RWTH AACHEN UNIVERSITY

AARHUS UNIVERSITY

Efficient Processing of Multiple DTW Queries in Time Series Databases

Hardy Kremer[1] Stephan Günnemann[1]
Anca-Maria Ivanescu[1] Ira Assent[2] Thomas Seidl[1]

[1] RWTH Aachen University, Germany

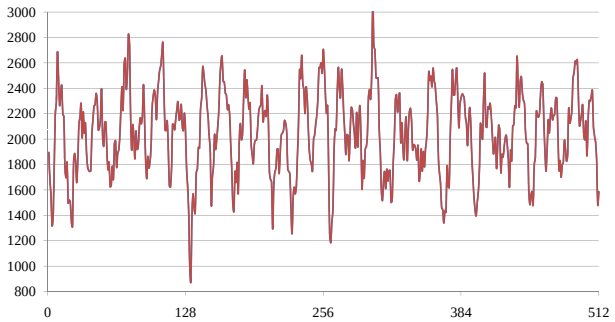
[2] Aarhus University, Denmark

SSDBM 2011, Portland, Oregon, USA
July 20-22, 2011

Time Series Similarity Search

Time series

- Sequence of time related values
- Stock data, sensor data, EEG measurements, climate data, ...

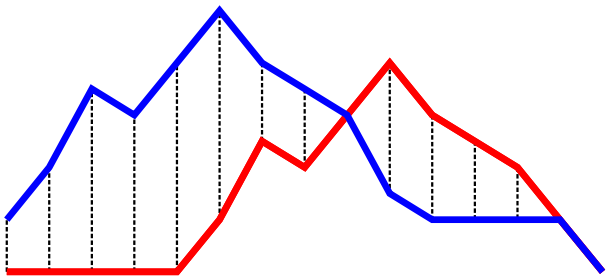


Similarity search

Find time series with similar patterns over time

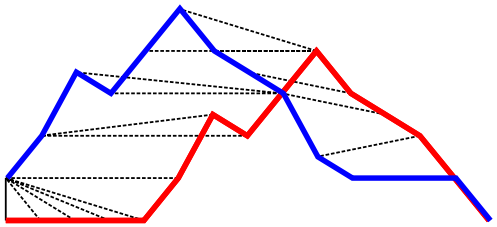
with e.g., Euclidean Distance

only corresponding Time Points are compared

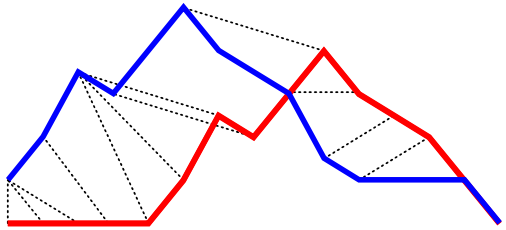


But: In many applications, time series are out-of-sync.

Dynamic Time Warping



Dynamic Time Warping



DTW with k-Band

- DTW: Distance between time series is based on new alignment.

$\Rightarrow$ Matching by stretching & scaling along time axis.

- k-Band constraint avoids degenerate warpings.

Efficient processing

- Quadratic complexity in length of time series
- There are many efficient query processing algorithms

$\Rightarrow$ **good for single, ad-hoc queries**

Multiple Query Processing for DTW

Today's applications

massive amounts of queries need to be processed in **limited time**:

- Sensor networks ⇒ timely reaction to events
- Data Mining applications ⇒ Scalability
- Interactive Visualization ⇒ Interactiveness

Often, **queries may be similar or even share subsequences**.

- E.g., determine the transitive closure in density-based clustering.

Definition: Multiple DTW Range Query

Input:

- set of time series queries $Q = \{q^1, \ldots, q^c\}$, database DB, range ϵ

Output:

- multiple result sets $Res^i(DB) = \{t \in DB | DTW(q, t) \leq \epsilon\}$ with $i = 1, \ldots, c$.

Later on: kNN-queries

Multiple Query Processing for DTW (2)

Definition: Multiple DTW Range Query

Input:

- set of time series queries $Q = \{q^1, \ldots, q^c\}$, database DB, range ϵ

Output:

- multiple result sets $Res^i(DB) = \{t \in DB | DTW(q, t) \leq \epsilon\}$ with $i = 1, \ldots, c$.

Our novel approach...

- exploits the **similarity among queries**, for **combined pruning**.
- uses a nested **hierarchy of query subgroups**, for pruning of smaller and more similar query subsets.
- is **combinable** with existing single DTW query approaches.
- guarantees **exact** results.

Overview

1. Preliminaries
2. Processing of Multiple DTW Queries
3. Experiments
4. Conclusion

DTW Definition

k-band DTW

$$DTW([s_1, ..., s_n],\ [t_1, ..., t_m]) =$$

$$dist_{band}(s_n,\ t_m) + \min \begin{cases} DTW([s_1, ..., s_{n-1}],\ [t_1, ..., t_{m-1}]) \\ DTW([s_1, ..., s_n],\ [t_1, ..., t_{m-1}]) \\ DTW([s_1, ..., s_{n-1}],\ [t_1, ..., t_m]) \end{cases}$$

with

$$dist_{band}(s_i,\ t_j) = \begin{cases} dist(s_i, t_j) & |i-j| \leq k \\ \infty & else \end{cases}$$

$$DTW(\emptyset, \emptyset) = 0, \quad DTW(x, \emptyset) = \infty, \quad DTW(\emptyset, y) = \infty$$

Distance between points is measured by **ground distance** $dist(s_i, t_j)$

Efficient, exact single query DTW processing

- DTW is computationally expensive
- Many approaches use multistep filter-and-refine architecture
- If filter lower bounds DTW $\Rightarrow$ lossless
- There are filters that achieve substantial speed-ups, e.g. LBKeogh

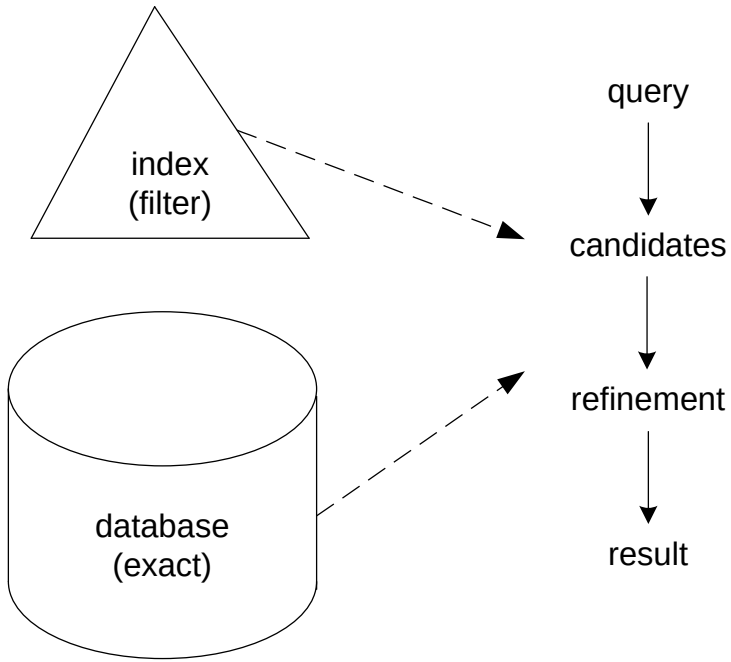
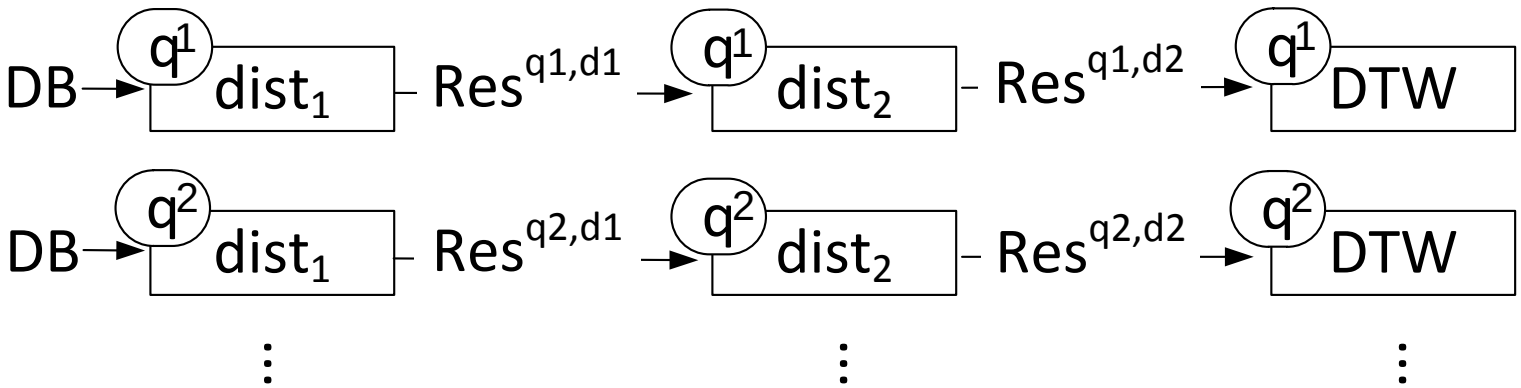


Figure: Multistep filter-and-refine architecture

Baseline Solution for Multiple DTW Processing

- Independent processing of each query of query set Q
- I.e., each single query passes through filter cascade w.r.t. whole DB



- does not exploit knowledge about whole query set Q.

$\Rightarrow$ In our approach, queries are processed simultaneously.

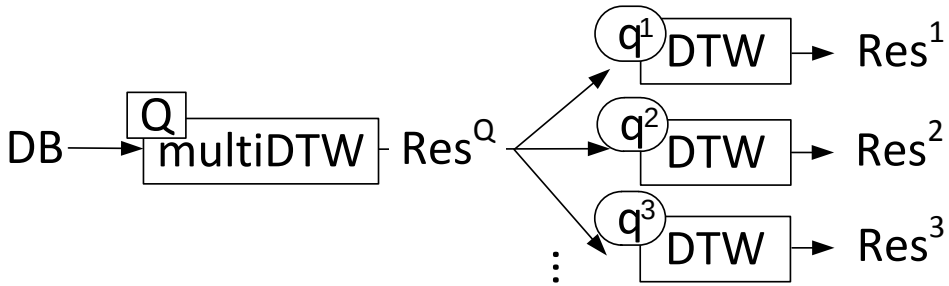
Overview

1. Preliminaries
2. Processing of Multiple DTW Queries
3. Experiments
4. Conclusion

Multiple Query Distance Function: Idea

- For simultaneous processing, we need a distance function between several query objects and a database object.

$\Rightarrow$ One single calculation for several queries.

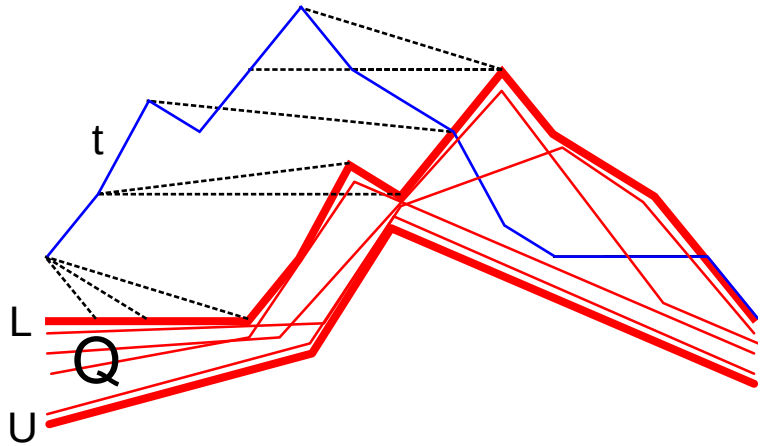


- No matching query in Q if $multiDTW(Q, t) > \epsilon$ $\Rightarrow$ pruning.
- Shared Lower Bounding Property for preventing false dismissals: for all $Q, t : \forall q \in Q : multiDTW(Q, t) \leq DTW(q, t)$

Multiple Query Distance Function: Solution

- We use a compact, single representation of a time series query set, called Multiple Query Bounding Box:

  $multiBox(Q) = [(L_1, U_1), \ldots, (L_n, U_n)] = [B_1, \ldots, B_n]$
  with $L_i = \min_{q \in Q} q_i$ and $U_i = \max_{q \in Q} q_i$



- Final multiDTW function uses the minimal distance from time series t to a MultiBox B as **DTW ground distance**:

$$dist(B_i,\ t_j) = dist((L_i, U_i),\ t_j) = \begin{cases} |t_j - U_i| & \text{if } t_j > U_i \\ |t_j - L_i| & \text{if } t_j < L_i \\ 0 & \text{otherwise} \end{cases}$$

Multiple Query Tree

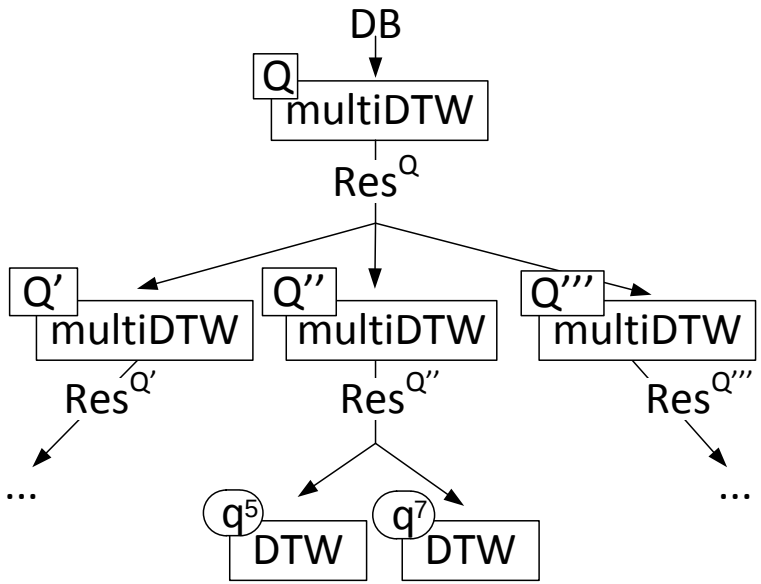
- one single group = one single query to be processed

⇒ BUT: we have a relatively large intermediate result set Res^Q.

- smaller, more similar groups could reduce the intermediate result size.

⇒ We introduce a hierarchy of query sets with:

- $\forall Q, Q' \subseteq Q, p: \ multiDTW(Q, p) \leq multiDTW(Q', p)$



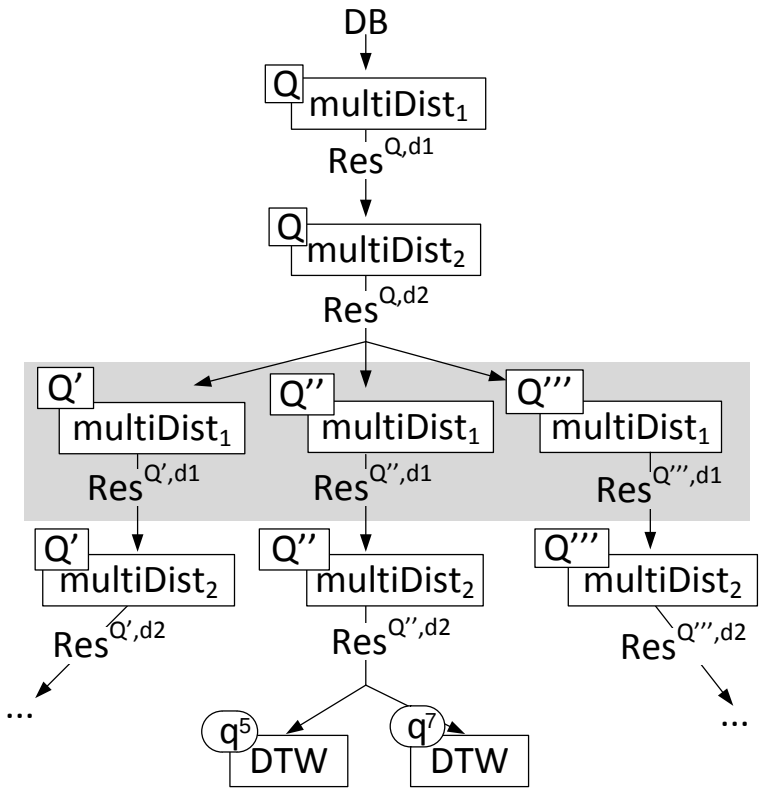
- Trade-off: smaller intermediate sets vs. higher computational costs

Filter-supported Hierarchical Multiple DTW Query

- Existing filter techniques are orthogonal to our approach.

$\Rightarrow$ Flexibility w.r.t. pruning. In each node of the tree we can either

- go to new query granularity, i.e. by splitting the query group, or
- use a new lower bound filter.



Multiple DTW kNN Query (1)

Definition: Multiple DTW kNN Query

Input:

- set of time series queries $Q = \{q^1, \ldots, q^c\}$, database DB, k

Output:

- multiple result sets $Res^i(DB)$ with $|Res^i(DB)| = k$ and
  $\forall t \in Res^i(DB) \forall s \in DB \backslash Res^i(DB)$:
  $DTW(q^i, t) \leq DTW(q^i, s)$ for all $i = 1, \ldots, c$.

Multiple DTW *k*NN Query (2)

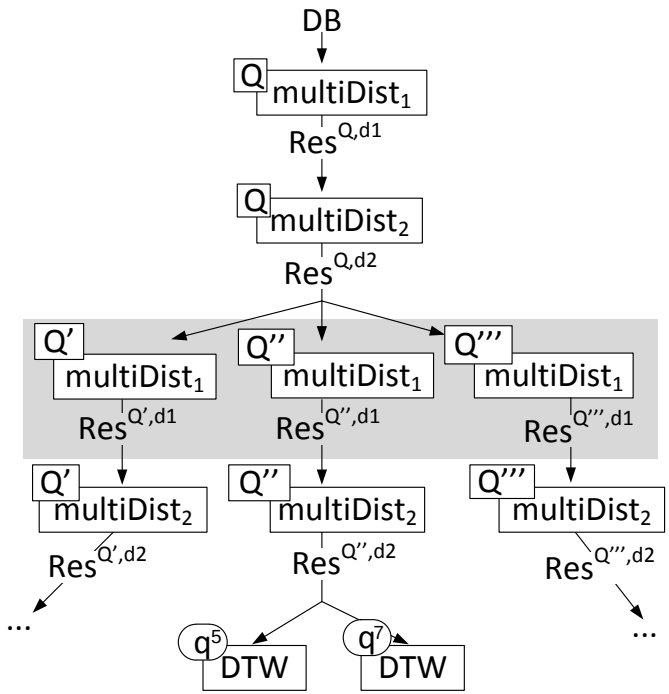
*k*NN Processing

- For *k*NN queries the ϵ range for pruning is not known a priori.

$\Rightarrow$ We need one moving threshold per query subgroup

$\Rightarrow$ for each query we have a current set of *k* result

- both are constantly updated



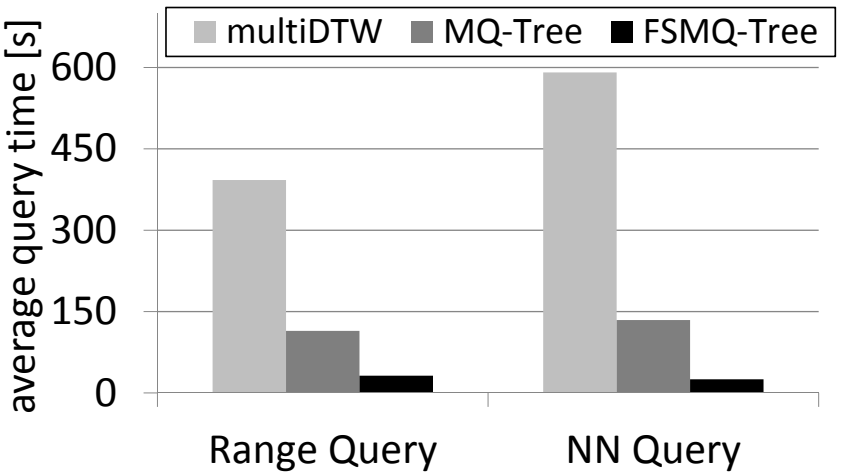
Overview

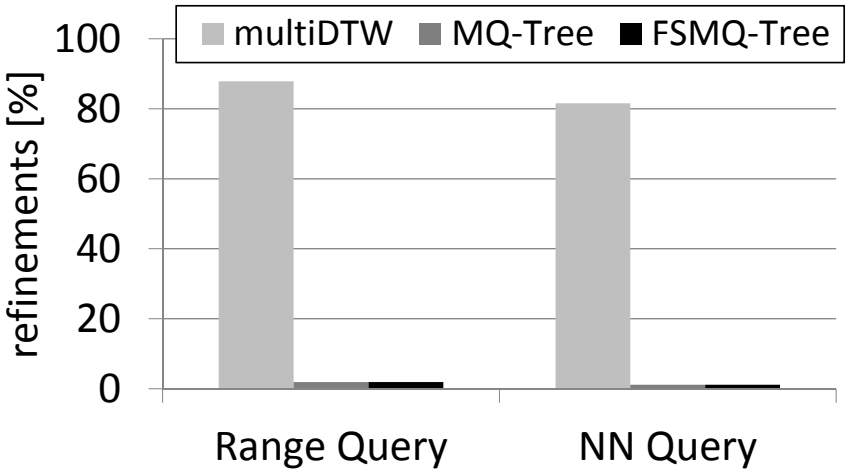
1. Preliminaries
2. Processing of Multiple DTW Queries
3. Experiments
4. Conclusion

Experimental Evaluation: Setup

- Our approach:
  Linear scan to process database time series, but usage of an index & dimensionality reduction is also possible
- Measurements:
  - Wall clock time averaged over 10 queries
  - relative number of exact refinements
- Multiple Query generator:
  - for each Multiple Query, we select S random seeds
  - for each seed, we generate g queries deviating from the seed by a standard deviation of 10%
  - As default, we use 8 seeds and 5 generated queries per seed, resulting in 40 queries per multiple query.

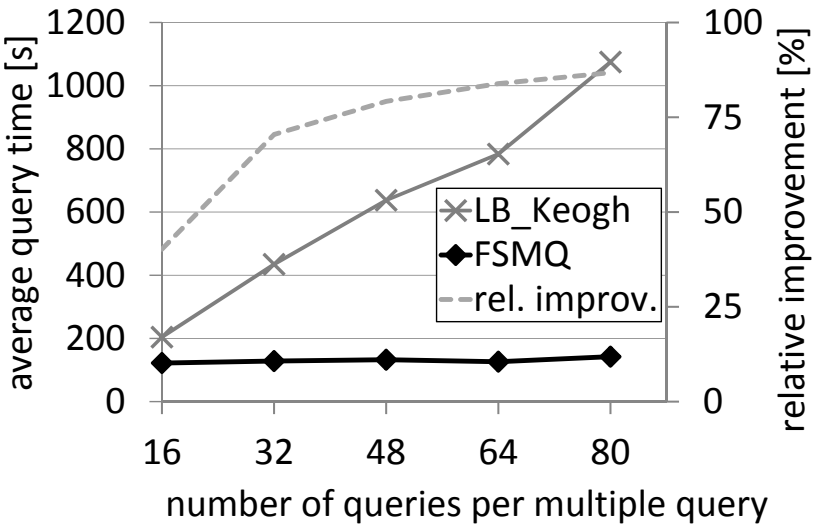
Comparison of the three variants of our method

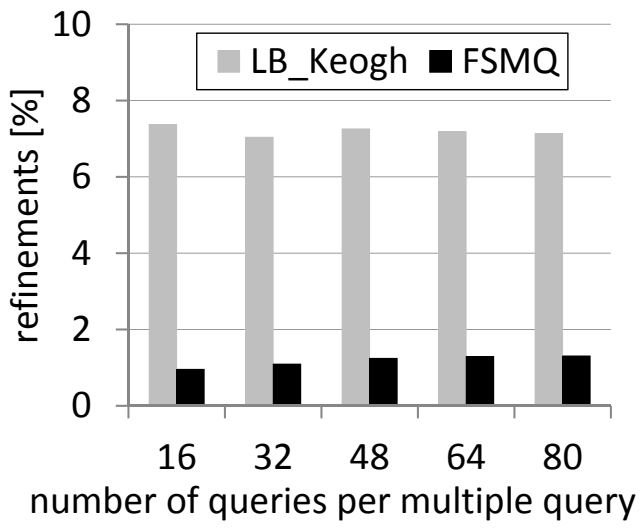




- Random Walk data
- both MQ-tree and the FSMQ-tree dramatically reduce the number of exact DTW calculations.
- Filter support has significant influence on efficiency.

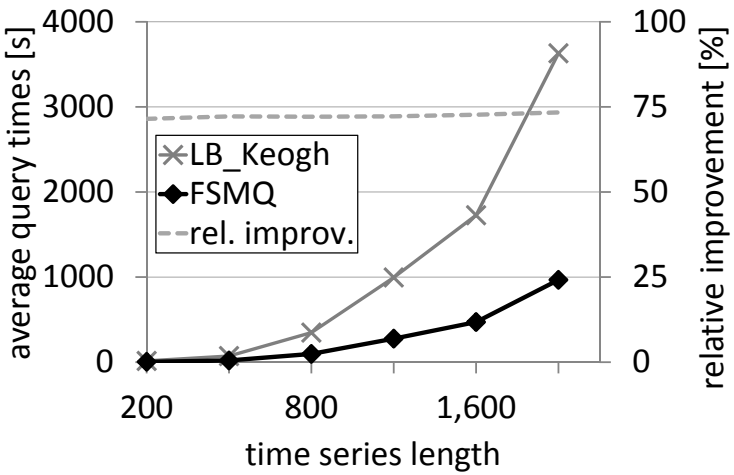
Number of individual queries per multiple query

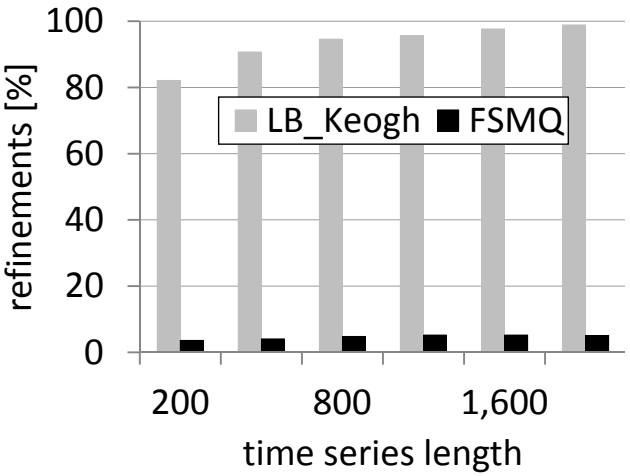




- EEG real world data, *k*NN queries
- the runtime of the single query solution increases much faster.

Scalabilty: Time Series Length





- EEG real world data, *k*NN queries
- Filter LBKEogh has problems with the large scatter in EEG data.
- Grouping by similarity is the method of choice here.

Overview

1. Preliminaries
2. Processing of Multiple DTW Queries
3. Experiments
4. Conclusion

Conclusion

In this talk, I discussed an approach that

- ... processes multiple DTW queries
- ... exploits the **similarity among queries**.
- ... uses a nested **hierarchy of query subgroups**.
- ... is **orthogonal** to existing single DTW query approaches.
- ... guarantees **exact** results.
- ... **outperforms** the baseline solution.

Conclusion

In this talk, I discussed an approach that

- ... processes multiple DTW queries
- ... exploits the **similarity among queries**.
- ... uses a nested **hierarchy of query subgroups**.
- ... is **orthogonal** to existing single DTW query approaches.
- ... guarantees **exact** results.
- ... **outperforms** the baseline solution.

Thank you for your attention.

Questions?