CSCI 599: Multidimensional Databases

Efficient Similarity Search In Sequence Databases

Rakesh Agrawal, Christos Faloutsos, Arun Swami

Selina Chu
University of Southern California
Fall 2003

Outline of Talk

- Background on similarity search in time series
  - Distance measures
  - Dimensionality reduction
  - Indexing
- Proposed method
- Experiments and results
- Conclusion

What are Time Series?

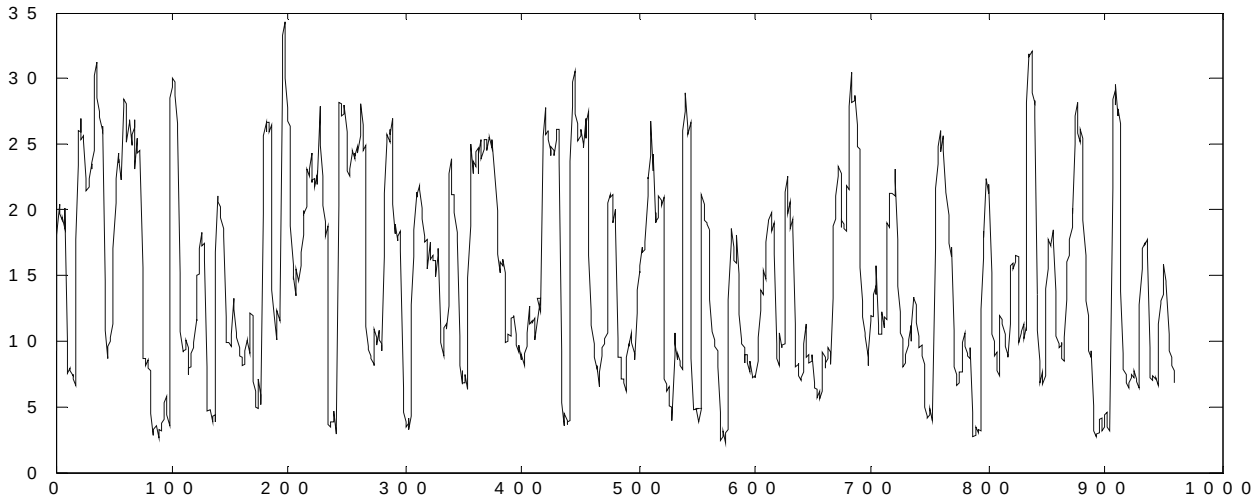
18.1750
18.2180
18.1800
18.1800
18.2750
18.3180
18.3500
18.3500
18.4000
18.4000
18.3180
18.2180
18.2000
18.1750

••

••

17.6180
17.6750
17.6750
17.6180
17.6180
17.6180
17.6750
17.7500

A time series is a collection of observations made sequentially in time.



Data Mining Applications of Time Series

Classification: Classify patients based on their ECG patterns

Clustering: Group patients together with similar irregularities in their ECGs

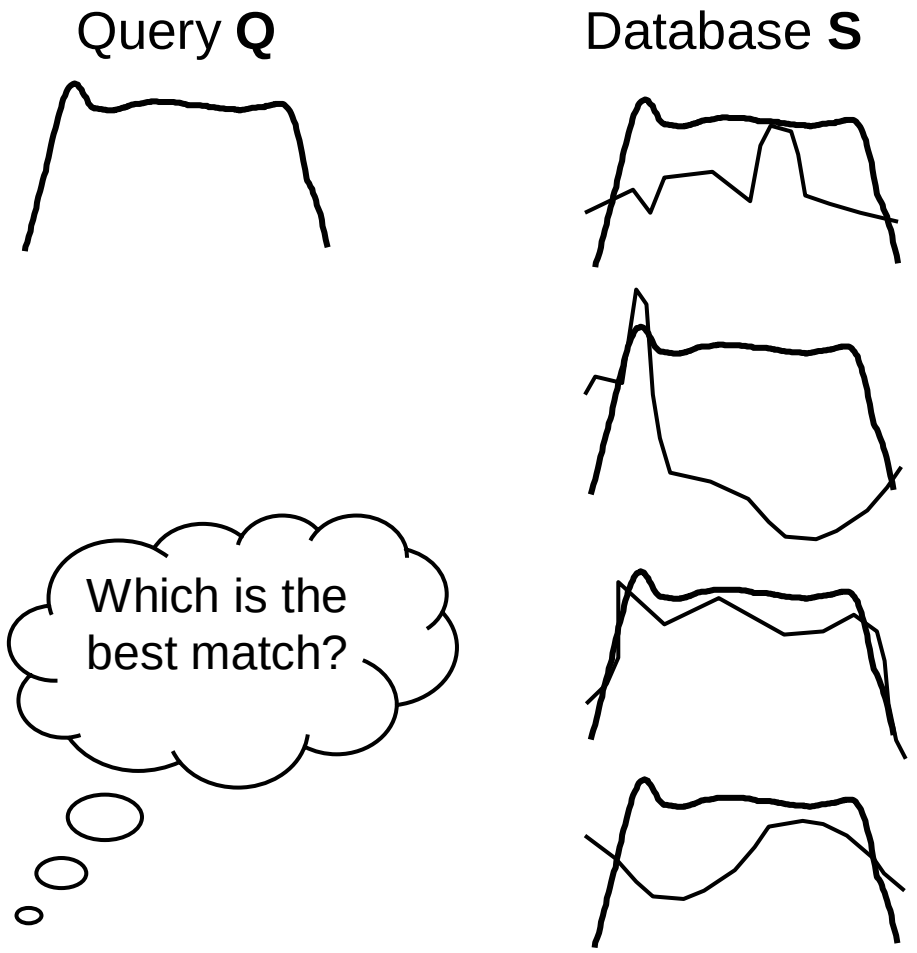
Association Rules: a peak followed by a plateau implies a downward trend with a confidence of 0.4 and a support of 0.2

Query by Content: Find other patients with similar abnormalities in their ECG patterns

What is Similarity Search?

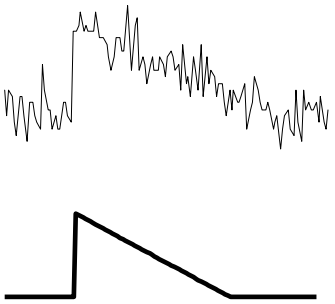
(In the Context of Time Series)

Given a query **Q**, a reference database **S** and a distance measure, find the sequence in **S** that best matches **Q** and return its position.



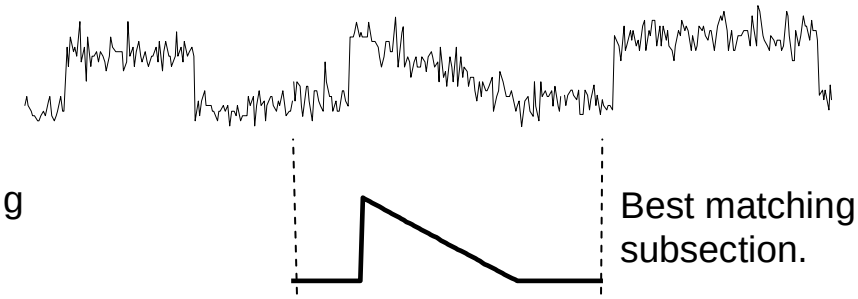
Two Categories of Similarity Matching

1: Whole Matching



Best matching same length sequence

2:Subsequence Matching



Best matching subsection.

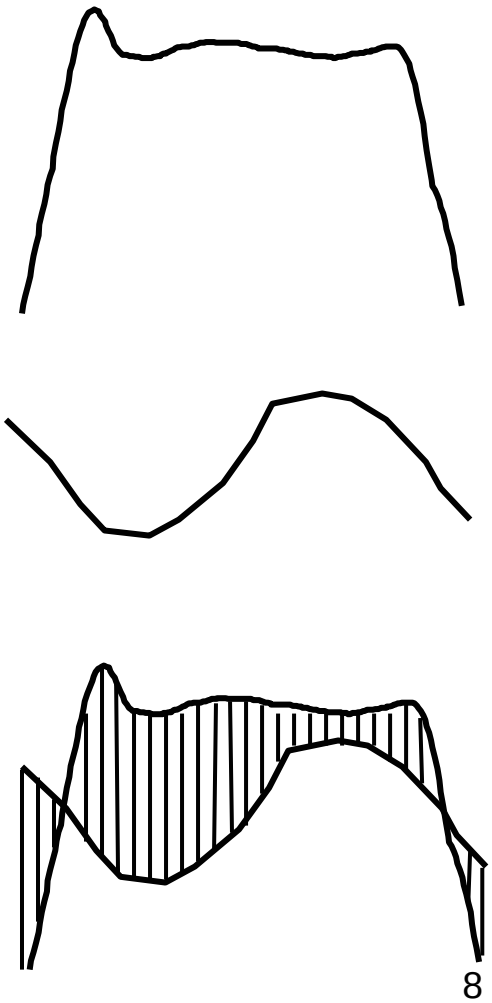
Outline of Talk

- Background on similarity search in time series
  - Distance measures
  - Dimensionality reduction
  - Indexing
- Proposed method
- Experiments and results
- Conclusion

How do we measure similarity?

Euclidean Distance between two time series $Q = \{q_1 \ldots q_n\}$ and $S = \{s_1 \ldots s_n\}$ is defined as:

$$D(Q,S) \equiv \sqrt{\sum_{i=1}^{n} (q_i - s_i)^2}$$



Desired properties of similarity measures:

D(A,B) = D(B,A) *Symmetry*

D(A,A) = 0 *Constancy of Self-Similarity*

D(A,B) = 0, Iff A=B *Positivity (Separation)*

D(A,B) ≤ D(A,C) + D(B,C) *Triangular Inequality*

All measures for indexing require the triangular inequality to hold.

Three Types of Similarity Queries

- **Nearest Neighbor Query** – Find the nearest Starbucks to LAX

- **Range Query** – Find all Starbucks that are within 5 miles of LAX

- **All-Pairs Query** (spatial joint) – Find all Starbucks that are within 5 miles of a Burger King

Outline of Talk

- Background on similarity search in time series
  - Distance measures
  - Dimensionality reduction
  - Indexing
- Proposed method
- Experiments and results
- Conclusion

Why an Approximate Representation of the Data?

- One Hour of ECG data: 1 Gigabyte.
- Typical Web-Log: 5 Gigabytes per week.
- Space Shuttle Database: 158 Gigabytes and growing.
- Macho Database: 2 Terabytes, updated with 3 gigabytes per day.

We need a representation of the data that we can efficiently manipulate.

In most cases, patterns, not individual points, are of interest.

DFT: Discrete Fourier Transform

- **Basic Idea:** Represent the time series as a linear combination of sines and cosines, but keep only the first *n*/2 coefficients
- **Why n/2 coefficients?** Because each sine wave requires 2 numbers, for the phase and amplitude

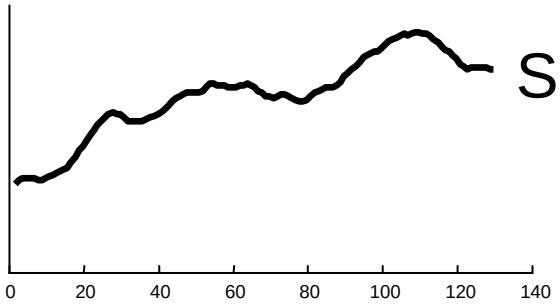
$$X_f = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t e^{-j 2\pi f t / n} \qquad f = 0,1,\ldots n-1$$

$$e^{j\theta} = \cos(\theta) + j \sin(\theta)$$

Most popular implementation : Fast Fourier Transform (FFT) - O(*n*log*n*)

- Actually O(*n*log*n*) if n is a power of two, otherwise O(n^2).

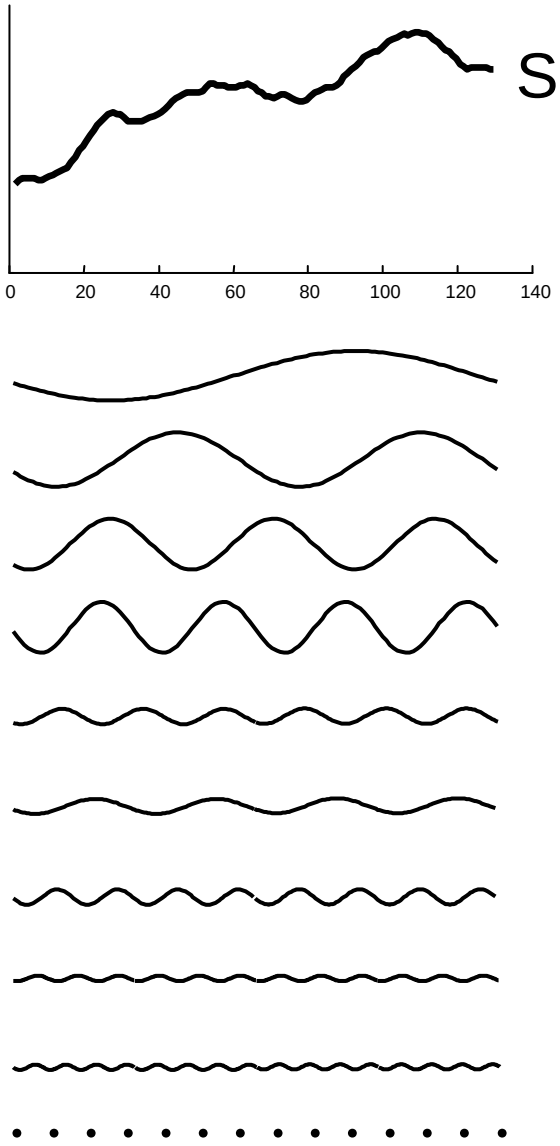
Example of Dimensionality Reduction Using DFT



Raw Data

0.4995
0.5264
0.5523
0.5761
0.5973
0.6153
0.6301
0.6420
0.6515
0.6596
0.6672
0.6751
0.6843
0.6954
0.7086
0.7170
0.7412
0.7595
0.7780
0.7956
0.8115
0.8177
0.8345
0.8407
0.8431
0.8423
0.8387
...
...

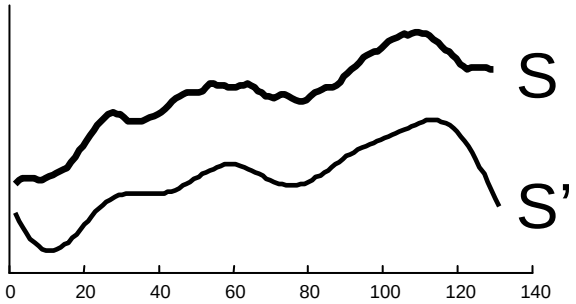
Example of Dimensionality Reduction Using DFT

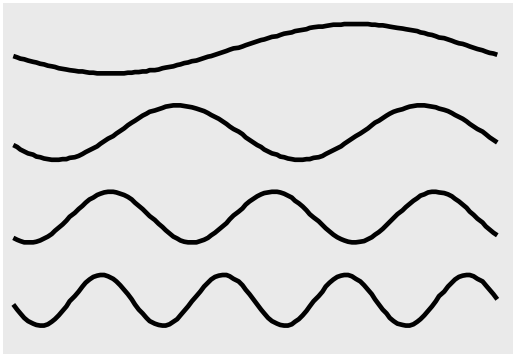


Raw Data
0.4995
0.5264
0.5523
0.5761
0.5973
0.6153
0.6301
0.6420
0.6515
0.6596
0.6672
0.6751
0.6843
0.6954
0.7086
0.7170
0.7412
0.7595
0.7780
0.7956
0.8115
0.8177
0.8345
0.8407
0.8431
0.8423
...

Fourier Coefficients	
0.2464	1.4810
0.4231	0.7142
0.2423	0.4611
0.2179	0.3449
0.2020	0.2428
0.1805	0.1992
0.1818	0.1556
0.1745	0.1337
0.1922	0.0964
0.1716	0.0629
0.1748	0.0520
0.1683	0.0294
0.1817	0.0211
0.1817	0.0211
0.1683	0.0294
0.1748	0.0520
0.1716	0.0629
0.1922	0.0964
0.1745	0.1337
0.1818	0.1556
0.1805	0.1992
0.2020	0.2428
0.2179	0.3449
0.2423	0.4611
0.4231	0.7142
0.2464	1.4810
...	

Example of Dimensionality Reduction Using DFT





Raw Data
0.4995
0.5264
0.5523
0.5761
0.5973
0.6153
0.6301
0.6420
0.6515
0.6596
0.6672
0.6751
0.6843
0.6954
0.7086
0.7170
0.7412
0.7595
0.7780
0.7956
0.8115
0.8177
0.8345
0.8407
0.8431
0.8423
...

Fourier Coefficients	
0.2464	1.4810
0.4231	0.7142
0.2423	0.4611
0.2179	0.3449
0.2020	0.2428
0.1805	0.1992
0.1818	0.1556
0.1745	0.1337
0.1922	0.0964
0.1716	0.0629
0.1748	0.0520
0.1683	0.0294
0.1817	0.0211
0.1817	0.0211
0.1683	0.0294
0.1748	0.0520
0.1716	0.0629
0.1922	0.0964
0.1745	0.1337
0.1818	0.1556
0.1805	0.1992
0.2020	0.2428
0.2179	0.3449
0.2423	0.4611
0.4231	0.7142
0.2464	1.4810
...	

Reduced set of Fourier Coefficients

0.2464	1.4810
0.4231	0.7142
0.2423	0.4611
0.2179	0.3449

We have discarded $\frac{9}{10}$ of the data

How do we search in a large database to find the best match to a query?

Trivial solution: Search every item in the database (Sequential search)

Problem: Do not scale well to large datasets

A better solution…

How do we search in a large database to find the best match to a query?

Trivial solution: Search every item in the database (Sequential search)

Problem: Do not scale well to large datasets

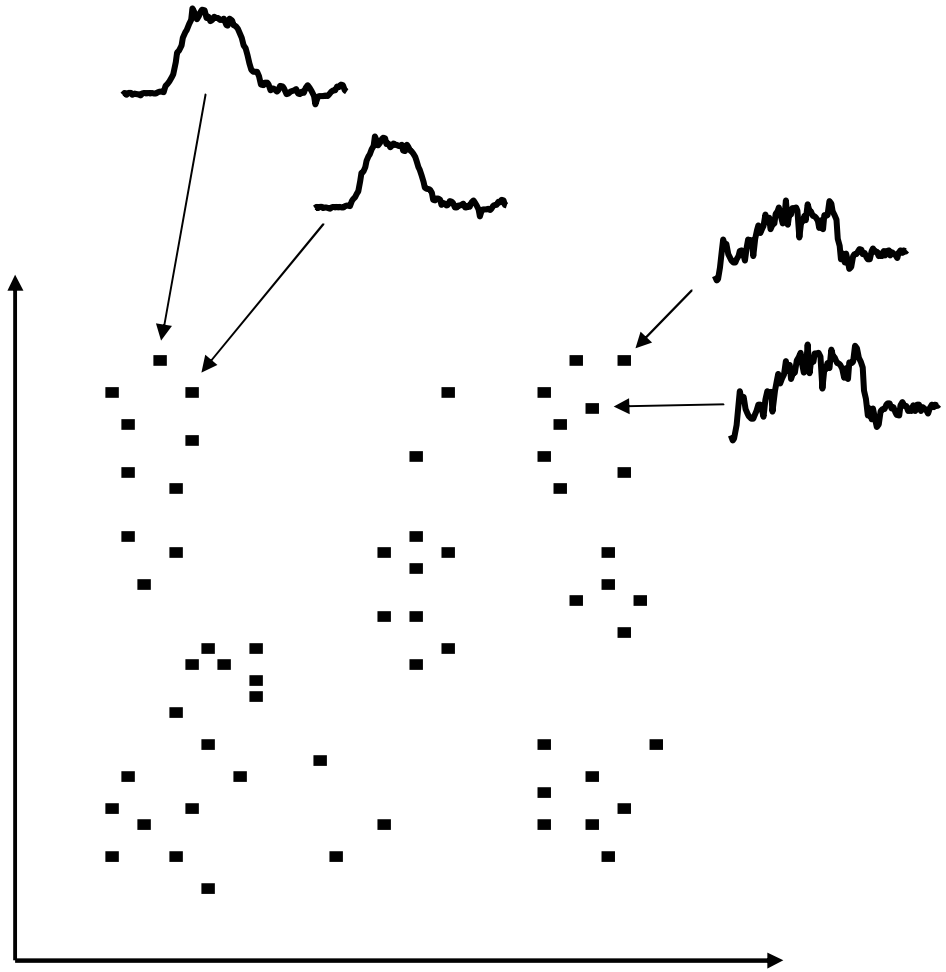
A better solution…

We can index the database using a multidimensional index structure, allowing efficient similarity search

Outline of Talk

- Background on similarity search in time series
  - Distance measures
  - Dimensionality reduction
  - Indexing
- Proposed method
- Experiments and results
- Conclusion

Indexing Time Series



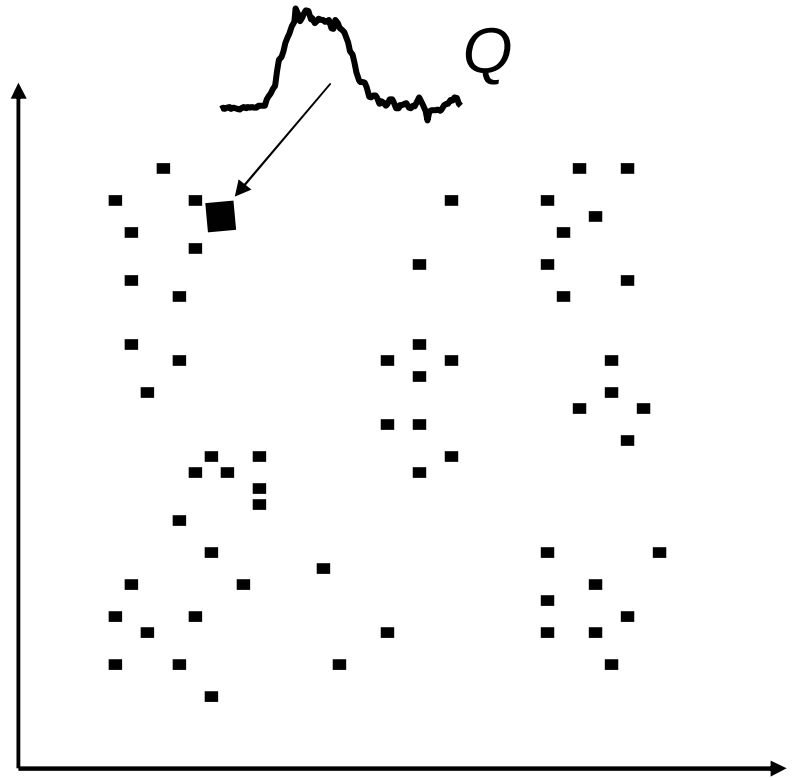
n-dimensional space

We can project time series of length *n* into *n*-dimension space.

The first value in *S* is the X-axis, the second value in *S* is the Y-axis, etc.

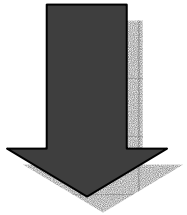
One advantage of doing this is that we have abstracted away the details of “time series”, now all query processing can be imagined as finding points in space...

…we can also project the query *Q* into the same *n*-dimension space and simply look for the nearest points.

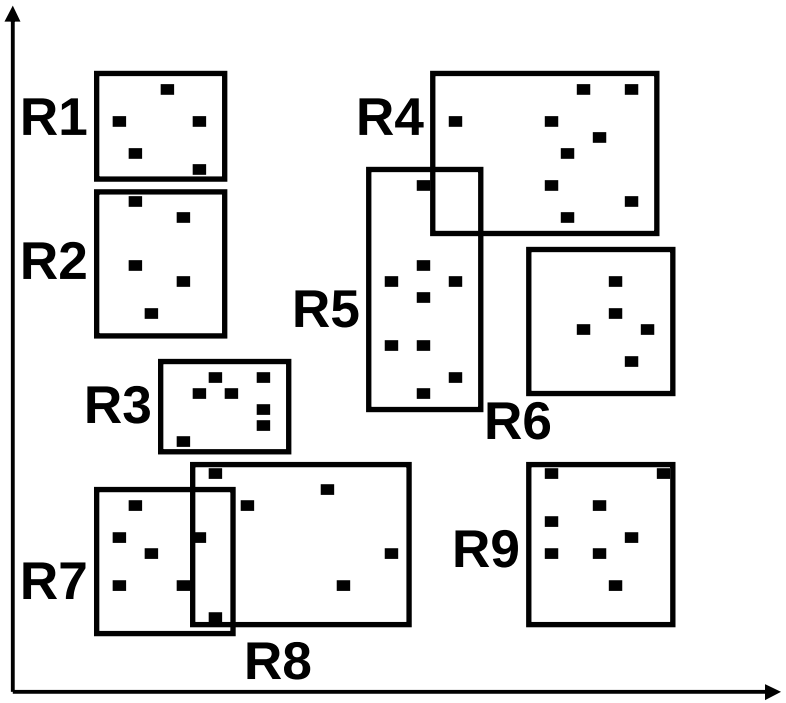


…the problem is that we have to look at every point to find the nearest neighbor..

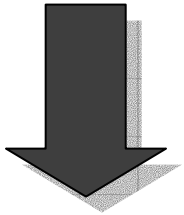
Solution? We can use Spatial Access Methods (SAMs). Of which R-tree is the most famous example…



We can group clusters of datapoints with “boxes”, called Minimum Bounding Rectangles (MBR).

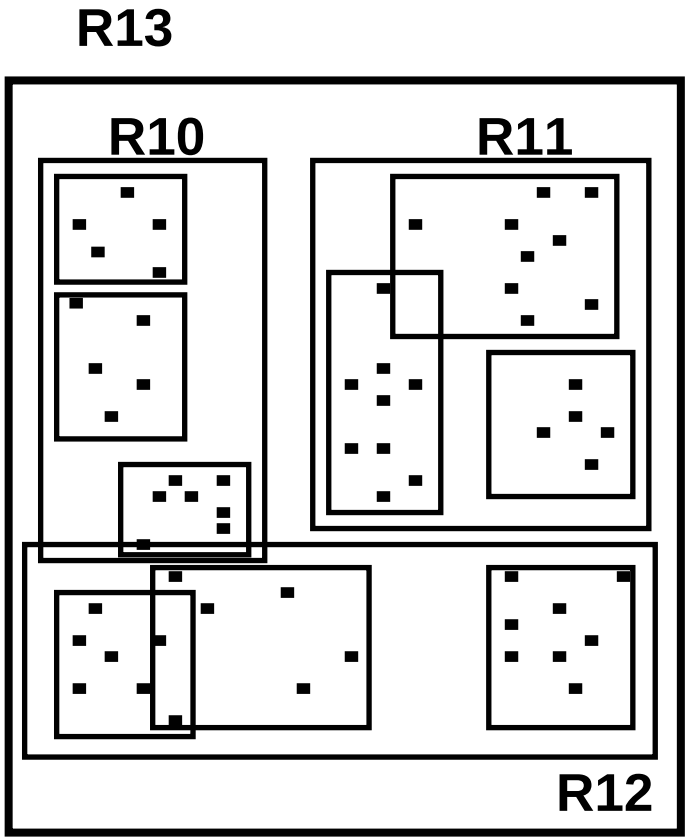


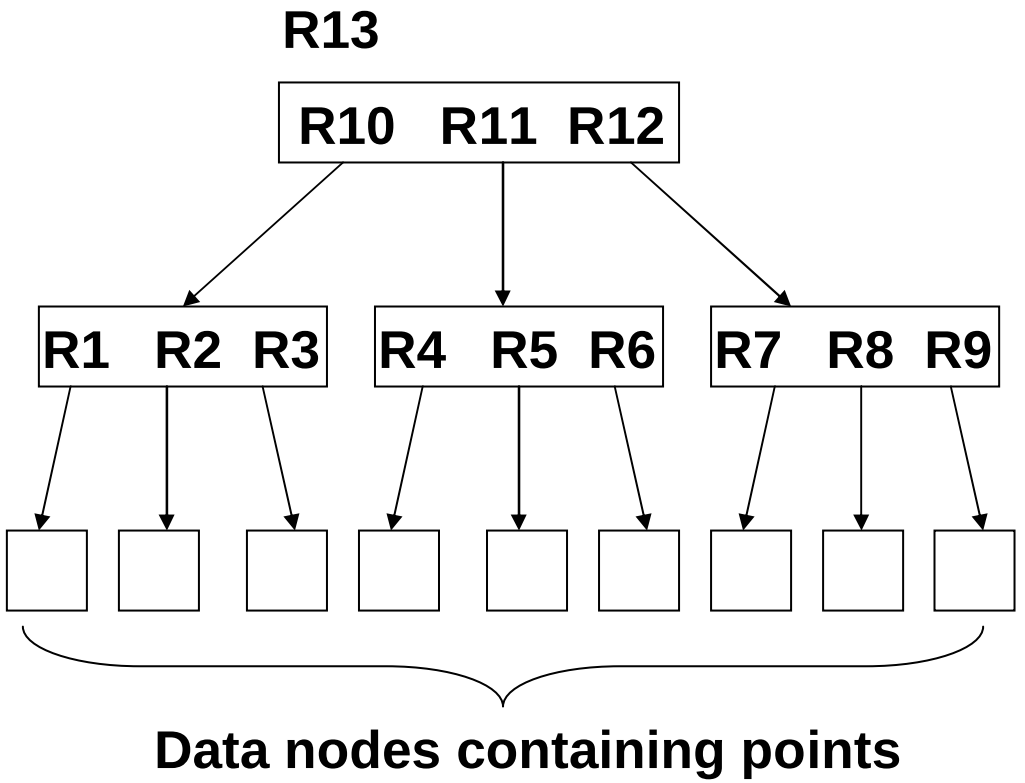
We can further recursively group MBRs into larger MBRs….



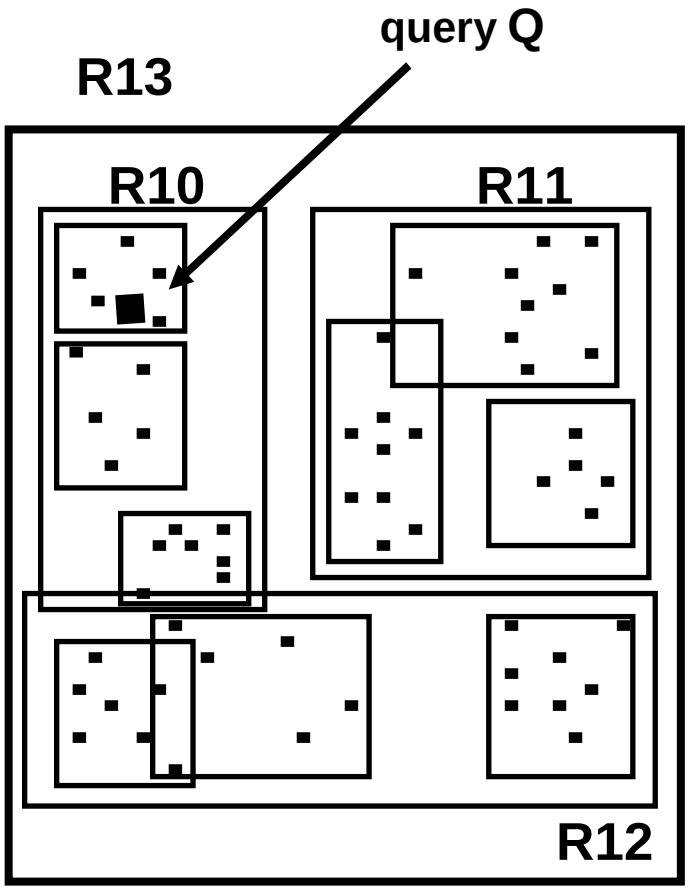
…these nested MBRs are organized as a tree (called a spatial access tree or a multidimensional tree).

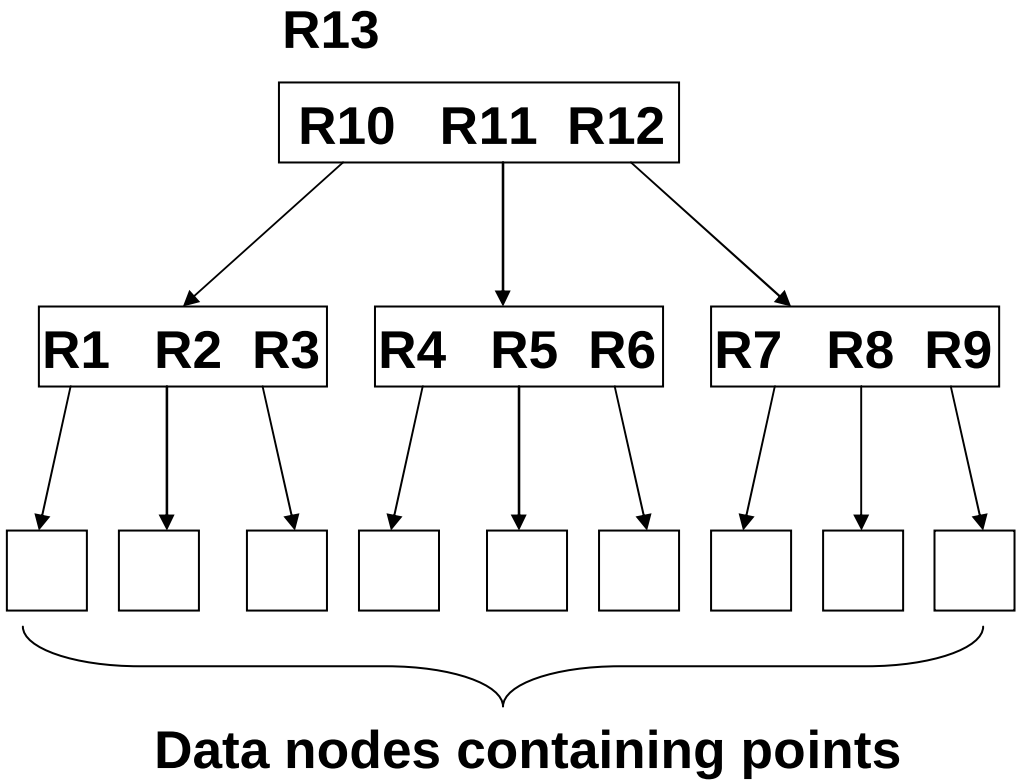
Examples include R-tree, X-tree, Hybrid-tree etc.



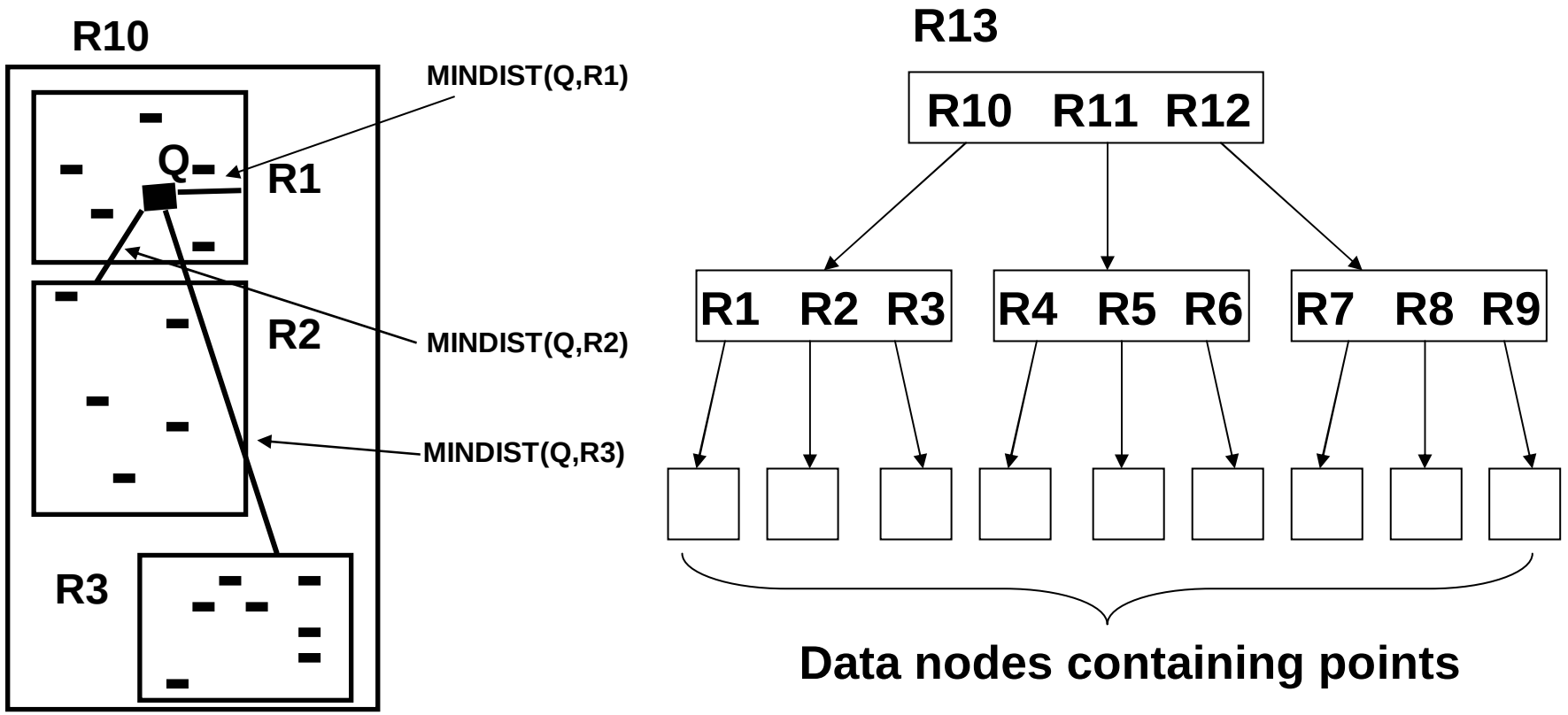


Projecting query Q into the same n-dimensional space

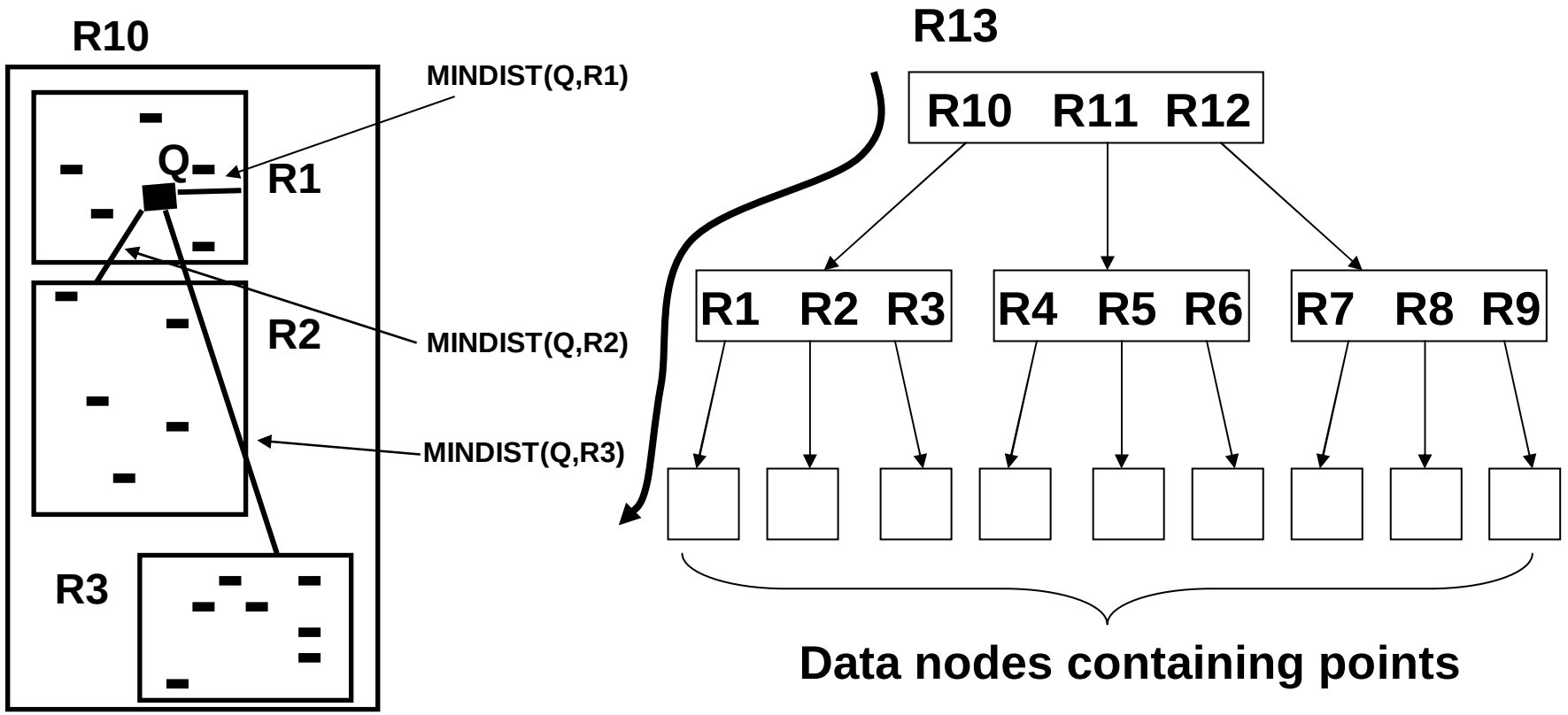




We can define a function, **MINDIST(*point*, *MBR*)**, which tells us the minimum possible distance between any point and any MBR, at any level of the tree.



We can define a function, **MINDIST(*point*, *MBR*)**, which tells us the minimum possible distance between any point and any MBR, at any level of the tree.



This allows us to search through the database efficiently

Once the query is completed…

Post-process the answer set :

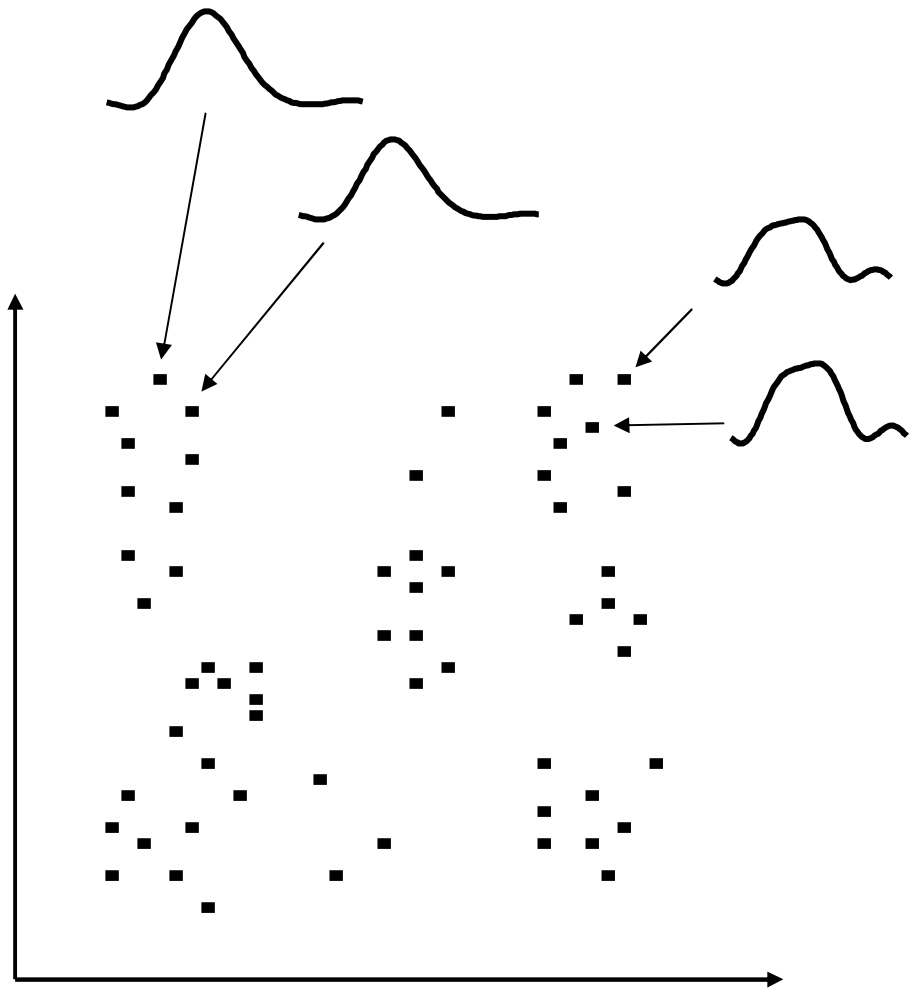
- Compute the actual distance between the query and each sequence from the answer set, in the time domain

- Prune sequences that are greater than ε

Outline of Talk

- Background on similarity search in time series
  - Distance measures
  - Dimensionality reduction
  - Indexing
- Proposed method
- Experiments and results
- Conclusion

F-Index



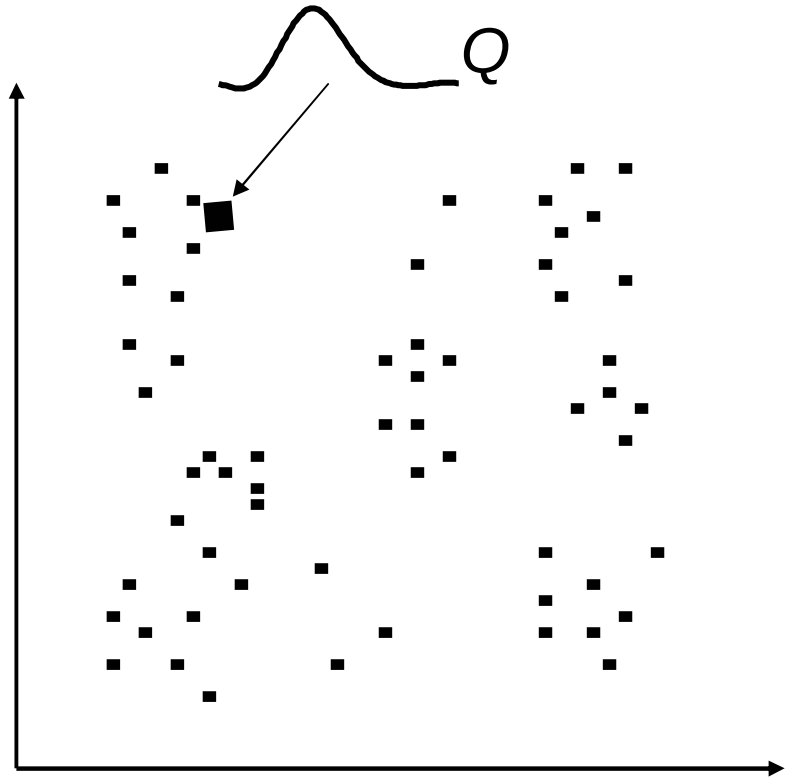
n’-dimensional space

Obtain coefficients of DFT for every sequence in the database

Build the multidimensional index using the first few Fourier coefficients of every sequence in the database

We can project the Fourier coefficients into *n’*-dimension space.
(*n*’ = # of Fourier coefficients used)

Querying in F-Index



And then we use DFT to transform query Q into frequency domain and uses the first few Fourier coefficients to project into the same *n'*-dimensional space.

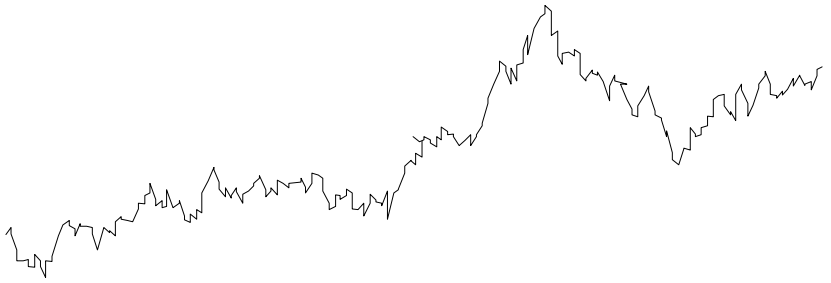
Again, we can use Spatial Access Methods to perform the similarity search.

Outline of Talk

- Background on similarity search in time series
  - Distance measures
  - Dimensionality reduction
  - Indexing
- Proposed method
- Experiments and results
- Conclusion

Experimental Setup

Dataset: Random Walk



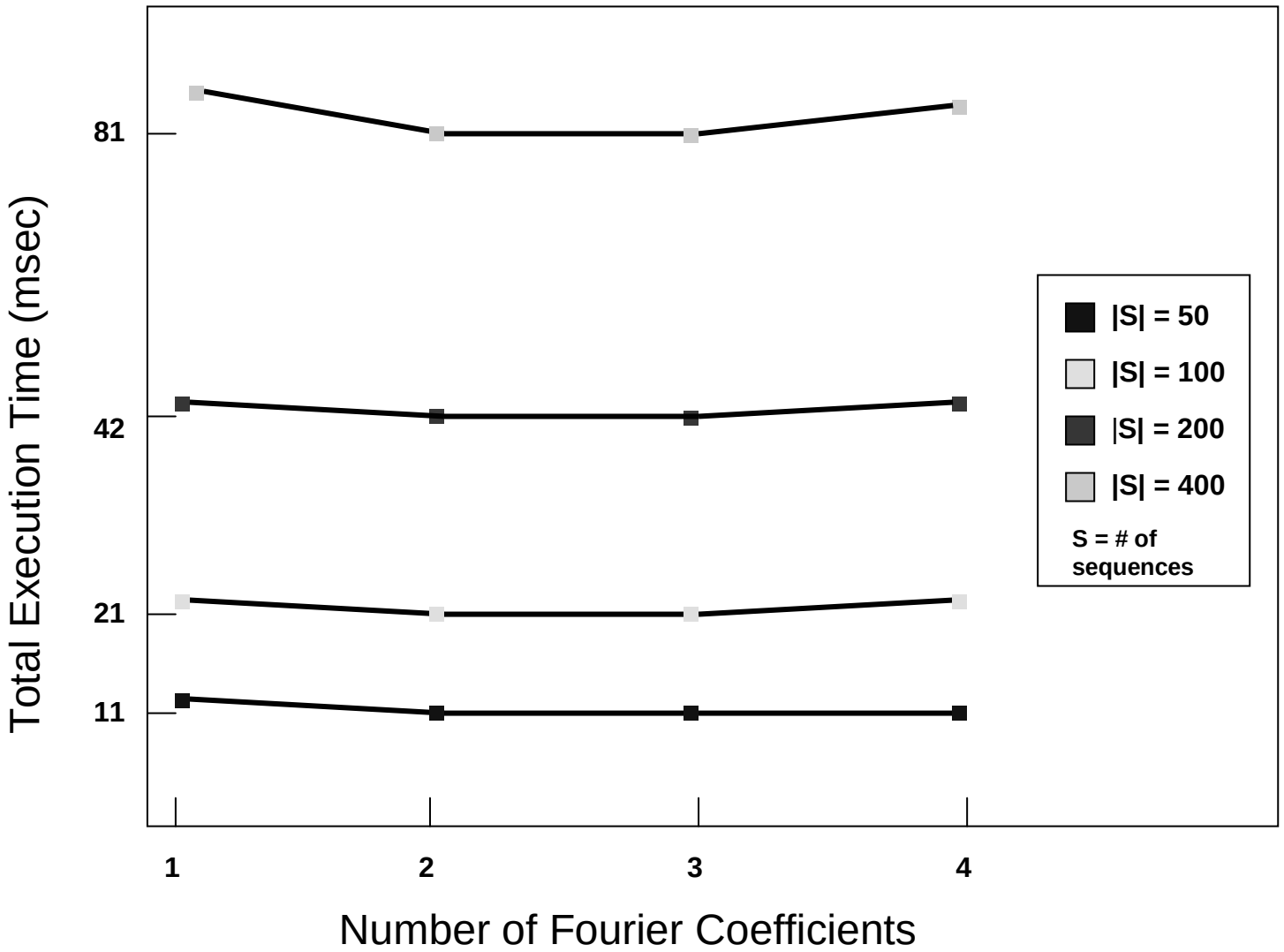
Compare F-index and sequential scanning

Perform range and all-pairs queries

# of Fourier coefficients used: 1 - 4

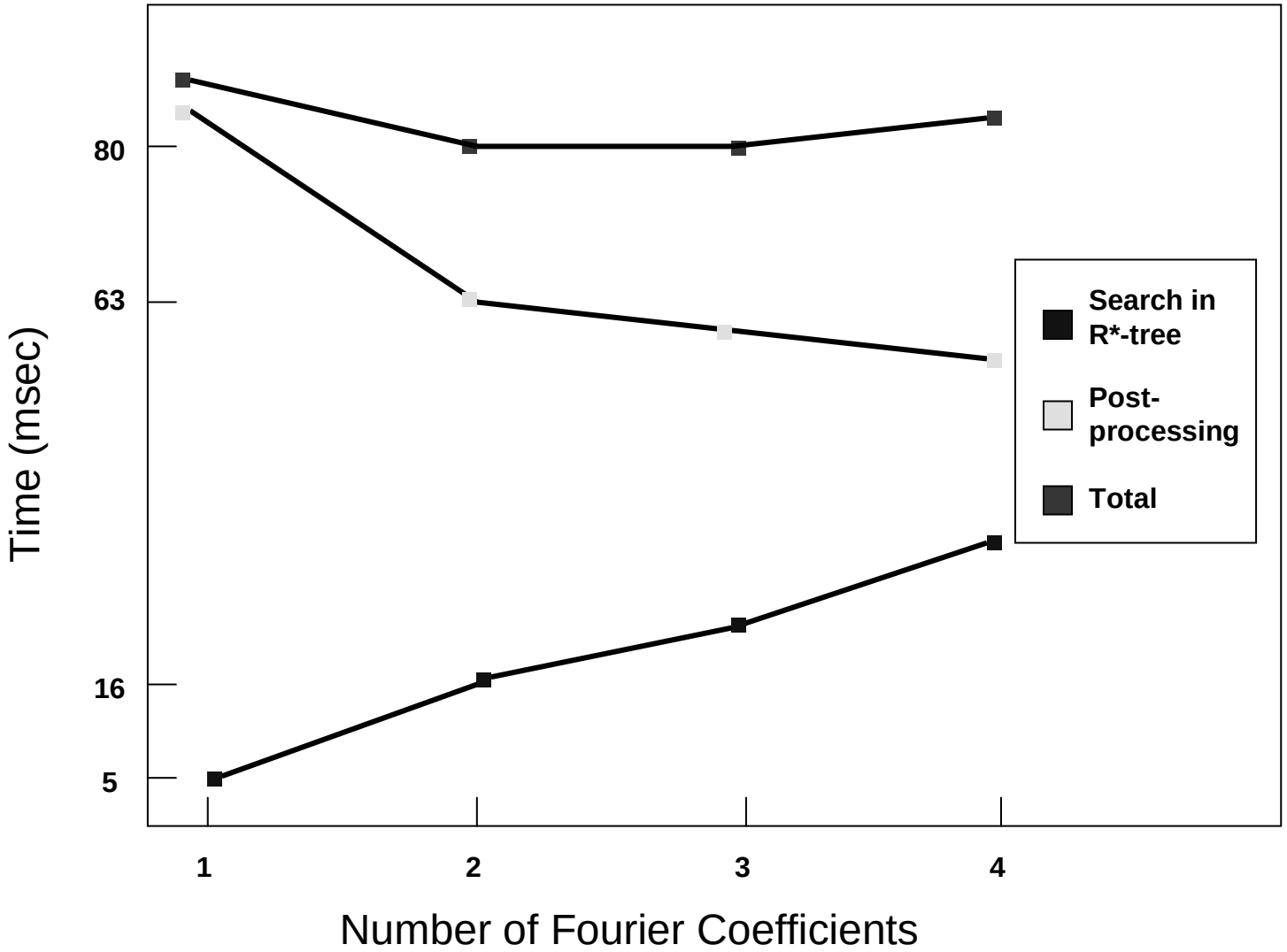
Time per query vs. # Fourier coefficients

(For range queries)



Breakdown of the execution time

(For range queries, |s|=400)



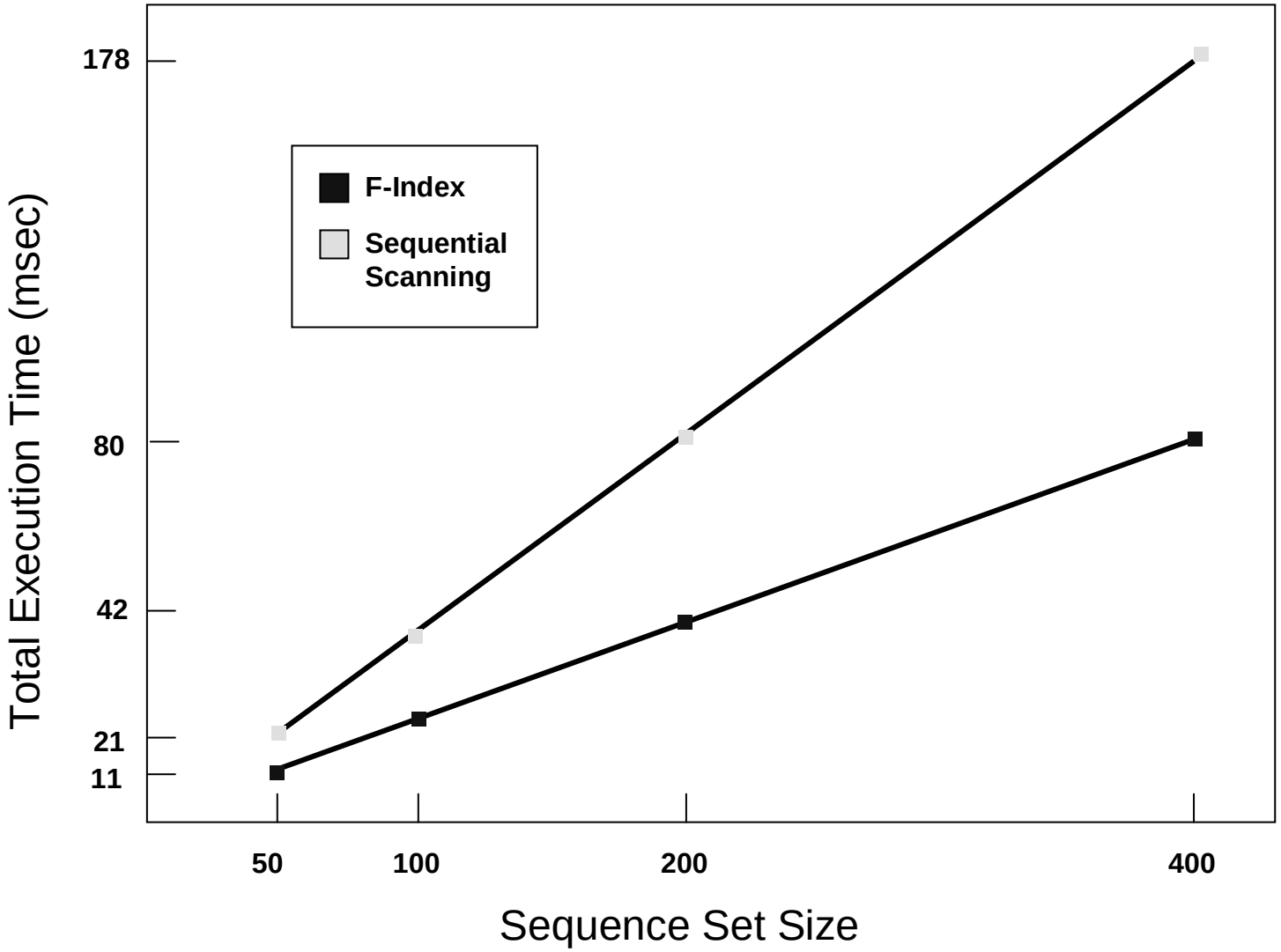
What does the results show us?

If we increase the number of Fourier coefficients (f_c)…

- ⬆ dimensionality of R*-tree,
  (each additional fc will increase dimensionality of R*-tree by 2)
- ⬆ search time in the tree,
- ⬇ false hits, less post-processing time

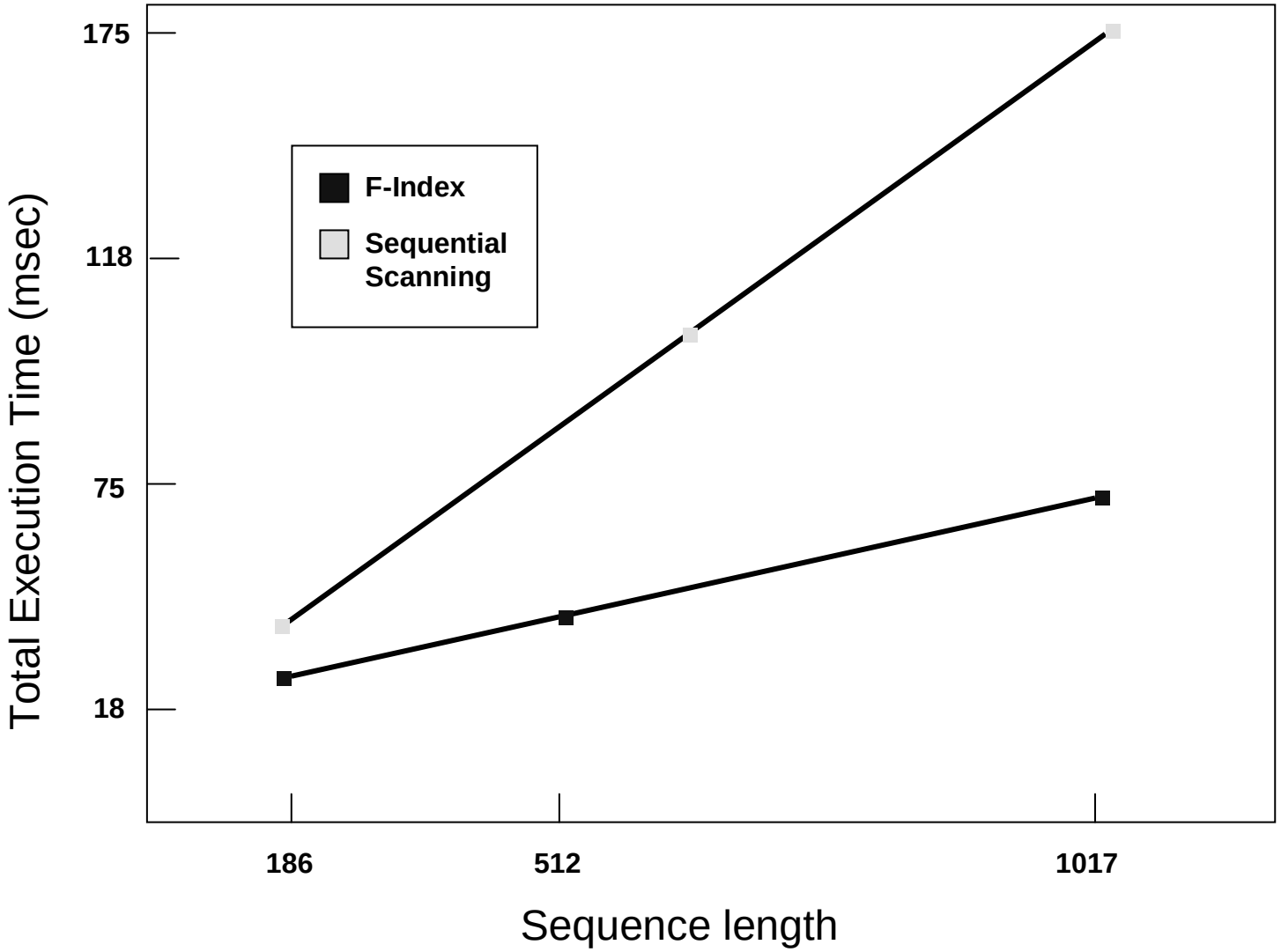
Time per query vs. # Sequences per dataset

(For range queries, # of Fourier coefficients=2)



Time per query vs. Sequence length

(For range queries, # of Fourier coefficients=2)



Conclusion

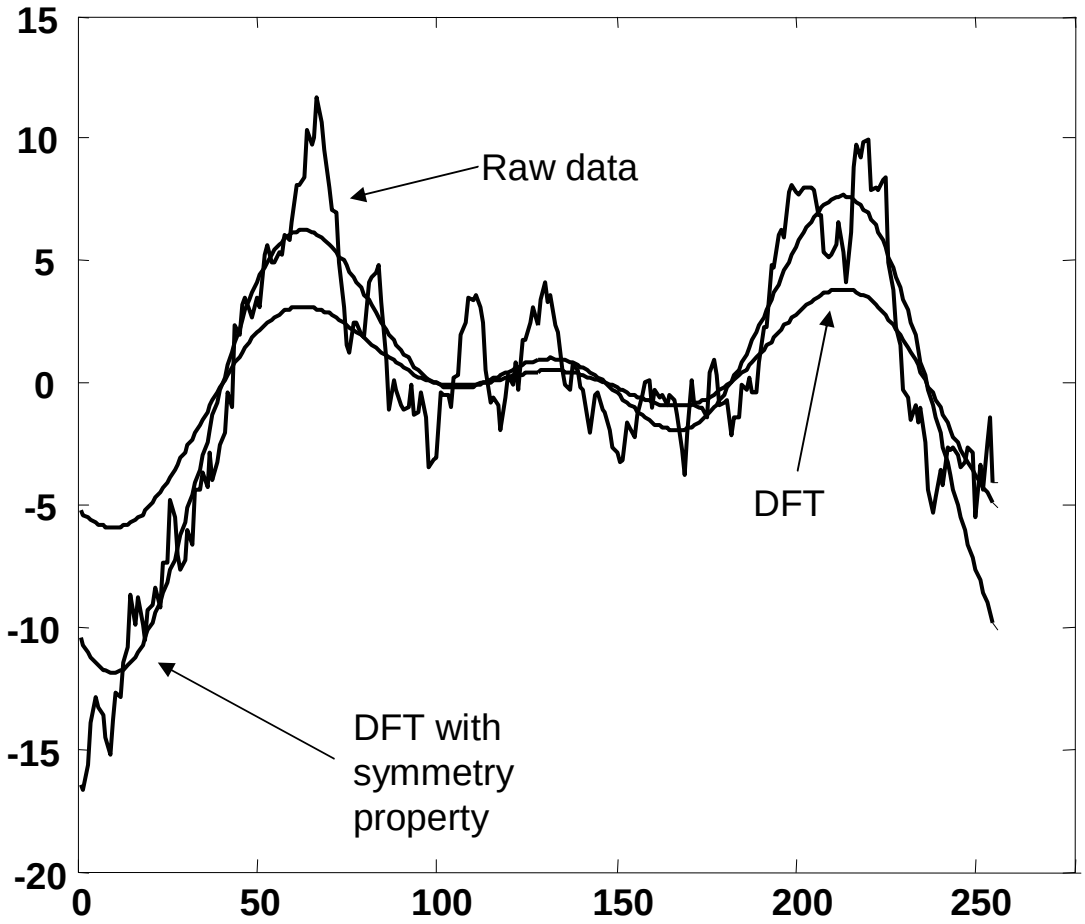
- Proposed indexing method for similarity search in time series

- Used DFT to reduce the dimensionality of the data for indexing

- Displayed significant performance gain over sequential scanning

Developments in the last ten years

- This paper did not use symmetry property of DFT (Davood Rafiei pointed this out in 1997)



Developments in the last ten years
(continue)

- There have been some advances in SAM's, so instead of 1-4 dimensions, we can now use 10-16.

- There is now a widespread belief that Euclidean distance may be too brittle for real world problems, some people are using DTW (Dynamic Time Warping) instead.

- The authors only used one dataset to test on! It turn outs the data can have a major impact on the performance.

Questions??

Triangular Inequality

- Querying cost is expensive
- Using the Triangular Inequality property allows us to reduce the number of calls

$D(A,B) \leq D(A,C) + D(B,C)$

If we know:

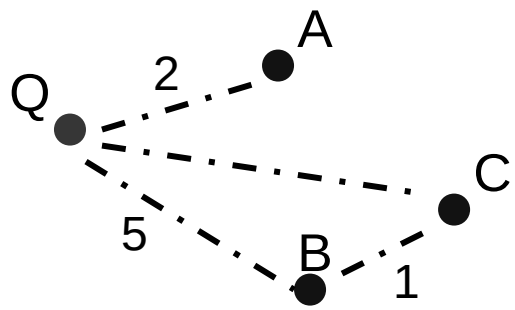
$D(Q,A) = 2$ (best so far)

$D(Q,B) = 5$ and $D(B,C)=1$

By Triangular Inequality, we do not need to look at C because:

$D(Q,B) - D(B,C) \leq D(Q,C)$

$4 \leq D(Q,C)$



Dimensionality Curse
(in indexing)

- To search, we have to examine its neighbors
  - 1 dimension, need to search 2 MBRs
  - 2 dimension, need to search 8 MBRs
  - etc…
- When dimensionality increases, MBRs in non-leaf nodes increase in size, resulting in a decreased fan-out.
- Could become worst than sequential scanning

R-tree

- R-tree is a dynamic B-tree like structure
  - has the self-balancing property (all the leaf nodes appear at the same level)
- Achieves an average of about 50% node utilization.
- Spatial objects can be represented in their original data space
  - Objects are represented by the MBRs in which they are contained.

referenced from:
http://goanna.cs.rmit.edu.au/~santhas/research/paper1/node4.html

R*-tree

- Two improvements over the R-tree
  - new node splitting policy
    - results in minimized overlap and better storage utilization
    - reduce dead space in bounding rectangles and their perimeter.
  - Forces some aging objects to be re-inserted,
    - changing the shape of the tree dynamically
    - results in better search performance.

referenced from:
http://goanna.cs.rmit.edu.au/~santhas/research/paper1/node4.html

This paper was extended by the following:

- Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. In Proceedings of the SIGMOD. (Gemini)
- Loh, W., Kim, S & Whang, K. (2000). Index interpolation: an approach to subsequence matching supporting normalization transform in time-series databases. Proceedings 9th International Conference on Information and Knowledge Management.
- Chu, K & Wong, M. (1999). Fast time-series searching with scaling and shifting. Proceedings of the 18th ACM Symposium on Principles of Database Systems, Philadelphia.
- Refiei, D. (1999). On similarity-based queries for time series data. Proc of the 15th IEEE International Conference on Data Engineering. Sydney, Australia.

Papers on some other dimensionality reduction techniques:

Discrete Wavelet Transform (DWT):

- Chan, K. & Fu, W. (1999). Efficient time series matching by wavelets. Proceedings of the 15th IEEE International Conference on Data Engineering.
- Wu, Y., Agrawal, D. & Abbadi, A.(2000). A Comparison of DFT and DWT based Similarity Search in Time-Series Databases. Proceedings of the 9th International Conference on Information and Knowledge Management.
- Kahveci, T. & Singh, A (2001). Variable length queries for time series data. Proceedings 17th International Conference on Data Engineering. Heidelberg, Germany.

Singular Value Decomposition (SVD):

- Korn, F., Jagadish, H & Faloutsos. C. (1997). Efficiently supporting ad hoc queries in large datasets of time sequences. Proceedings of SIGMOD, Tucson, AZ, pp 289-300.

Piecewise Polynomial Approximations:

- Keogh, E,. Chakrabarti, K,. Pazzani, M. & Mehrotra (2001) Dimensionality reduction for fast similarity search in large time series databases. Knowledge and Information Systems. Volume 3, Number 3, August.
- Yi, B,K., & Faloutsos, C.(2000). Fast time sequence indexing for arbitrary Lp norms. Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt.

Piecewise Linear Representation

- Keogh, E., Chu, S., Hart, D., Pazzani, M. (2001) An Online Algorithm for Segmenting Time Series. Keogh, E., Chu, S., Hart, D., Pazzani, M. In The IEEE International Conference on Data Mining (ICDM), 2001.

Inner Product Approximation

- Egecioglu, O., & Ferhatosmanoglu, H. (2000). Dimensionality reduction and similarity distance computation by inner product approximations. Proceedings of the 9th ACM International Conference on Information and Knowledge Management (CIKM), pp. 219-226.

About using different types of data:

- Keogh, E. and Kasetty, S. (2002). On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. In the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. July 23 - 26, 2002. Edmonton, Alberta, Canada. pp 102-111.

datasets:

http://kdd.ics.uci.edu/

http://wwwmacho.mcmaster.ca/Project/Overview/status.html