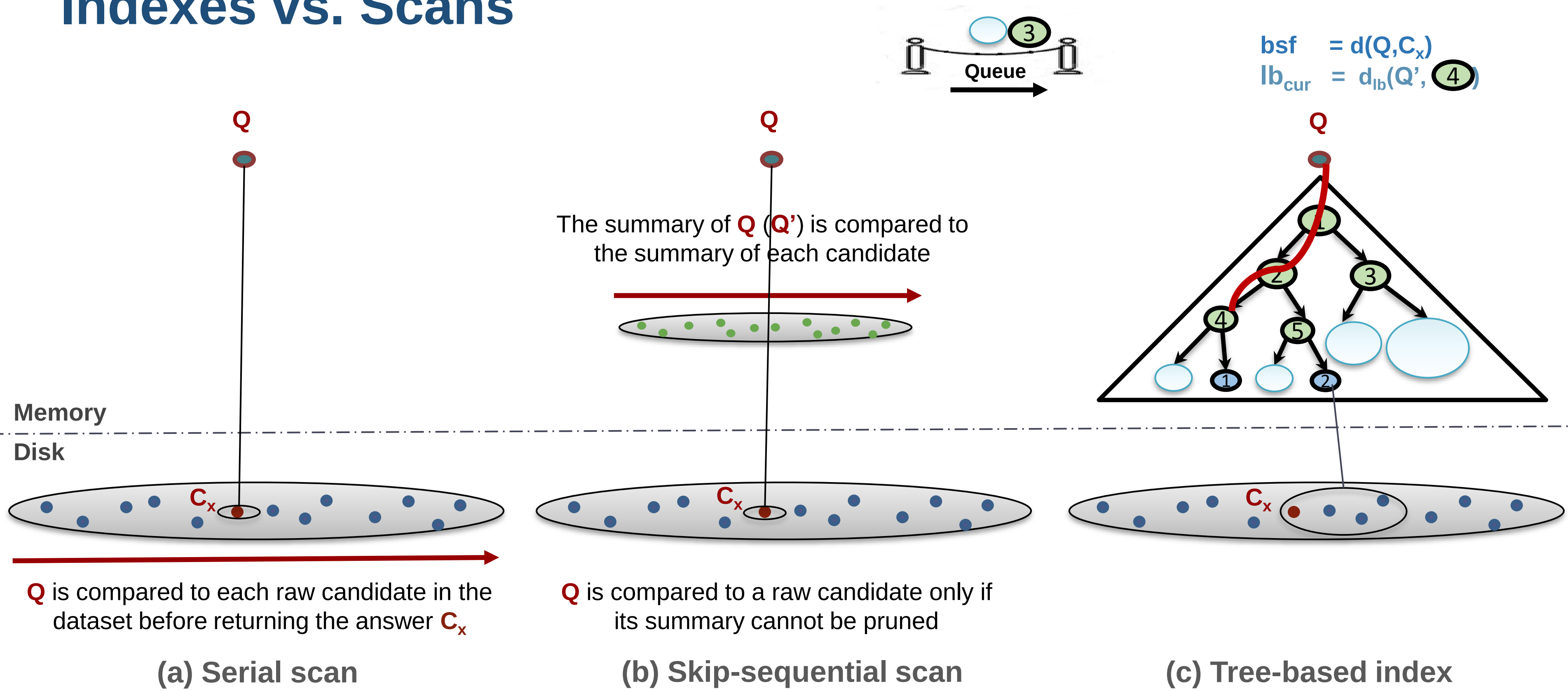
Indexes vs. Scans



(a) Serial scan (b) Skip-sequential scan (c) Tree-based index

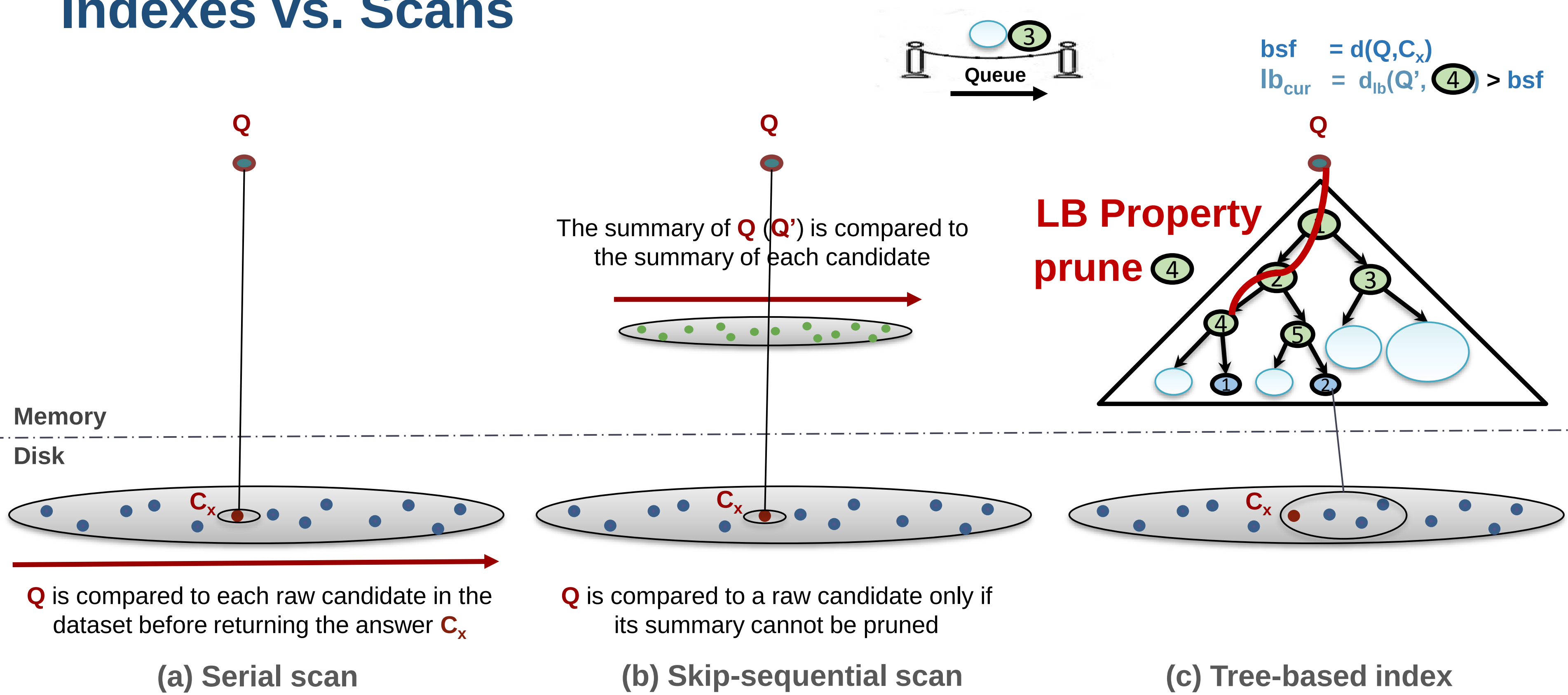
Answering a similarity search query using different access paths

Indexes vs. Scans



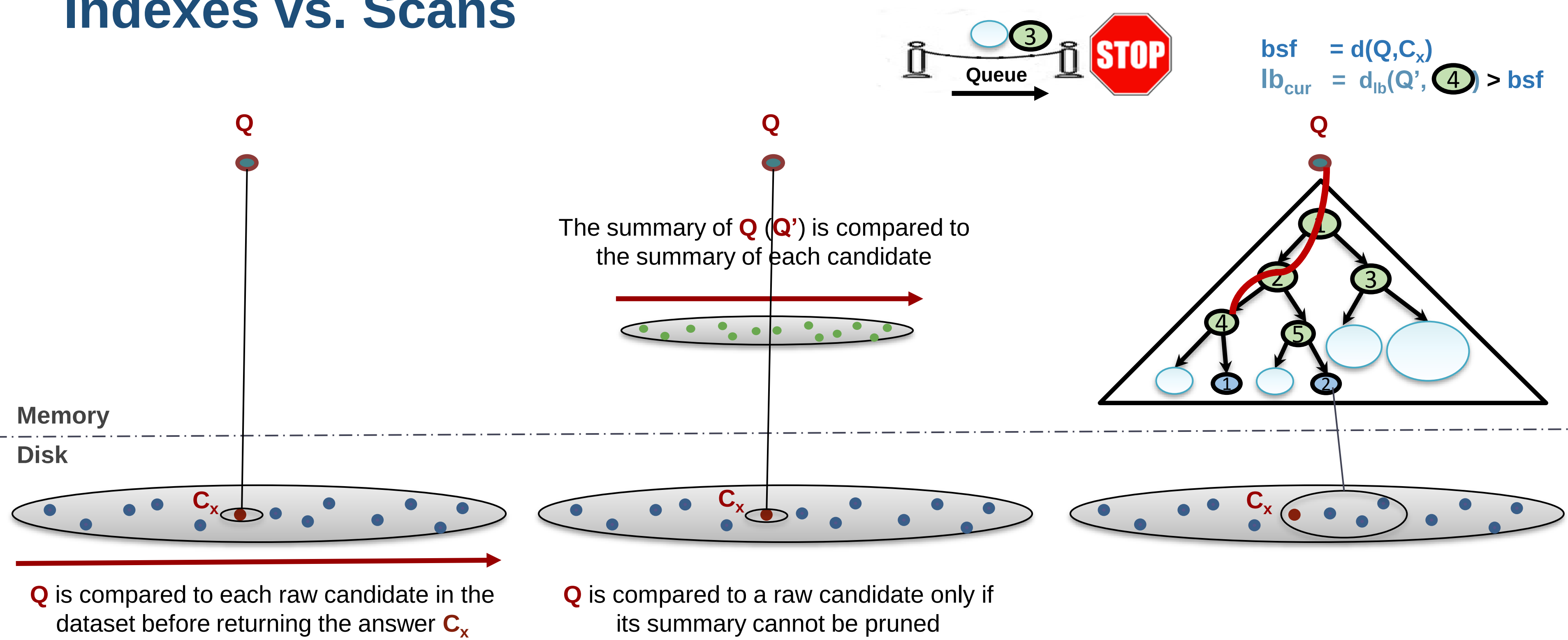
Answering a similarity search query using different access paths

Indexes vs. Scans



Answering a similarity search query using different access paths

Indexes vs. Scans



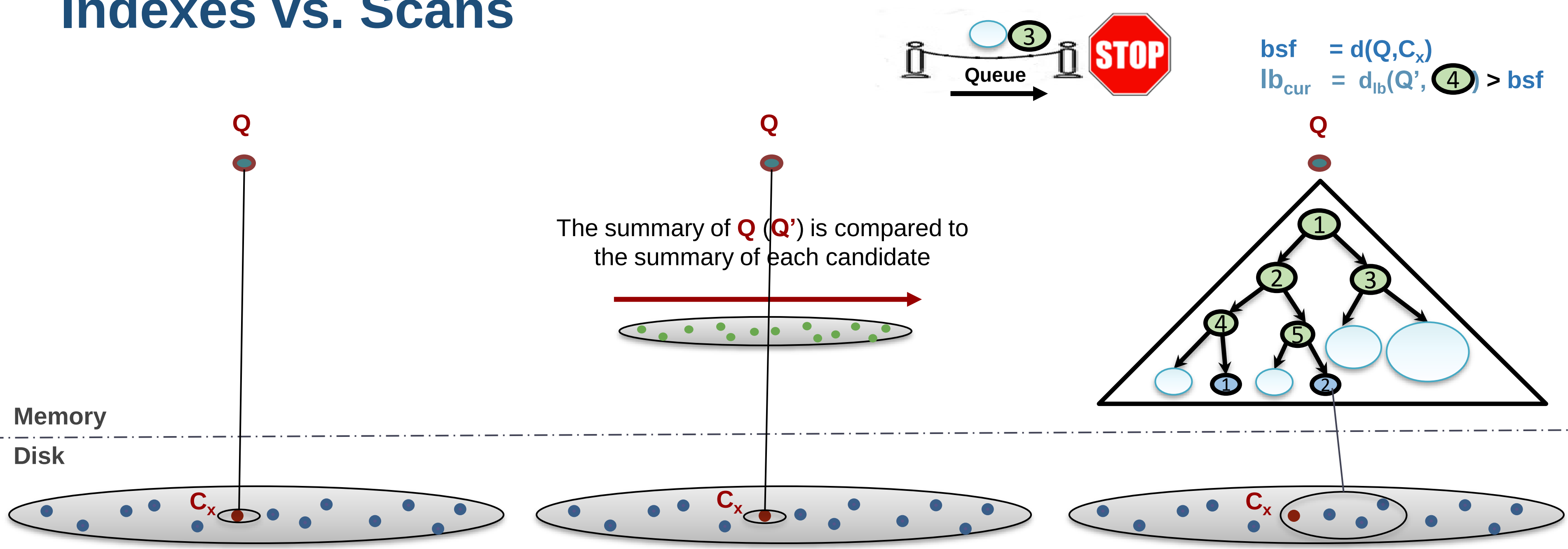
(a) Serial scan

(b) Skip-sequential scan

(c) Tree-based index

Answering a similarity search query using different access paths

Indexes vs. Scans



Q is compared to each raw candidate in the dataset before returning the answer C_x

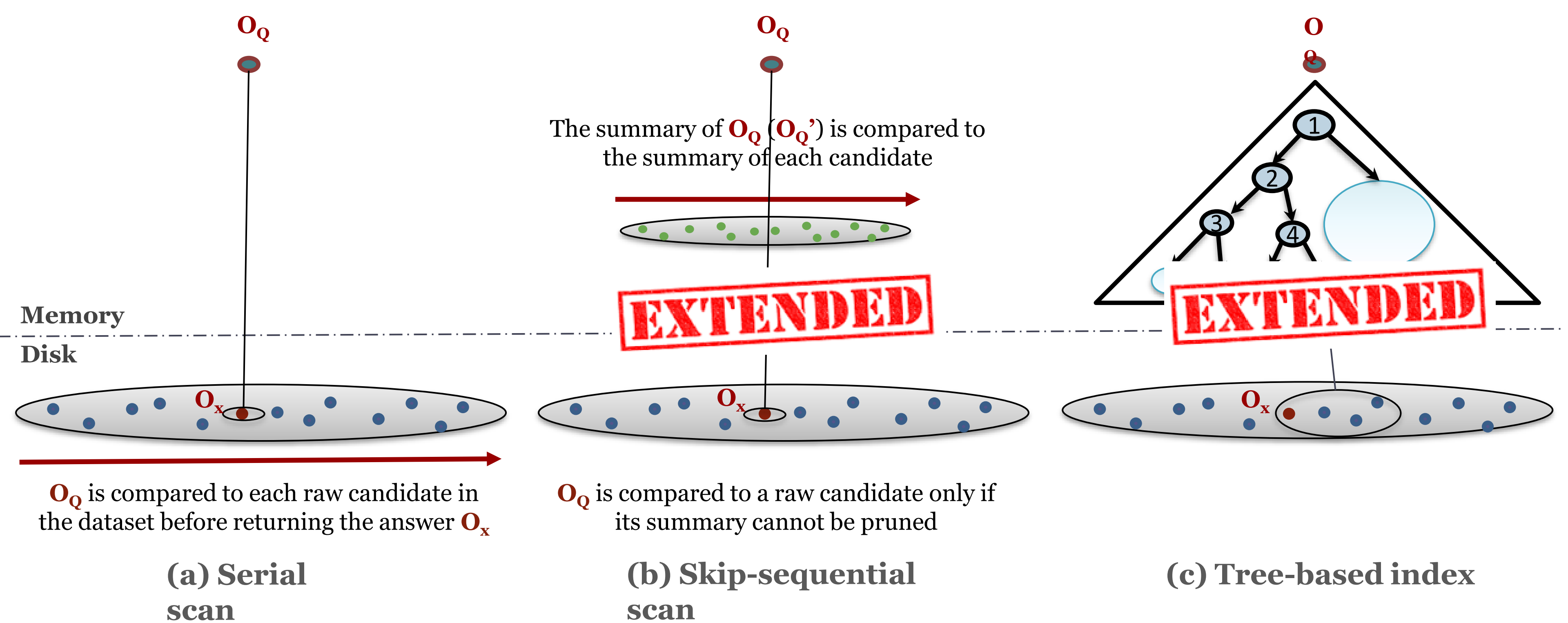
Q is compared to a raw candidate only if its summary cannot be pruned

(a) Serial scan **(b) Skip-sequential scan** **(c) Tree-based index**

Answering a similarity search query using different access paths

Similarity Search
Data Series Extensions

Access Paths

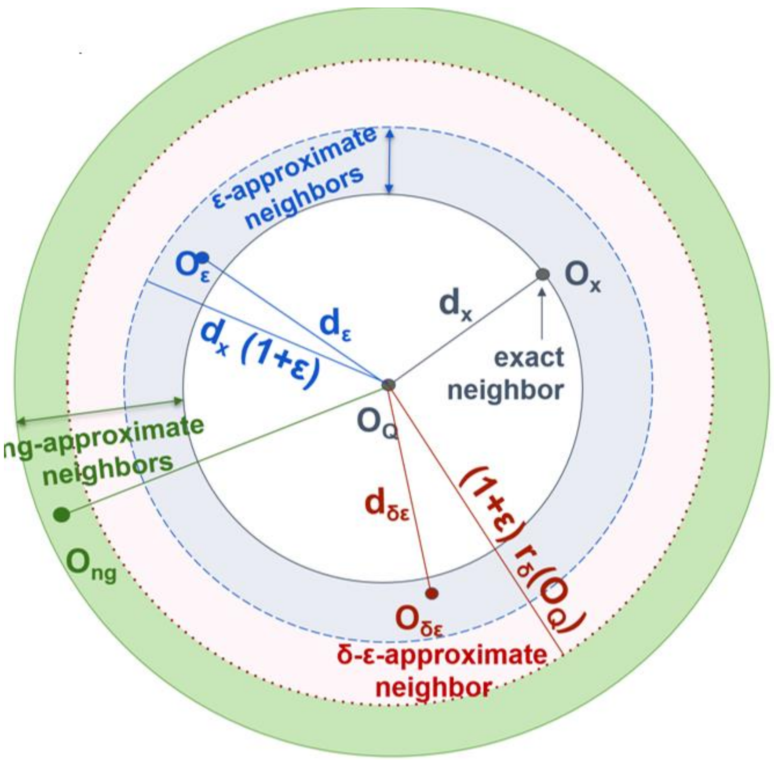


Answering a similarity search query using different access paths

Extensions: Skip-Sequential Scans

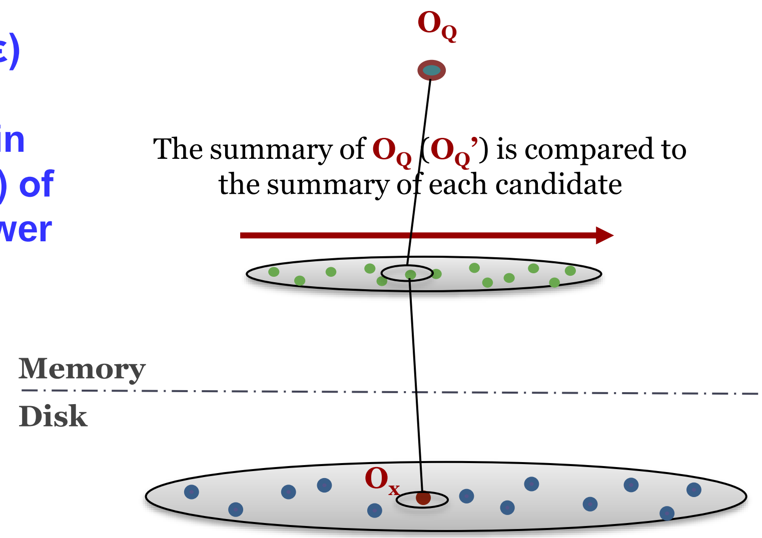
bsf $= d(O_Q, O_1)$

$lb_{cur} = d_{lb}(O_Q', O_x') <$ bsf

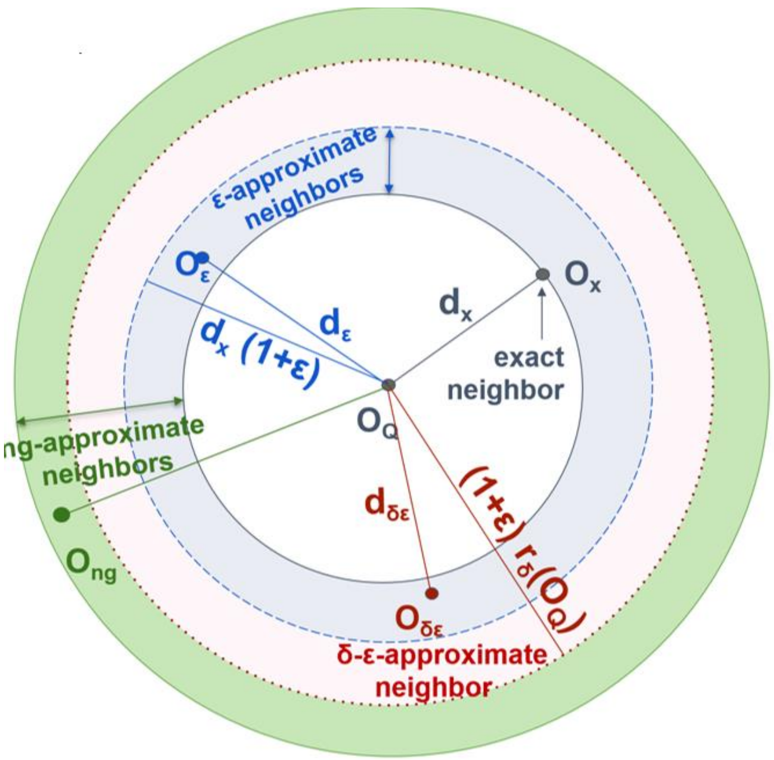


$d_\varepsilon \le d_x (1+\varepsilon)$

Result is within distance (1+ ε) of the exact answer



Extensions: Skip-Sequential Scans



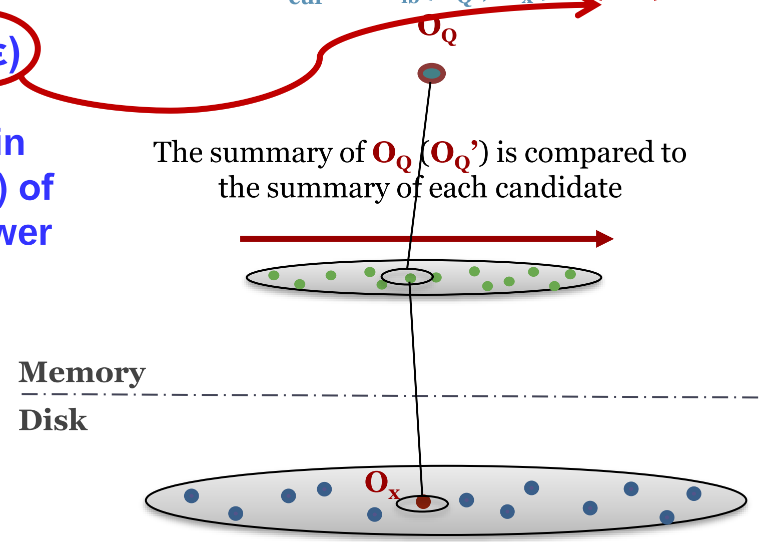
d_ε <= $d_x(1+\varepsilon)$

Result is within distance (1+ ε) of the exact answer

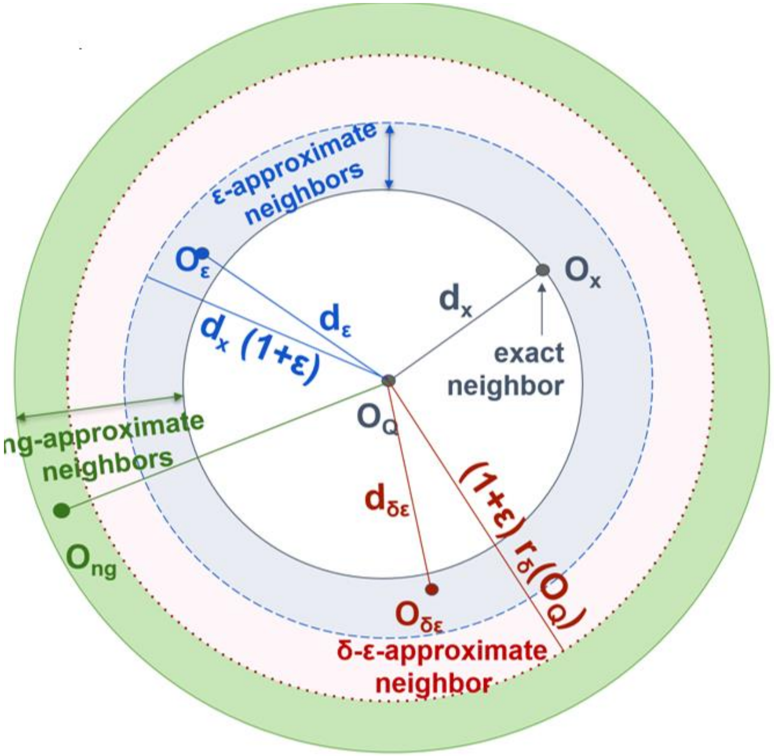
$bsf = d(O_Q, O_1)$

$\cancel{lb_{cur} = d_{lb}(O_Q', O_x') < bsf}$

$lb_{cur} = d_{lb}(O_Q', O_x') < (1+\varepsilon)\ bsf$



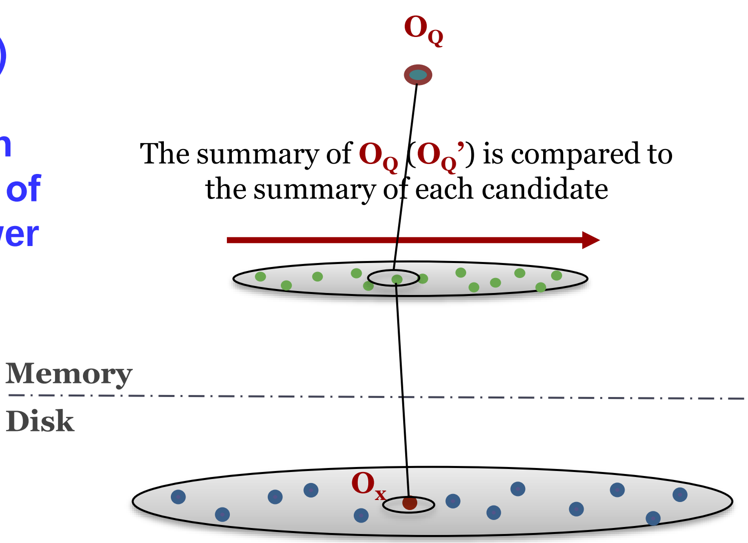
Extensions: Skip-Sequential Scans



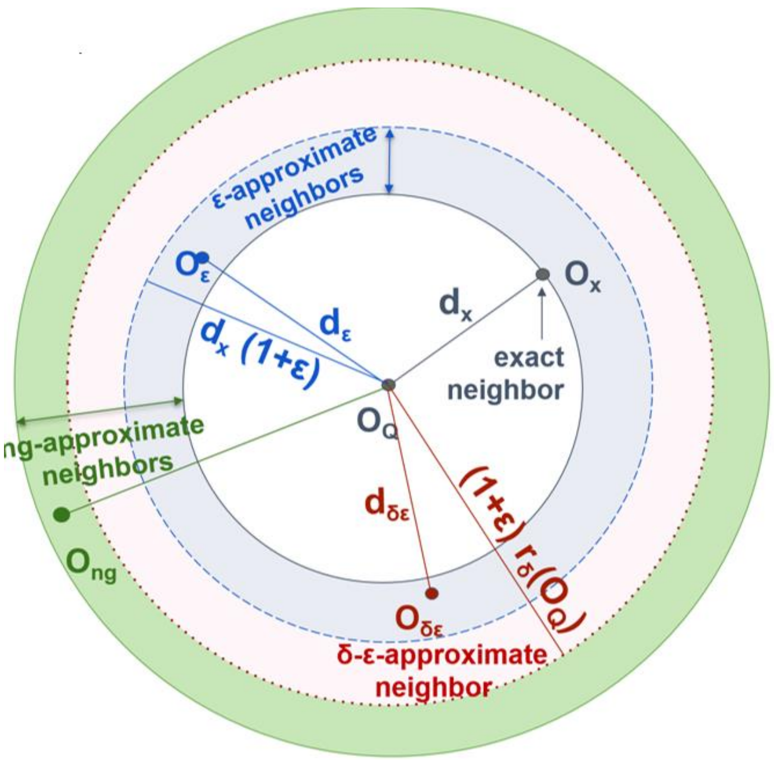
$d_\varepsilon <= d_x (1+\varepsilon)$

Result is within distance (1+ ε) of the exact answer

bsf = $d(O_Q, O_1)$

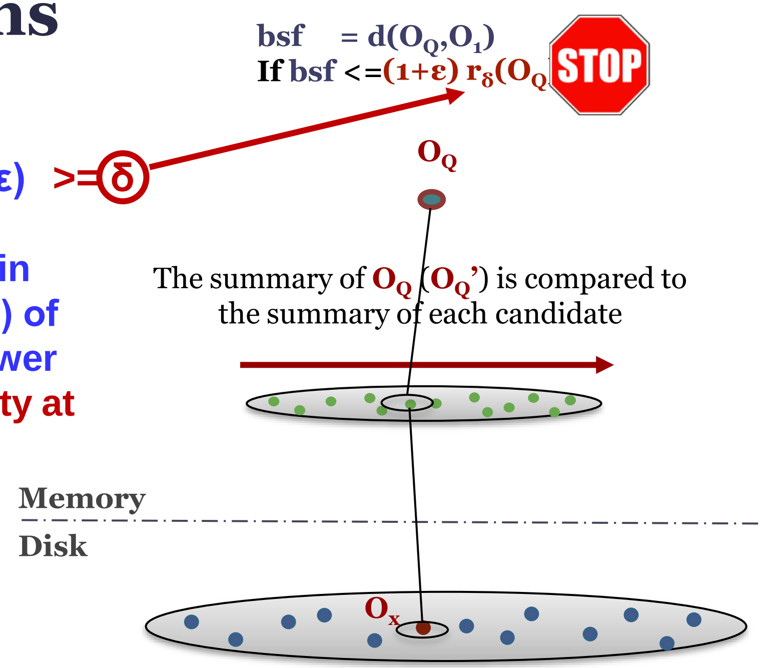


Extensions: Skip-Sequential Scans

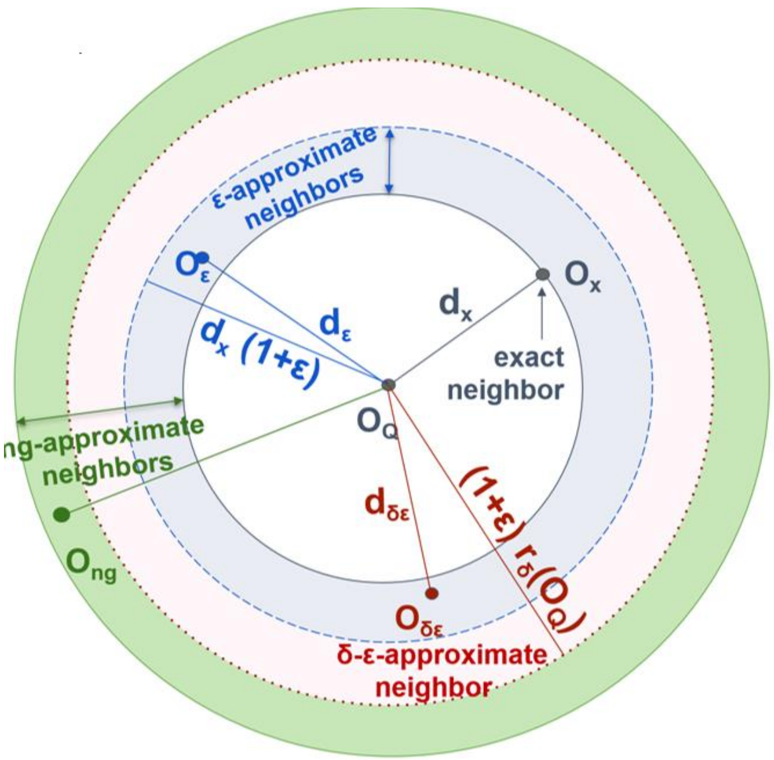


$P\{d_\varepsilon <= d_x\,(1+\varepsilon)\} >= \delta$

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

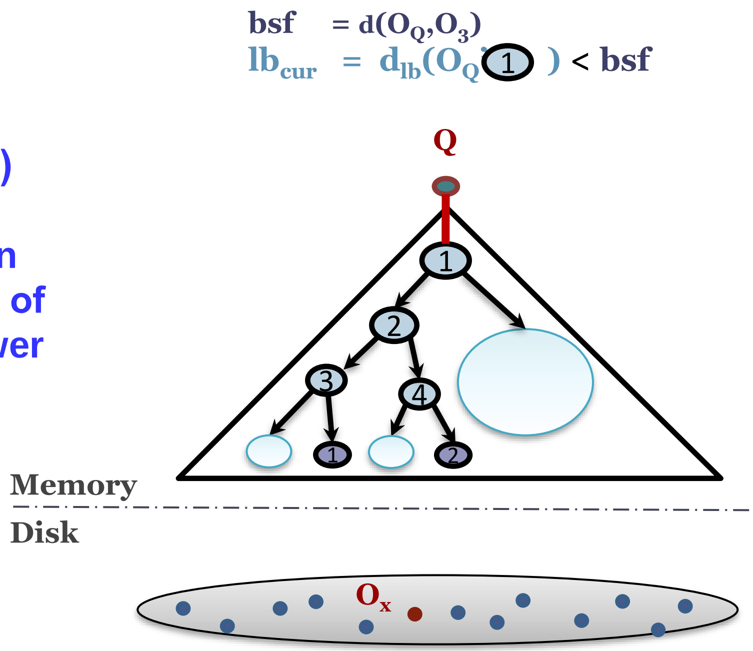


Extensions: Indexes

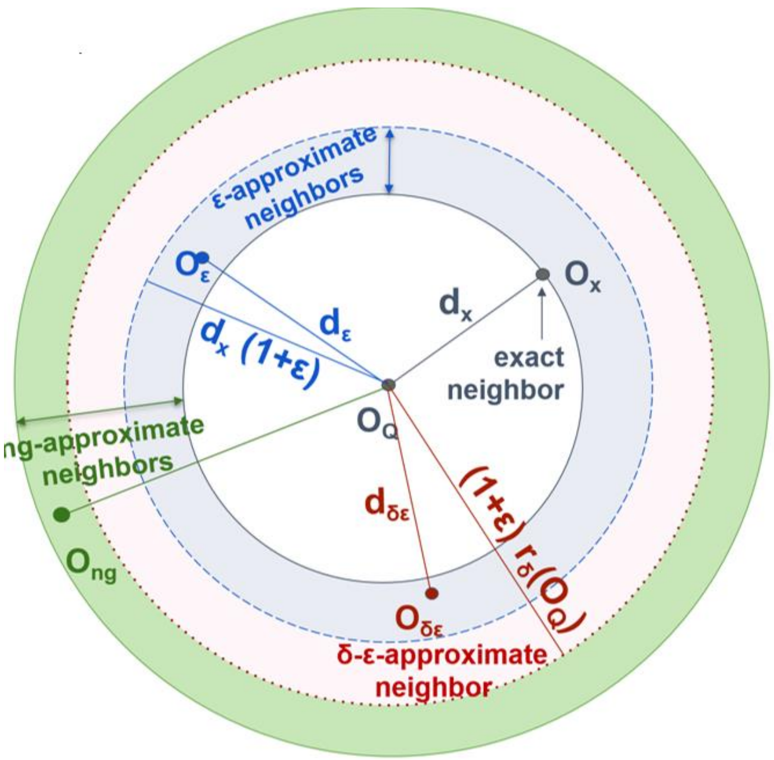


d_ε <= $d_x\,(1+\varepsilon)$

Result is within distance (1+ ε) of the exact answer

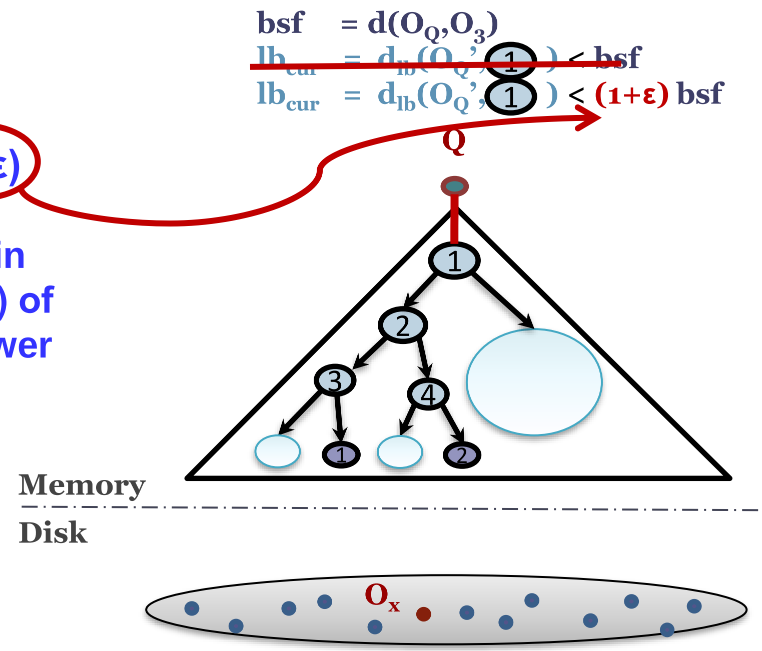


Extensions: Indexes

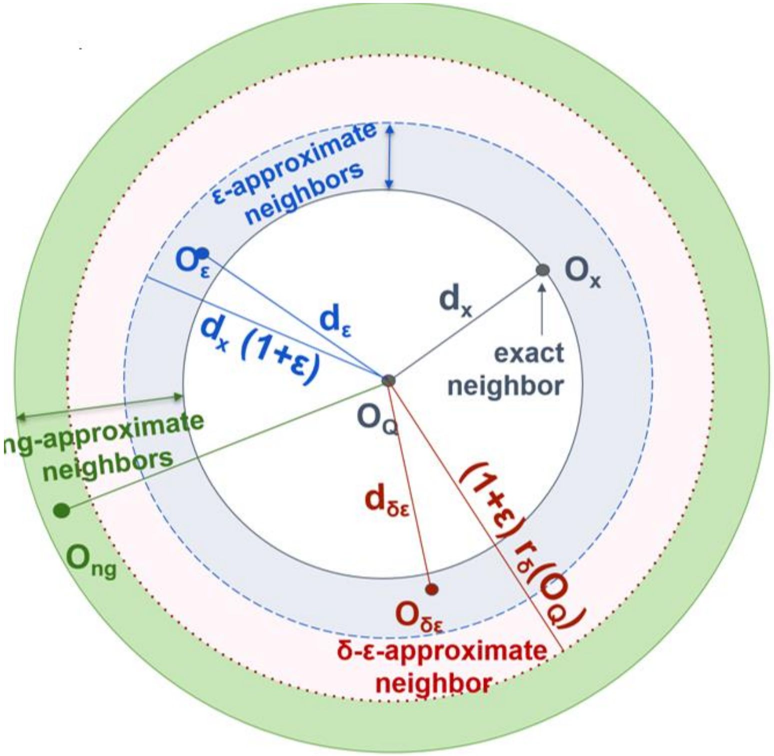


d_ε <= d_x(1+ε)

Result is within distance (1+ ε) of the exact answer

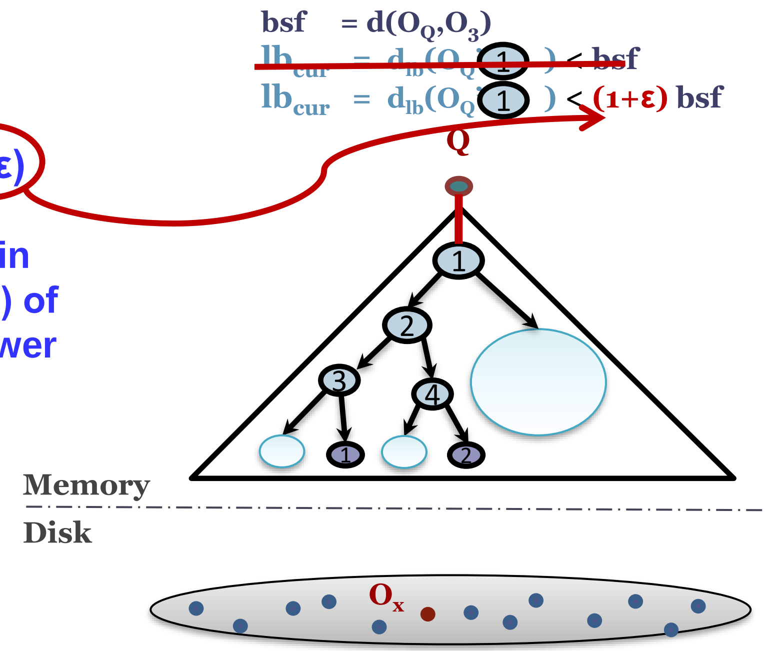


Extensions: Indexes

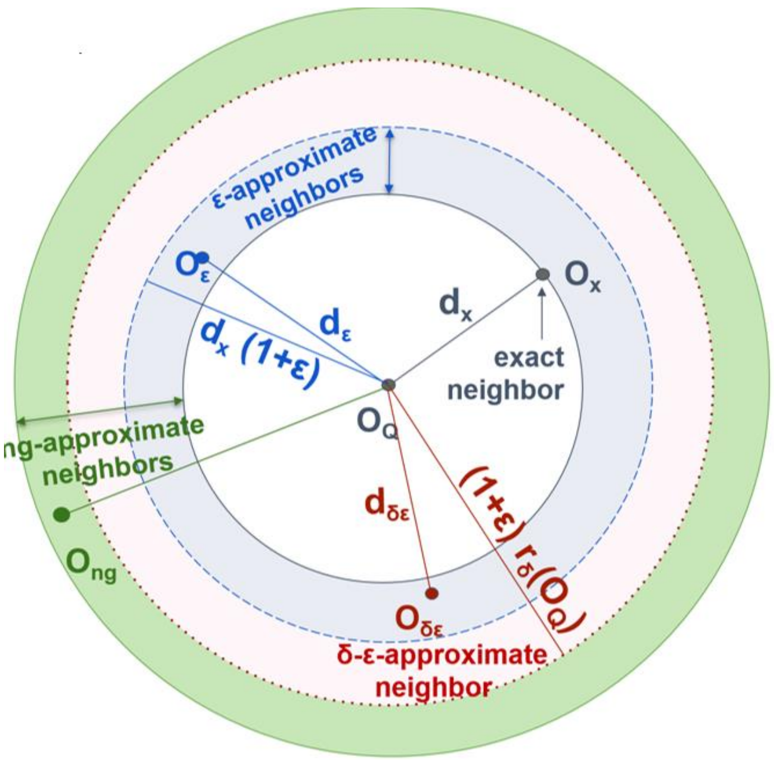


d_ε <= d_x (1+ε)

Result is within distance (1+ ε) of the exact answer

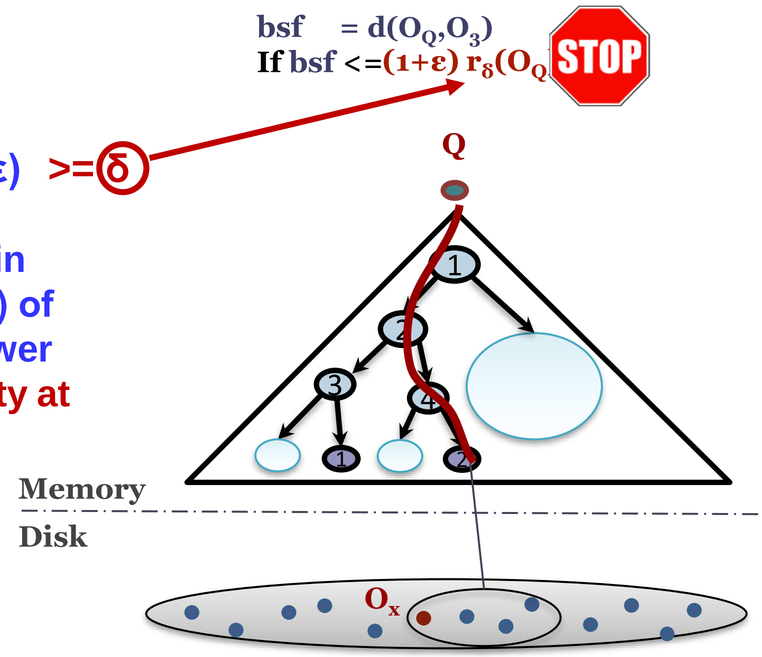


Extensions: Indexes



$P\{d_\varepsilon <= d_x (1+\varepsilon) >= \delta$

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



Questions?

Data Series Similarity Search State-of-the-Art Methods

Publications

Echihabi-EDBT'21

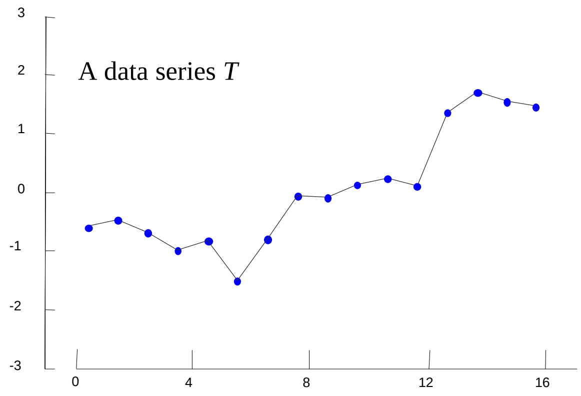
Data Series Similarity Search State-of-the-Art Methods

for a more complete and detailed presentation, see tutorial:

Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas. Big Sequence Management: Scaling Up and Out. EDBT 2021
http://helios.mi.parisdescartes.fr/~themisp/publications.html#tutorials

iSAX Summarization

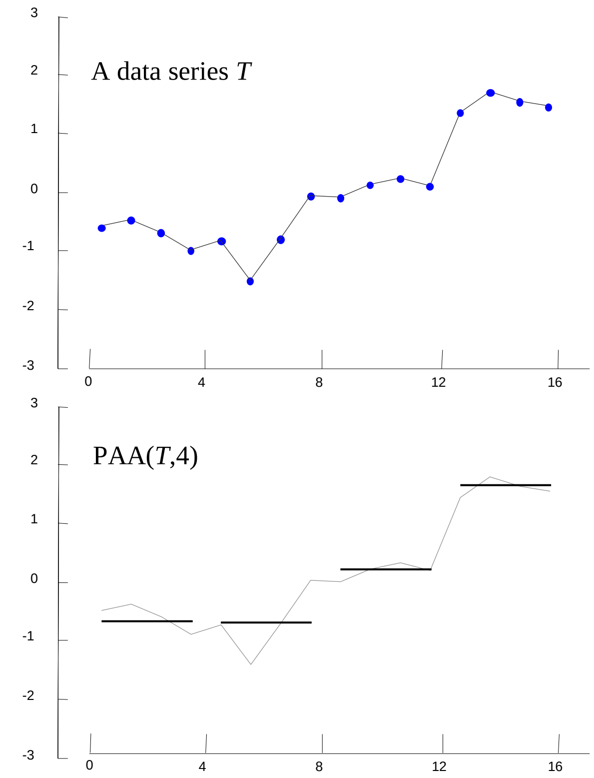
- **i**ndexable **S**ymbolic **A**ggregate appro**X**imation (SAX)
  - **(1)** Represent data series *T* of length ***n*** with ***w*** segments using Piecewise Aggregate Approximation (PAA)



iSAX Summarization

- **i**ndexable **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 $\mu = 0, \sigma = 1$
    - $\text{PAA}(T,w) = \bar{T} = \bar{t}_1, \ldots, \bar{t}_w$

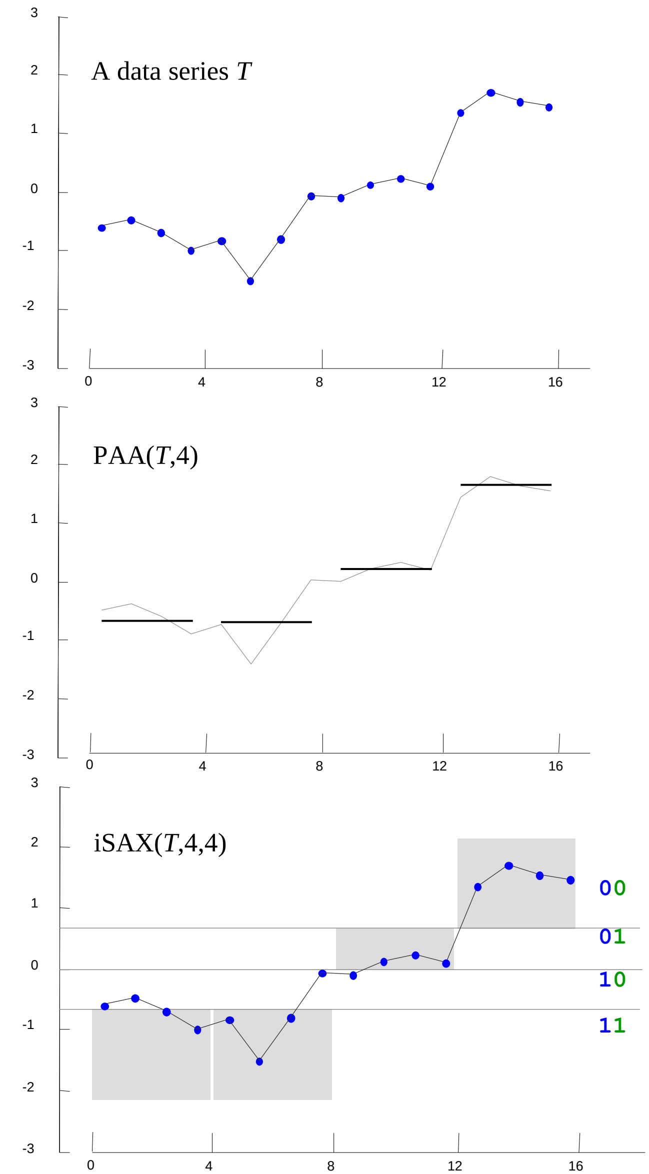
      where $\bar{t}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} T_j$



iSAX Summarization

- **i**ndexable **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 $\mu = 0, \sigma = 1$
    - $\text{PAA}(T,w) = \bar{T} = \bar{t}_1, \ldots, \bar{t}_w$

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

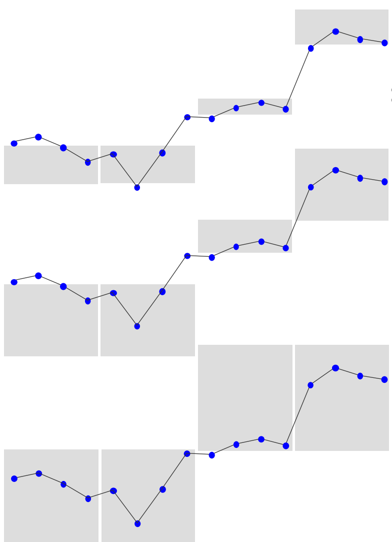


iSAX Summarization
iSAX Summarization

Publications

Shieh-KDD'08

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



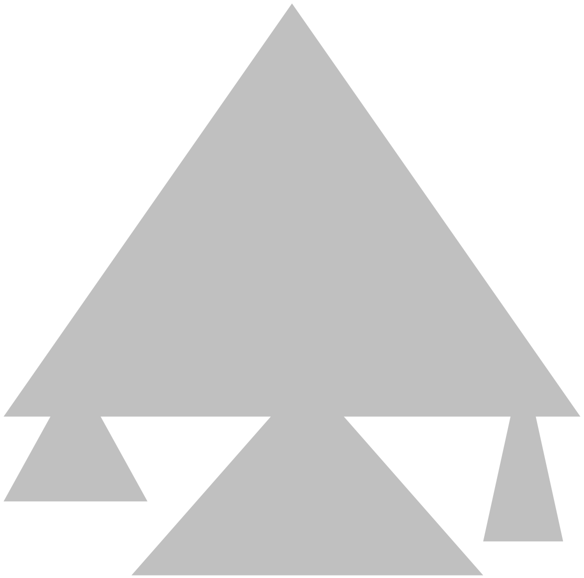
= { 6, 6, 3, 0} = {110 ,110 ,0111 ,000}

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

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

*i*SAX Index Family

- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***

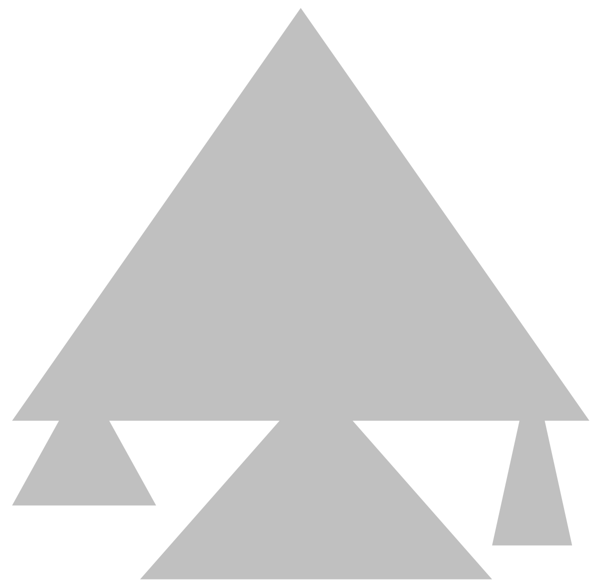


*i*SAX Index Family

- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***

e.g., th=4, w=4, b=1

$$\left[\begin{array}{cccc} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{array}\right.$$



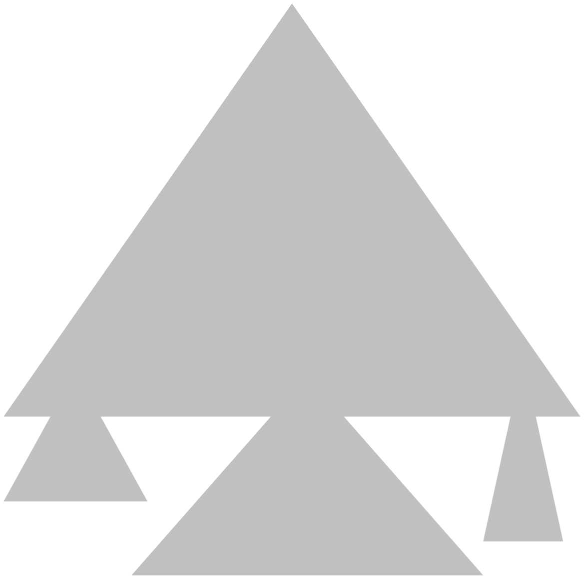
*i*SAX Index Family

- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***

e.g., th=4, w=4, b=1

Insert:
1 1 1 0

1 1 1 0
1 1 1 0
1 1 1 0
1 1 1 0



*i*SAX Index Family

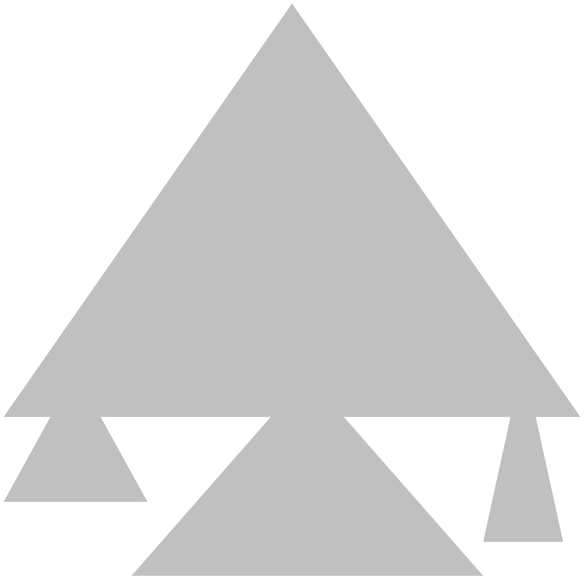
- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***

e.g., th=4, w=4, b=1

1 1 **1** 0

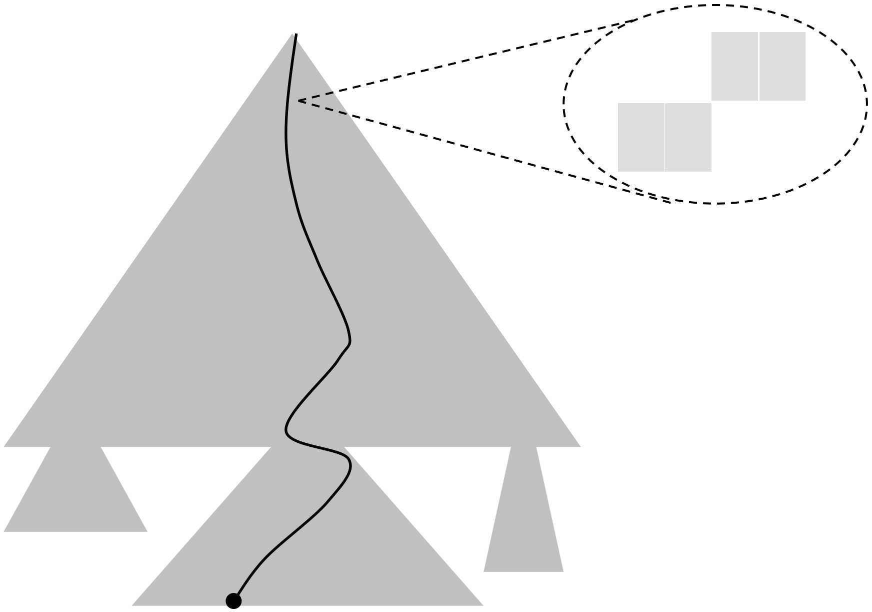
1 1 1**0** 0
1 1 1**0** 0

1 1 1**1** 0
1 1 1**1** 0
1 1 1**1** 0



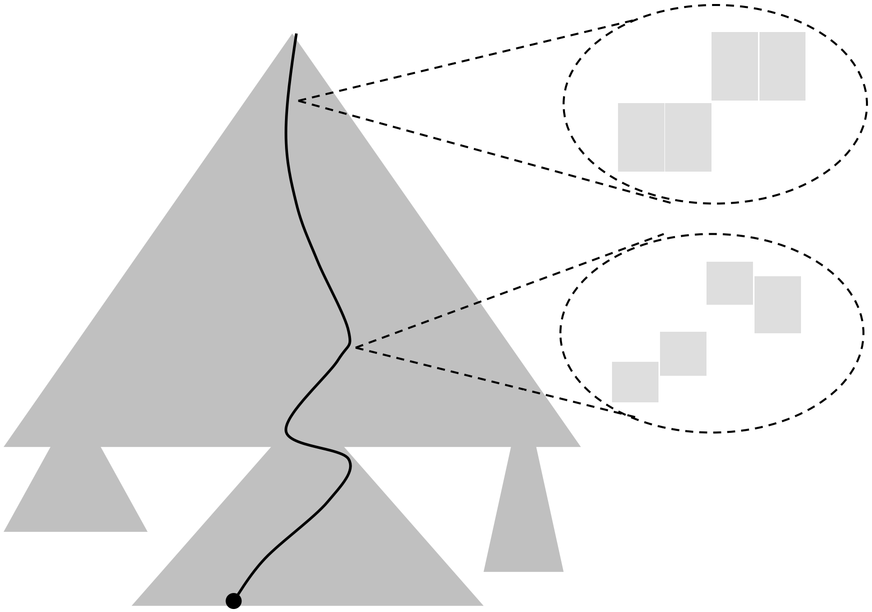
*i*SAX Index Family

- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***



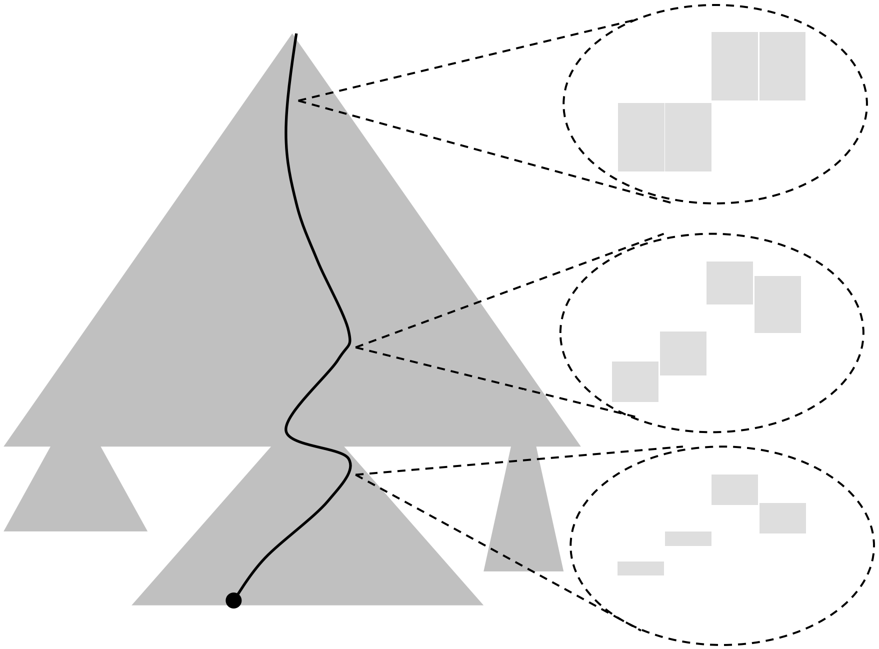
*i*SAX Index Family

- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***



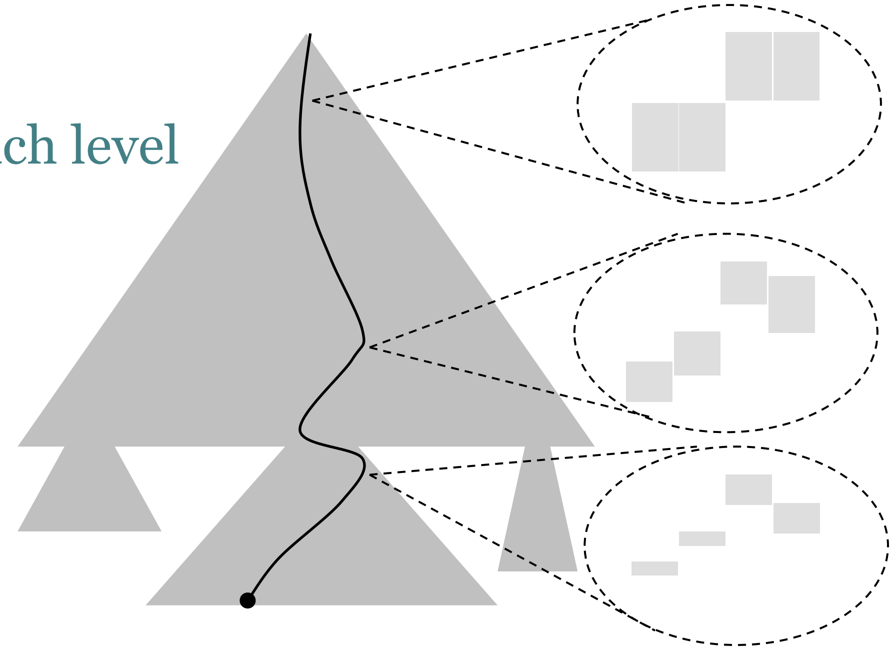
*i*SAX Index Family

- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***



*i*SAX Index Family

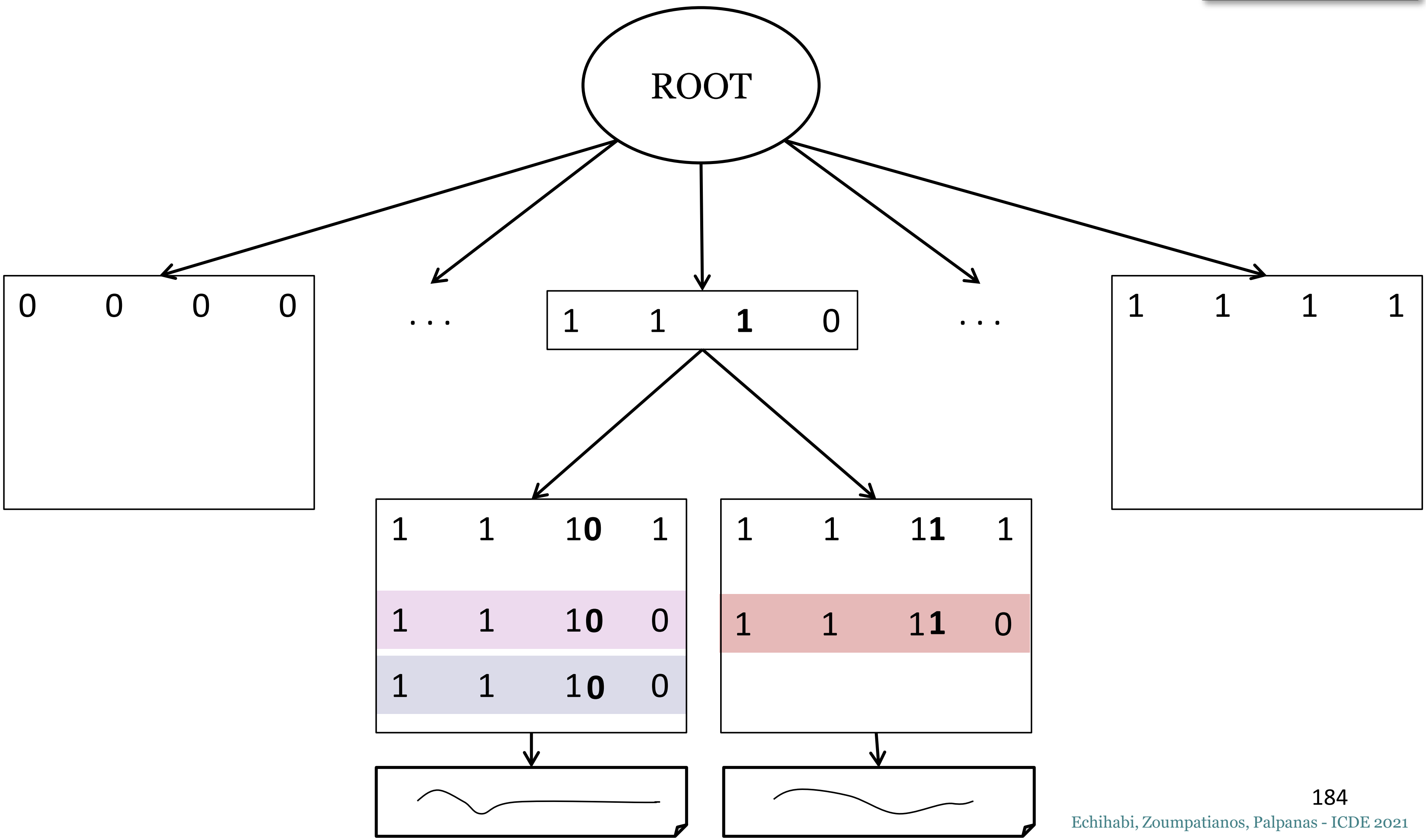
- non-balanced tree-based index with non-overlapping regions, and controlled fan-out rate
  - base cardinality ***b*** (optional), segments ***w***, threshold ***th***
  - hierarchically subdivides SAX space until num. entries ≤ ***th***
- Approximate Search
  - Match *i*SAX representation at each level
- Exact Search
  - Leverage approximate search
  - Prune search space
    - Lower bounding distance



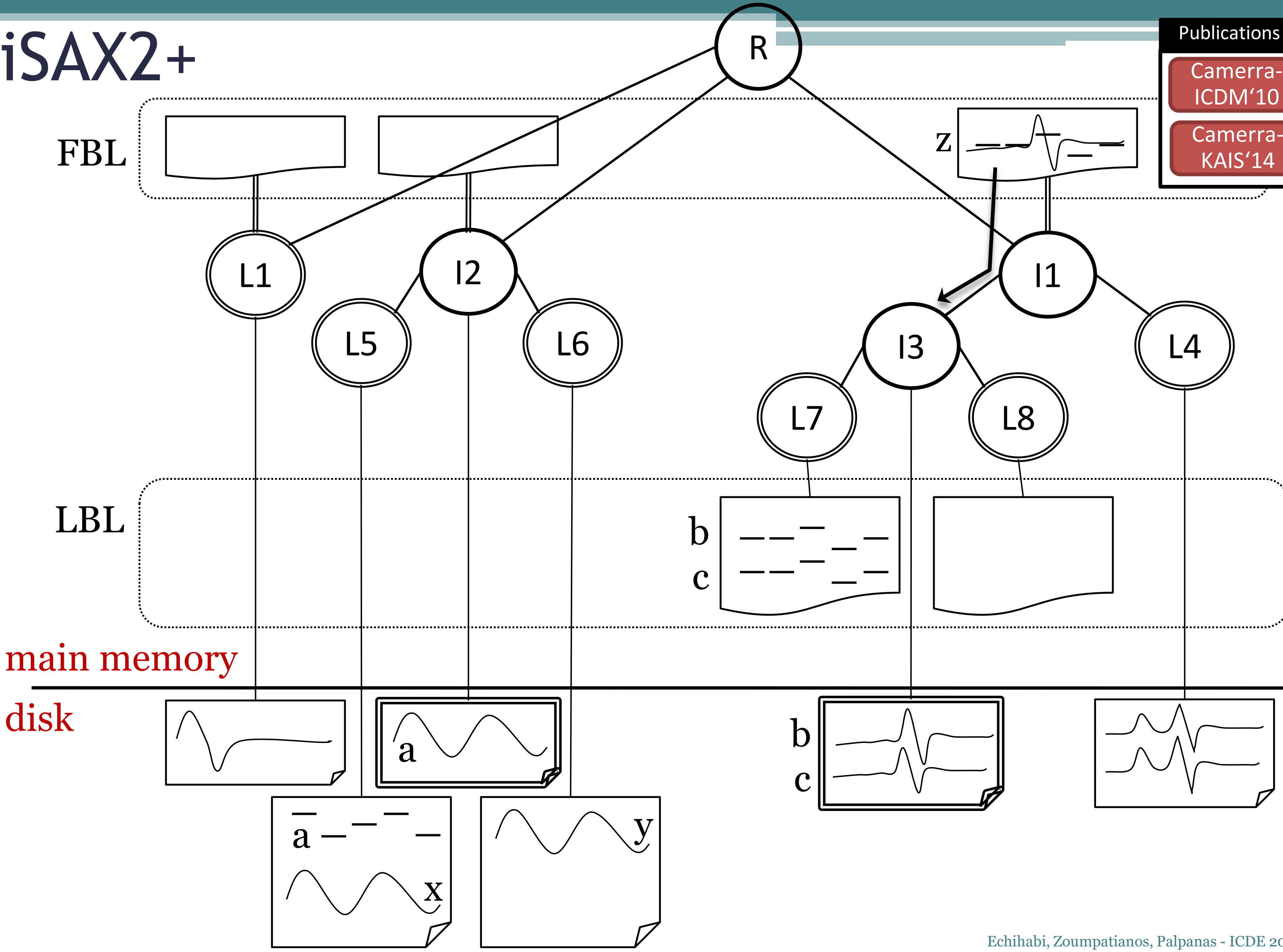
iSAX

Publications

Shieh-KDD'08



iSAX2+



Publications

Camerra-ICDM'10

Camerra-KAIS'14

Publications

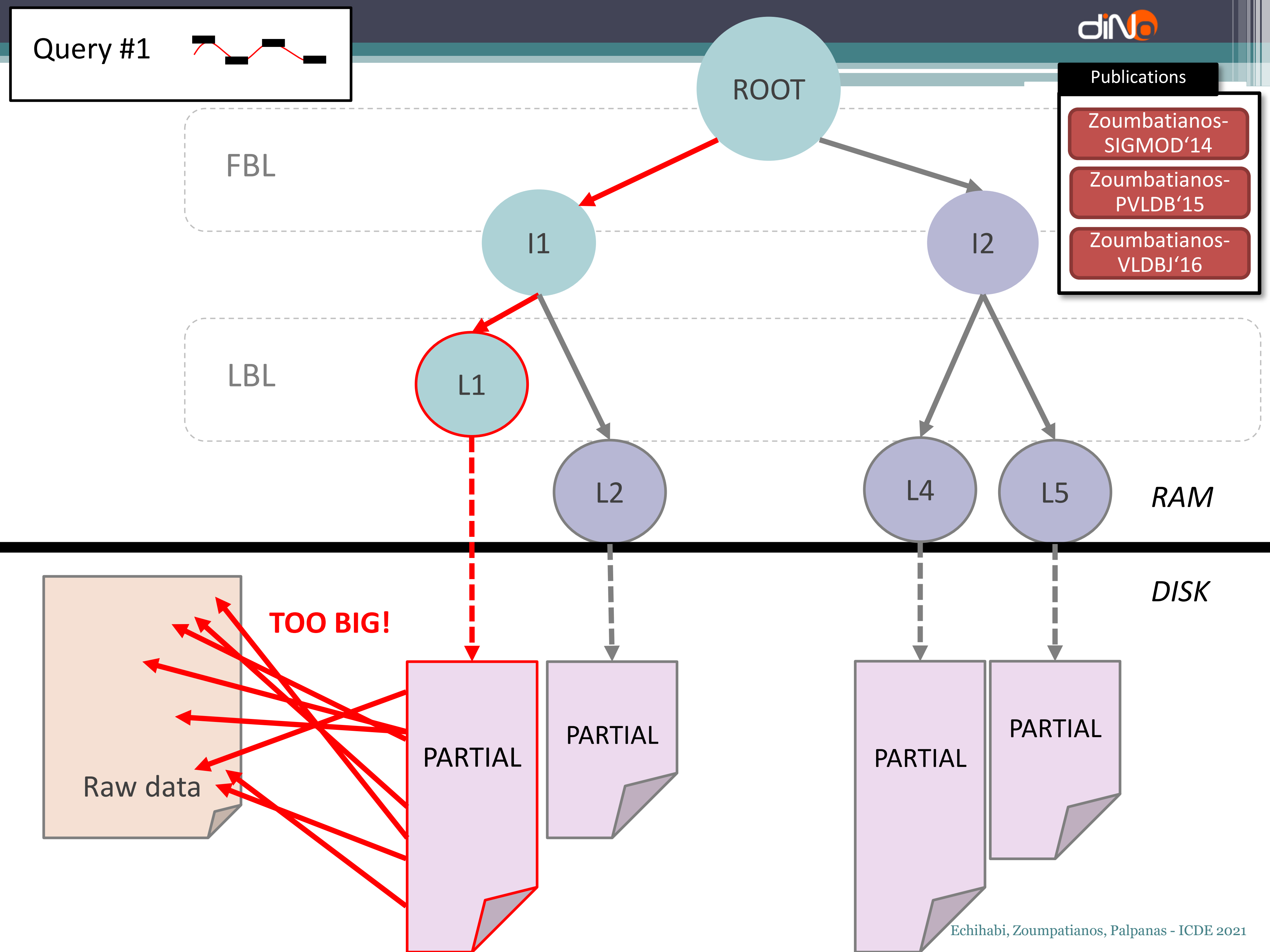
Zoumbatianos-SIGMOD'14

Zoumbatianos-PVLDB'15

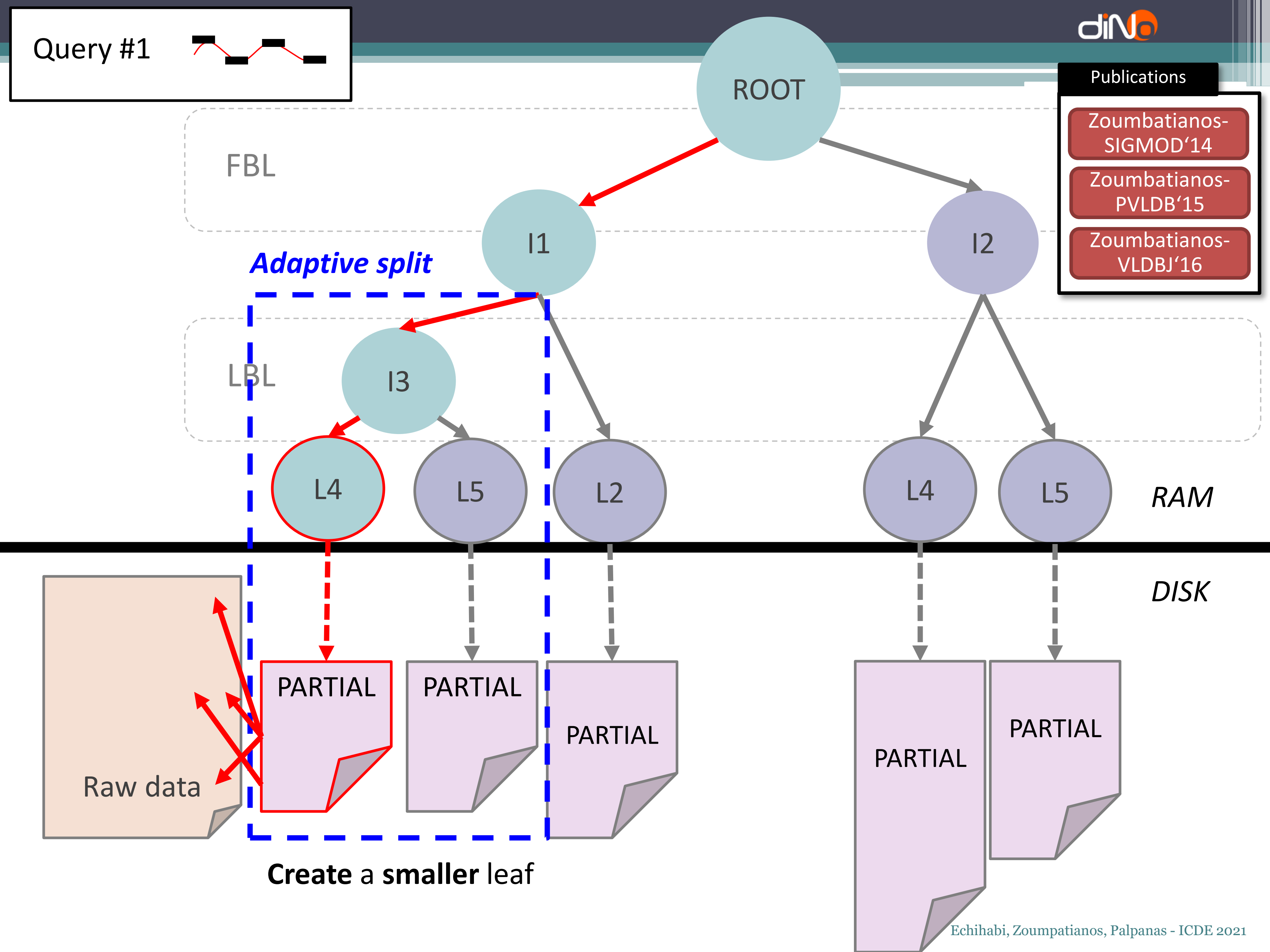
Zoumbatianos-VLDBJ'16

ADS+

- novel paradigm for building a data series index
  - does not build entire index and then answer queries
  - starts answering queries by building the part of the index needed by those queries
- still guarantees correct answers
- intuition for proposed solution
  - builds index using only *i*SAX summaries; uses large leaf size
  - postpones 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)

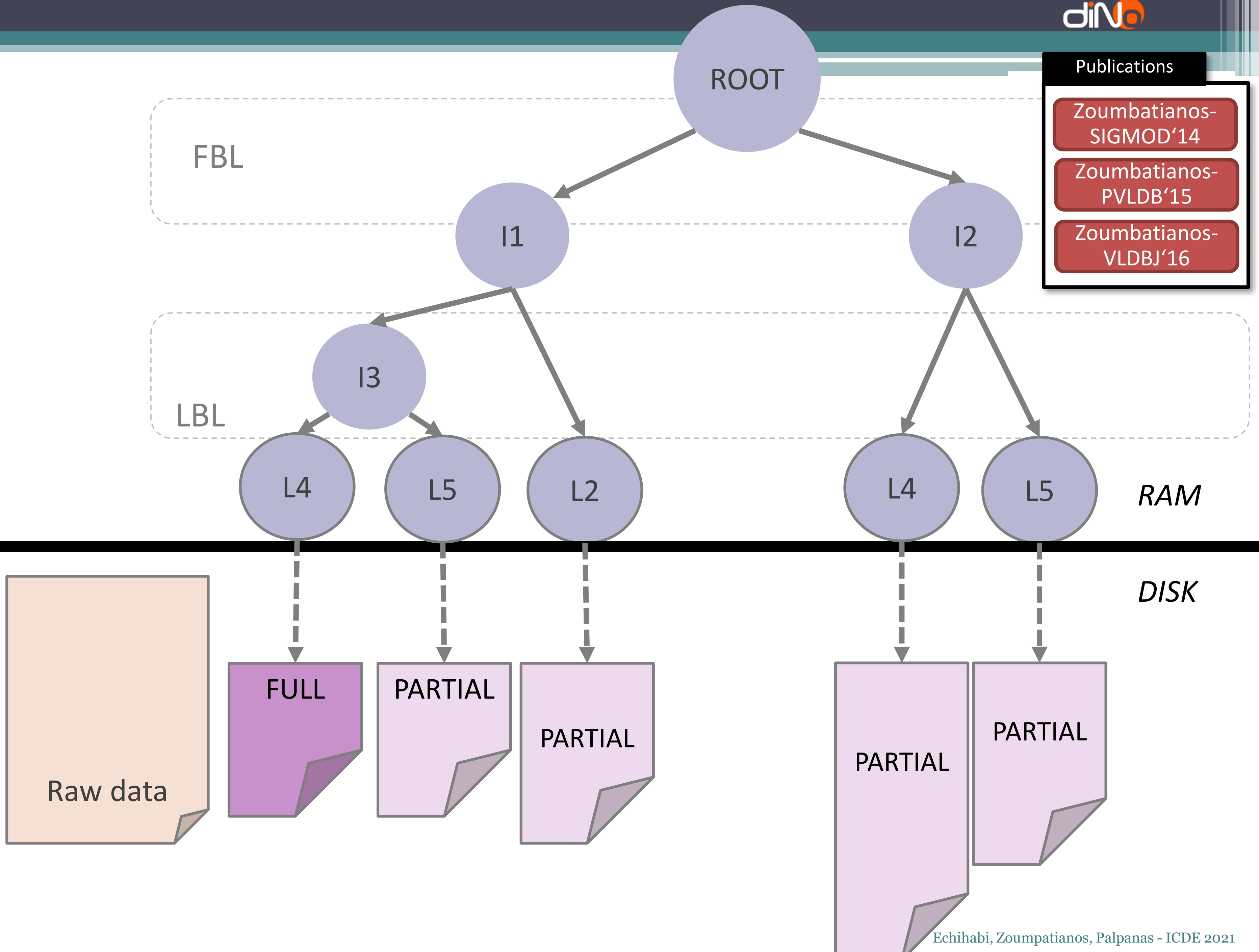


Query #1
Publications
Zoumbatianos-SIGMOD'14
Zoumbatianos-PVLDB'15
Zoumbatianos-VLDBJ'16
ROOT
FBL
I1
I2
LBL
L1
L2
L4
L5
RAM
DISK
TOO BIG!
Raw data
PARTIAL
PARTIAL
PARTIAL
PARTIAL

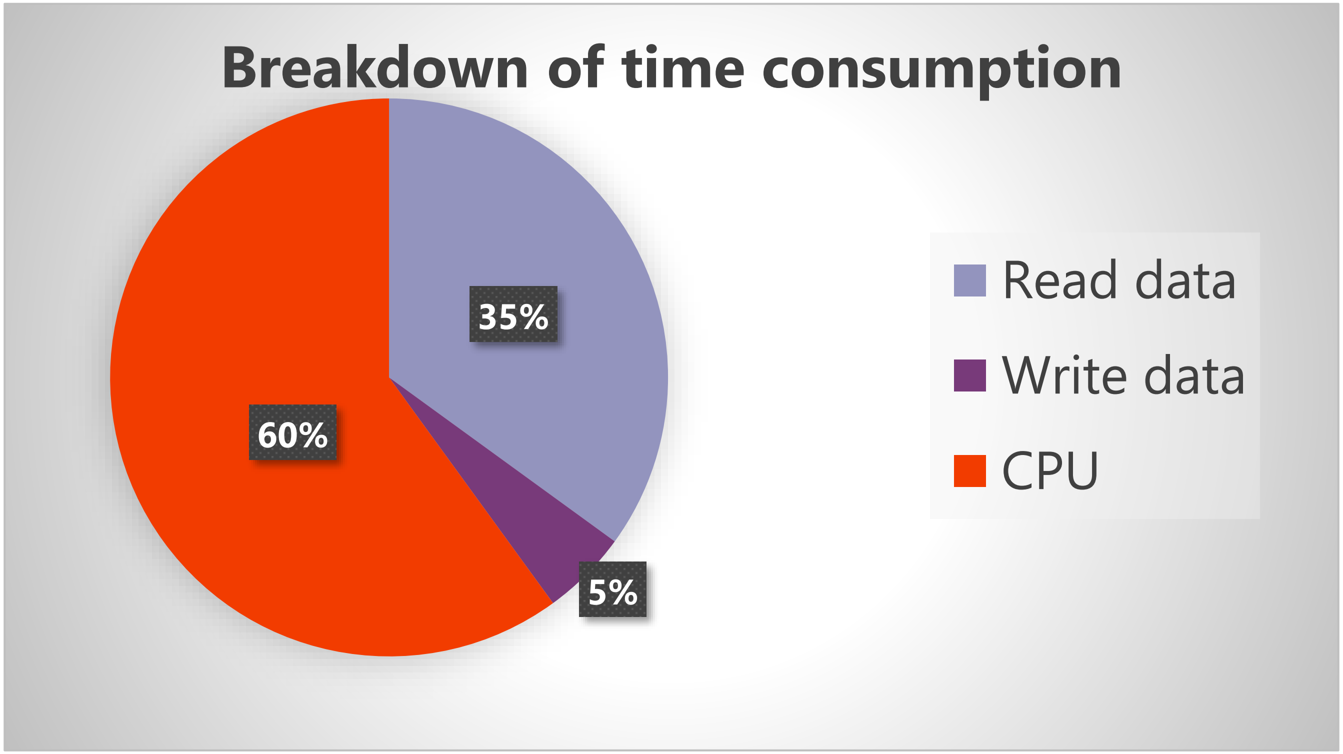

Query #1
Publications
Zoumbatianos-SIGMOD‘14
Zoumbatianos-PVLDB‘15
Zoumbatianos-VLDBJ‘16
ROOT
FBL
I1
I2
Adaptive split
LBL
I3
L4
L5
L2
L4
L5
RAM
DISK
Raw data
PARTIAL
PARTIAL
PARTIAL
PARTIAL
PARTIAL
Create a **smaller** leaf

Publications

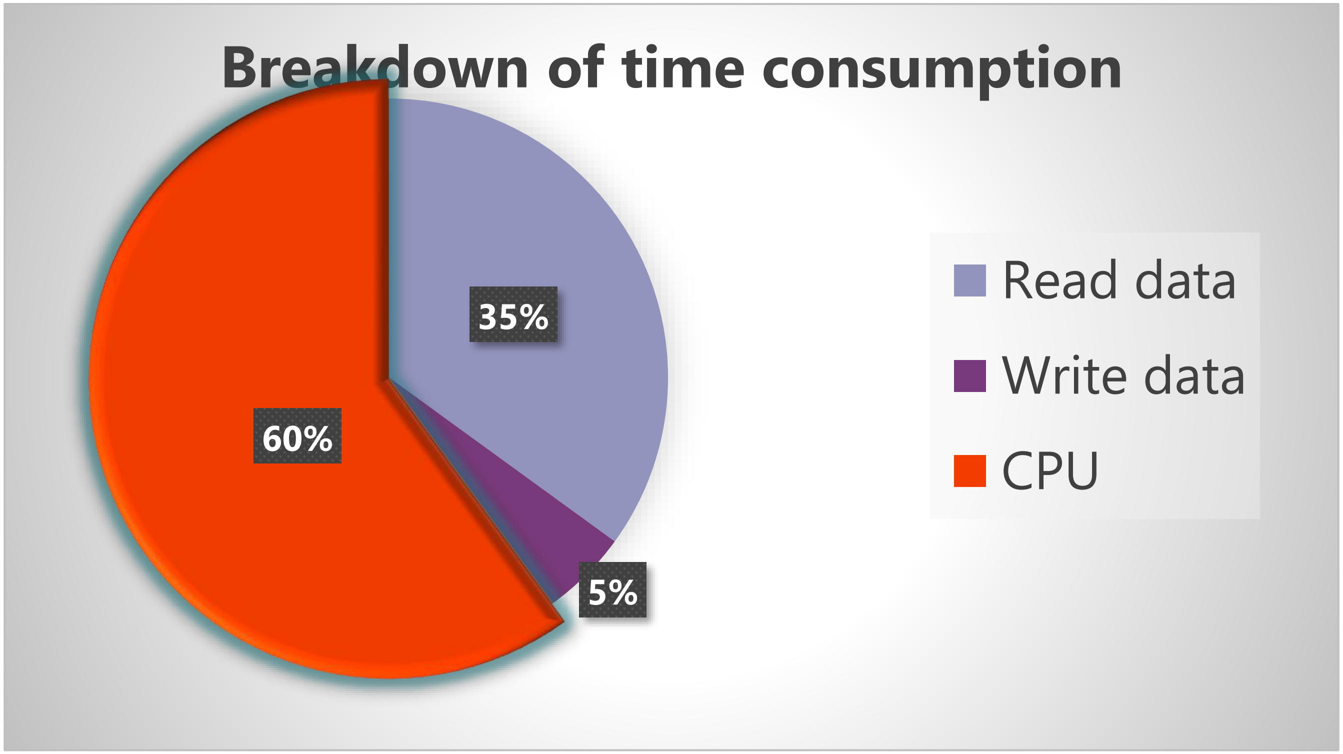
- Zoumbatianos-SIGMOD‘14
- Zoumbatianos-PVLDB‘15
- Zoumbatianos-VLDBJ‘16



ADS Index creation



ADS Index creation



~60% of time spent in CPU: potential for improvement!

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20

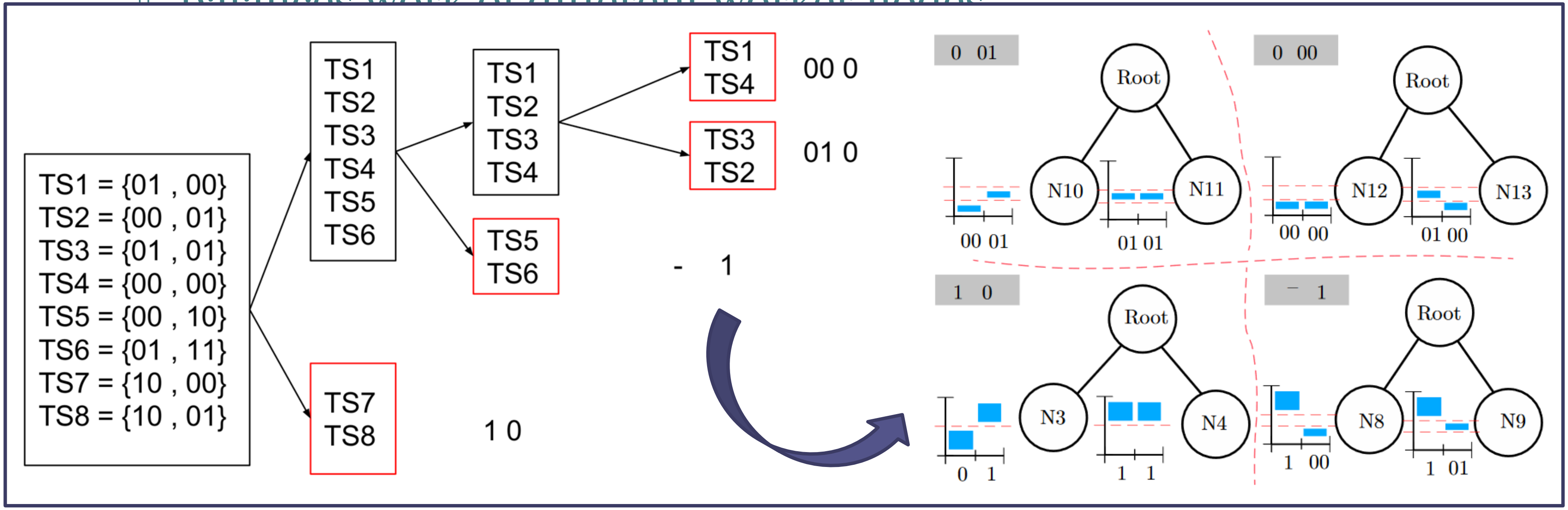
- DPiSAX: current solution for distributed processing (Spark)
  - balances work of different worker nodes

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20

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



Parallelization/Distribution

Publications

Yagoubi-ICDM'17

Yagoubi-TKDE'18

Lavchenko-KAIS'20

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

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21

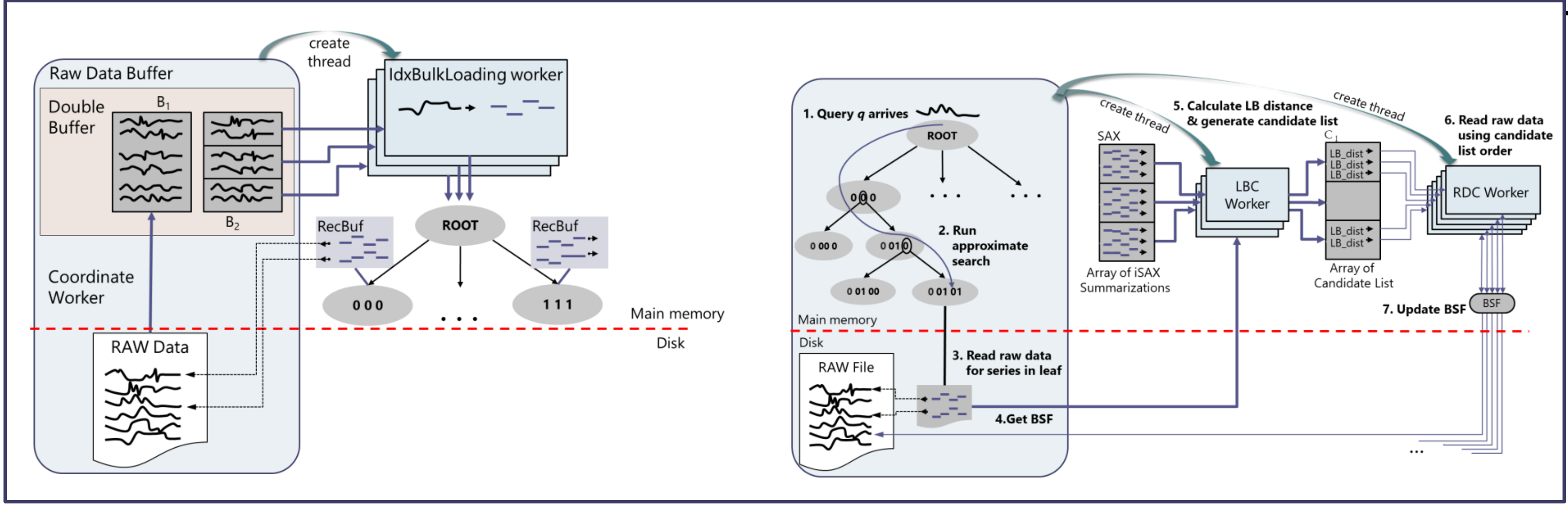
- DPiSAX: current solution for distributed processing (S
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized solution
- ParIS+: current solution for modern hardware
  - completely masks out the CPU cost

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21

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



Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21

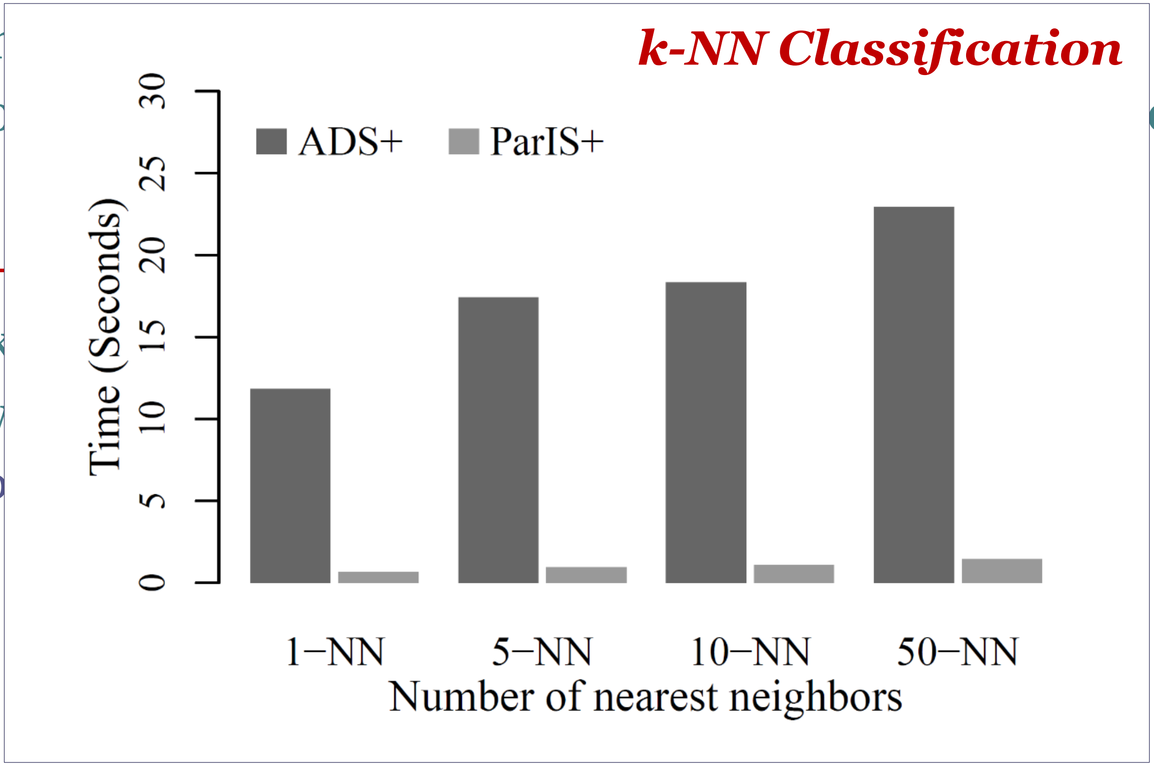
- DPiSAX: current solution for distributed processing (S
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized solution
- ParIS+: current solution for modern hardware
  - masks out the CPU cost
  - answers exact queries in the order of a few secs
    -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

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21

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



Publications

Yagoubi-ICDM'17

Yagoubi-TKDE'18

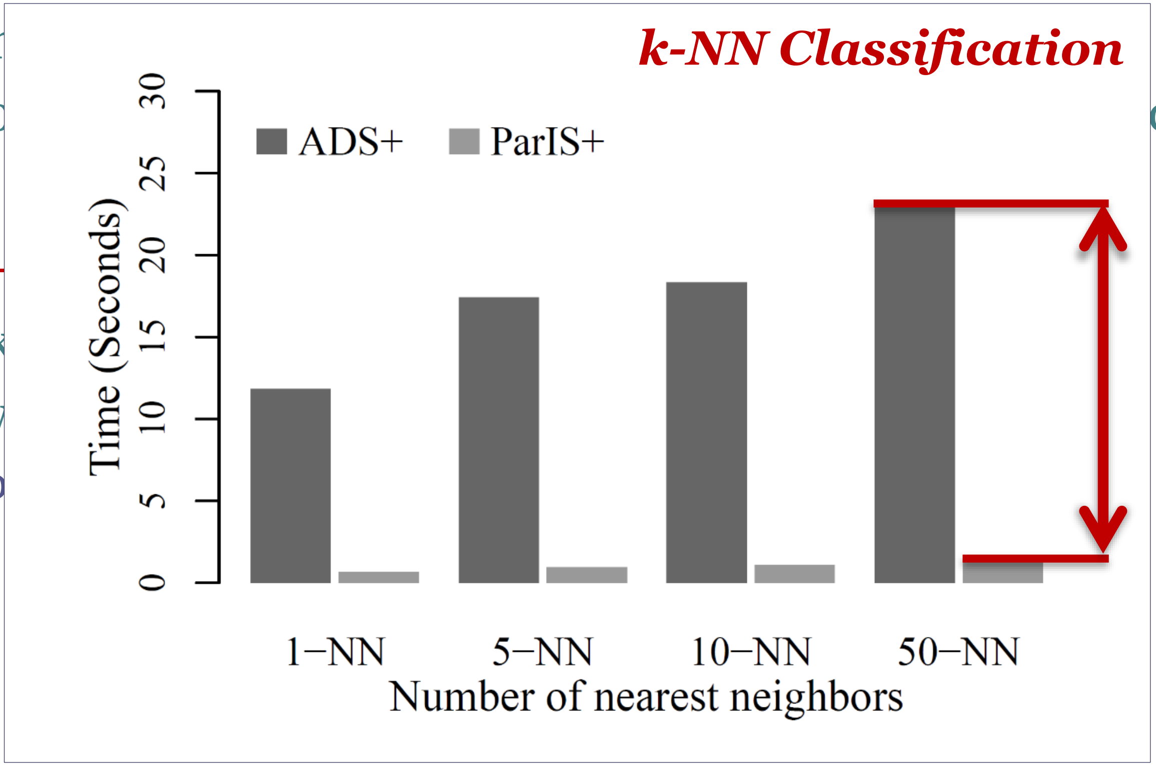
Lavchenko-KAIS'20

Peng-BigData'18

Peng-TKDE'21

Parallelization/Distribution

- DPiSAX: current solution for distributed processing (S
  - balar
  - perfo ... d solution
- ParIS+
  - mask
  - answ
    - 3 o



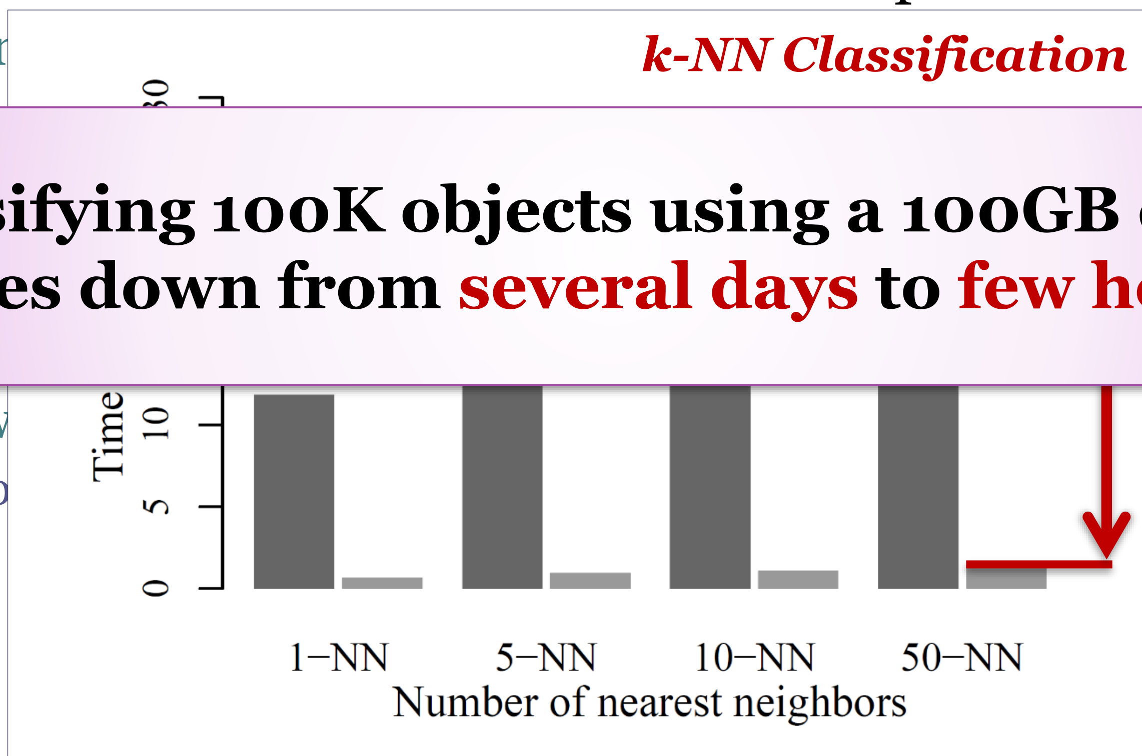
18x faster

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21

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



18x faster

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

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21
- Peng-ICDE'20
- Peng-VLDBJ'21

- DPiSAX: current solution for distributed processing (Spar[k)]
  - balances work of different worker nodes
  - performs 2 orders of magnitude faster than centralized soluti[ons]
- 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

Parallelization/Distribution

Publications

- Yagoubi-ICDM'17
- Yagoubi-TKDE'18
- Lavchenko-KAIS'20
- Peng-BigData'18
- Peng-TKDE'21
- Peng-ICDE'20
- Peng-VLDBJ'21
- Peng-ICDE'21

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

Extensions...

Publications

Kondylakis-PVLDB'18

Kondylakis-SIGMOD'19

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

Extensions...

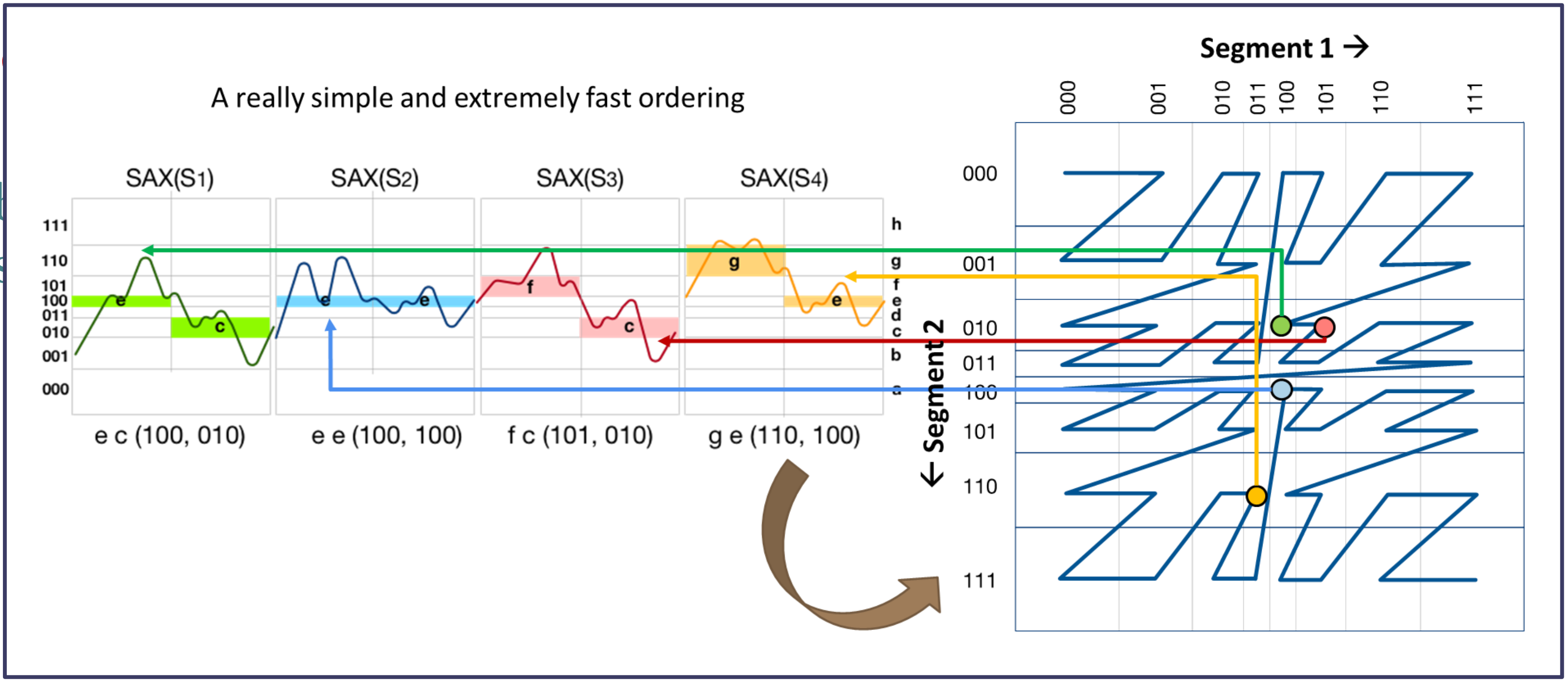
Publications

Kondylakis-PVLDB'18

Kondylakis-SIGMOD'19

• Co

▫ I
S



Publications

Kondylakis-PVLDB'18

Kondylakis-SIGMOD'19

Extensions…

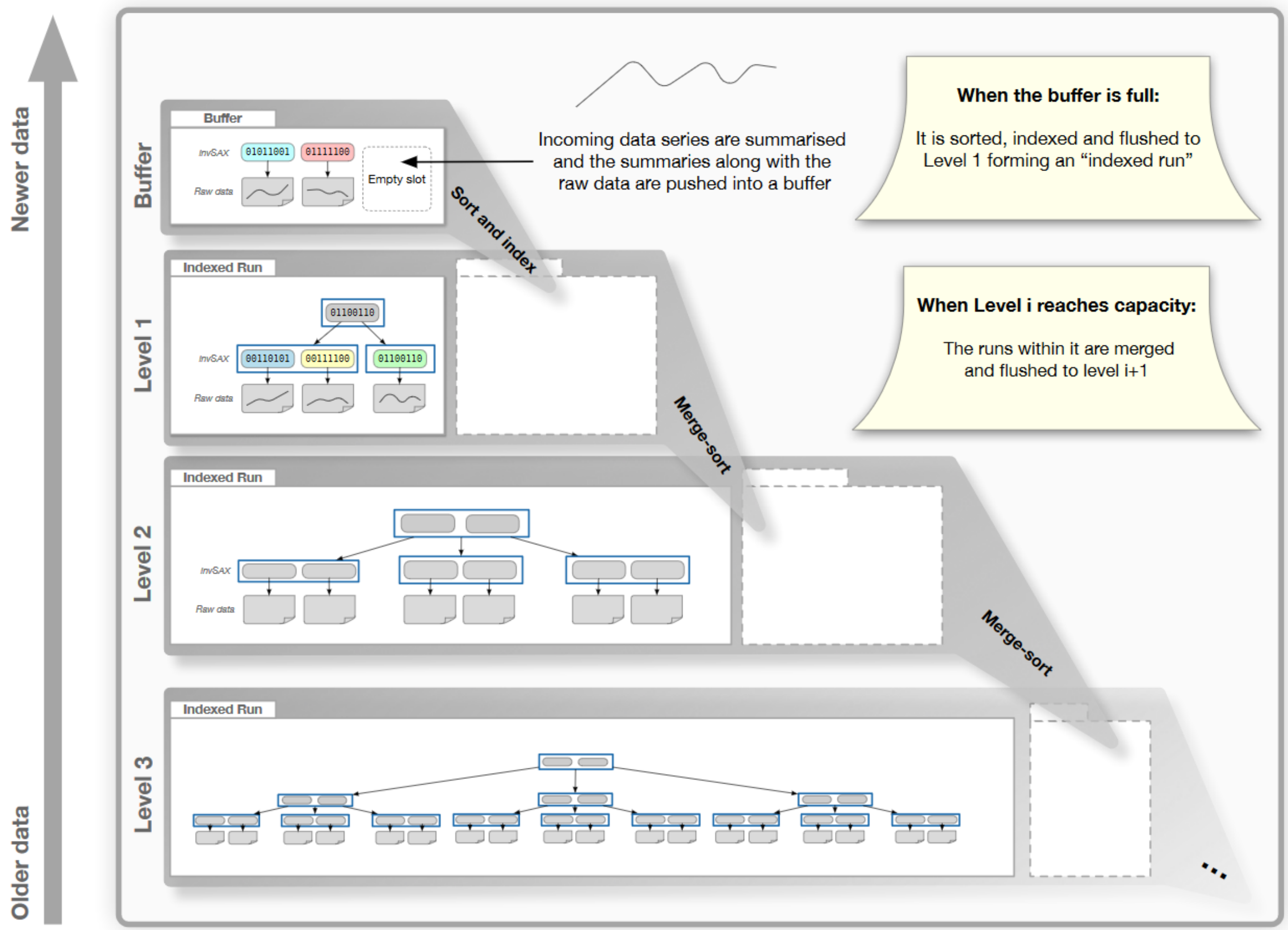
- Coconut: current solution for limited memory devices and streaming time series
  - bottom-up, succinct index construction based on sortable summarizations
  - outperforms state-of-the-art in terms of index space, index construction time, and query answering time

Coconut-LSM

Extensions...

Publications

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



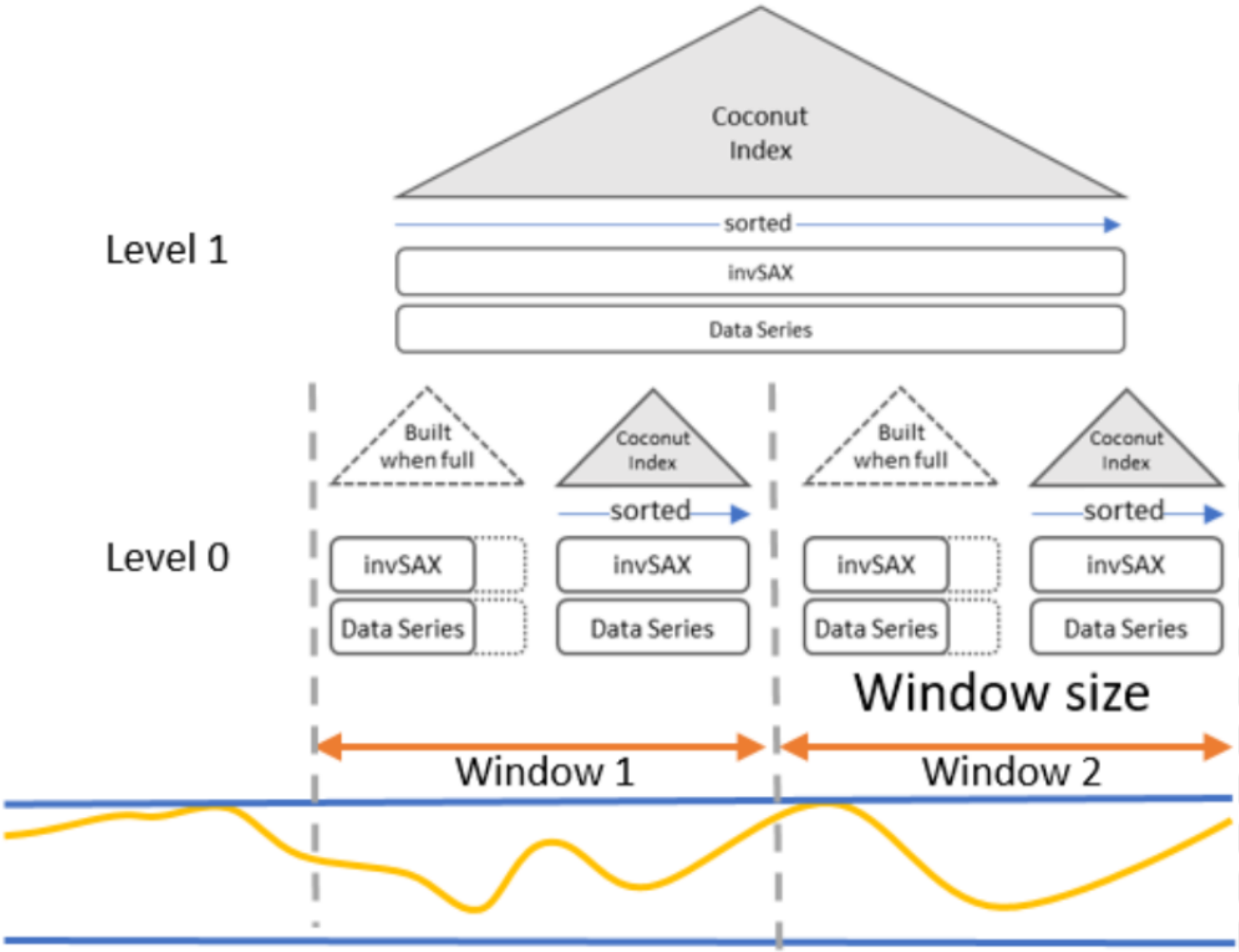
Extensions…

Publications

Kondylakis-PVLDB'18

Kondylakis-SIGMOD'19

Kondylakis-VLDBJ'20



Extensions…

Publications

- Kondylakis-PVLDB'18
- Kondylakis-SIGMOD'19
- Kondylakis-VLDBJ'20
- Linardi-ICDE'18
- Linardi-PVLDB'19
- Linardi-VLDBJ'20

- 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

Extensions…

Publications

Kondylakis-PVLDB‘18

Kondylakis-SIGMOD’19

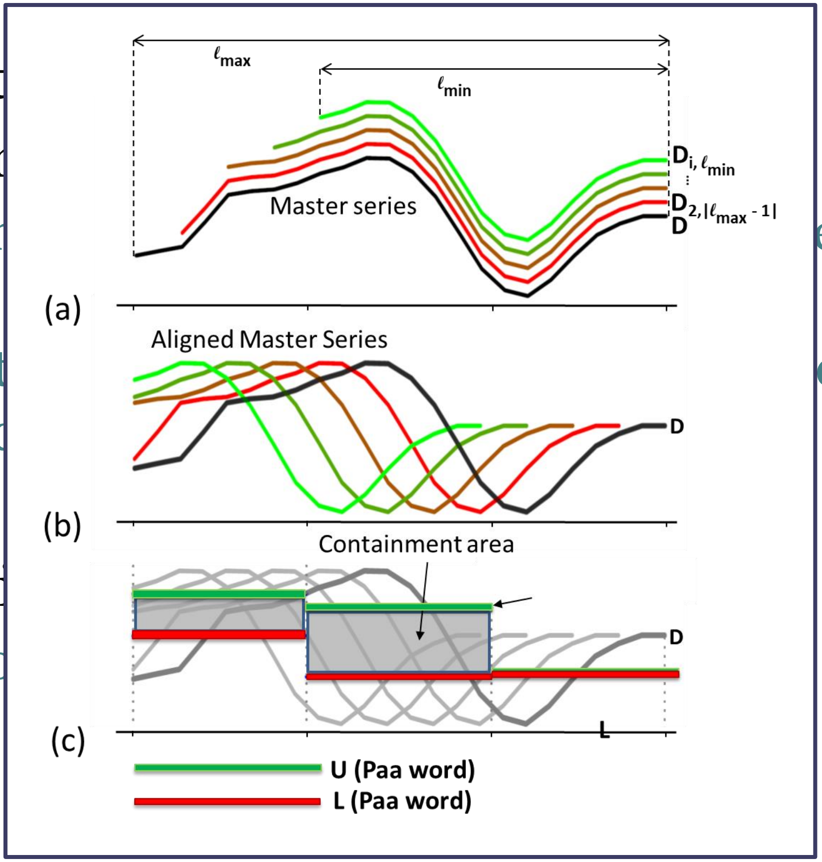
Kondylakis-VLDBJ’20

Linardi-ICDE’18

Linardi-PVLDB‘19

Linardi-VLDBJ‘20

- Coconut: current solut… and streamin…
  - bottom-up, succinct in… e summarizations
  - outperforms state-of-t… ex construction time, and…
- ULISSE: current soluti…
  - single-index support c…



Extensions...

Publications

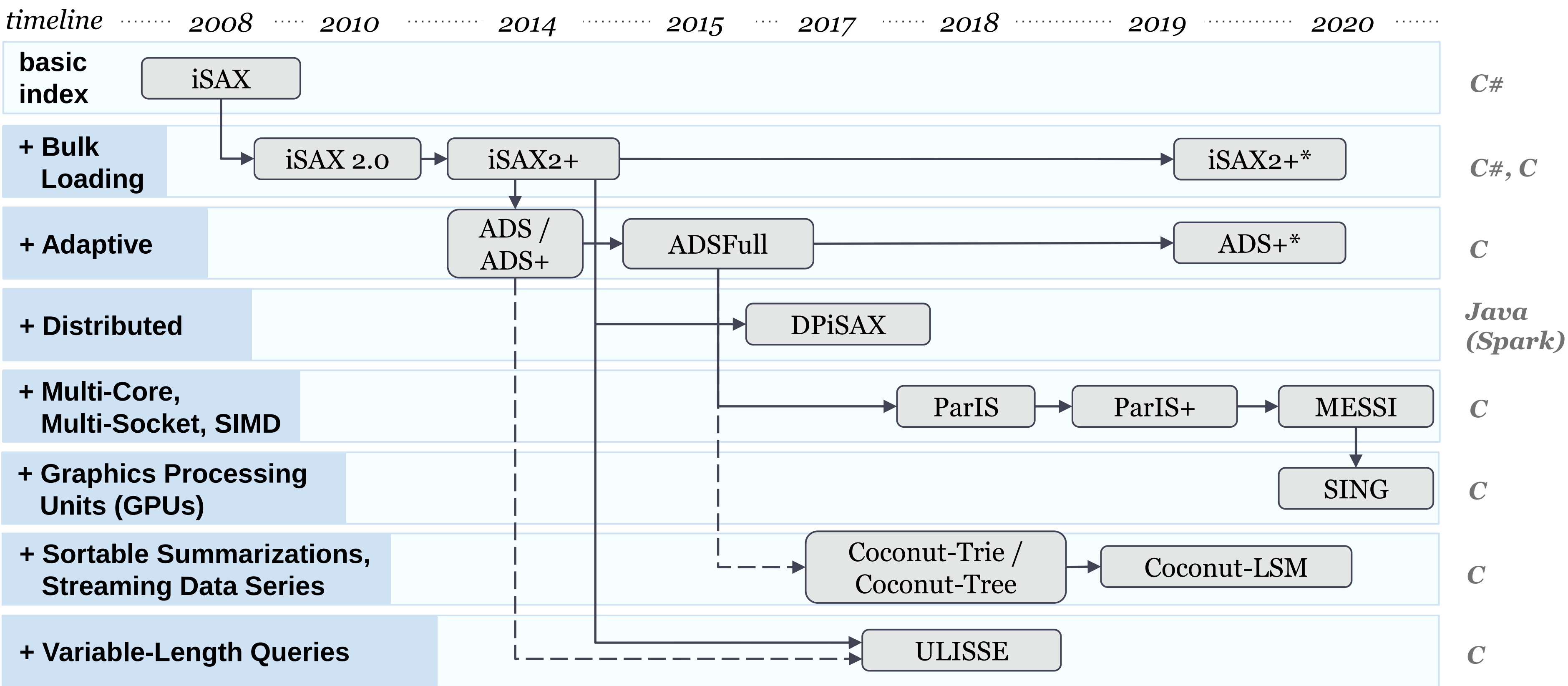
- Kondylakis-PVLDB'18
- Kondylakis-SIGMOD'19
- Kondylakis-VLDBJ'20
- Linardi-ICDE'18
- Linardi-PVLDB'19
- Linardi-VLDBJ'20

- 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

iSAX Index Family

Publications

Palpanas-ISIP'19

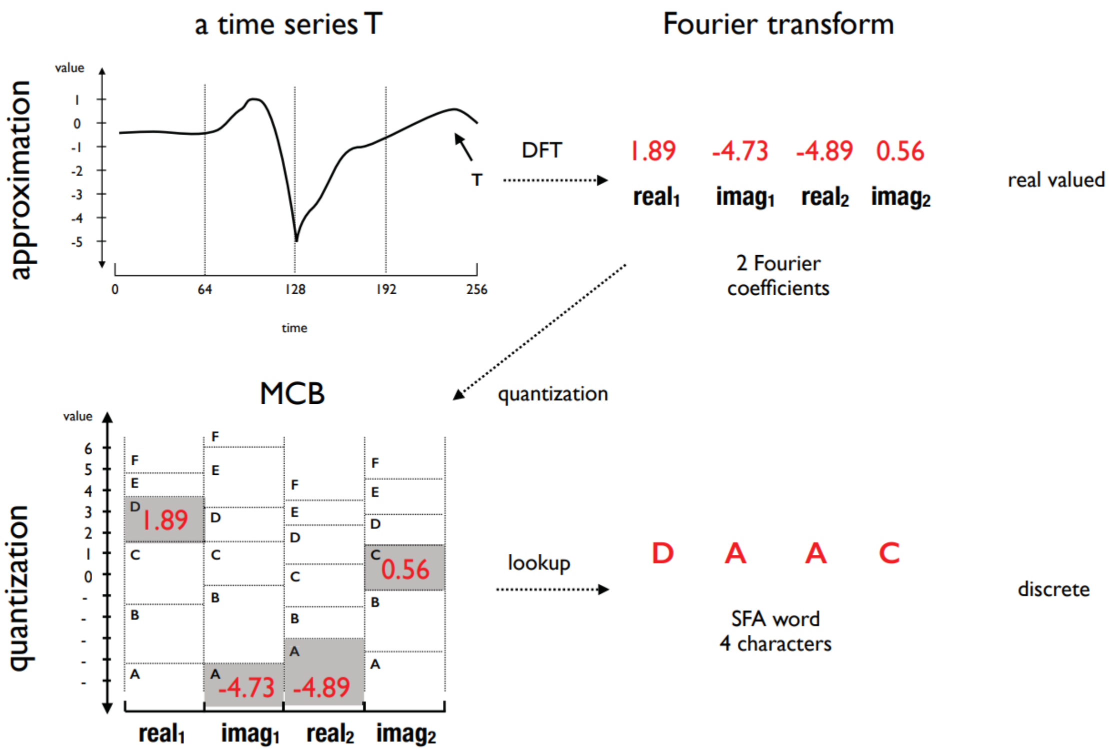


Timeline depicted on top; implementation languages marked on the right. Solid arrows denote inheritance of index design; dashed arrows denote inheritance of some of the design features; two new versions of iSAX2+/ADS+ marked with asterisk support approximate similarity search with deterministic and probabilistic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

Symbolic Fourier Approximation (SFA)
Summarization

Publications

Schafer-ICDE'12



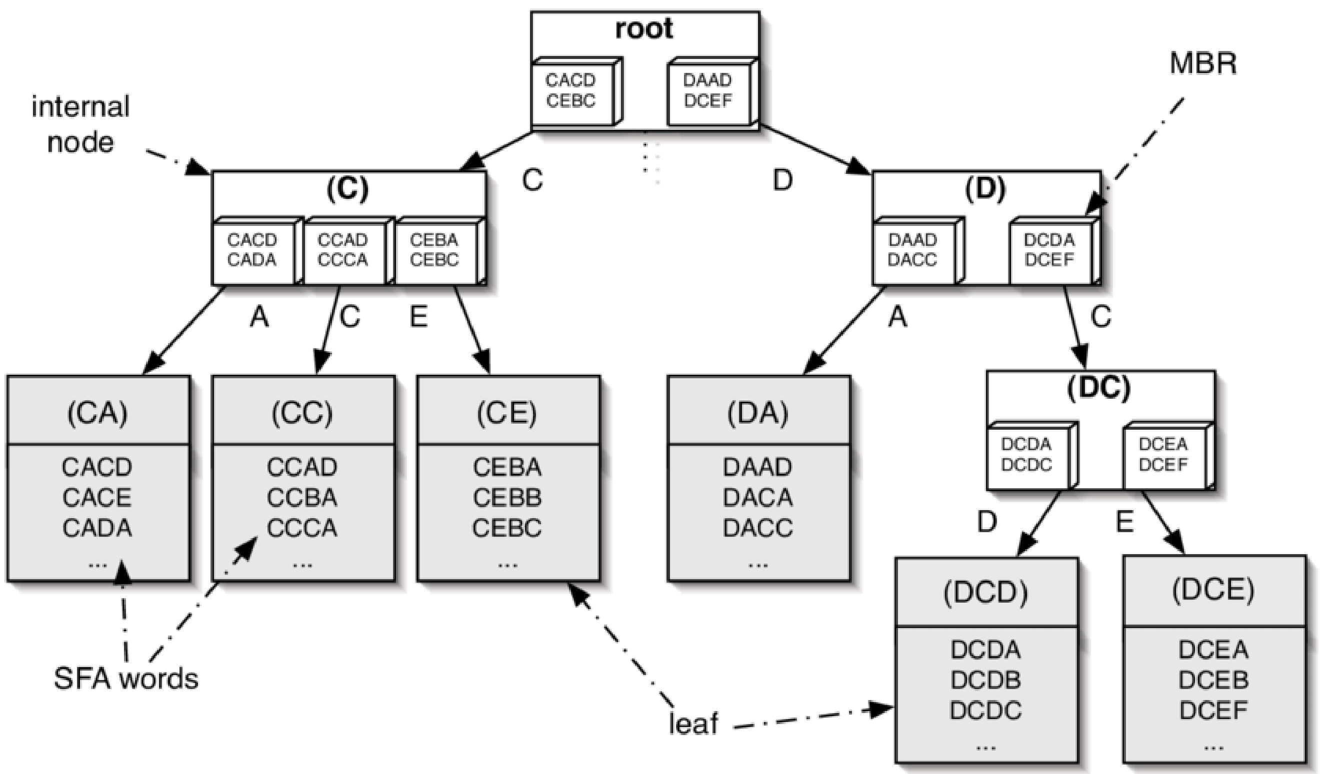
The SFA representation*

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

SFA
Indexing

Publications

Schafer-ICDE‘12

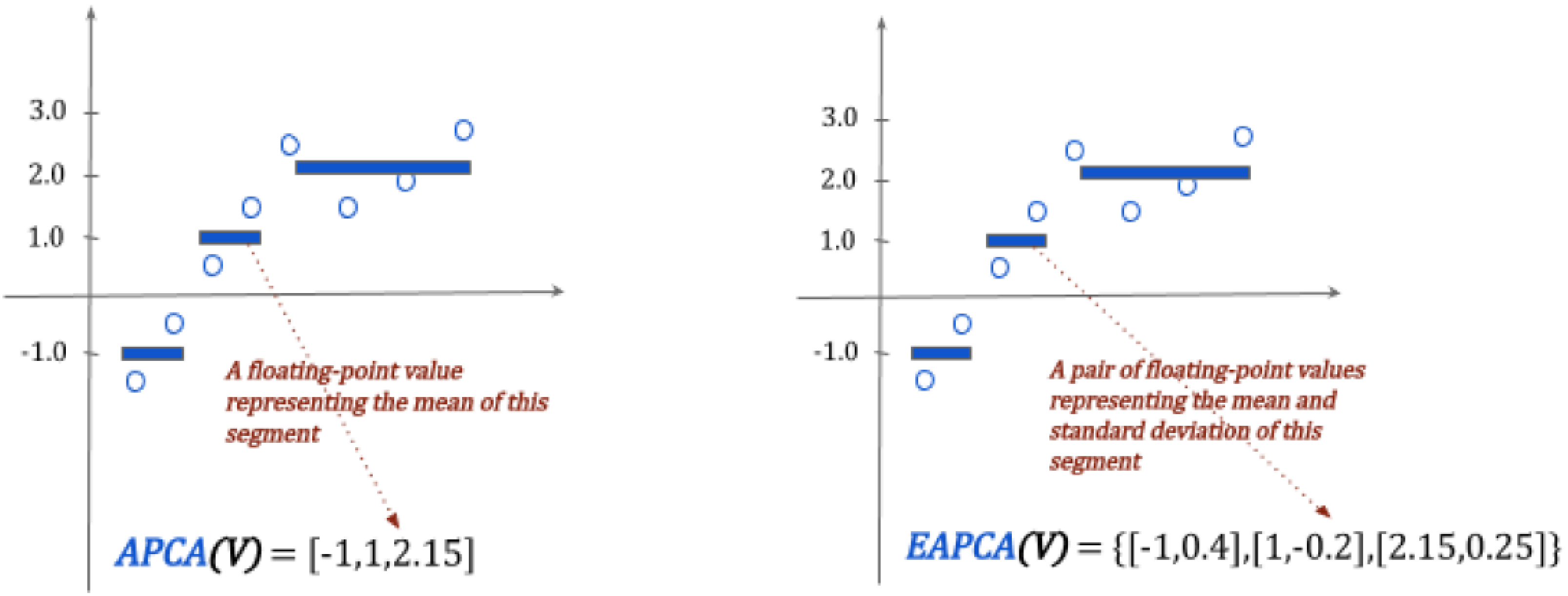


<u>The SFA Trie*</u>

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

DSTree
Summarization

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



(a) APCA

(b) EAPCA

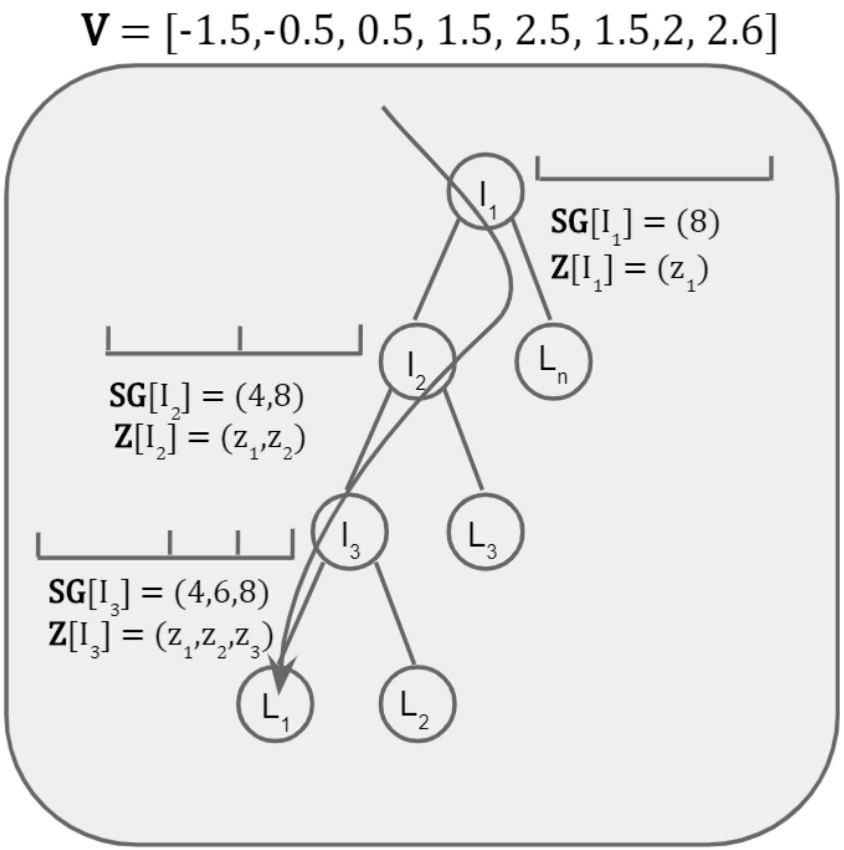
Intertwined with indexing

<u>The APCA and EAPCA representations</u>

DSTree
Indexing

Publications

Wang-PVLDB'13



Each node contains
- ❑ # vectors
- ❑ segmentation **SG**
- ❑ synopsis **Z**

Each Leaf node also :
- ❑ stores its raw vectors in a separate disk file

ParSketch

Publications

Cole et al. KDD'05

Yagoubi et al. DMKD'18

- solution for distributed processing (Spark)
  - represents data series using sketches
    - using a set of random vectors (Johnson-Lindenstrauss lemma)

ParSketch

Publications

Cole et al. KDD‘05

Yagoubi et al. DMKD‘18

- solution for distributed processing (Spark)
  - represents data series using sketches
    - using a set of random vectors (Johnson-Lindenstrauss lemma)

$$x = (x_1, x_2, x_3, \dots x_n)$$
$$y = (y_1, y_2, y_3, \dots y_n)$$
$$z = (z_1, z_2, z_3, \dots z_n)$$

$$R1 = (r_1 1, r_1 2, r_1 3, \dots r_1 w)$$
$$R2 = (r_2 1, r_2 2, r_2 3, \dots r_2 w)$$
$$R3 = (r_3 1, r_3 2, r_3 3, \dots r_3 w)$$
$$R4 = (r_4 1, r_4 2, r_4 3, \dots r_4 w)$$

$$(xsk1, xsk2, xsk3, xsk4)$$
$$(ysk1, ysk2, ysk3, ysk4)$$
$$(zsk1, zsk2, zsk3, zsk4)$$

ParSketch

Publications

Cole et al. KDD'05

Yagoubi et al. DMKD'18

- solution for distributed processing (Spark)
  - represents data series using sketches
    - using a set of random vectors (Johnson-Lindenstrauss lemma)

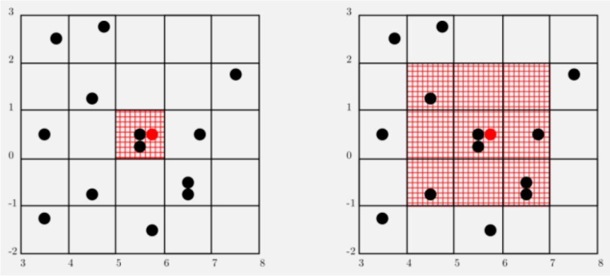
$$x = (x_1, x_2, x_3, \dots x_n)$$
$$y = (y_1, y_2, y_3, \dots y_n)$$
$$z = (z_1, z_2, z_3, \dots z_n)$$

$$R1 = (r_1 1, r_1 2, r_1 3, \dots r_1 w)$$
$$R2 = (r_2 1, r_2 2, r_2 3, \dots r_2 w)$$
$$R3 = (r_3 1, r_3 2, r_3 3, \dots r_3 w)$$
$$R4 = (r_4 1, r_4 2, r_4 3, \dots r_4 w)$$

$$(xsk1, xsk2, xsk3, xsk4)$$
$$(ysk1, ysk2, ysk3, ysk4)$$
$$(zsk1, zsk2, zsk3, zsk4)$$

  - define groups of dimensions in sketches
  - store the values of each group in a grid (in parallel)
    - each grid is kept by a node

node 1



node 2

ParSketch

Publications

Cole et al. KDD'05

Yagoubi et al. DMKD'18

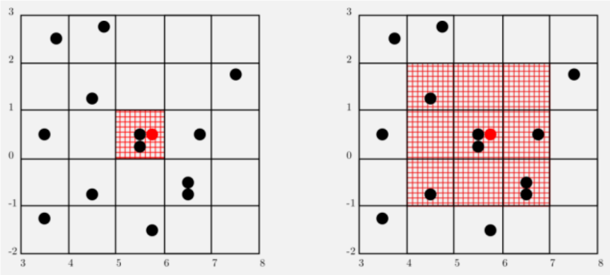
- solution for distributed processing (Spark)
  - represents data series using sketches
    - using a set of random vectors (Johnson-Lindenstrauss lemma)

$$x = (x_1, x_2, x_3, \dots x_n) \quad R1 = (r_1 1, r_1 2, r_1 3, \dots r_1 w)$$
$$y = (y_1, y_2, y_3, \dots y_n) \quad R2 = (r_2 1, r_2 2, r_2 3, \dots r_2 w)$$
$$z = (z_1, z_2, z_3, \dots z_n) \quad R3 = (r_3 1, r_3 2, r_3 3, \dots r_3 w)$$
$$R4 = (r_4 1, r_4 2, r_4 3, \dots r_4 w)$$

$$\Rightarrow \quad (xsk1, xsk2, xsk3, xsk4) \quad (ysk1, ysk2, ysk3, ysk4) \quad (zsk1, zsk2, zsk3, zsk4)$$

  - define groups of dimensions in sketches
  - store the values of each group in a grid (in parallel)
    - each grid is kept by a node

node 1



node 2

  - for ng-approximate query answering (originally proposed for ε-range queries)
    - find in the grids time series that are close to the query
    - finally, check the real similarity of candidates to find the results
  - performs well for high-frequency series

Publications

Zhang-ICDE'19

Wu-ICDE'19

Feng-IEEE Access'20

- other techniques, not covered here:
  - TARDIS
  - KV-Match
  - L-Match

Publications

- Zhang-ICDE'19
- Wu-ICDE'19
- Feng-IEEE Access'20
- Echihabi-EDBT'21

- other techniques, not covered here:
  - TARDIS
  - KV-Match
  - L-Match

- for a more complete and detailed presentation, see tutorial:
  - *Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas. Big Sequence Management: Scaling Up and Out. EDBT 2021*

Questions?

High-d Vector Similarity Search State-of-the-Art Methods

High-d Vector Similarity Search Methods

- Tree-Based Methods
- Hash-Based Methods
- Quantization-Based Methods
- Graph-Based Methods

High-d Vector Similarity Search State-of-the-Art Methods

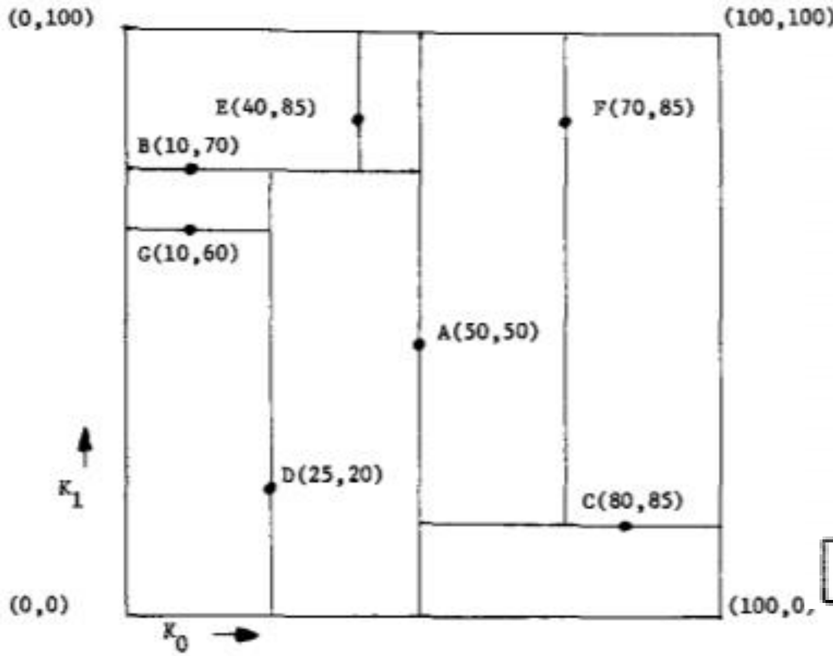
Tree-Based Methods

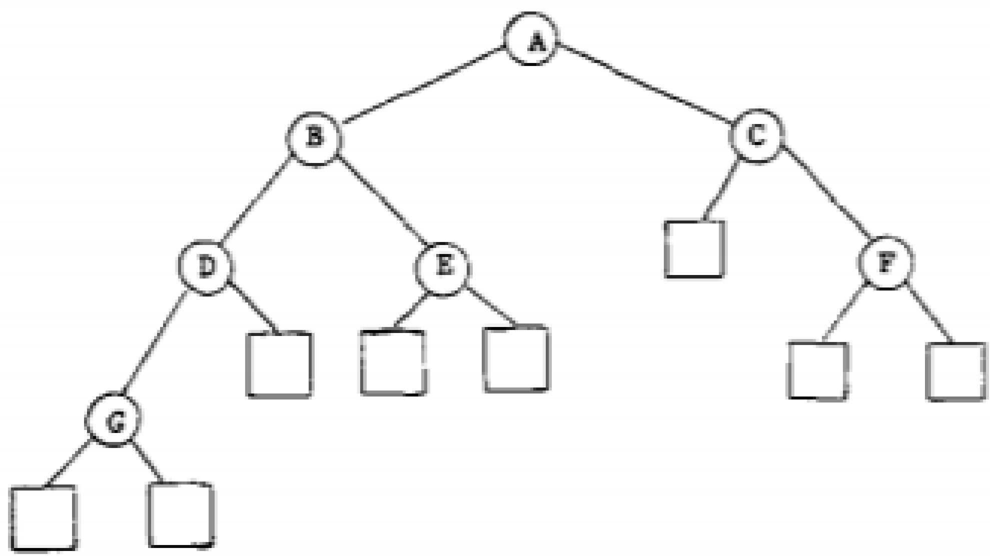
KD-tree

Publications

Bentley CACM'75

- Solution for exact kNN search



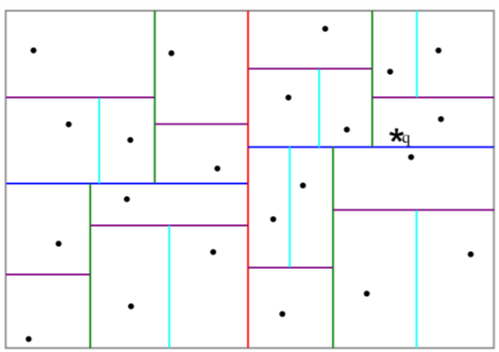


Randomized KD-tree

Publications

Silpa-Anan CVPR'08

- Solution for ng-approximate kNN search
  - Multiple randomized kd-trees with a small set of dimensions with highest variance
  - Concurrent search on the forest of kd-trees



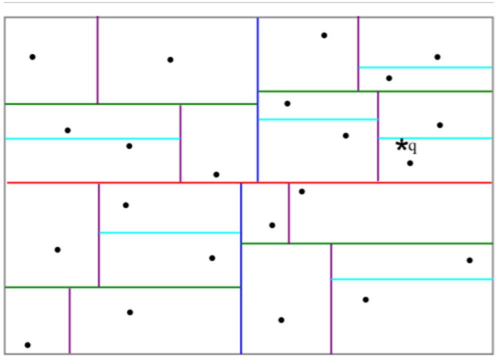


Figure from Muja et al. VISAPP"09

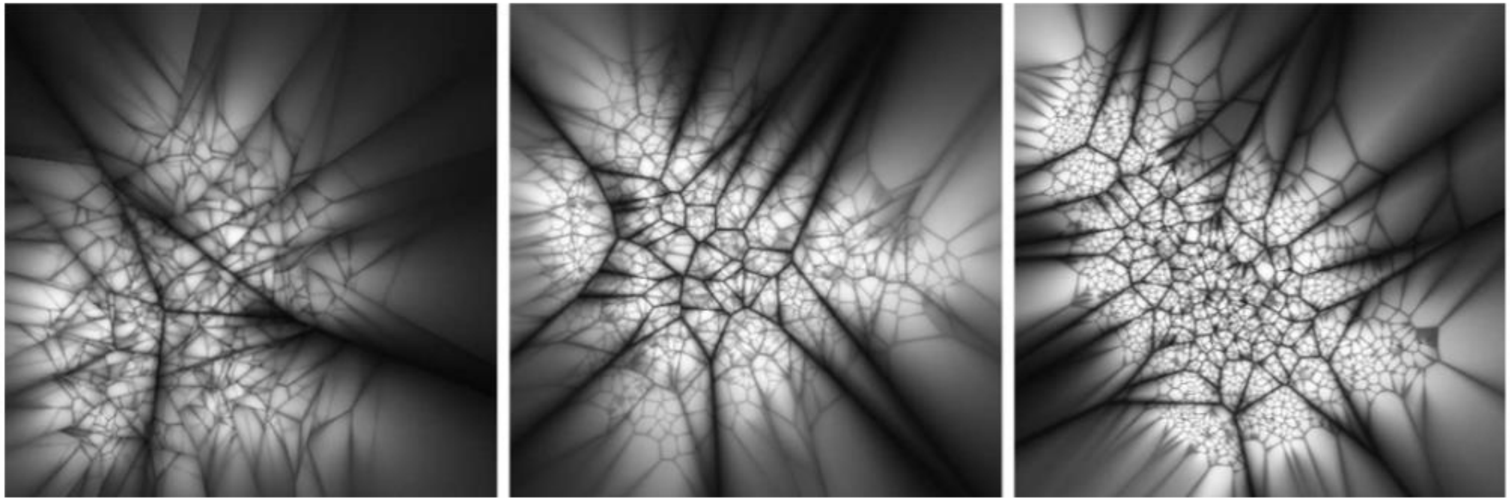
Example of randomized kd-trees. The nearest neighbor is across a decision boundary from the query point in the first decomposition, however is in the same cell in the second decomposition.

Flann

Publications

Muja et al. VISAPP'09

- Solution for ng-approximate kNN search
  - Randomized kd-tree
  - Hierarchical k-means



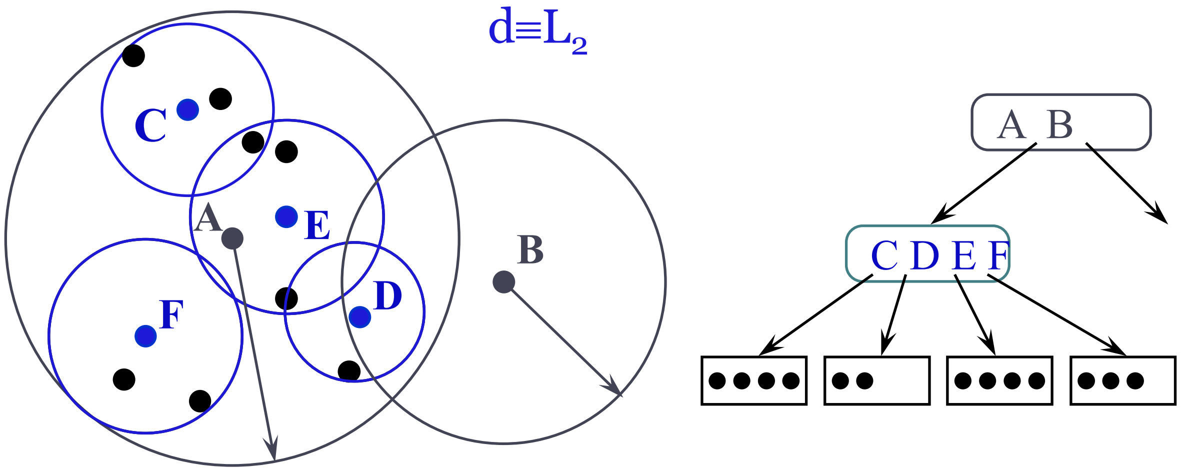
Projections of priority search k-means trees constructed using different branching factors: 4, 32, 128

MTree

Publications

Ciaccia et al. VLDB'97

Ciaccia et al. ICDE'00



- Solution for exact and δ-ε-approximate kNN search
- Each node N of the tree has an associated region, Reg(N), defined as

  $$\mathbf{Reg(N) = \{p: p \in U\ ,\ d(p,v_N) \leq r_N\}}$$

  where:
  - v_N (the "center") is also called a routing object, and
  - r_N is called the (covering) radius of the region
- The set of indexed points p that are reachable from node N are guaranteed to have $d(p,v_N) \leq r_N$

Slide by M. Patella.

MTree

Publications

Ciaccia et al. VLDB'97

Ciaccia et al. ICDE'00

- Each node N stores a variable number of *entries*

Leaf node:

- An entry E has the form E=(ObjFeatures,distP,TID), where
  - ObjFeatures are the feature values of the indexed object
  - distP is the distance between the object and its parent routing object (i.e, the routing object of node N)

Internal node:

- E=(RoutingObjFeatures,CoveringRadius,distP,PID), where
  - RoutingObjFeatures are the feature values of the routing object
  - CoveringRadius is the radius of the region
  - distP is the distance between the routing object and its parent routing object (undefined for entries in the root node)

Slides by M. Patella.

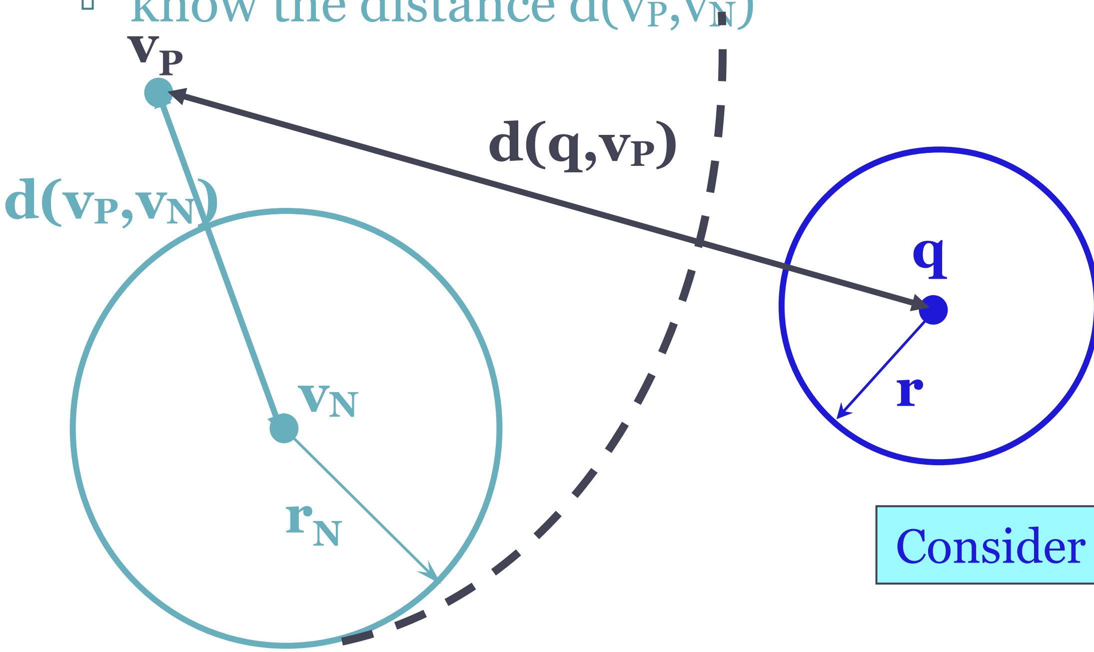
Mtree- Fast pruning based on distP

Publications

Ciaccia et al. VLDB'97

Ciaccia et al. ICDE'00

- Pre-computed distances distP are exploited during query execution to save distance computations
- Let v_P be the parent (routing) object of v_N
- When we come to consider the entry of v_N, we
  - have already computed the distance $d(q,v_P)$ between the query and its parent
  - know the distance $d(v_P,v_N)$



From the triangle inequality it is:
$d(q,v_N) \geq |d(q,v_P) - d(v_P,v_N)|$

Thus we can prune node N ***without computing*** $d(q,v_N)$ if

$$|d(q,v_P) - d(v_P,v_N)| > r_N + r$$

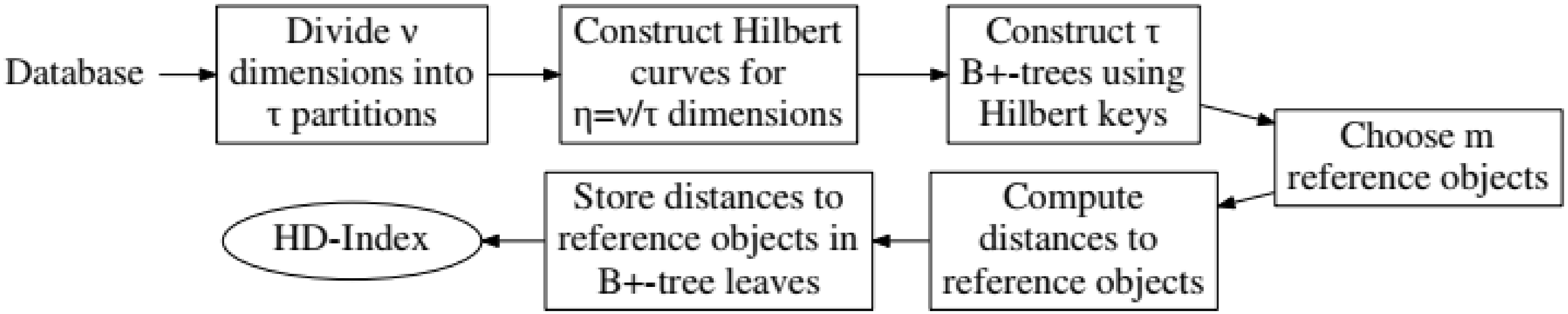
Consider a range query $\{p: d(p,q) \leq r\}$

HD-Index

Publications

Arora et al. PVLDB'18

- Solution for ng-approximate kNN search
  - Index creation



- Dimensions are partitioned
- For each partition, a space-filling (Hilbert) curve is passed
- Hilbert keys are indexed using a modified B+-tree
- Reference objects are chosen
- Leaves of B+-trees contain distance to reference objects in the full-dimensional space
- Modified B+-trees are called Reference Distance B+-trees (RDB-trees)
- Collection of RDB-trees form High-Dimensional Index (HD-Index)

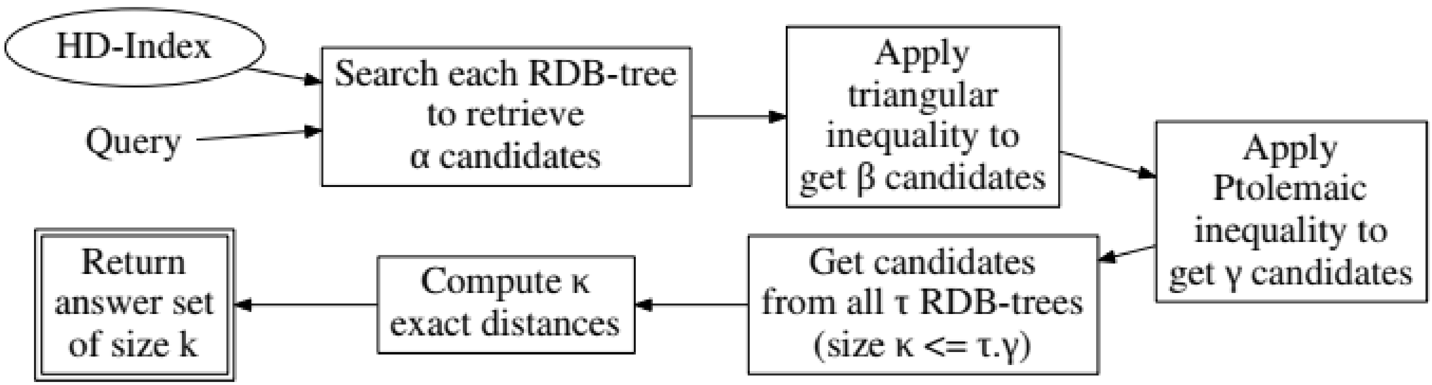
Slide by A. Bhattacharya

HD-Index

Publications

Arora et al. PVLDB'18

- Solution for ng-approximate kNN search
  - Query answering



- Query Q partitioned into same subspaces
- For each RDB-tree, initial search retrieves α candidates
  - $\alpha/2$ on each side of the query Hilbert key
- Candidates are refined successively to β and γ candidates using triangular and Ptolemaic inequalities
- Collection of all such candidates form the final candidate set of size κ
- *Exact* distance computations are done with these κ candidates to return top-k

Slide by A. Bhattacharya

High-d Vector Similarity Search
State-of-the-Art Methods

Hash-Based Methods

Locality Sensitive Hashing (LSH)

Publications

Indyk et al. STOC'98

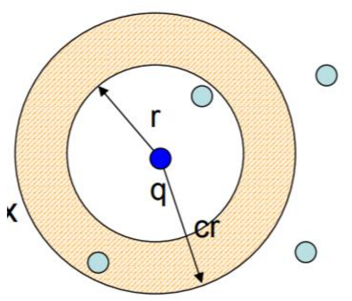
- Solution for δ-ε-approximate kNN search $\delta < 1$
- Random projections into a lower dimensional space using hashing
- Probability of collisions increases with locality

- c-Approximate r-Near Neighbor: build data structure which, for any query q:
  - If there is a point $p \in P$, $||p-q|| \leq r$ Then return $p' \in P$, $||p-q|| \leq c\,r$

- c-approximate nearest neighbor reduces to c-approximate near neighbor
  - Enumerate all approximate near neighbors

- Find a vector in a preprocessed set $S \subseteq \{0, 1\}$ d that has minimum Hamming distance to a query vector $y \in \{0, 1\}$ d

(r_1, r_2, p_1, p_2)-sensitive [IM98]
- $\Pr[\ h(x) = h(y)\ ] \geq p_1$, if $dist(x, y) \leq r_1$
- $\Pr[\ h(x) = h(y)\ ] \leq p_2$, if $dist(x, y) \geq r_2$



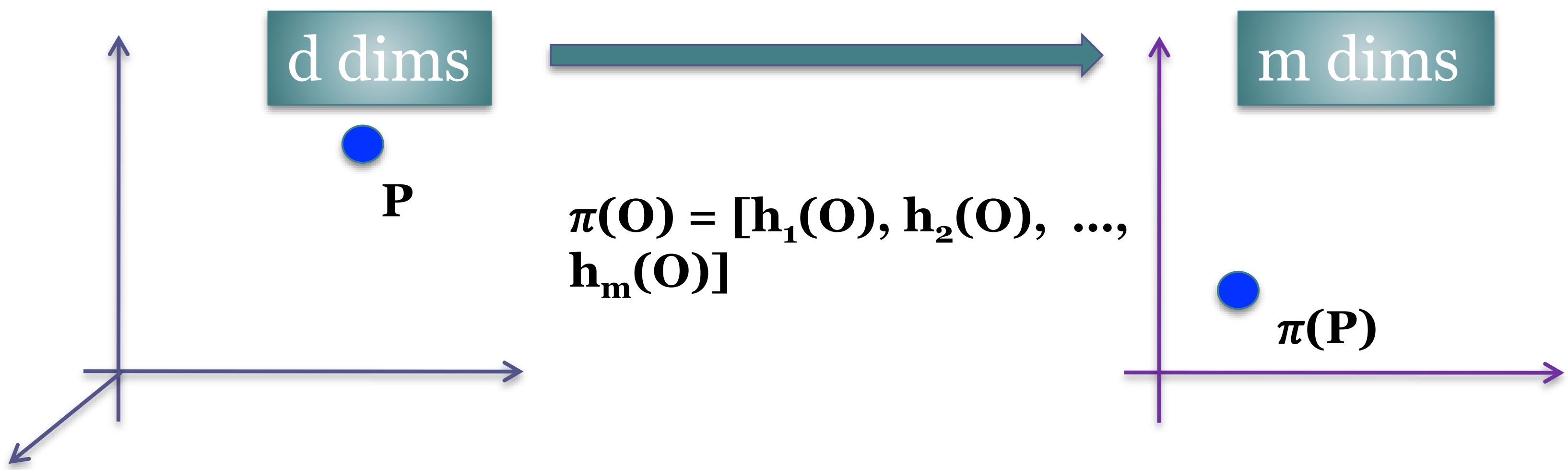
Locality Sensitive Hashing (LSH)

Publications

Andoni et al. CACM’08

- A large family
  - Different distance measures:
    - Hamming distance
    - L_p ($0 < p \leq 2$): use p-stable distribution to generate the projection vector
    - Angular distance (simHash)
    - Jaccard distance (minhash)
  - Tighter Theoretical Bounds
  - Better query efficiency/smaller index size

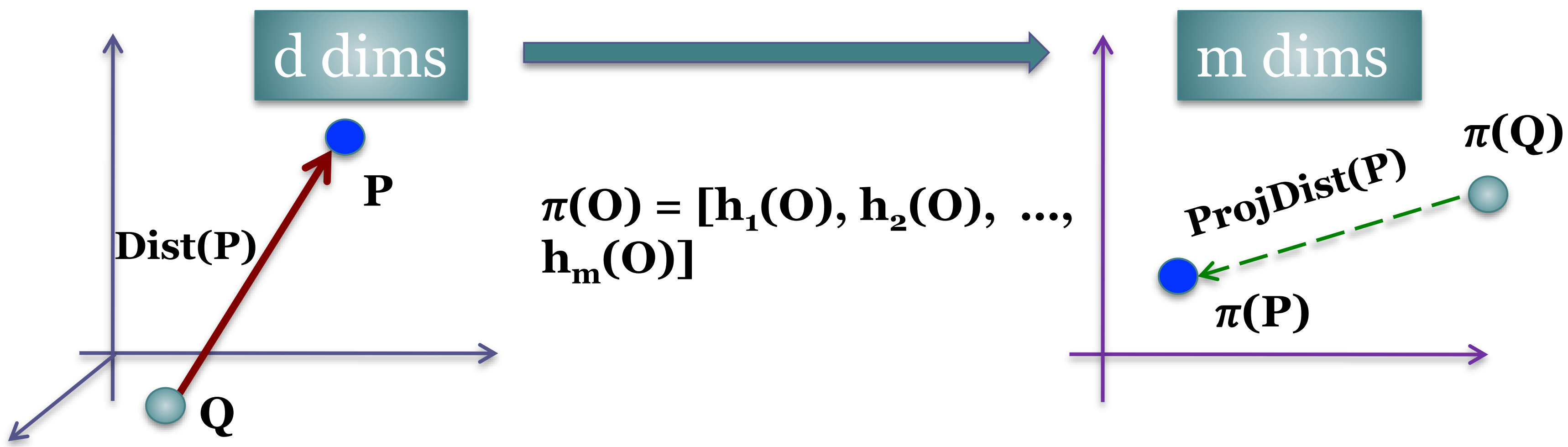
Probabilistic Mapping



- Probabilistic, linear mapping from the original space to the projected space

Slide by W. Wang

Probabilistic Mapping



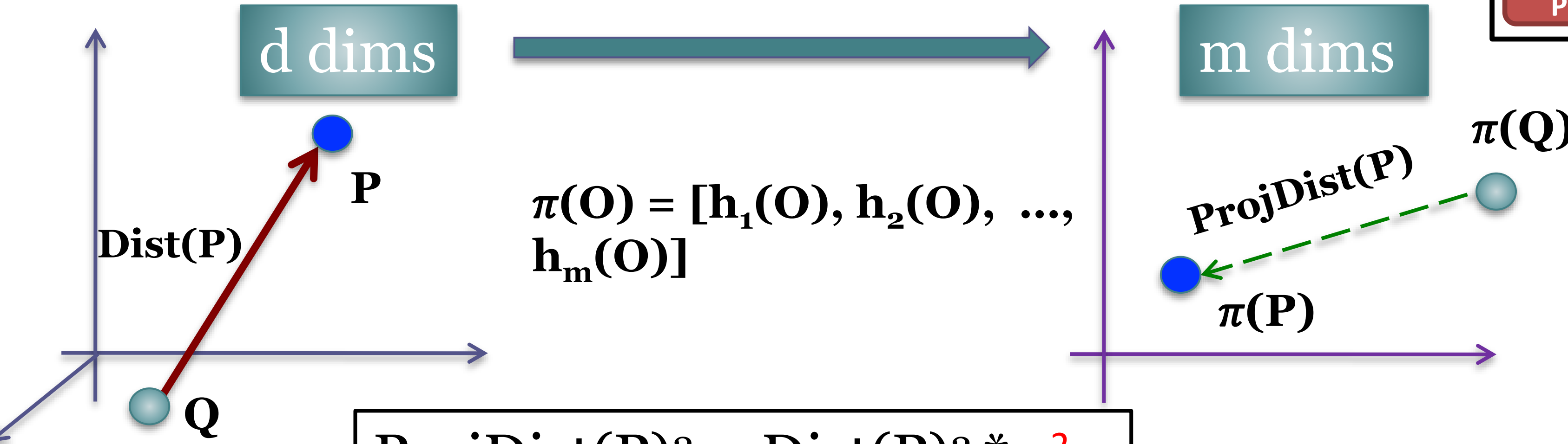
- Probabilistic, linear mapping from the original space to the projected space
- What about the distances (wrt Q or π(Q)) in these two spaces?

Slide by W. Wang

SRS

Publications

Sun et al. PVLDB' 14



$\pi(O) = [h_1(O), h_2(O), \ldots, h_m(O)]$

$$\text{ProjDist}(P)^2 \sim \text{Dist}(P)^2 * \chi^2_m$$

- Given that ProjDist(P) ≤ r, what can we infer about Dist(P)?
  - If Dist(P) ≤ R, then Pr[ProjDist(P) ≤ r] ≥ $\Psi_m((r/R)^2)$
  - If Dist(P) > cR, then Pr[ProjDist(P) ≤ r] ≤ $\Psi_m((r/cR)^2) = t$
  - (some probability) at most O(tn) points with ProjDist ≤ R
  - (constant probability) one of the O(tn) points has Dist ≤ R

- This solves the so-called (R, c)-NN queries → returns a c^2 ANN
- Using another algorithm & proof → returns a c-ANN

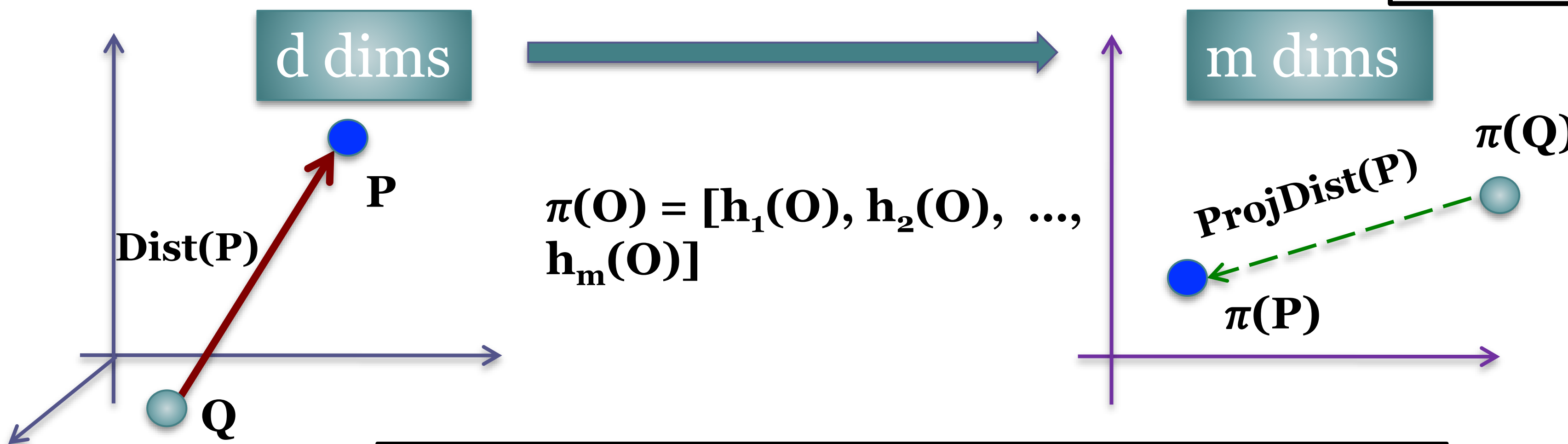
Slide by W. Wang

C2LSH/QALSH

Publications

Huang et al. PVLDB' 15

Gan et al. SIGMOD'12



$\boxed{\text{Collision wrt w: if } |h_i(P) - h_i(Q)| \leq w}$

- Given that P's #collision ≥ αm, what can we infer about Dist(P)?
  - If Dist(P) ≤ R, then Pr[#collision ≥ αm] ≥ γ_1
  - If Dist(P) > cR, then Pr[#collision ≥ αm] ≤ γ_2
  - (some probability) at most O(γ_2*n) points with #collision ≥ αm
  - (constant probability) one of the O(γ_2*n) points has #collision ≥ αm

Slide by W. Wang

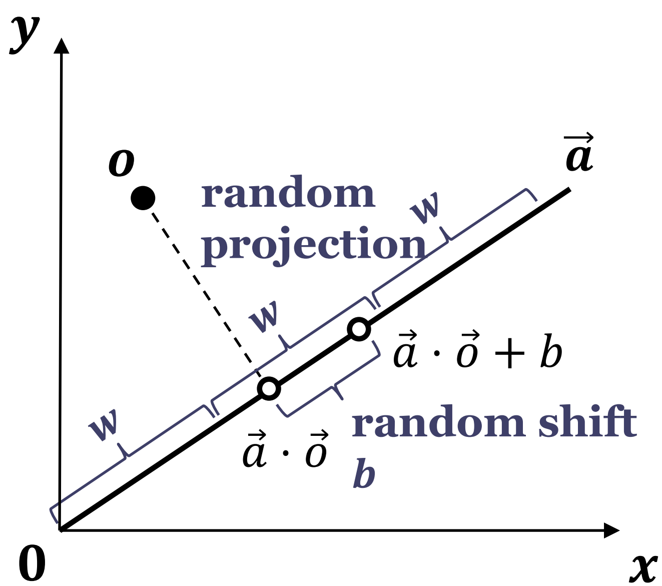
Query-oblivious LSH functions

Publications

Huang et al. PVLDB'15

- **The query-oblivious LSH functions for Euclidean distance:**

$$h_{\vec{a},b}(o) = \left\lfloor \frac{\vec{a}\cdot\vec{o}+b}{w} \right\rfloor$$



Query-Oblivious Bucket Partition:

- Buckets are ***statically*** determined before any query arrives;
- Use the **origin (i.e., "o") as anchor**;
- If $\boldsymbol{h_{\vec{a},b}(o) = h_{\vec{a},b}(q)}$, we say ***o* and *q* collide** under $h_{\vec{a},b}(\cdot)$.

Slide by Q. Huang

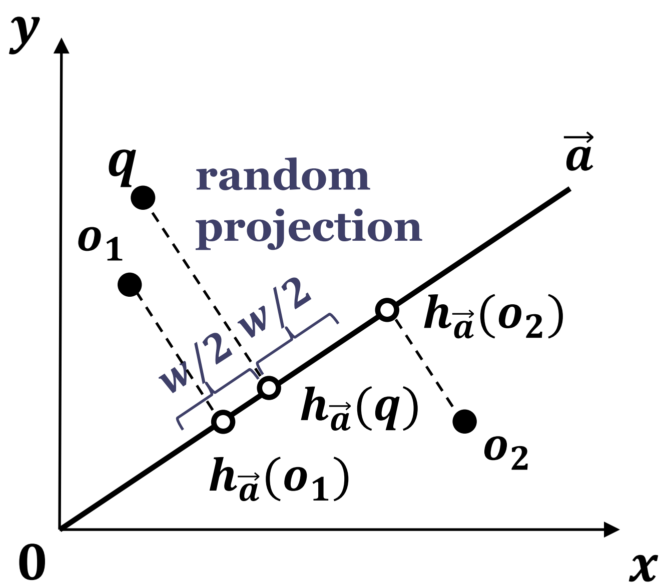
QALSH

Publications

Huang et al. PVLDB' 15

- **Query-aware LSH function = random projection + query-aware bucket partition**

$$h_{\vec{a}}(o) = \vec{a} \cdot \vec{o}$$



Query-Aware Bucket Partition:

– Buckets are **dynamically** determined when q arrives;

– Use "$h_{\vec{a}}(q)$" **as anchor** ;

– If an object o falls into the **anchor bucket**, i.e., $|h_{\vec{a}}(o) - h_{\vec{a}}(q)| \leq \frac{w}{2}$, we say **$o$ and q collide** under $h_{\vec{a}}(\cdot)$.

Slide by Q. Huang

VHP

Publications

Lu et al. PVLDB' 20

- Solution for δ-ε-approximate kNN search
  - Indexing:
    - Store LSH projections with independent B+ trees.
  - Querying
    - Impose a virtual hypersphere in the original high-d space
    - Keep enlarging the virtual hypersphere to accommodate more candidate until the success probability is met

Slide by W. Wang

Some Comparisons

Publications

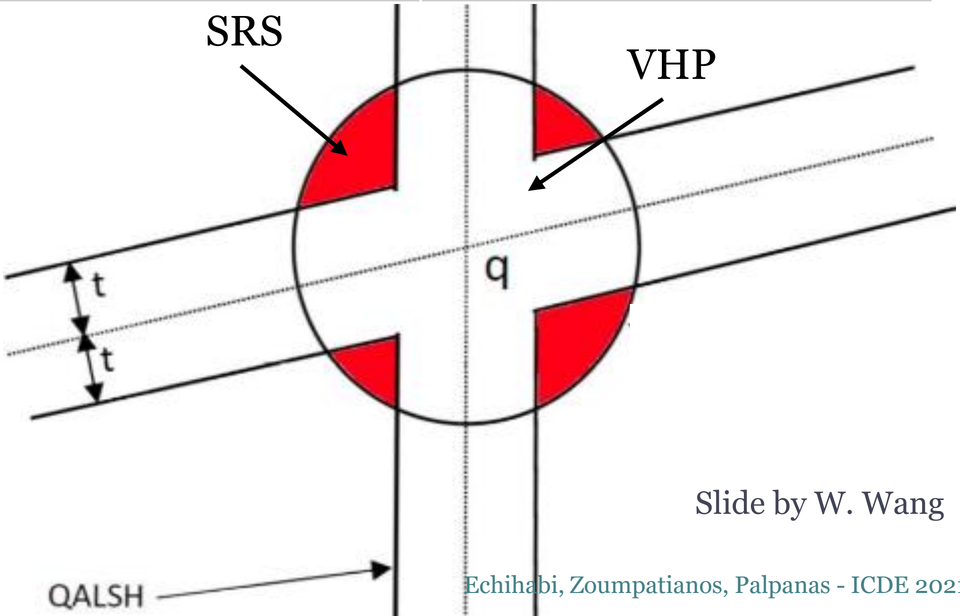
Huang et al. PVLDB' 15

Candidate Conditions

Method	Collision Count	(Observed) Distance	Max Candidates
SRS	= m	$\leq$ r	T
QALSH	$\geq \alpha$m	n/a	βn
VHP	$\geq$ i (i = 1, 2, ..., m)	$\leq l_i$	βn

Candidate Regions

VHP = SRS ∩ QALSH



Slide by W. Wang

High-d Vector Similarity Search State-of-the-Art Methods

Quantization-Based Methods

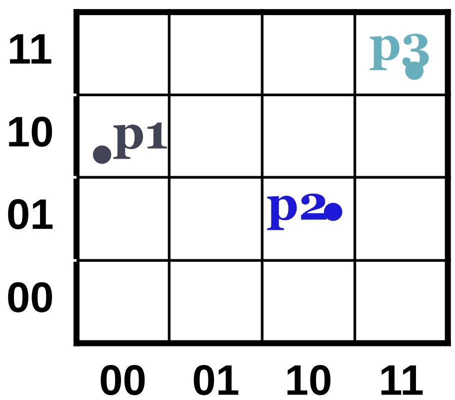
VA-file

Publications

Blott et. al VLDB'98

- A solution for exact kNN search
- The basic idea of the VA-file is to speed-up the sequential scan by exploiting a "Vector Approximation"
- Each dimension of the data space is partitioned into 2^{b_i} intervals using b_i bits
  - E.g.: the 1st coordinate uses 2 bits, which leads to the intervals 00,01,10, and 11
- Thus, each coordinate of a point (vector) requires now b_i bits instead of 32
- The VA-file stores, for each point of the dataset, its approximation, which is a vector of $\sum_{i=1,D} b_i$ bits

Data space



Feature values

p1	0.1 0.6
p2	0.7 0.4
p3	0.9 0.3

VA-file

p1	00 10
p2	10 01
p3	11 11

Slides by M. Patella.

VA-file

Publications

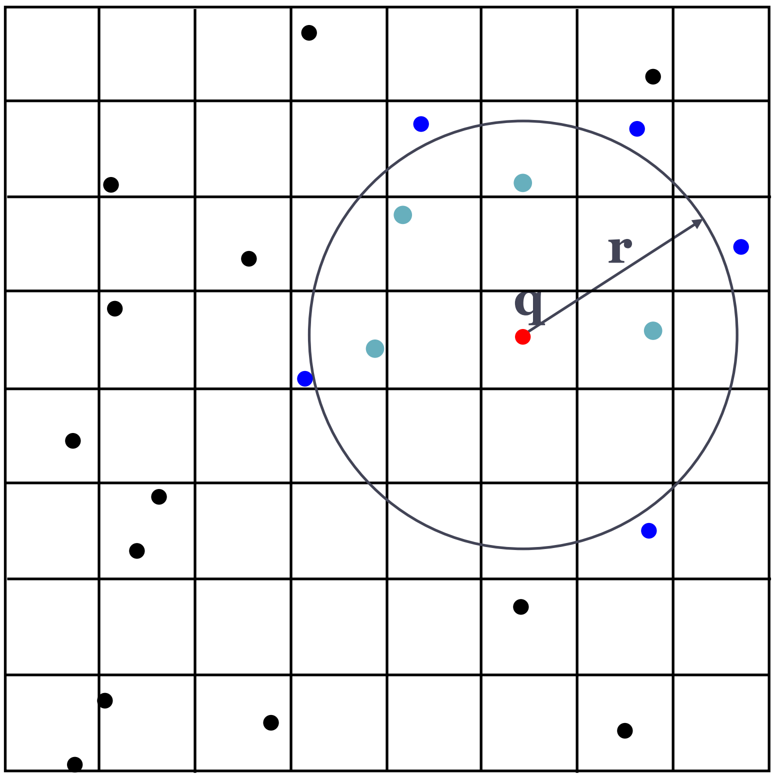
Blott et. al VLDB'98

- Query processing with the VA-file is based on a filter & refine approach
- For simplicity, consider a range query

  Filter: the VA file is accessed and only the points in the regions that intersect the query region are kept

  Refine: the feature vectors are retrieved and an exact check is made

actual results
false drops
excluded points



Slides by M. Patella.

VA+file

Publications

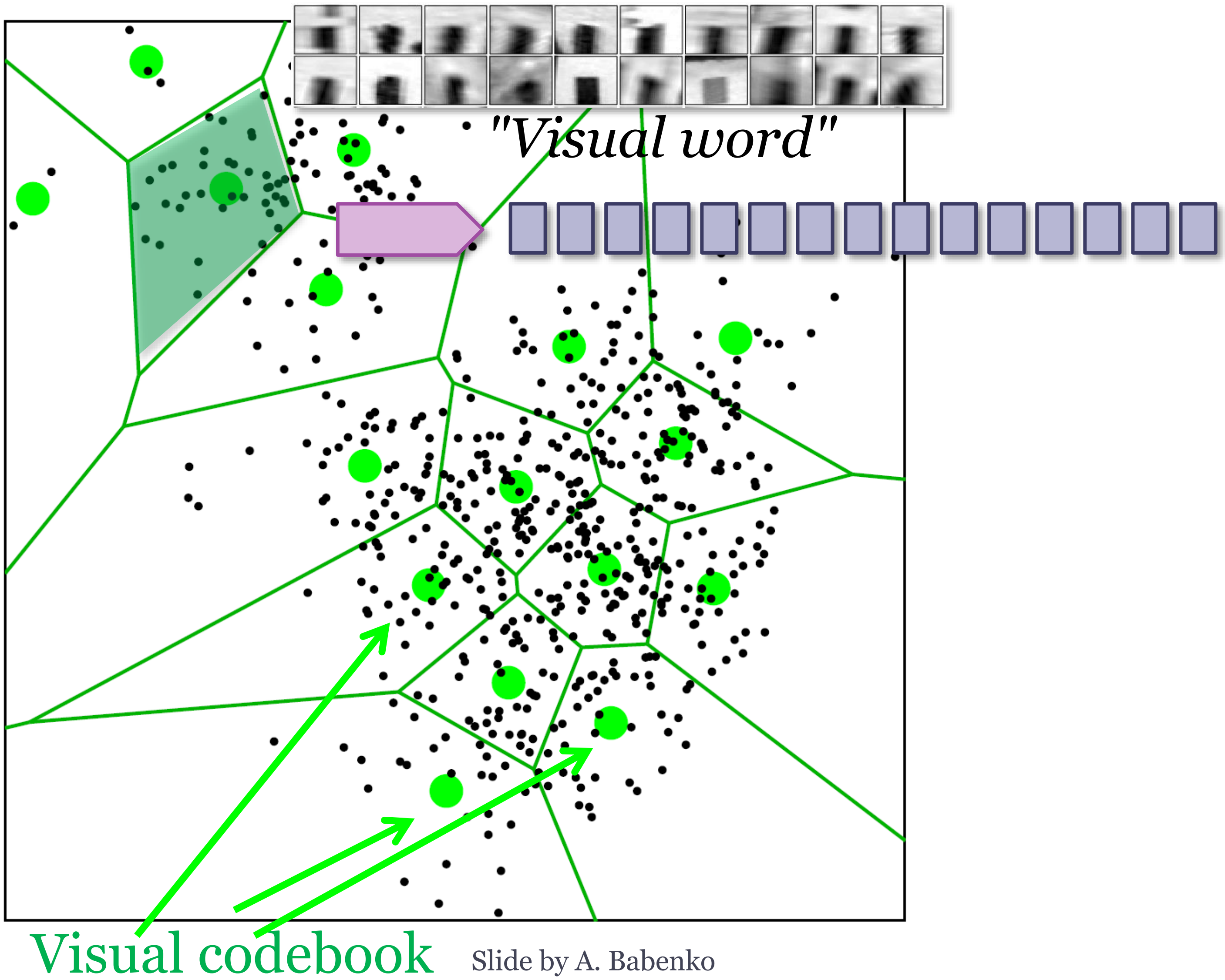
Ferhatosmanoglu et al. CIKM'00

- Solution for exact kNN search
- An improvement of the VA-file method:
  - Does not assume that neighboring dimensions are uncorrelated
  - Decorrelates the data using KLT
  - Allocates bits per dimension in a non-uniform fashion
  - Partitions each dimension using k-means instead of equi-depth

The Inverted Index

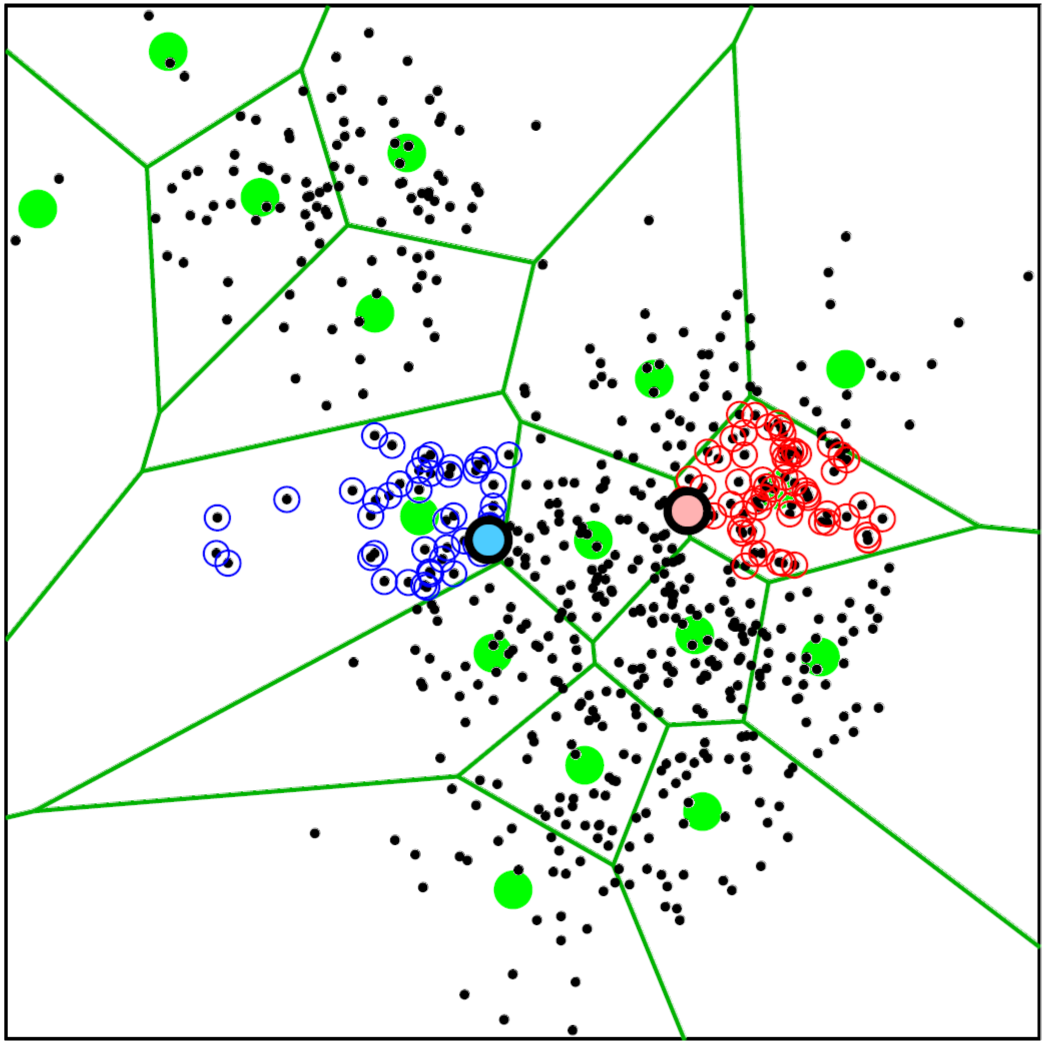
Publications

Sivic et al. ICCV' 03



Slide by A. Babenko

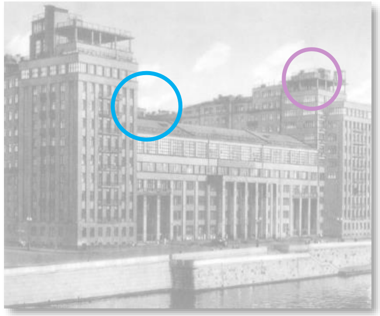
Querying the Inverted Index



Publications

Sivic et al. ICCV' 03

Query:



- Have to consider several words for best accuracy
- Want to use as big codebook as possible

conflict

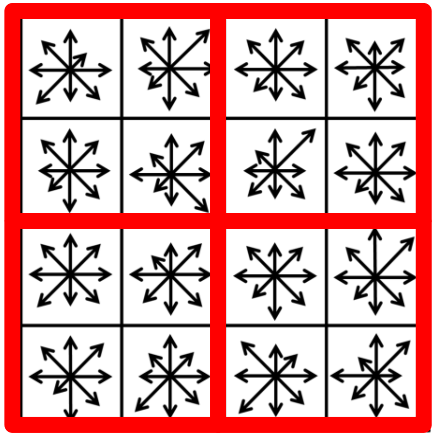
- Want to spend as little time as possible for matching to codebooks

Slide by A. Babenko

Product Quantization

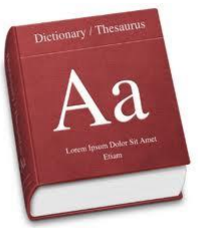
Publications

Jegou et al. TPAMI' 11



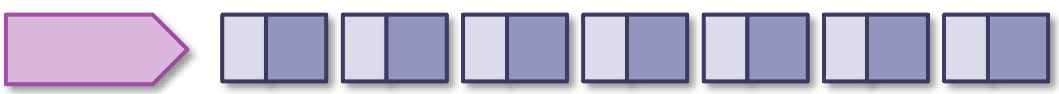
1. Split vector into correlated subvectors
2. use separate small codebook for each chunk

Quantization vs. Product quantization:



For a budget of 4 bytes per descriptor:

1. Can use a single codebook with 1 billion codewords
2. Can use 4 different codebooks with 256 codewords each



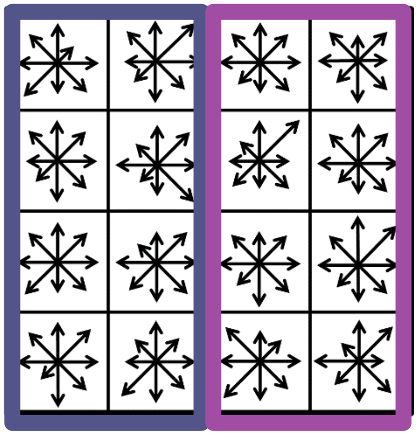
IVFADC+ variants (state-of-the-art for billion scale datasets) = inverted index for indexing + product quantization for reranking

Slide by A. Babenko

The Inverted Multi-Index

Publications

Babenko et al. TPAMI' 12

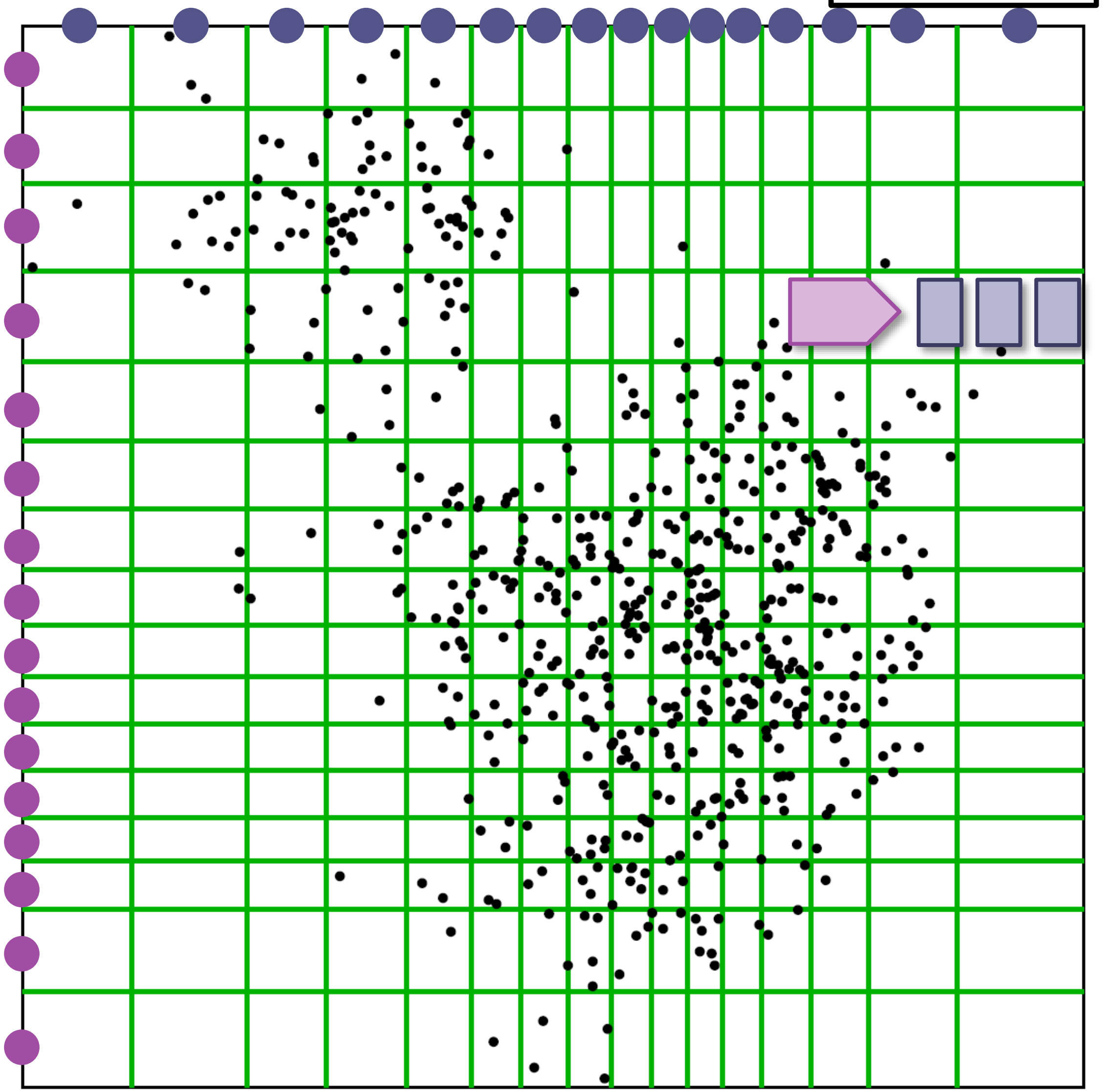


Idea: use product quantization for indexing

Main advantage:
For the same K, much finer subdivision achieved

Main problem:
Very non-uniform entry size distribution

Slide by A. Babenko

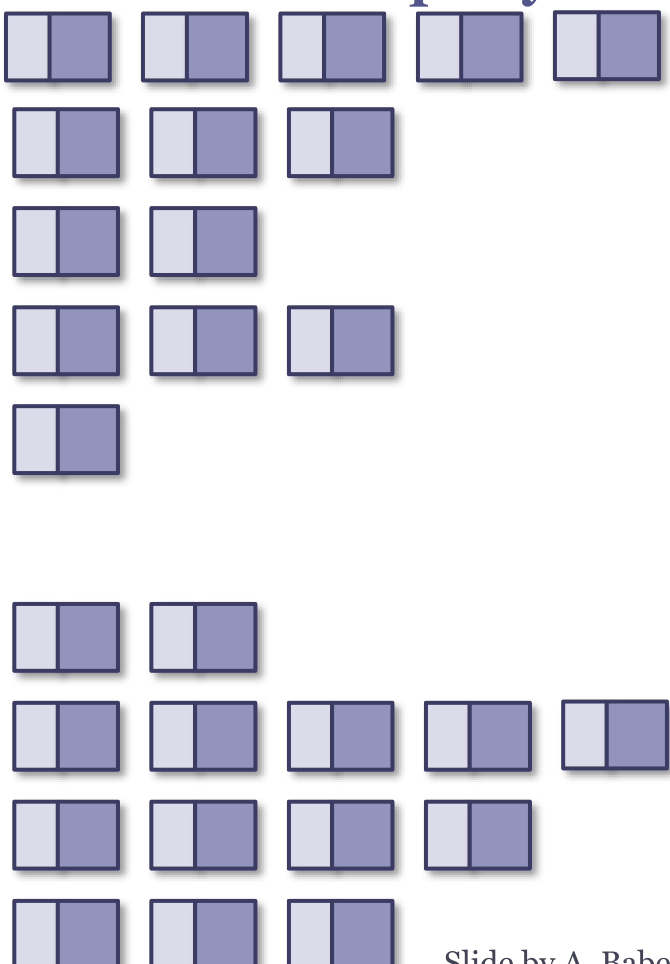


Querying the Inverted Multi-Index

Publications

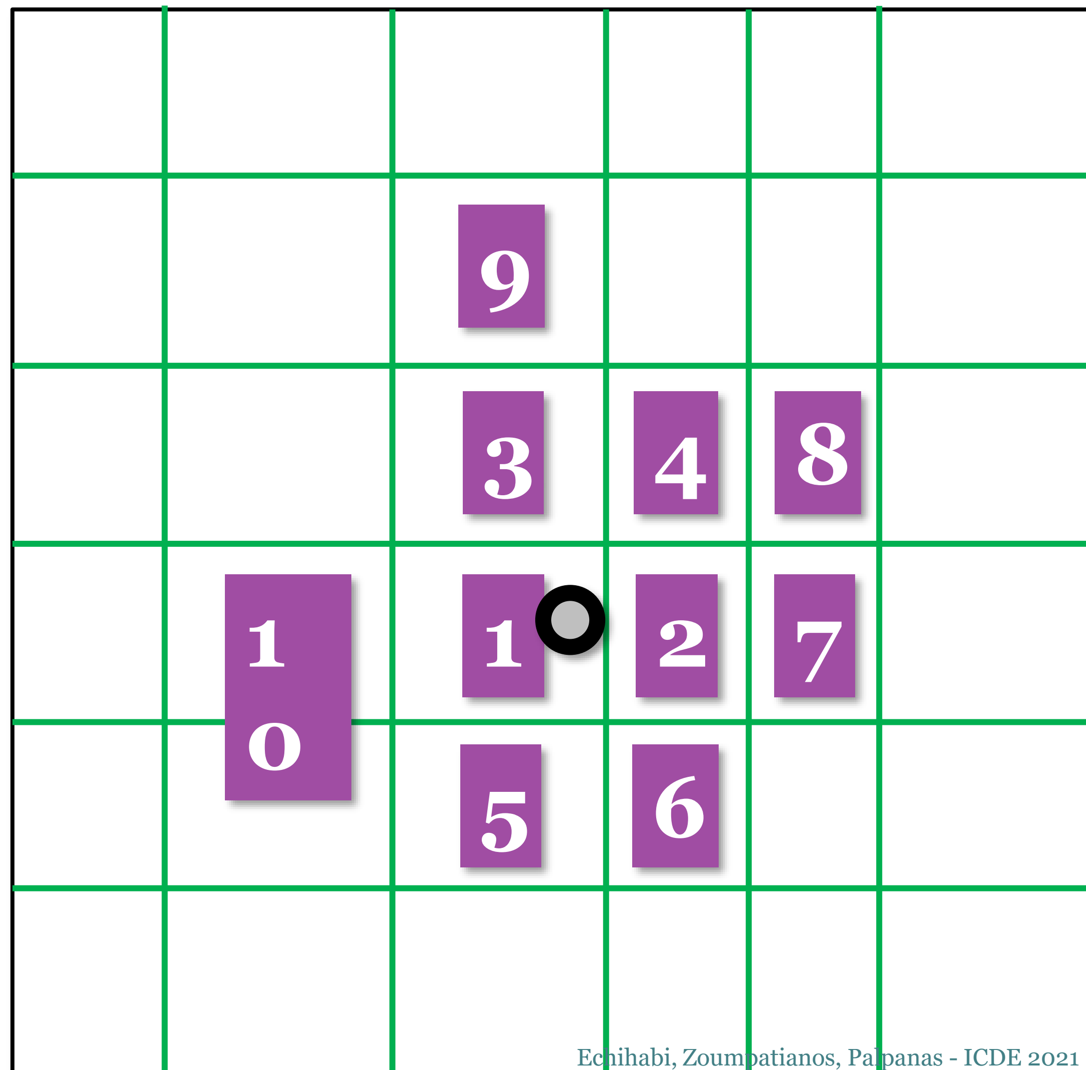
Babenko et al. TPAMI' 12

Answer to the query:



Input: query

Output: stream of entries



Slide by A. Babenko

High-d Vector Similarity Search State-of-the-Art Methods

Graph-Based Methods

Conceptual Graphs

- Voronoi/Delaunay Diagrams
- kNN Graphs
- Navigable Small World Graphs
- Relative Neighborhood graphs

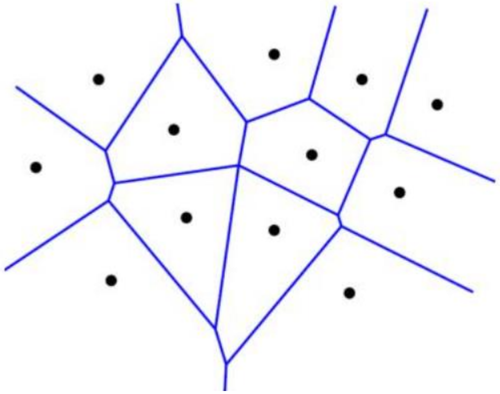
The Delaunay Diagram

Publications

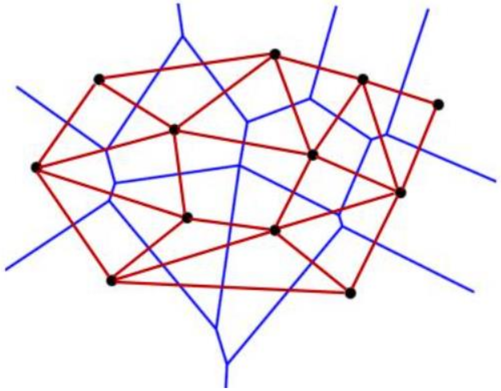
Delaunay
CSMN' 39

Delaunay Diagram – Dual of Voronoi Diagram

- The VD is constructed by decomposing the space using a finite number of points, called sites into regions, such that each site is associated to a region consisting of all points closer to it than to any other site.
- The DT is the dual of the VD, contructed by connecting sites with an edge if their regions share a side.



Voronoi Diagram



Delaunay Diagram

kNN Graphs

Publications

Anastasiu et al. CIKM' 15

Dong et al. WWW' 11

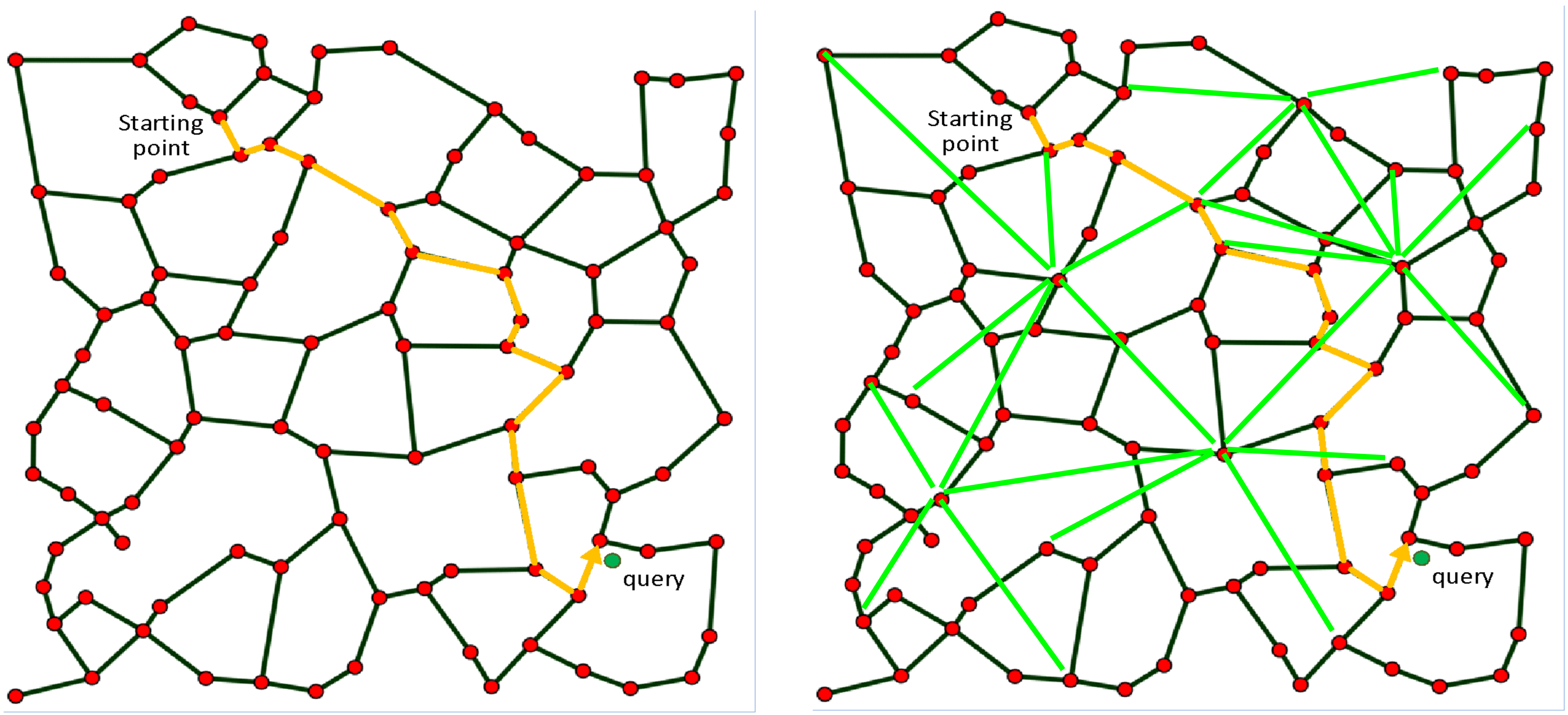
- Exact kNN graphs on n d-dimensional points:
  - Each point in the space is considered a node
  - A directed edge is added between nodes node A and B (A -=> B) if B is a k-nearest neighbor of A
  - $O(dn^2)$
  - Example: L2knng
- Approximate kNN Graphs:
  - LSH
  - Heuristics
    - Example: NN-Descent: "a neighbor of a neighbor is also likely to be a neighbor"

NSW Graphs

Publications

Kleinberg STOC' 00

- Augment approximate kNN graphs with long range links:
  - Milgram experiment
  - Shorten the greedy algorithm path to log(N)

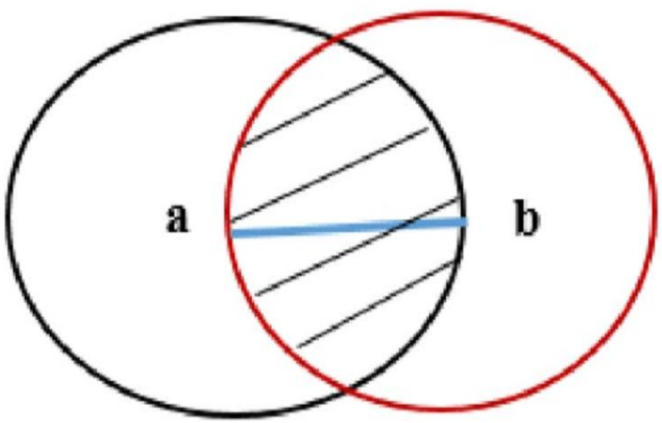


Relative Neighbourhood graph (RNG)

Publications

Toussaint
Pat. Recognit.'80

- A superset of the minimal spanning tree (MST) and a subset of the Delaunay Diagram.
- Two algorithms for obtaining the RNG of n points on the plane:
  - An algorithm for 1-d space in o(n2) time
  - Another algorithm for d-dimensional spaces running in o(n3).
- An edge is constructed between two vertices if there is no vertex in the intersection of the two balls

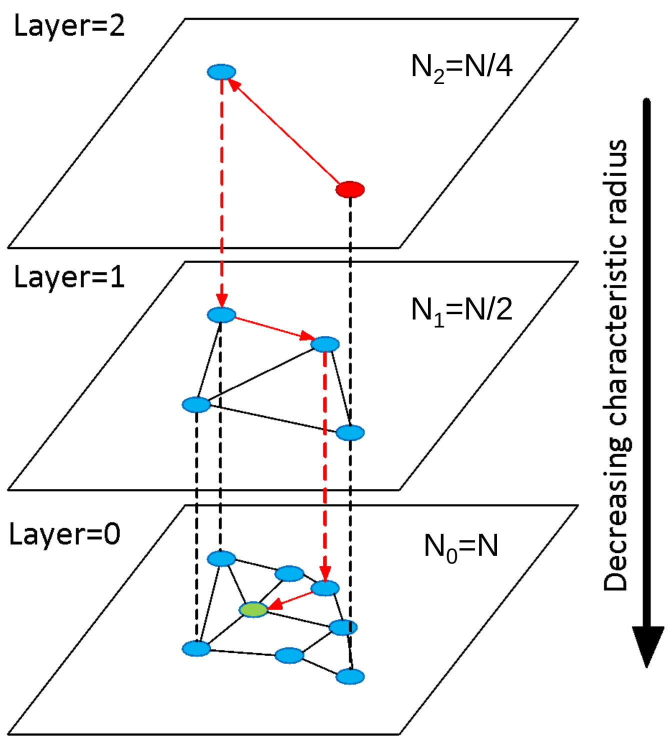


HNSW

Publications

Malkov et al. TPAMI' 20 Arxiv'16

- In HNSW we split the graph into layers (fewer elements at higher levels)
- Search starts for the top layer. Greedy routing at each level and descend to the next layer.
- Maximum degree is capped while paths ~ log(N) → log(N) complexity scaling.
- Incremental construction



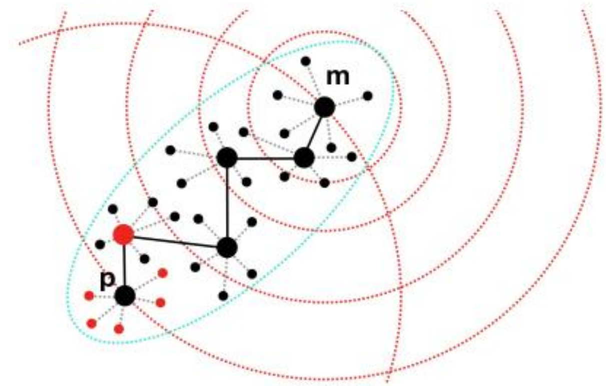
Slides by Malkov

Navigating Speading-out Graph (NSG)

Publications

Fu et al.
PVLDB' 19

- RNGs do not guarantee monotonic search
  - There exists at least one monotonic path. Following this path, the query can be approached with the distance decreasing monotonically
- Propose a Monotonic RNG (MRNG)
- Build an approximate *k*NN graph.
- Find the *Navigating Node*. (*A*ll search will start with this fixed node – center of the graph).
- For each node p, find a relatively small candidate neighbour set. (*sparse*)
- Select the edges for p according to the definition of MRNG. (*low complexity*)
- leverage Depth-First-Search tree (*connectivity*)



Slides by Fu

Questions?

Experimental Comparisons: Similarity Search Methods

How do similarity search methods compare?

- several methods proposed in last 3 decades by different communities
- never carefully compared to one another

- we now present results of extensive experimental comparison

Experimental Comparisons: A Taxonomy

Methods

Similarity Search Methods

Publications

Echihabi-PVLDB'18

Echihabi-PVLDB'19

Methods

Publications

Echihabi-PVLDB'18

Echihabi-PVLDB'19

Publications

Echihabi-PVLDB‘18

Echihabi-PVLDB‘19

Methods

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

Similarity Search Methods

δ,ε guarantees → **δ-ε-Approximate***

No guarantees → **ng-Approximate**

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

Methods

Publications

Echihabi-PVLDB'18

Echihabi-PVLDB'19

Similarity Search Methods

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

δ,ε guarantees → **δ-ε-Approximate***

$\delta < 1$, ε guarantee → **Probabilistic**

No guarantees → **ng-Approximate**

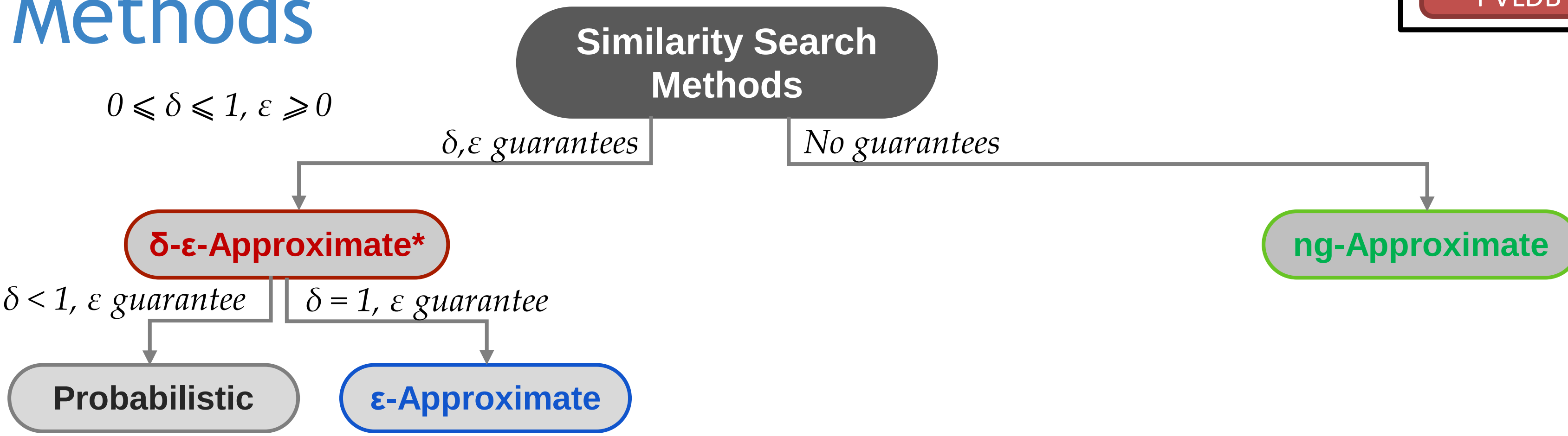
*** result is within distance (1+ ε) of the exact answer with probability δ**

Publications

Echihabi-PVLDB'18

Echihabi-PVLDB'19

Methods



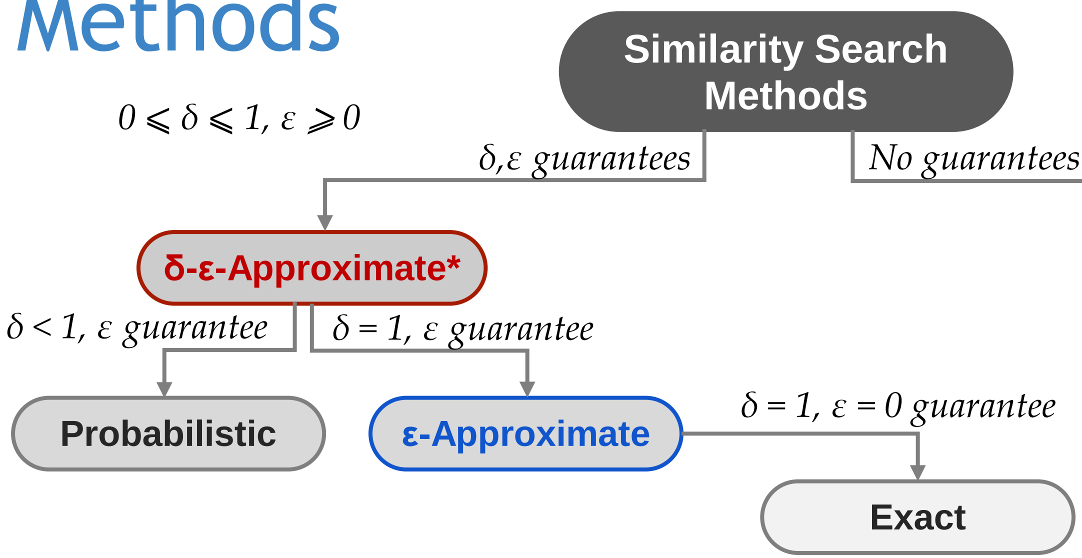
*** result is within distance (1+ ε) of the exact answer with probability δ**

Methods

Publications

Echihabi-PVLDB‘18

Echihabi-PVLDB‘19



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

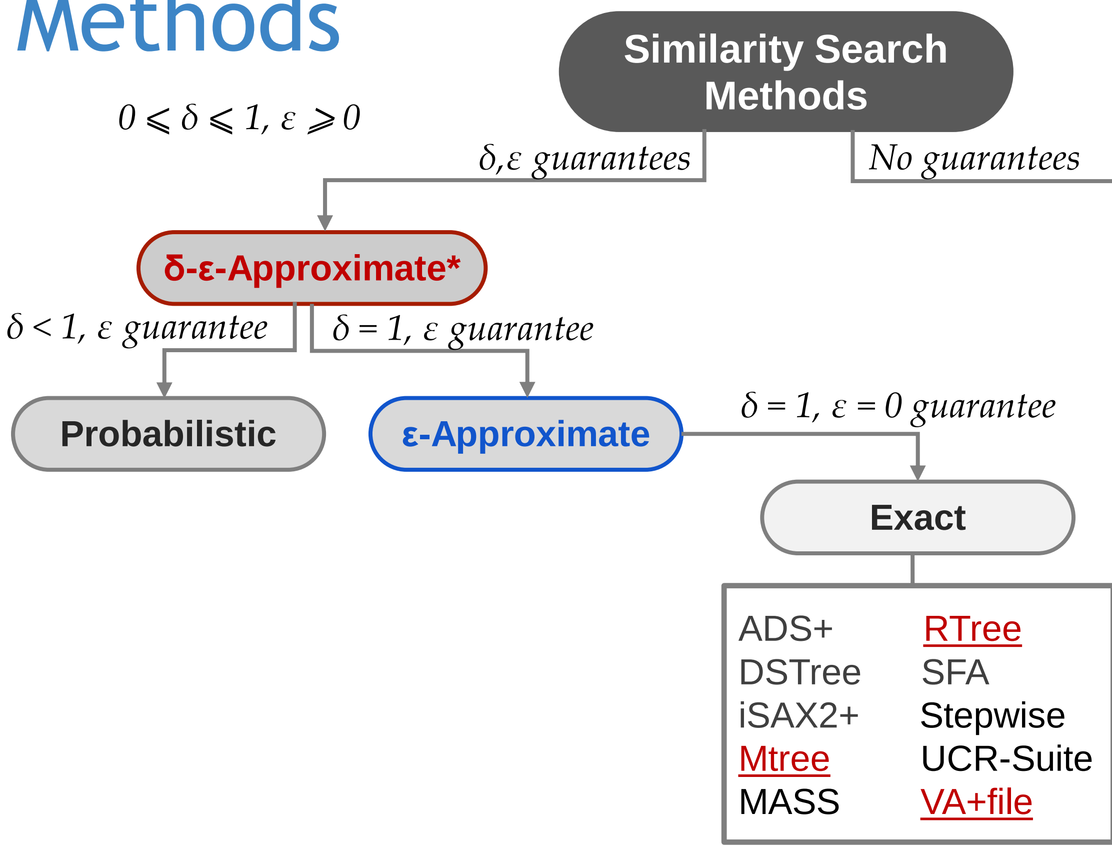
Publications

Echihabi-PVLDB'18

Echihabi-PVLDB'19

Techniques for data Series
Techniques for High-D vectors

Methods



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

Publications

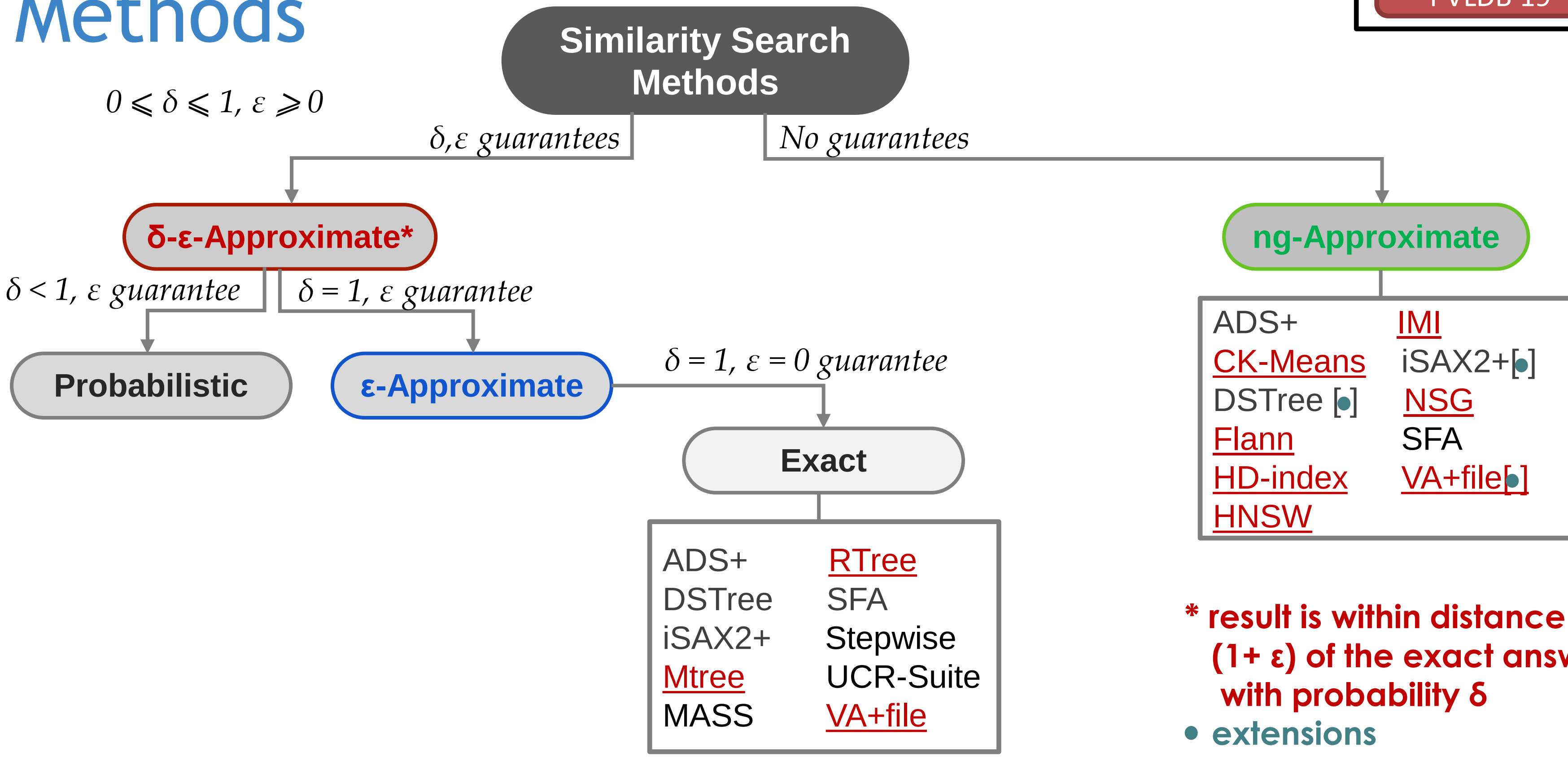
Echihabi-PVLDB'18

Echihabi-PVLDB'19

Techniques for data Series

Techniques for High-D vectors

Methods



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

• extensions

Publications

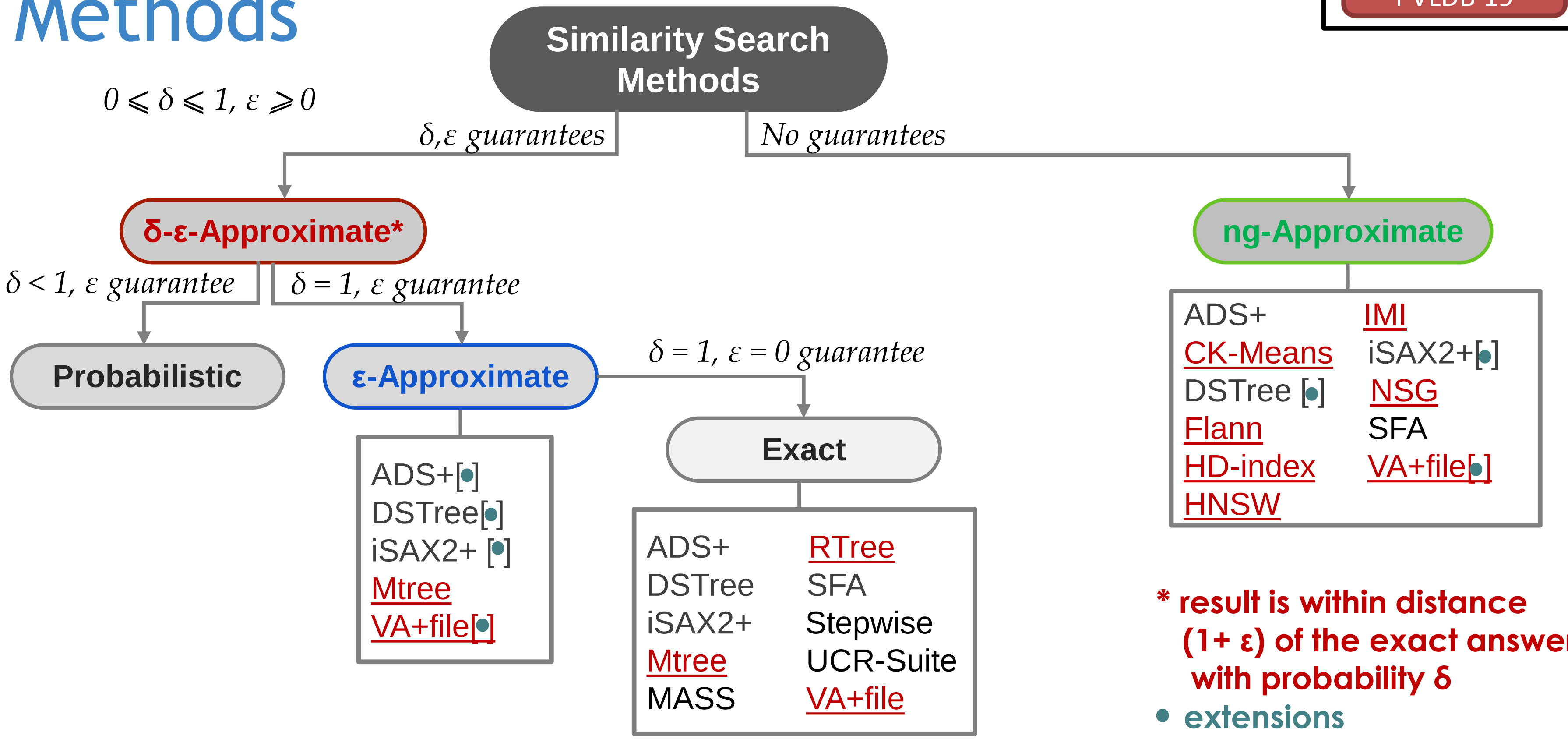
Echihabi-PVLDB‘18

Echihabi-PVLDB‘19

Techniques for data Series

Techniques for High-D vectors

Methods



Methods

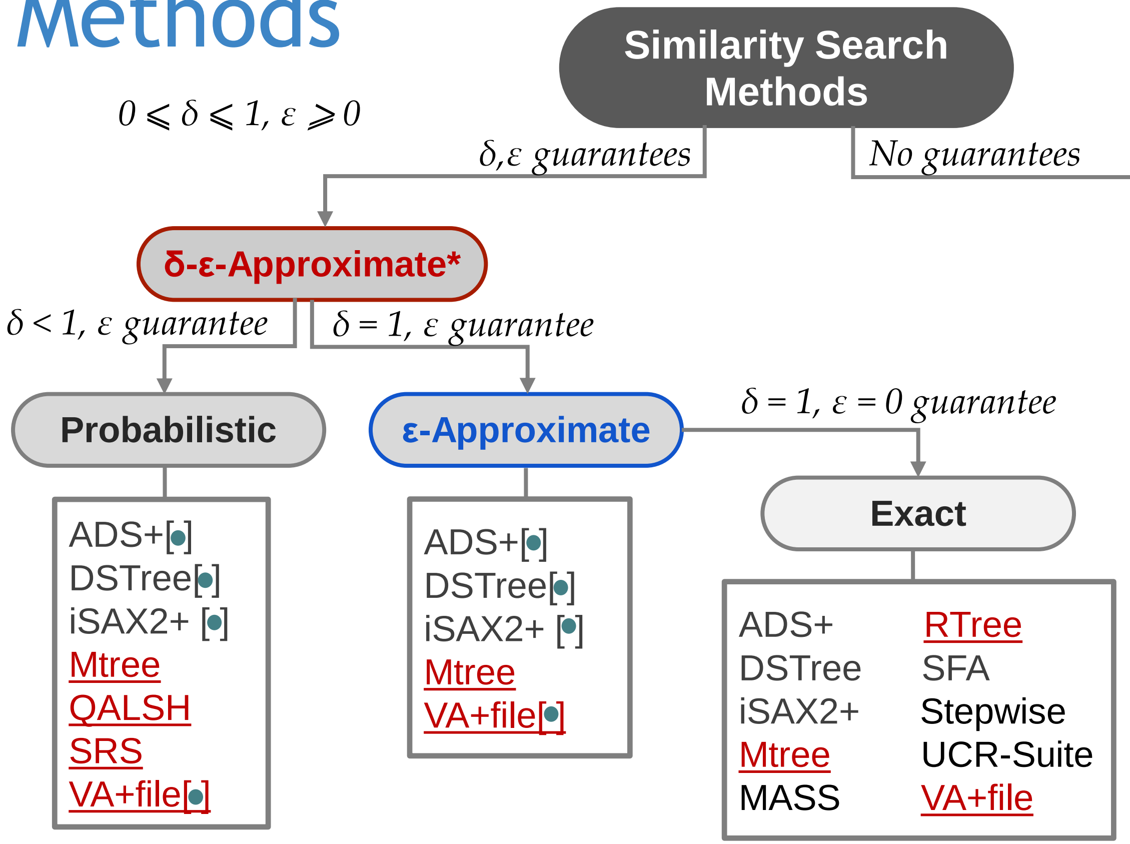
Publications

Echihabi-PVLDB‘18

Echihabi-PVLDB‘19

Techniques for data Series

Techniques for High-D vectors



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

• extensions

Experimental Comparisons: Exact Query Answering

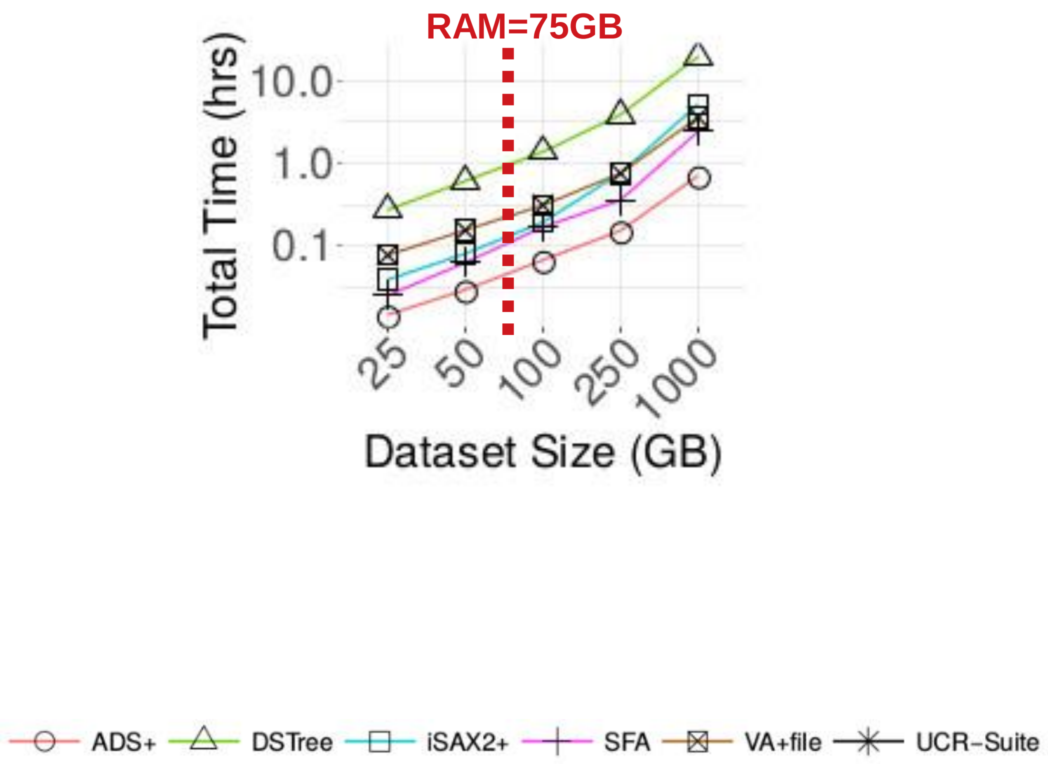
Experimental Framework

Publications

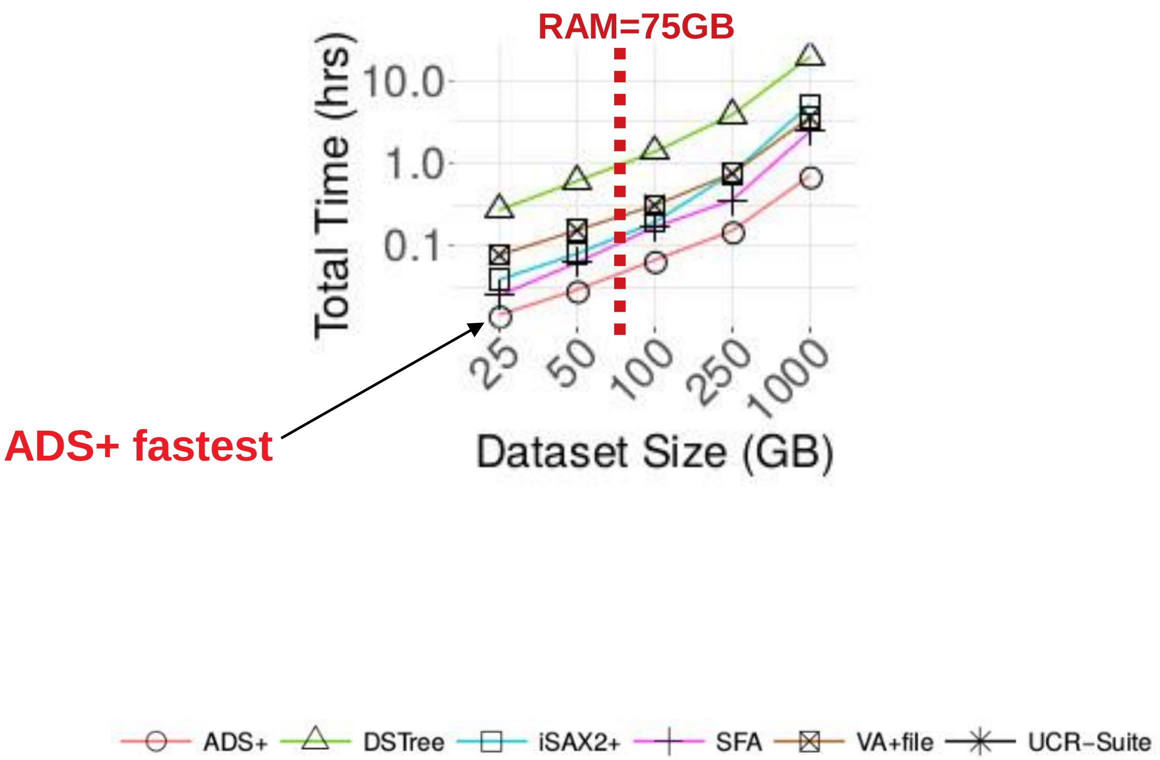
Echihabi-PVLDB'18

- 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

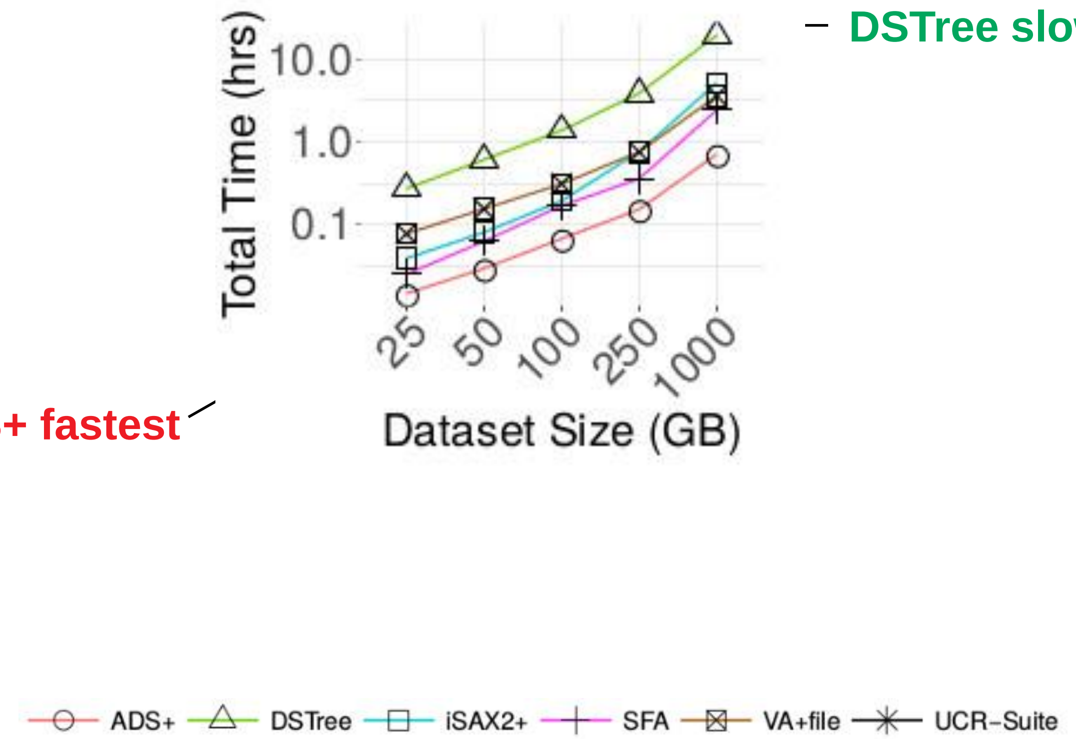
Time for Indexing (Idx) vs. Dataset Size



Time for Indexing (Idx) vs. Dataset Size



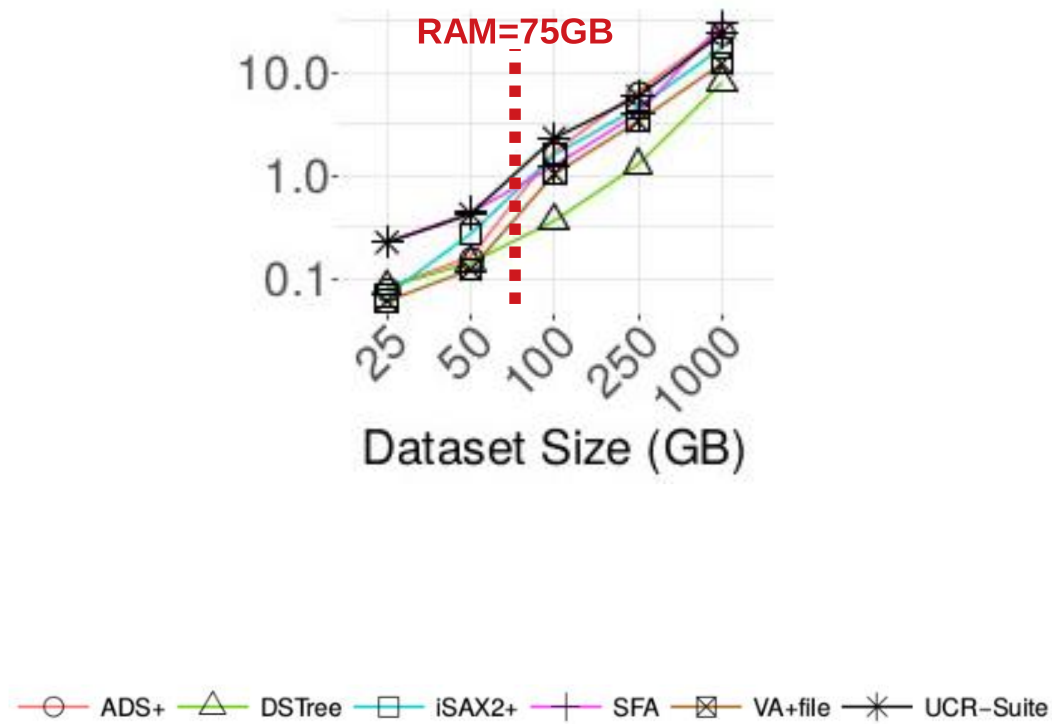
Time for Indexing (Idx) vs. Dataset Size



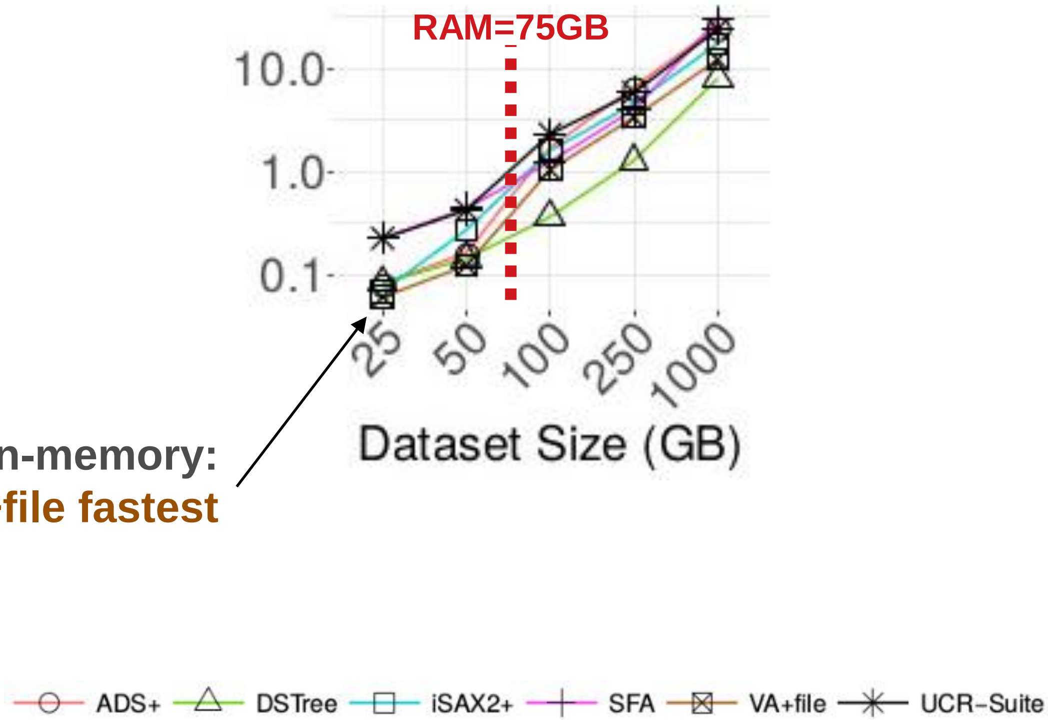
– DSTree slowest

ADS+ fastest

Time for 100 Exact Queries vs. Dataset size

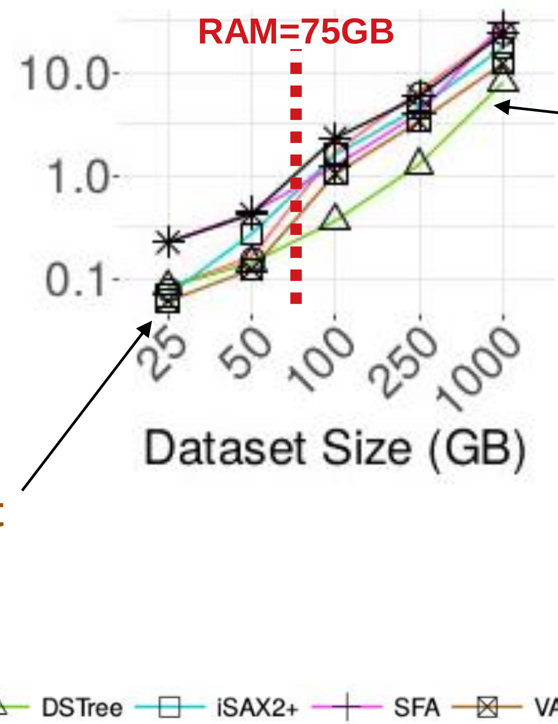


Time for 100 Exact Queries vs. Dataset size



In-memory:
VA+file fastest

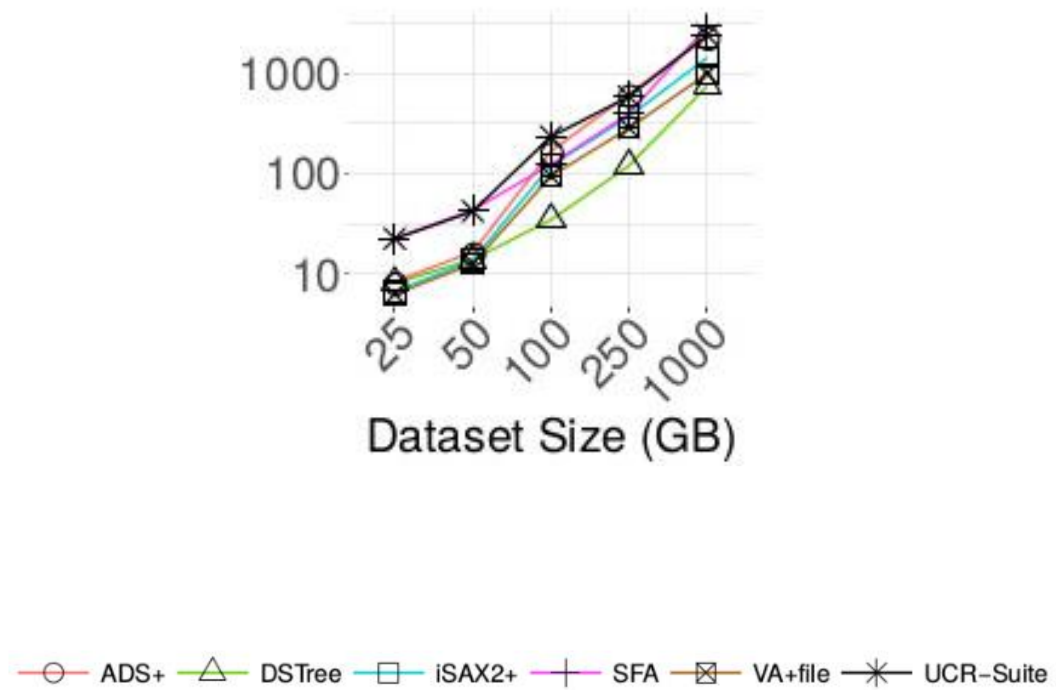
Time for 100 Exact Queries vs. Dataset size



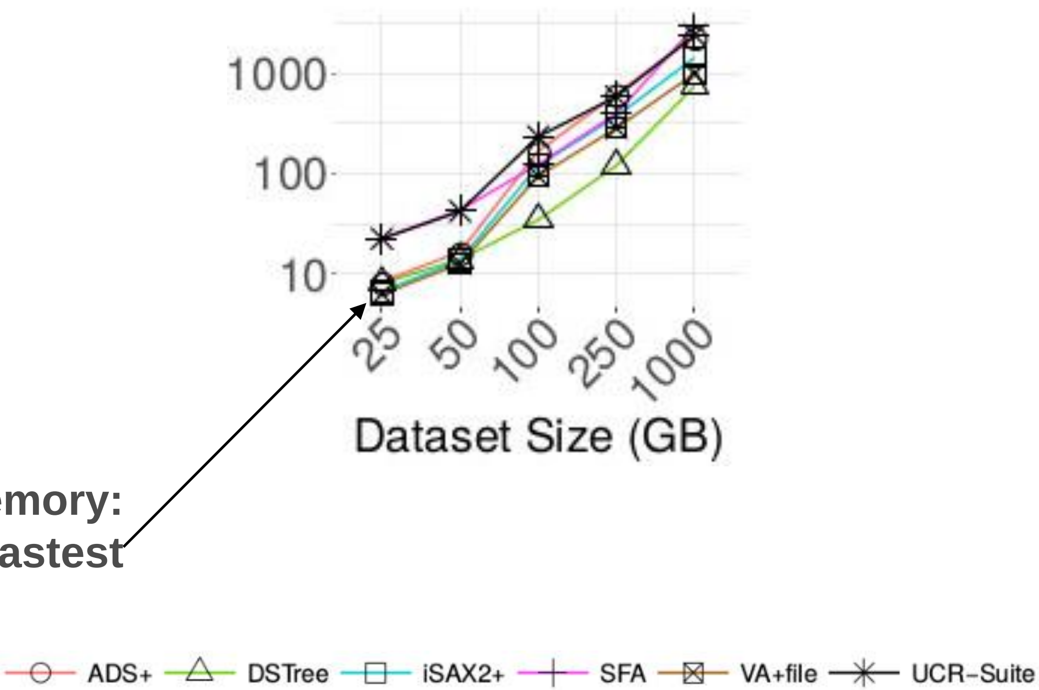
disk:
DSTree fastest

In-memory:
VA+file fastest

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

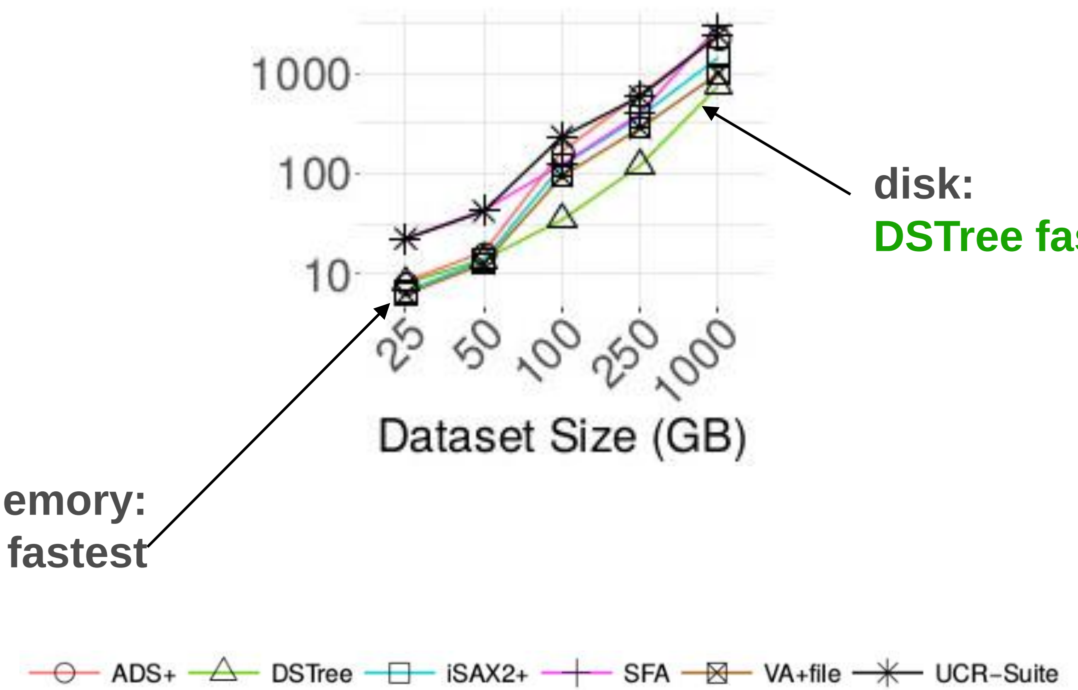


Time for Idx + 10K Exact Queries vs. Dataset size



In-memory:
VA+file fastest

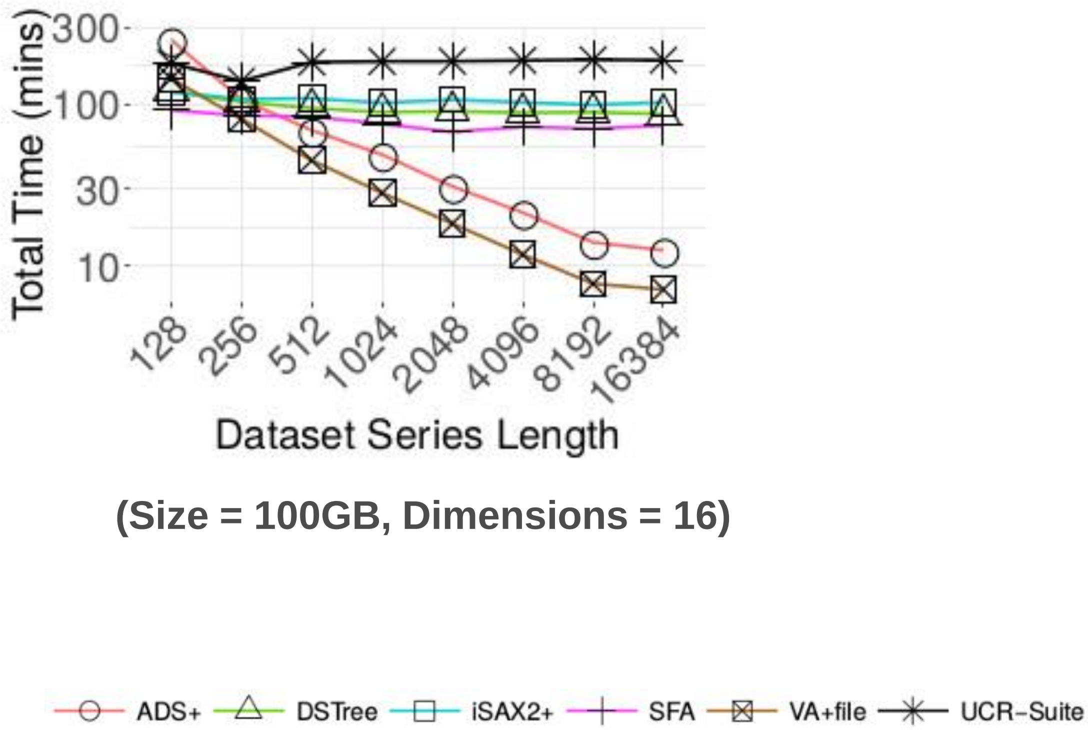
Time for Idx + 10K Exact Queries vs. Dataset size



disk:
DSTree fastest

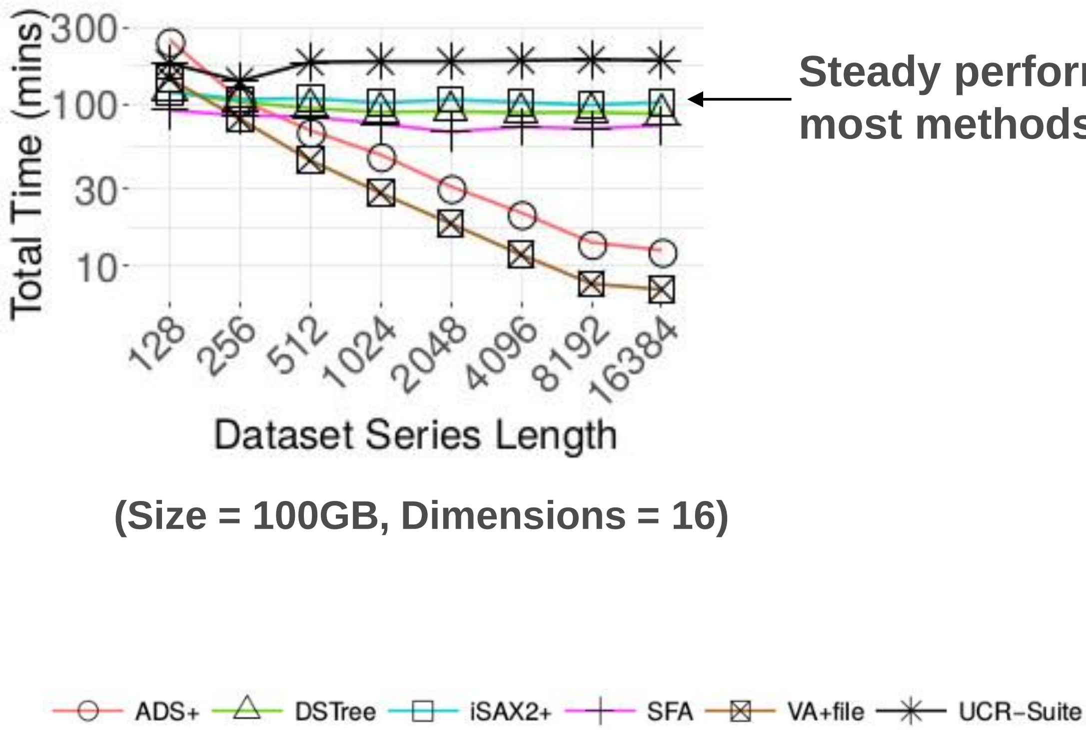
In-memory:
VA+file fastest

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



(Size = 100GB, Dimensions = 16)

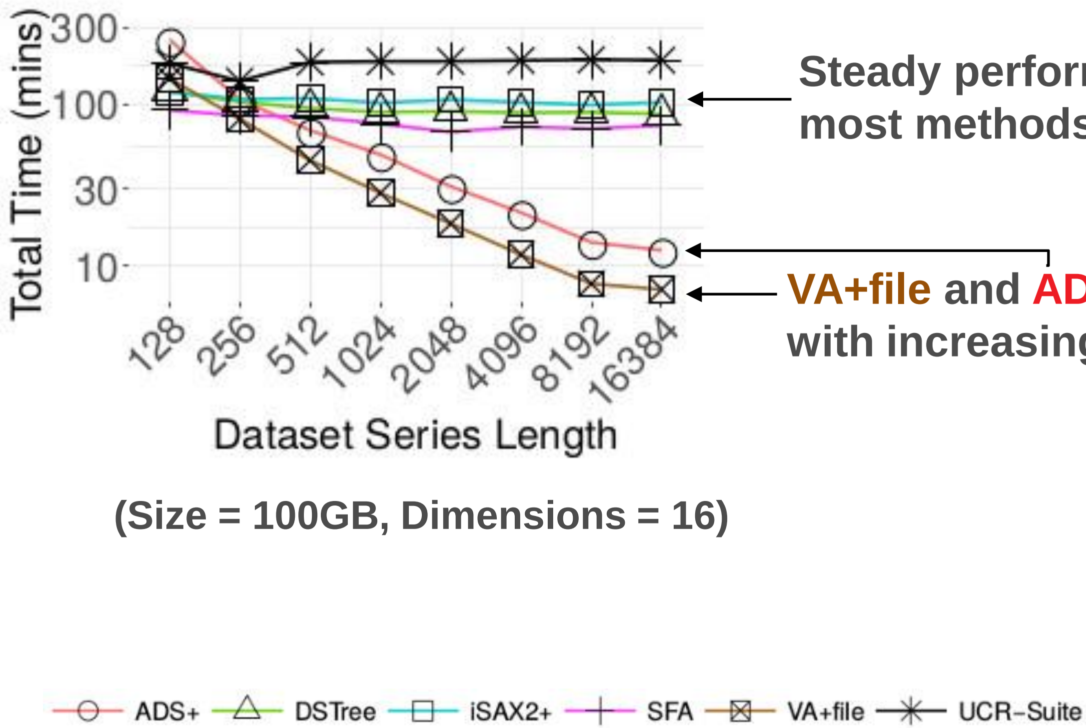
Time for Idx + 10K Exact Queries vs. Series Length



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)

Publications

Echihabi-PVLDB'18

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
  - New 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

Publications

Echihabi-PVLDB'18

TLB does not always predict performance

Publications

Echihabi-PVLDB'18

TLB does not always predict performance

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

Publications

Echihabi-PVLDB'18

TLB does not always predict performance

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

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

Publications

Echihabi-PVLDB'18

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

TLB does not always predict performance

Publications

Echihabi-PVLDB'18

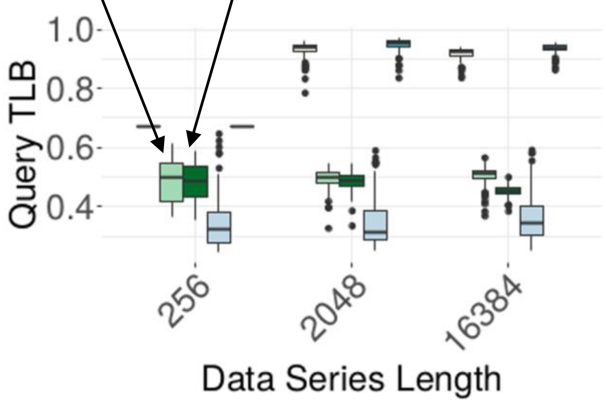
The TLB measures the quality of a summarization (higher is better)

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

worst

best

DSTree and **iSAX2+** have similar TLB



TLB does not always predict performance

Publications

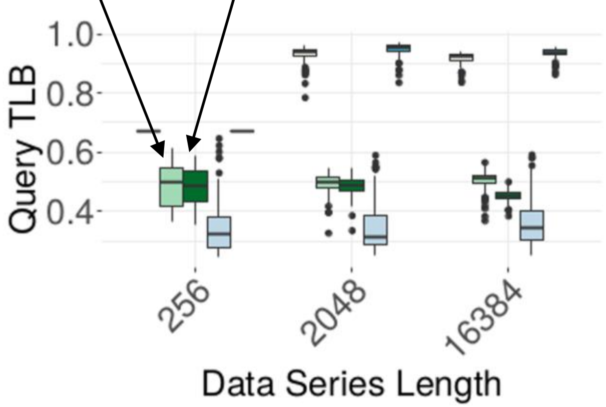
Echihabi-PVLDB'18

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

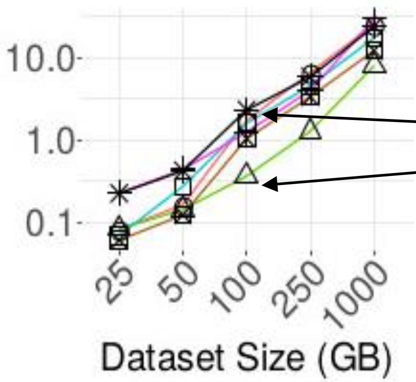
$$\mathbf{0} \leq \quad \text{TLB} = \frac{dist(Query, candidate)\ in\ reduced\ space}{dist(Query, candidate)\ in\ original\ space} \leq \mathbf{1}$$

worst best

DSTree and **iSAX2+** have similar TLB



YET



iSAX2+ 5x slower than **DSTree**

Publications

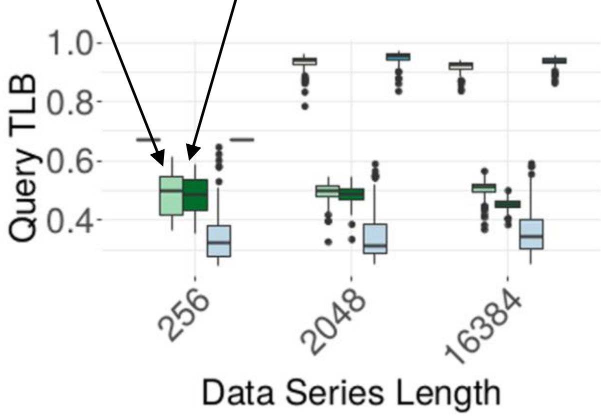
Echihabi-PVLDB'18

TLB does not always predict performance

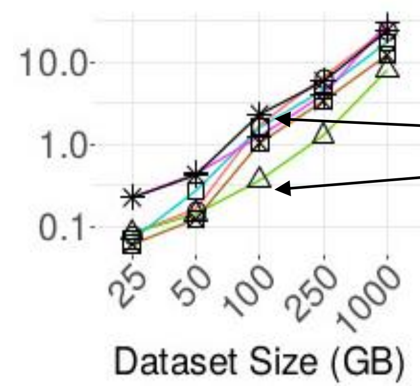
The TLB measures the quality of a summarization (higher is better)

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

DSTree and iSAX2+ have similar TLB



YET



iSAX2+ 5x slower than DSTree

No bias, same data and same implementation framework

Publications

Echihabi-PVLDB'18

Insights

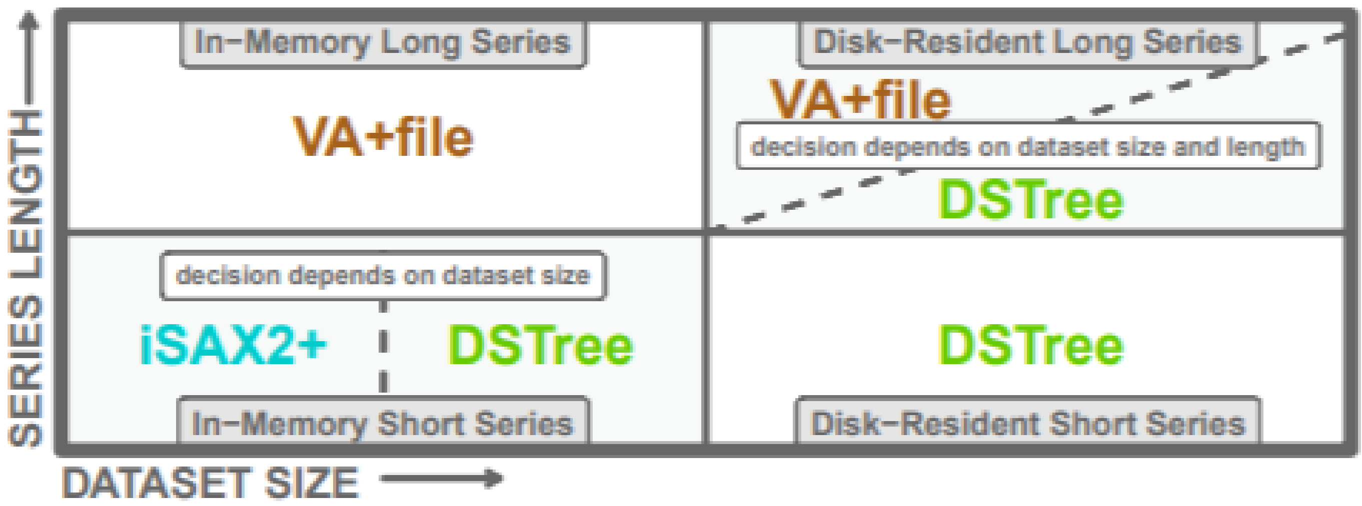
- Results are sensitive to:
  - Parameter tuning
  - Hardware setup
  - Implementation
  - Workload selection
- Results identify methods that would benefit from modern hardware

Recommendations

Publications

Echihabi-PVLDB'18

Scenario: Indexing and answering 10K exact queries on HDD



Experimental Comparisons: Approximate Query Answering

Experimental Framework

Publications

Echihabi-PVLDB'19

- Datasets
  - In-memory and disk-based datasets
  - Synthetic data modeling financial time series
  - Four real datasets from deep learning, computer vision, seismology, and neuroscience (25GB-250GB)
- Query Workloads
  - 100 – 10,000 kNN queries k in [1,100]
  - ng-approximate and δ-ε-approximate queries (exact queries used as yardstick)
- C/C++ methods (3 methods reimplemented from scratch)
- Performance measures
  - Efficiency: time, throughput, #disk accesses, % of data accessed
  - Accuracy: average recall, mean average precision, mean relative error
- Procedure:
  - Step 1: Parametrization
  - Step 2: Evaluation of indexing/query answering scalability in-memory
  - Step 3: Evaluation of indexing/query answering scalability on-disk
  - Step 4: Additional experiments with best-performing methods on disk

Approximate Methods Covered in Study

		Matching Accuracy				Representation		Implementation		
		exact	ng-appr.	ϵ-appr.	δ-ϵ-appr.	Raw	Reduced	Original	New	Disk-resident Data
Graphs	HNSW		[99]			✓		C++		
	NSG		[58]			✓		C++		

Approximate Methods Covered in Study

		Matching Accuracy				Representation		Implementation		
		exact	ng-appr.	ϵ-appr.	δ-ϵ-appr.	Raw	Reduced	Original	New	Disk-resident Data
Graphs	HNSW		[99]			✓		C++		
	NSG		[58]			✓		C++		
Inv. Indexes	IMI		[16, 60]				OPQ	C++		✓

Approximate Methods Covered in Study

		Matching Accuracy				Representation		Implementation		
		exact	ng-appr.	ϵ-appr.	δ-ϵ-appr.	Raw	Reduced	Original	New	Disk-resident Data
Graphs	HNSW		[99]			✓		C++		
	NSG		[58]			✓		C++		
Inv. Indexes	IMI		[16, 60]				OPQ	C++		✓
LSH	QALSH				[69]		Signatures	C++		
	SRS				[136]		Signatures	C++		

Approximate Methods Covered in Study

		Matching Accuracy				Representation		Implementation		
		exact	ng-appr.	ϵ-appr.	δ-ϵ-appr.	Raw	Reduced	Original	New	Disk-resident Data
Graphs	HNSW		[99]			✓		C++		
	NSG		[58]			✓		C++		
Inv. Indexes	IMI		[16, 60]				OPQ	C++		✓
LSH	QALSH				[69]		Signatures	C++		
	SRS				[136]		Signatures	C++		
Scans	VA+file	[55]	•	•	•		DFT	MATLAB	C	✓

- **Our extensions**

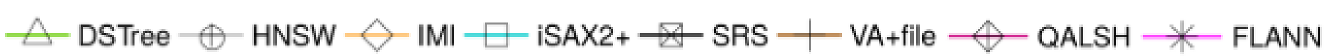
Approximate Methods Covered in Study

		Matching Accuracy				Representation		Implementation		
		exact	ng-appr.	ϵ-appr.	δ-ϵ-appr.	Raw	Reduced	Original	New	Disk-resident Data
Graphs	HNSW		[99]			✓		C++		
	NSG		[58]			✓		C++		
Inv. Indexes	IMI		[16, 60]				OPQ	C++		✓
LSH	QALSH				[69]		Signatures	C++		
	SRS				[136]		Signatures	C++		
Scans	VA+file	[55]	•	•	•		DFT	MATLAB	C	✓
Trees	Flann		[107]			✓		C++		
	DSTree	[146]	[146]	•	•		EAPCA	Java	C	✓
	HD-index		[11]				Hilbert keys	C++		✓
	iSAX2+	[30]	[30]	•	•		iSAX	C#	C	✓

- **Our extensions**

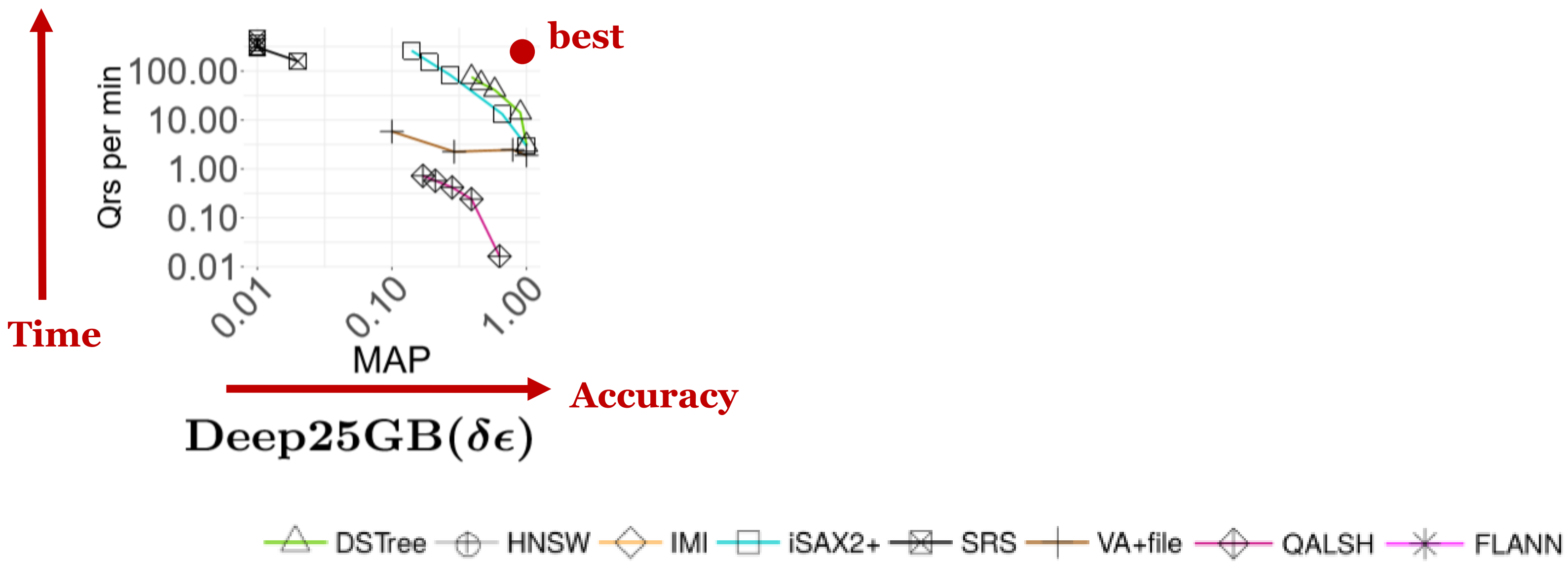
Unexpected Results

- New data series extensions are the overall winners even for general high-d vectors
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search)



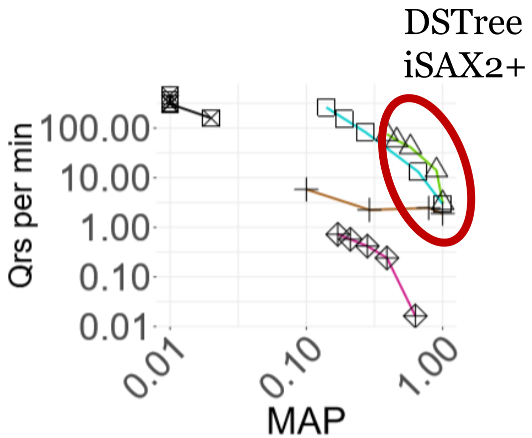
Unexpected Results

- New data series extensions are the overall winners even for general high-d vectors
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search)



Unexpected Results

- New data series extensions are the overall winners even for general high-d vectors
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory

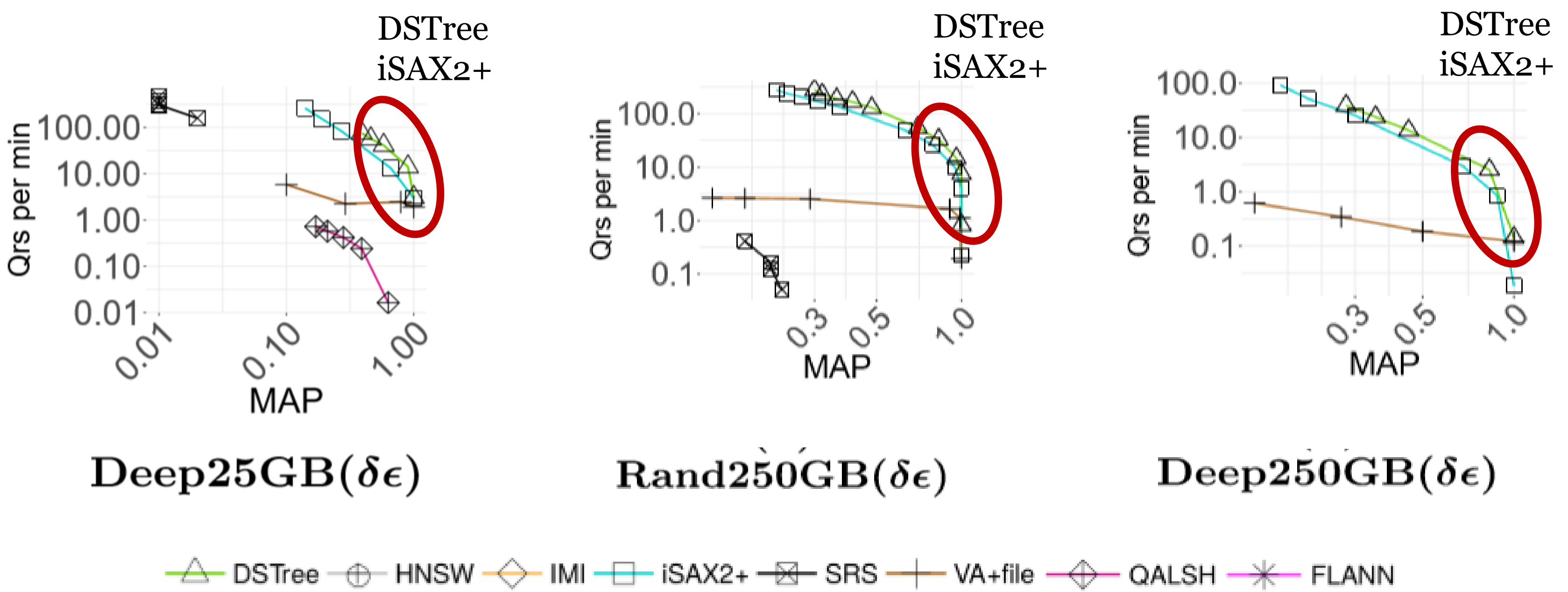


Deep25GB($\delta\epsilon$)

DSTree HNSW IMI iSAX2+ SRS VA+file QALSH FLANN

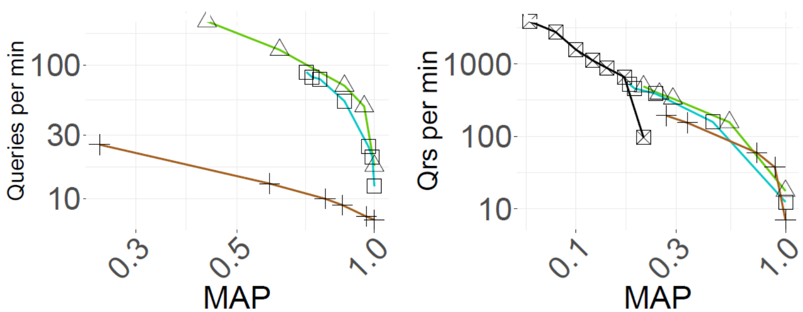
Unexpected Results

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



Unexpected Results

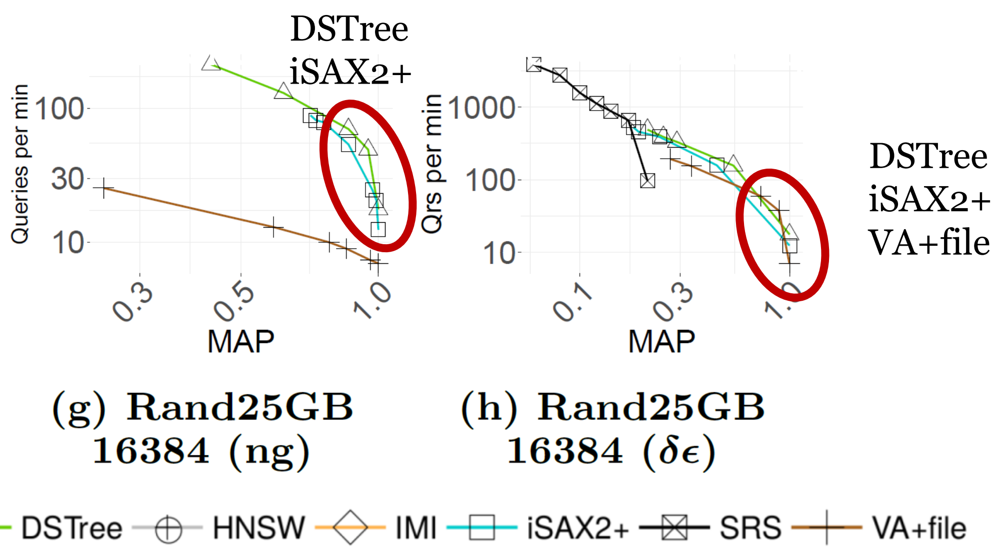
- Our new extensions are the overall winners even for general high-d vectors
  - perform the best for approximate queries with probabilistic guarantees (δ-ε-approximate search), in-memory and on-disk
  - perform the best for long vectors



(g) Rand25GB 16384 (ng)   **(h) Rand25GB 16384 ($\delta\epsilon$)**

Unexpected Results

- **Our new extensions are the overall winners** even for general high-d vectors
  - 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

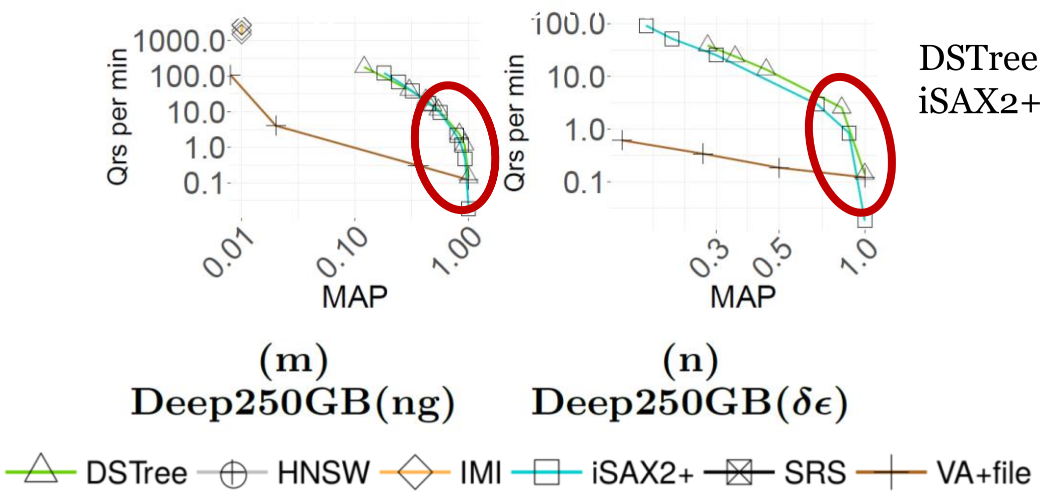


(g) Rand25GB 16384 (ng)

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

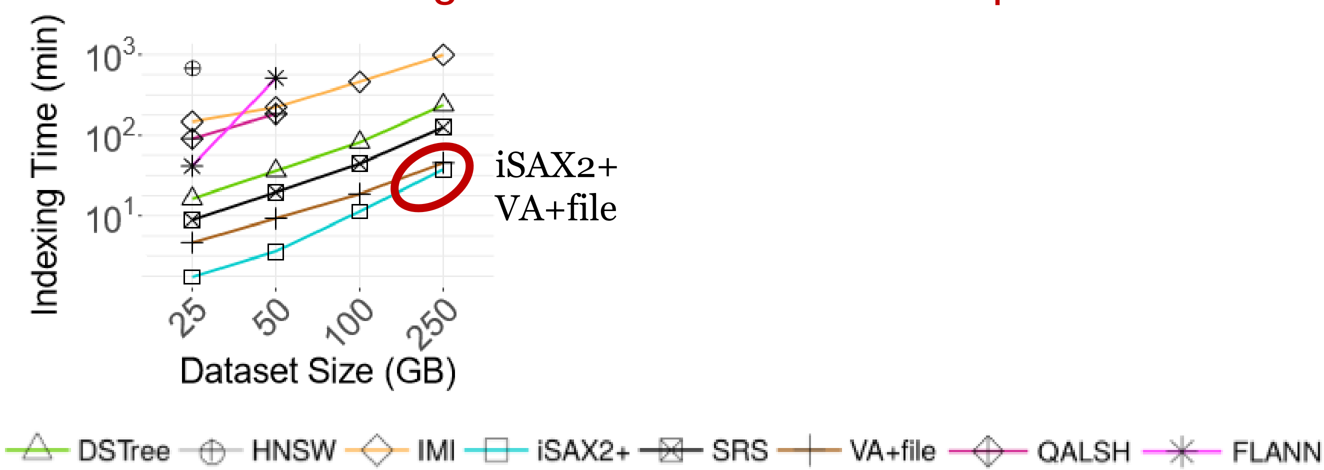
Unexpected Results

- Our new extensions are the overall winners even for general high-d vectors
  - 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



Unexpected Results

- New data series extensions are the overall winners even for general high-d vectors
  - 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
  - are fastest at indexing and have the lowest footprint



Insights

Exciting research direction for approximate similarity search in high-d spaces:

Insights

Exciting research direction for approximate similarity search in high-d spaces:

Currently two main groups of solutions exist:

approximate search solutions
without guarantees
relatively efficient

approximate search solutions
with guarantees
relatively slow

Insights

Exciting research direction for approximate similarity search in high-d spaces:

Currently two main groups of solutions exist:

approximate search solutions	approximate search solutions
without guarantees	with guarantees
relatively efficient	relatively slow

Insights

Exciting research direction for approximate similarity search in high-d spaces:

Currently two main groups of solutions exist:

approximate search solutions	approximate search solutions
without guarantees	with guarantees
relatively efficient	relatively slow

We show that it is possible to have efficient approximate algorithms with guarantees

Insights

Approximate state-of-the-art techniques for high-d vectors are not practical:

Insights

Approximate state-of-the-art techniques for high-d vectors are not practical:

- LSH-based techniques
  - slow, high-footprint, low accuracy (recall/MAP)

Insights

Approximate state-of-the-art techniques for high-d vectors are not practical:

- LSH-based techniques
  - slow, high-footprint, low accuracy (recall/MAP)
- kNNG-based techniques
  - slow indexing, difficult to tune, in-memory, no guarantees

Insights

Approximate state-of-the-art techniques for high-d vectors are not practical:

- LSH-based techniques
  - slow, high-footprint, low accuracy (recall/MAP)
- kNNG-based techniques
  - slow indexing, difficult to tune, in-memory, no guarantees
- Quantization-based techniques
  - slow indexing, difficult to tune, no guarantees

Insights

Approximate state-of-the-art techniques for high-d vectors are not practical:

- LSH-based techniques
  - slow, high-footprint, low accuracy (recall/MAP)
- kNNG-based techniques
  - slow indexing, difficult to tune, in-memory, no guarantees
- Quantization-based techniques
  - slow indexing, difficult to tune, no guarantees

All suffer a serious limitation: accuracy determined during <u>index-building</u> & query answering

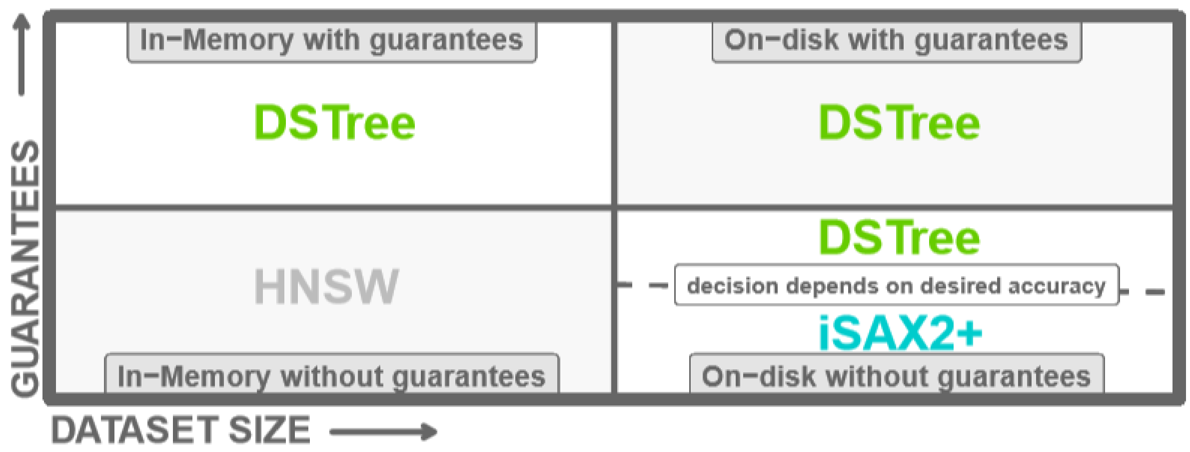
Recommendations for approx. techniques

Data series approaches are the overall winners!

The only exception is HNSW for in-memory ng-approximate queries using an existing index

Recommendations

Scenario: Answering a query workload using an existing index



Questions?

High-d Similarity Search: Challenges and Open Problems

Challenges and Open Problems

- we are still far from having solved the problem

- several challenges remain in terms of
  - usability, ease of use
  - scalability, distribution
  - benchmarking

- these challenges derive from modern data science applications

Challenges and Open Problems Outline

- benchmarking
- interactive analytics
- parallelization and distribution
- deep learning

Massive High-d Data Collections

Publications

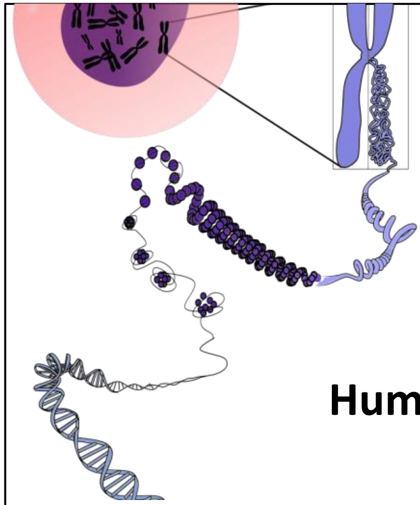
Palpanas-SIGREC'19

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

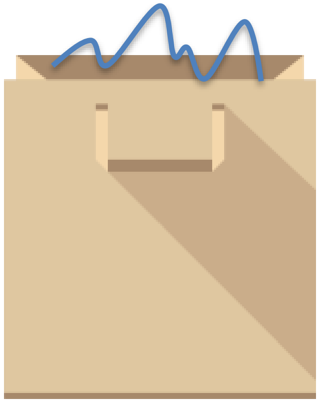
Challenges and Open Problems Outline

- benchmarking
- interactive analytics
- parallelization and distribution
- deep learning

Previous Studies

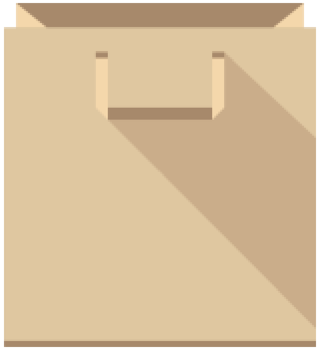
evaluate **performance** of **indexing methods** using **random queries**

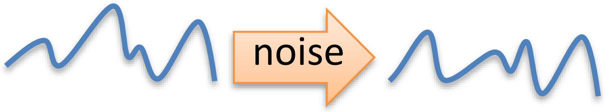
- chosen from the data (with/without noise)



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

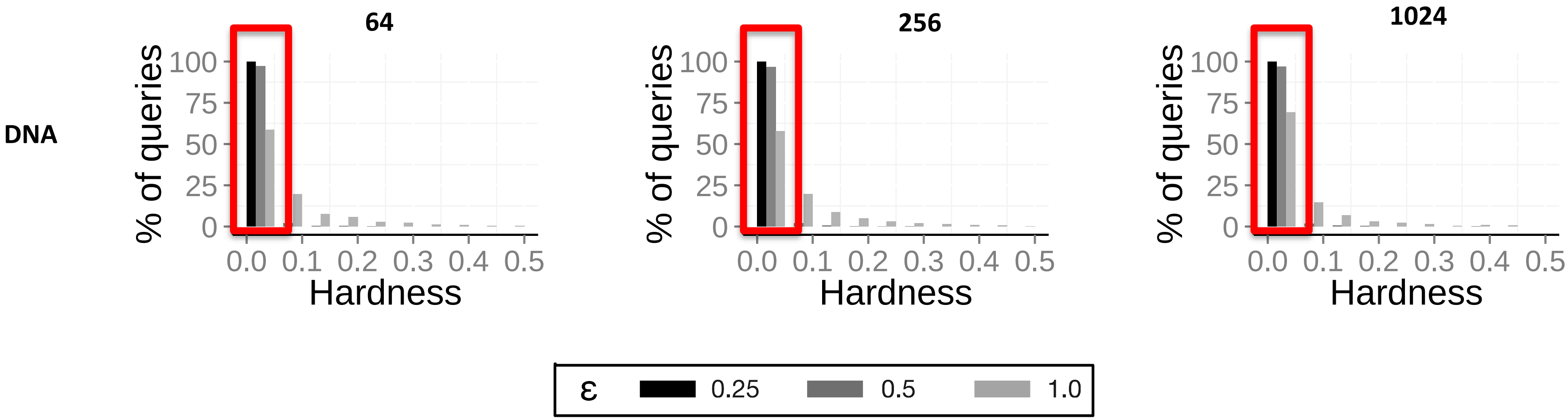
Publications

Zoumbatianos KDD '15

Zoumbatianos TKDE '18

Previous Workloads

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



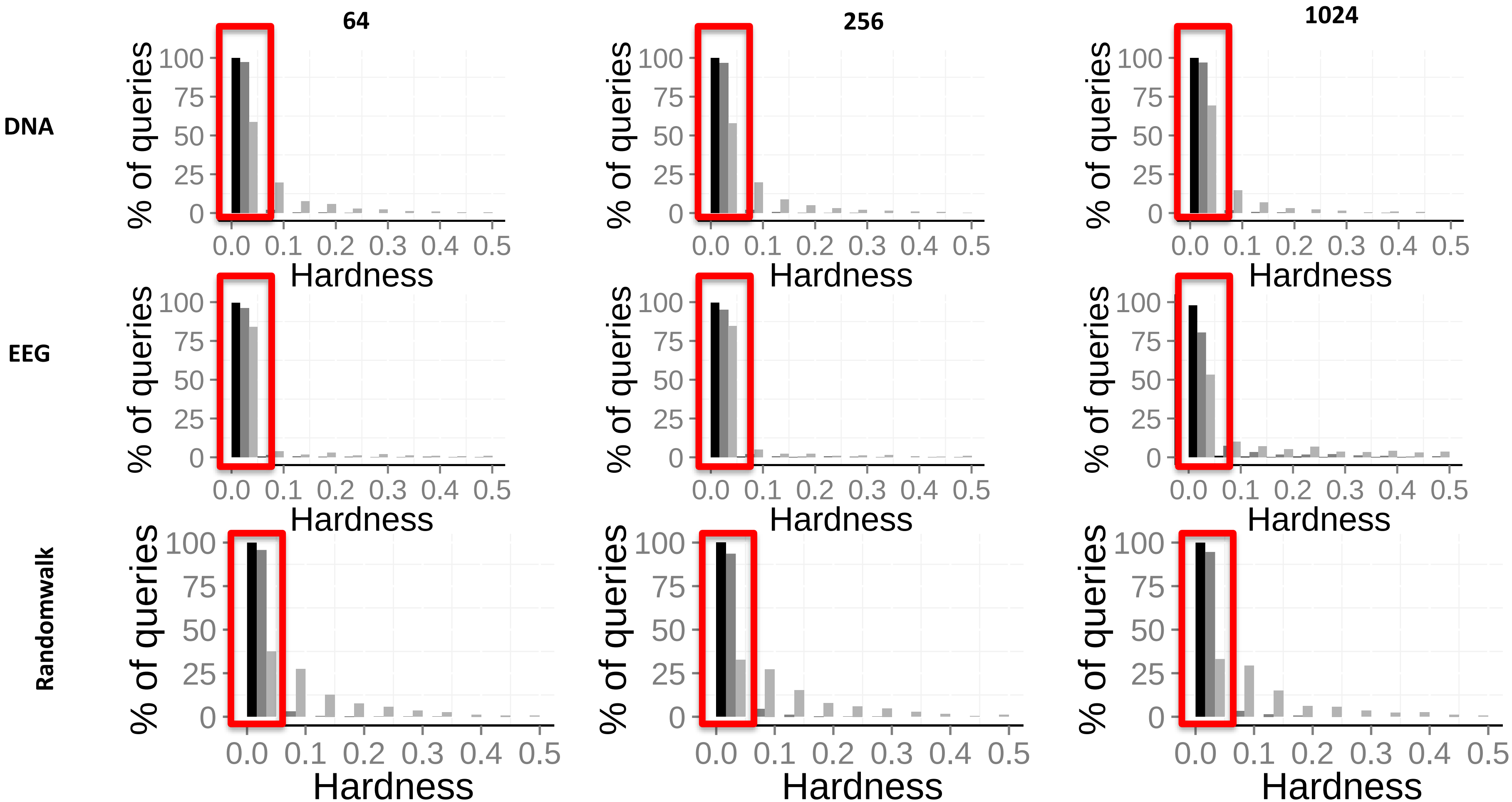
Previous Workloads

Publications

Zoumbatianos KDD '15

Zoumbatianos TKDE '18

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



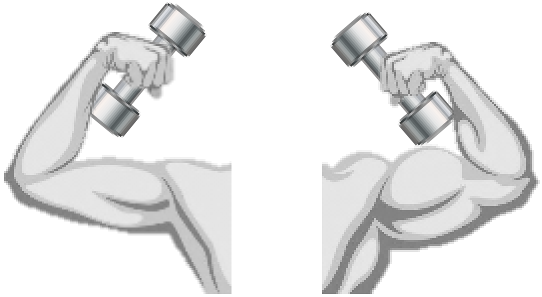
Publications

Zoumbatianos KDD '15

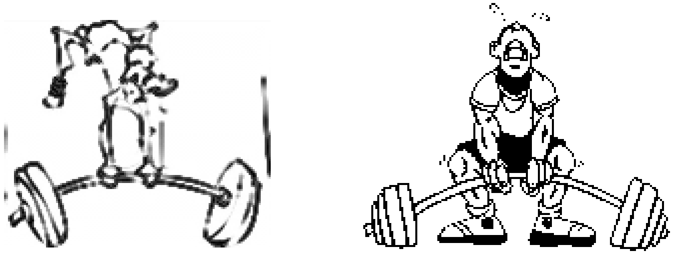
Zoumbatianos TKDE '18

Benchmark Workloads

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



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



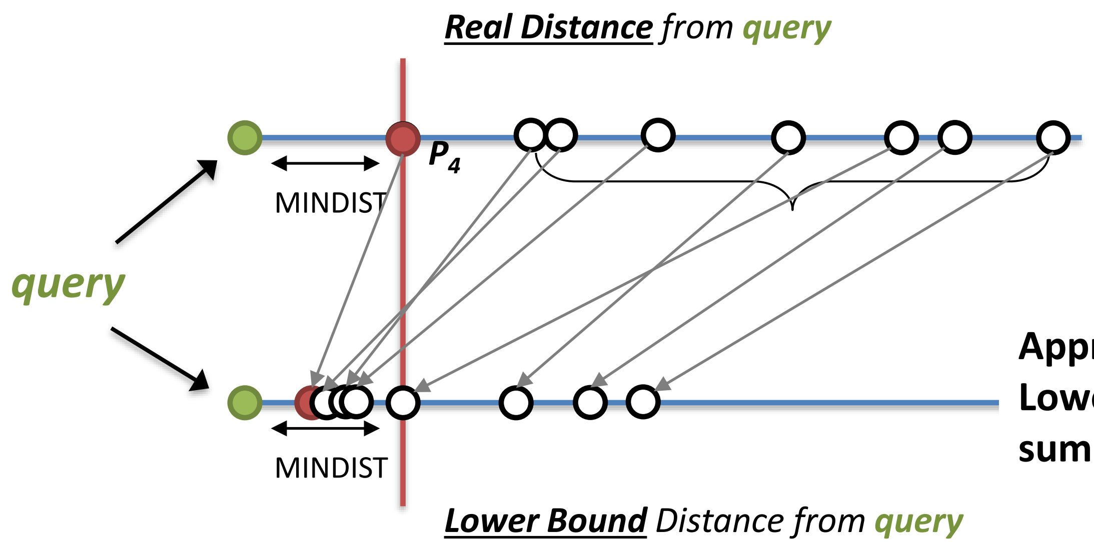
need **methods** for **generating** queries of **varying hardness**

Publications

Zoumbatianos KDD ‘15

Zoumbatianos TKDE ‘18

Characterizing Queries



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

Publications

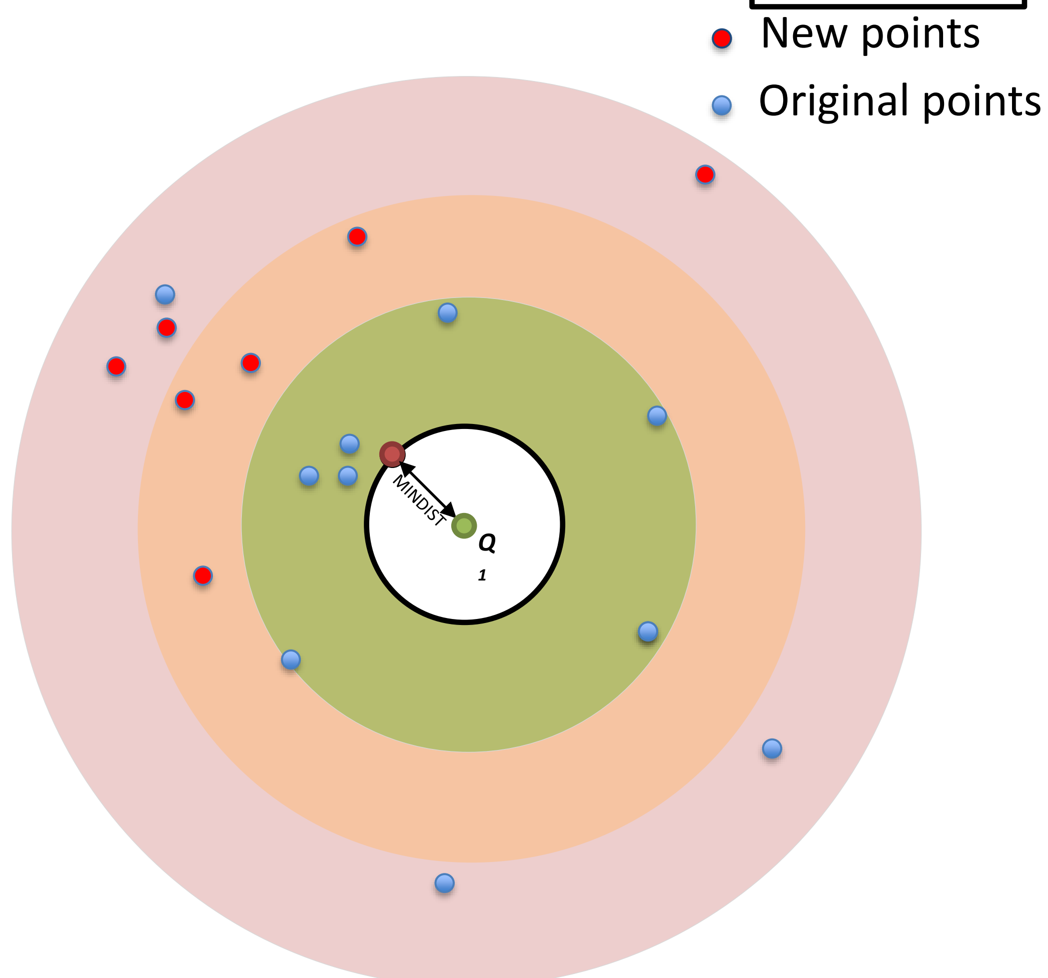
Zoumbatianos KDD ‘15

Zoumbatianos TKDE ‘18

Densification Method:
Equi-densification

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

Equal number of points in every “zone”



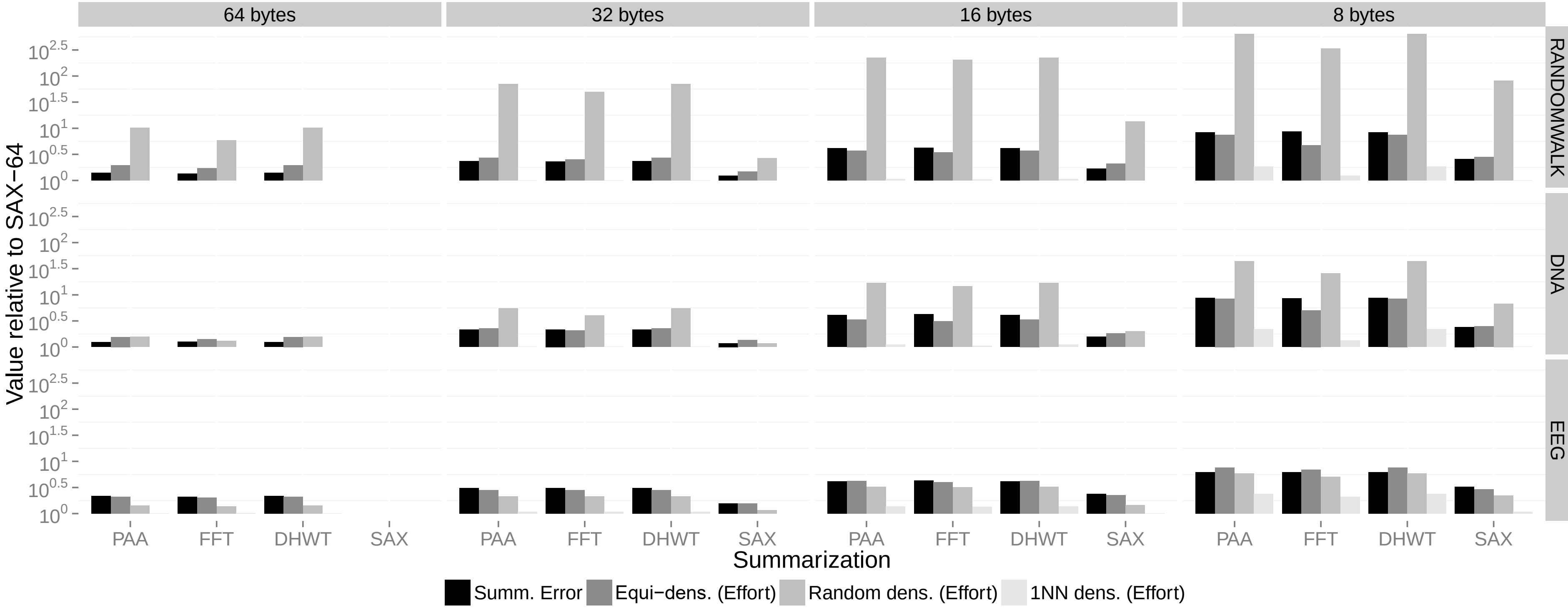
Experiments
Densification Methods

Publications
Zoumbatianos KDD '15
Zoumbatianos TKDE '18

Using all datasets of 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



Experiments

Densification Methods

Publications

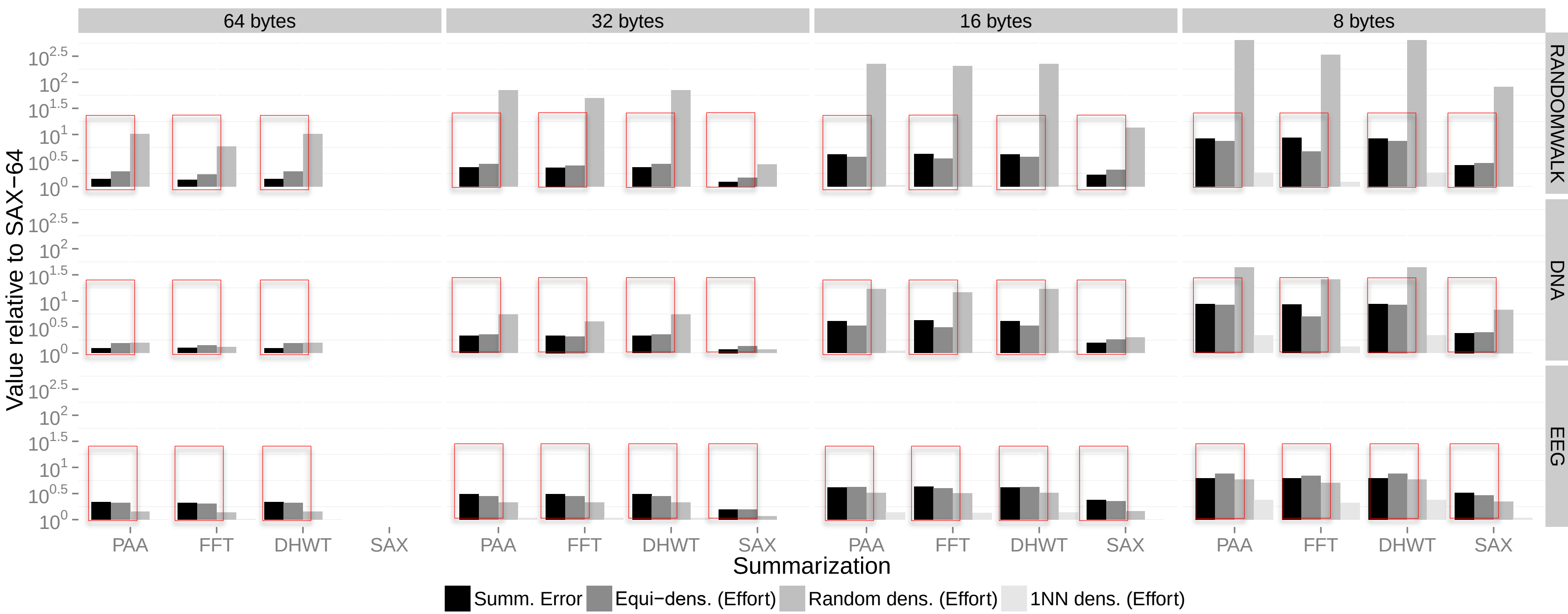
Zoumbatianos KDD '15

Zoumbatianos TKDE '18

For **equi-densification**

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

The worse a summarization the bigger effort it does



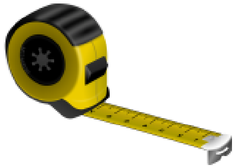
Publications

Zoumbatianos KDD ‘15

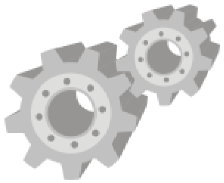
Zoumbatianos TKDE ‘18

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

Challenges and Open Problems Outline

- benchmarking
- interactive analytics
- parallelization and distribution
- deep learning

Interactive Analytics?

- analytics over high-d data 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

Gogolou-BigVis'19

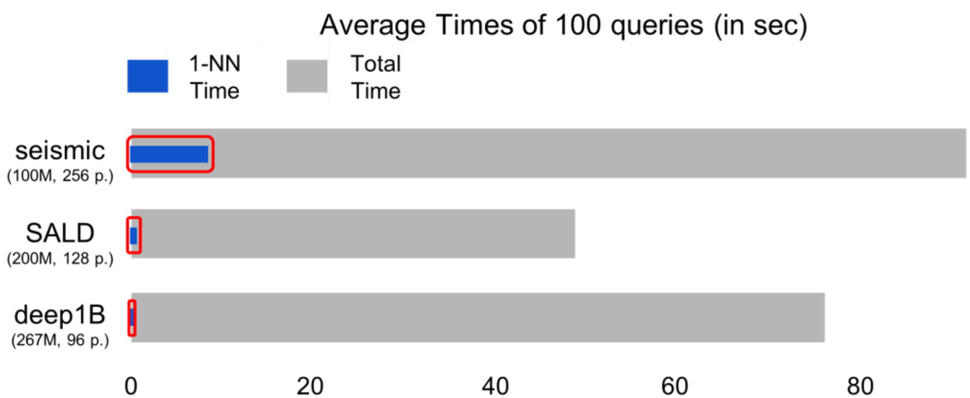
- interaction with users offers new opportunities
  - progressive answers
    - produce intermediate results
      - iteratively converge to final, correct solution

Need for Interactive Analytics

Publications

Gogolou-BigVis'19

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

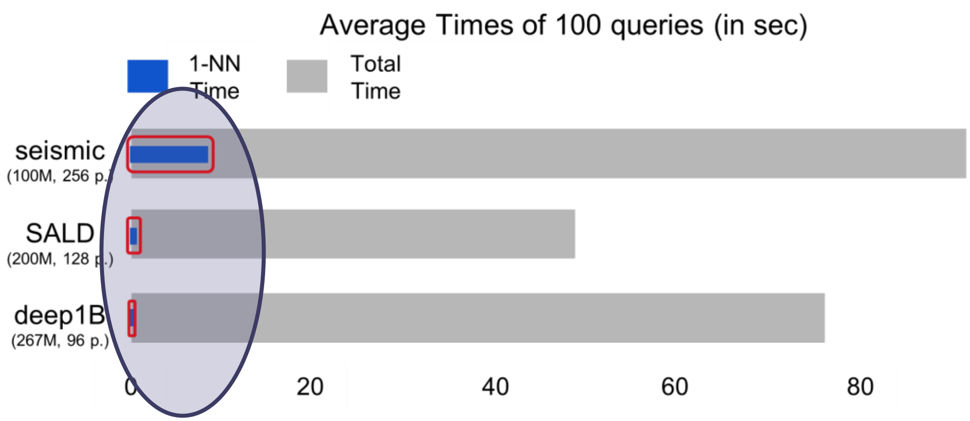


Need for Interactive Analytics

Publications

Gogolou-BigVis'19

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

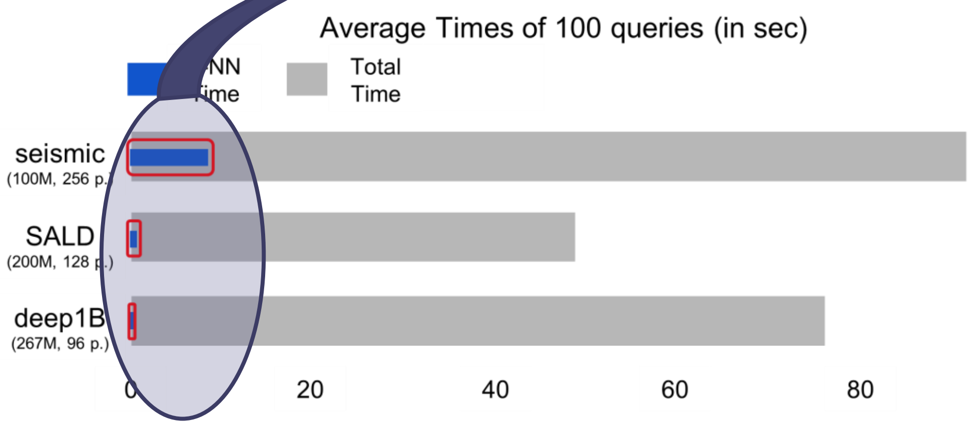


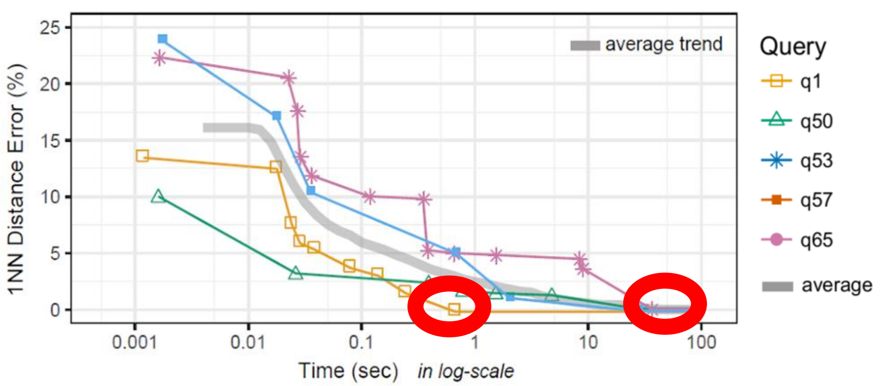
Need for Interactive Analytics

Publications

Gogolou-BigVis'19

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





Need for Interactive Analytics

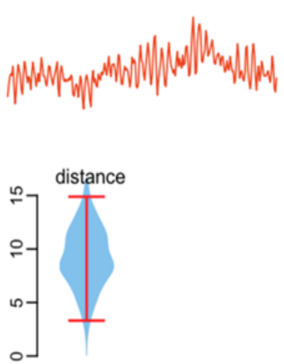
Publications

Gogolou-BigVis'19

Gogolou-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

Query & Initial Estimate



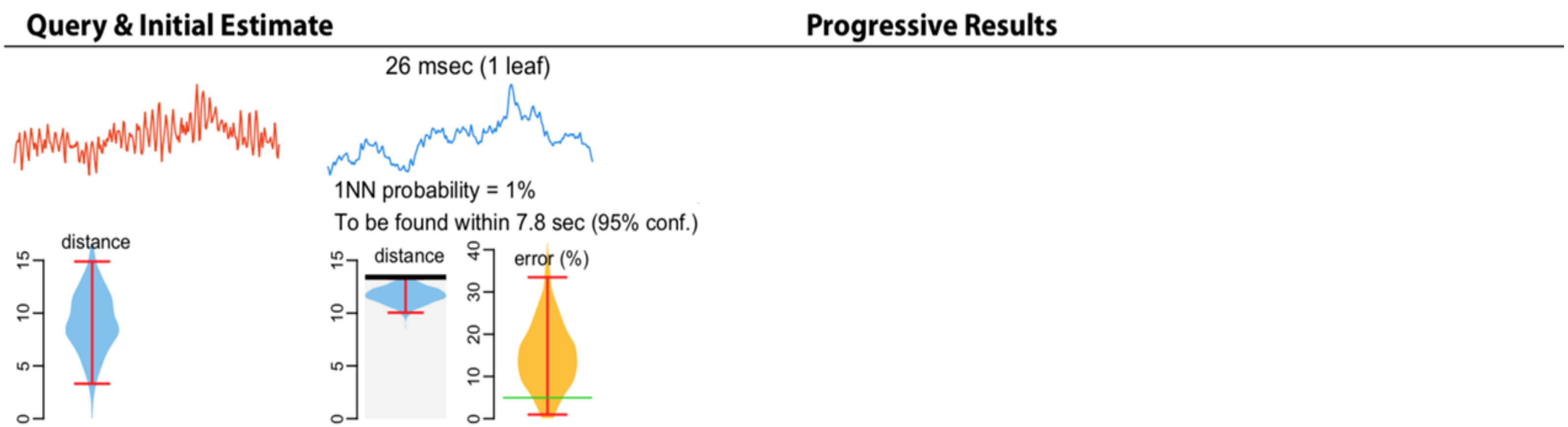
Need for Interactive Analytics

Publications

Gogolou-BigVis'19

Gogolou-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



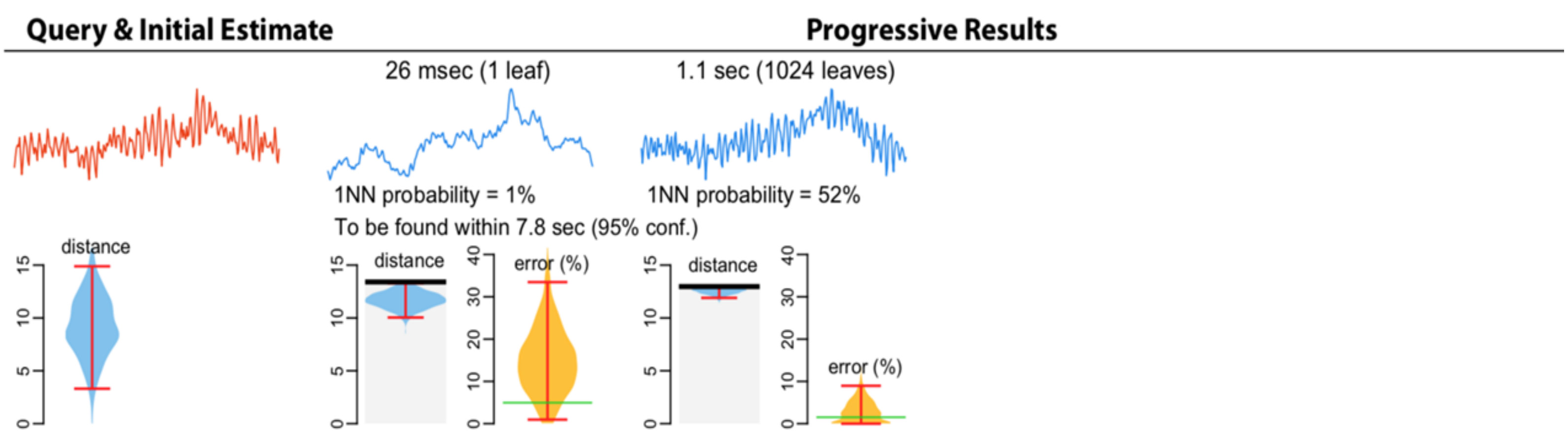
Need for Interactive Analytics

Publications

Gogolou-BigVis'19

Gogolou-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



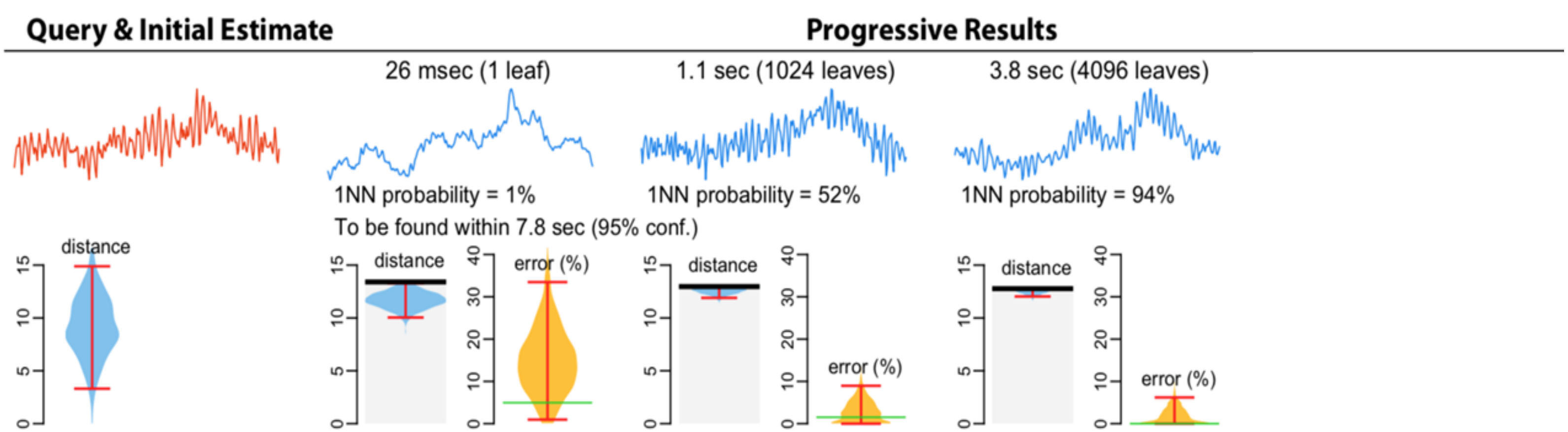
Need for Interactive Analytics

Publications

Gogolou-BigVis'19

Gogolou-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



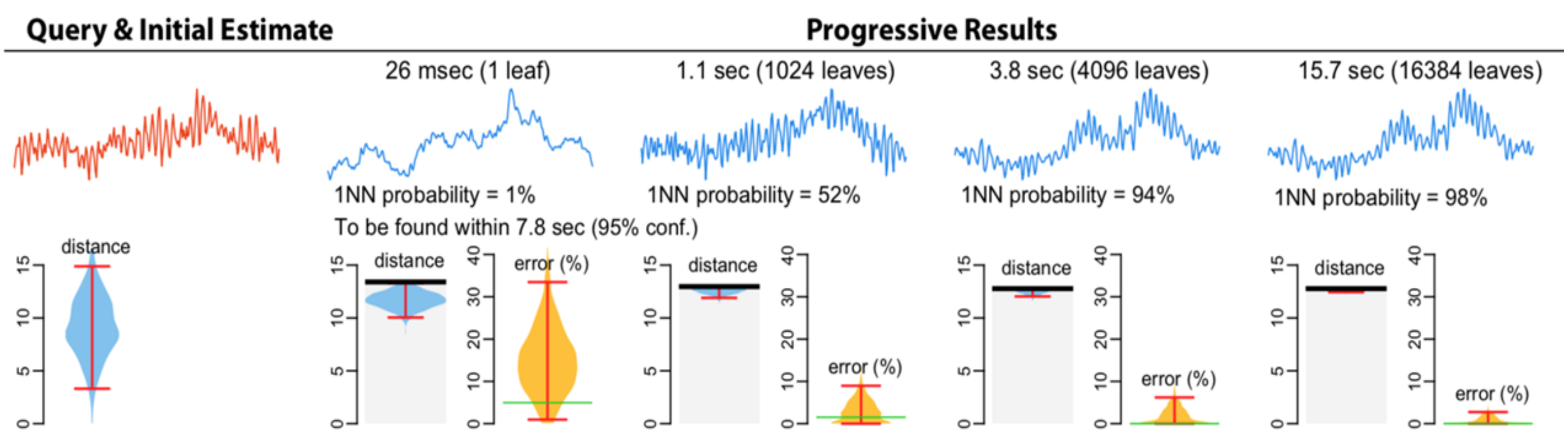
Need for Interactive Analytics

Publications

Gogolou-BigVis'19

Gogolou-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



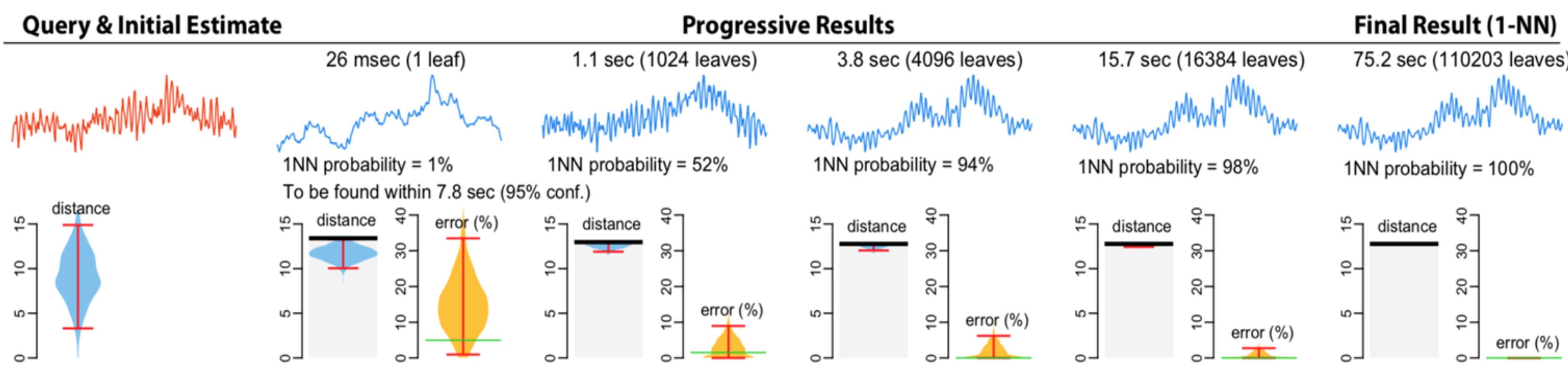
Need for Interactive Analytics

Publications

Gogolou-BigVis'19

Gogolou-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

- 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

Gogolou-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

Challenges and Open Problems Outline

- benchmarking
- interactive analytics
- parallelization and distribution
- deep learning

Publications

Palpanas-HPCS'17

Need for Parallelization/Distribution

- 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

Palpanas-HPCS'17

Need for Parallelization/Distribution

- further scale-up and scale-out possible!
  - techniques inherently parallelizable
    - across cores, across machines
- need to
  - propose methods for concurrent query answering
  - combine multi-core and distributed methods
  - examine FPGA and NVM technologies
- more involved solutions required when optimizing for energy
  - reducing execution time is relatively easy
  - minimizing total work (energy) is more challenging

Challenges and Open Problems Outline

- benchmarking
- interactive analytics
- parallelization and distribution
- deep learning

Connections to Deep Learning

- data series indexing for deep embeddings

Connections to Deep Learning

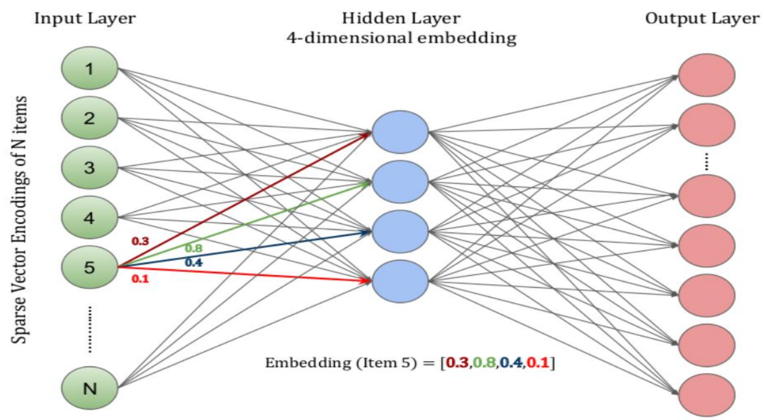
- data series indexing for deep embeddings

sequences
text
images
video
graphs
...

Connections to Deep Learning

- data series indexing for deep embeddings

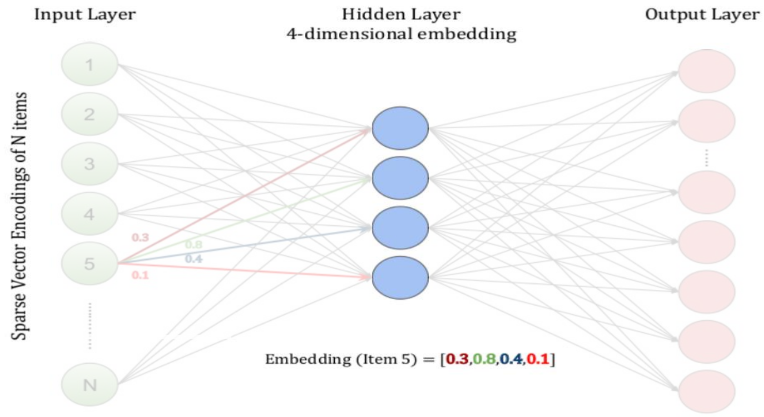
sequences
text
images
video
graphs
...



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 for high-d vectors
  - Different summarization for different high-d data types
  - 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 for high-d vectors
  - Different summarization for different high-d data types
  - 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 for high-d vectors
  - Different summarization for different high-d data types
  - 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

Overall Conclusions

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

Overall Conclusions

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

Overall Conclusions

- High-d data is a very common data type
    - across several different domains and applications
- Complex analytics on high-d data are challenging
    - have very high complexity
    - efficiency comes from data series management/indexing techniques
- Several exciting research opportunities

thank you!

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

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

References (chronological order)

- Delaunay, Boris (1934). "Sur la sphère vide". Bulletin de l'Académie des Sciences de l'URSS, Classe des Sciences Mathématiques et Naturelles. **6**: 793–800.
- Ramer, U. (1972). An iterative procedure for the polygonal approximation of planar curves. *Computer Graphics and Image Processing*. 1: pp. 244-256.
- 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.
- Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. Commun. ACM 18, 9 (Sept. 1975), 509–517.
- Pavlidis, T. (1976). Waveform segmentation through functional approximation. *IEEE Transactions on Computers*.
- Godfried T. Toussaint, The relative neighbourhood graph of a finite planar set, Pattern Recognition, Volume 12, Issue 4, 1980, Pages 261-268,
- Ishijima, M., et al. (1983). Scan-Along Polygonal Approximation for Data Compression of Electrocardiograms. *IEEE Transactions on Biomedical Engineering*. BME-30(11):723-729.
- 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.
- C. Faloutsos, M. Ranganathan, & Y. Manolopoulos. Fast Subsequence Matching in Time-Series Databases. In Proc. ACM SIGMOD Int'l Conf. on Management of Data, pp 419–429, 1994.
- 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.
- Koski, A., Juhola, M. & Meriste, M. (1995). Syntactic Recognition of ECG Signals By Attributed Finite Automata. *Pattern Recognition*, 28 (12), pp. 1927-1940.
- Seshadri P., Livny M. & Ramakrishnan R. (1995): SEQ: A Model for Sequence Databases. ICDE 1995: 232-239
- 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*.

References (chronological order)

- 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.
- 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*.
- Piotr Indyk, Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. STOC 1998.
- 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*.
- 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.
- Hunter, J. & McIntosh, N. (1999). Knowledge-based event detection in complex time series data. *Artificial Intelligence in Medicine*. pp. 271-280. Springer.
- K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "'nearest neighbor'" meaningful? In *ICDT*, 1999.
- 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*.
- 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.
- J. Kleinberg. The Small-world Phenomenon: An Algorithmic Perspective. In Proceedings of the Thirty- second Annual ACM Symposium on Theory of Computing, STOC '00, pages 163–170, New York, NY, USA, 2000. ACM

References (chronological order)

- 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.
- 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*.
- 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.
- C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional spaces. In *ICDT*, 2001
- Ge, X. & Smyth P. (2001). Segmental Semi-Markov Models for Endpoint Detection in Plasma Etching. To appear in *IEEE Transactions on Semiconductor Engineering*.
- 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)
- Sivic and Zisserman, "Video Google: a text retrieval approach to object matching in videos," *Proceedings Ninth IEEE International Conference on Computer Vision*, Nice, France, 2003, pp. 1470-1477 vol.2
- 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.
- E. Keogh. Tutorial on Data Mining and Machine Learning in Time Series Databases. KDD 2004.
- Richard Cole, Dennis E. Shasha, Xiaojian Zhao: Fast window correlations over uncooperative time series. KDD 2005: 743-749
- 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)
- Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, Eamonn J. Keogh: Querying and mining of time series data: experimental comparison of representations and distance measures. Proc. VLDB Endow. 1(2): 1542-1552 (2008)
- Stephen Blott and Roger Weber. 2008. What's wrong with high-dimensional similarity search? Proc. VLDB Endow. 1, 1 (August 2008), 3.

References (chronological order)

- C. Silpa-Anan and R. Hartley, "Optimised KD-trees for fast image descriptor matching," 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 2008, pp. 1-8
- Alexandr Andoni and Piotr Indyk. 2008. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Commun. ACM 51, 1 (January 2008), 117–122.
- 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
- 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)
- Hervé Jégou, Matthijs Douze, Cordelia Schmid: Product Quantization for Nearest Neighbor Search. IEEE Trans. Pattern Anal. Mach. Intell. 33(1): 117-128 (2011)
- Wei Dong, Charikar Moses, and Kai Li. 2011. Efficient k-nearest neighbor graph construction for generic similarity measures. In Proceedings of the 20th international conference on World wide web (WWW '11).
- P. Schafer and M. Hogvist. Sfa: A symbolic fourier approximation and index for similarity search in high dimensional datasets. ICDE 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.
- J. He, S. Kumar, and S.-F. Chang. On the difficulty of nearest neighbor search. In *ICML*, 2012.

References (chronological order)

- Junhao Gan, Jianlin Feng, Qiong Fang, and Wilfred Ng. 2012. Locality-sensitive hashing scheme based on dynamic collision counting. In SIGMOD.
- Babenko, Artem & Lempitsky, Victor. (2012). The Inverted Multi-Index. IEEE Transactions on Pattern Analysis and Machine Intelligence. 37. 3069-3076.
- 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.
- 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
- 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
- Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov. Approximate nearest neighbor algorithm based on navigable small world graphs. Information Systems (IS), 45:61 – 68, 2014.
- NSW IS'14: Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov: Approximate nearest neighbor algorithm based on navigable small world graphs, Inf. Syst., vol. 45, pp. 61–68, 2014.
- T. Ge, K. He, Q. Ke, and J. Sun. Optimized Product Quantization. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI), 36(4):744–755, Apr. 2014
- A. Babenko and V. Lempitsky. The Inverted MultiIndex. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 37(6):1247–1260, June 2015.

References (chronological order)

- 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
- David C. Anastasiu and George Karypis. 2015. L2Knng: Fast Exact K-Nearest Neighbor Graph Construction with L2-Norm Pruning. In CIKM '15.
- 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)
- 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
- Djamel Edine Yagoubi, Reza Akbarinia, Florent Masseglia, Themis Palpanas: DPiSAX: Massively Distributed Partitioned iSAX. ICDM 2017: 1135-1140
- 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.
- Katsiaryna Mirylenka, Michele Dallachiesa, Themis Palpanas: Correlation-Aware Distance Measures for Data Series. ICDE 2017: 502-505
- Katsiaryna Mirylenka, Michele Dallachiesa, Themis Palpanas: Data Series Similarity Using Correlation-Aware Measures. SSDBM 2017: 11:1-11:12
- Kostas Zoumpatianos, Themis Palpanas: Data Series Management: Fulfilling the Need for Big Sequence Analytics. ICDE 2018: 1677-1678
- 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

References (chronological order)

- Michele Linardi, Themis Palpanas: ULISSE: ULtra Compact Index for Variable-Length Similarity Search in Data Series. ICDE 2018: 1356-1359
- J. Wang, T. Zhang, j. song, N. Sebe, and H. T. Shen. A survey on learning to hash. TPAMI, 40(4): 769-790 (2018).
- Kostas Zoumpatianos, Yin Lou, Ioana Ileana, Themis Palpanas, Johannes Gehrke: Generating data series query workloads. VLDB J. 27(6): 823-846 (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)
- 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)
- 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)
- Botao Peng, Panagiota Fatourou, Themis Palpanas: ParIS: The Next Destination for Fast Data Series Indexing and Query Answering. BigData 2018: 791-800
- Akhil Arora, Sakshi Sinha, Piyush Kumar, Arnab Bhattacharya: HD-Index: Pushing the Scalability-Accuracy Boundary for Approximate kNN Search in High-Dimensional Spaces. PVLDB. 11(8): 906-919 (2018).
- D.E. Yagoubi, R. Akbarinia, B. Kolev, O. Levchenko, F. Masseglia, P. Valduriez, D. Shasha. ParCorr: efficient parallel methods to identify similar time series pairs across sliding windows. Data Mining and Knowledge Discovery (DMKD), 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)

References (chronological order)

- 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
- 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)
- Danila Piatov, Sven Helmer, Anton Dignös, Johann Gamper: Interactive and space-efficient multi-dimensional time series subsequence matching. Inf. Syst. 82: 121-135 (2019)
- 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)
- 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.
- 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)
- John Paparrizos, Michael J. Franklin: GRAIL: Efficient Time-Series Representation Learning. Proc. VLDB Endow. 12(11): 1762-1777 (2019)
- Cong Fu, Chao Xiang, Changxu Wang, and Deng Cai. 2019. Fast approximate nearest neighbor search with the navigating spreading-out graph. Proc. VLDB Endow. 12, 5 (January 2019), 461–474.
- Jiaye Wu, Peng Wang, Ningting Pan, Chen Wang, Wei Wang, Jianmin Wang: KV-Match: A Subsequence Matching Approach Supporting Normalization and Time Warping. ICDE 2019: 866-877
- Liang Zhang, Noura Alghamdi, Mohamed Y. Eltabakh, Elke A. Rundensteiner: TARDIS: Distributed Indexing Framework for Big Time Series Data. ICDE 2019: 1202-1213
- Anna Gogolou, Theophanis Tsandilas, Karima Echihabi, Anastasia Bezerianos, Themis Palpanas: Data Series Progressive Similarity Search with Probabilistic Quality Guarantees. SIGMOD Conference 2020: 1857-1873
- Themis Palpanas. Evolution of a Data Series Index - The iSAX Family of Data Series Indexes. CCIS, 1197 (2020)
- 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)

References (chronological order)

- Botao Peng, Panagiota Fatourou, Themis Palpanas: MESSI: In-Memory Data Series Indexing. ICDE 2020: 337-348
- Kefeng Feng, Peng Wang, Jiaye Wu, Wei Wang: L-Match: A Lightweight and Effective Subsequence Matching Approach. IEEE Access 8: 71572-71583 (2020)
- Chen Wang, Xiangdong Huang, Jialin Qiao, Tian Jiang, Lei Rui, Jinrui Zhang, Rong Kang, Julian Feinauer, Kevin Mcgrail, Peng Wang, Diaohan Luo, Jun Yuan, Jianmin Wang, Jiaguang Sun: Apache IoTDB: Time-series database for Internet of Things. Proc. VLDB Endow. 13(12): 2901-2904 (2020)
- Michele Linardi, Themis Palpanas. Scalable Data Series Subsequence Matching with ULISSE. VLDBJ 2020
- John Paparrizos, Chunwei Liu, Aaron J. Elmore, Michael J. Franklin: Debunking Four Long-Standing Misconceptions of Time-Series Distance Measures. SIGMOD Conference 2020
- Karima Echihabi, Kostas Zoumpatianos, and Themis Palpanas. Scalable Machine Learning on High-Dimensional Vectors: From Data Series to Deep Network Embeddings. In WIMS, 2020
- Oleksandra Levchenko, Boyan Kolev, Djamel-Edine Yagoubi, Reza Akbarinia, Florent Masseflia, Themis Palpanas, Dennis Shasha, Patrick Valduriez. BestNeighbor: Efficient Evaluation of kNN Queries on Large Time Series Databases. Knowledge and Information Systems (KAIS), 2020
- Kejing Lu, Hongya Wang, Wei Wang, Mineichi Kudo. VHP: Approximate Nearest Neighbor Search via Virtual Hypersphere Partitioning. PVLDB, 13(9): 1443-1455, 2020
- Yury A. Malkov, D. A. Yashunin: Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. IEEE Trans. Pattern Anal. Mach. Intell. 42(4): 824-836 (2020)
- Botao Peng, Panagiota Fatourou, Themis Palpanas. Paris+: Data series indexing on multi-core architectures. TKDE, 2021
- Botao Peng, Panagiota Fatourou, Themis Palpanas. SING: Sequence Indexing Using GPUs. ICDE, 2021
- Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas. Big Sequence Management: Scaling Up and Out. EDBT 2021
- Botao Peng, Panagiota Fatourou, Themis Palpanas. Fast Data Series Indexing for In-Memory Data. VLDBJ 2021