Scalable Model-Based Management of Correlated Dimensional Time Series in ModelarDB$_+$

37th IEEE International Conference on Data Engineering, 2021

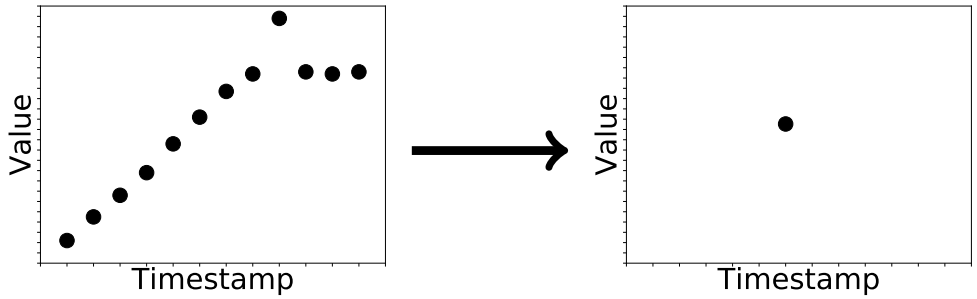
Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen
{skj,tbp,chr}@cs.aau.dk

Center for Data Intensive Systems, Daisy
Department of Computer Science
Aalborg University
Denmark

Motivation

From meetings with manufacturers, owners, and energy traders:

- Turbines have high-quality sensors with wired power and connectivity
- The storage needed makes storing high-frequency sensor data infeasible
- Simple aggregates (e.g. 10-minute averages) are stored instead of the high-frequent series, thereby removing useful fluctuations and outliers:



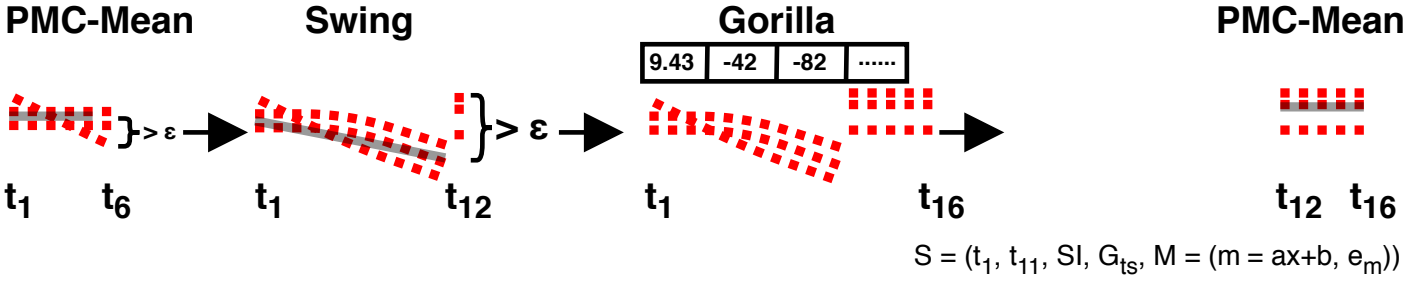
- Many of the collected time series are correlated with each other
- They can be stored within a user-defined error bound (possibly 0%)
- Metadata is also stored and aggregates are the primary query type

Contributions

- Compression of time series groups using multiple model types, we call this type of compression Multi-Model Group Compression (**MMGC**)
- **GOLEMM** the first Multi-Model Group Compression method for time series and model types extended to compress time series groups
- **Primitives** for users to effectively group time series, and a method that **automatically groups time series** using their metadata as dimensions
- Algorithms for executing simple and multi-dimensional **aggregate queries on models** representing values from time series groups
- **ModelarBD$_+$** a version of the open-source distributed model-based time series management systems ModelarDB with our methods added:
  - Available at `github.com/skejserjensen/ModelarDB` under Apache 2.0

Ingestion using GOLEMM

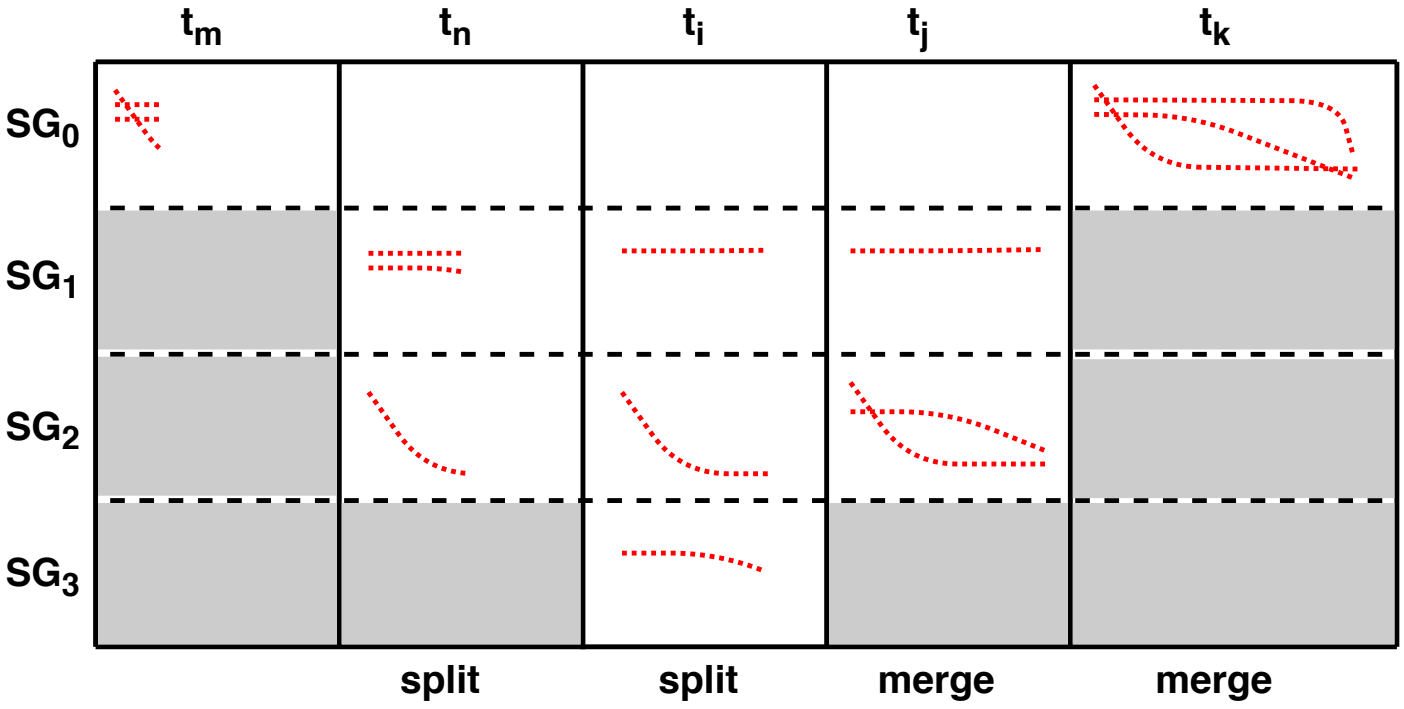
- A model is any representation of a time series group's values from which the values can be reconstructed within a user-defined error bound
- Users can use their domain knowledge, analyze historical data, or use ModelarDB+'s automatic grouping method built on the primitives
- Multiple model types fit models to the data points, e.g., a constant (PMC-Mean), linear (Swing), and lossless (Gorilla) model type:



- The model that provides the best compression ratio is written to disk
- Thus each time series group is split into variable-length segments

Dynamic Grouping

- The time series in a group might temporarily not be correlated:
  - For example, a temperature sensor can be obscured by clouds
- This can be efficiently detected as the compression ratio is reduced
- Dynamically splitting and merging groups remedies this problem:



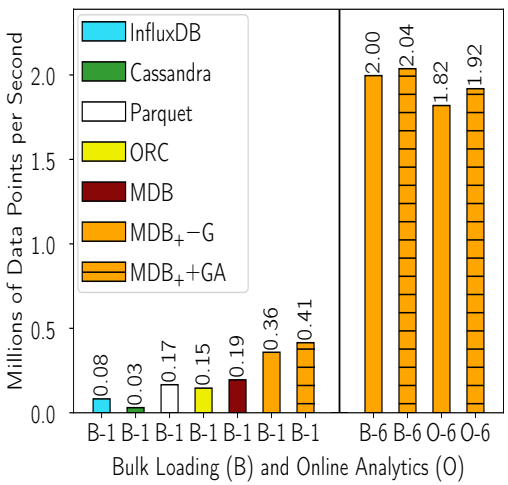
Query Processing

- ModelarDB+ can execute SQL queries on data points and segments
- `<Dimensions>` are denormalized user-defined dimensions
- Data Point View:
  - Executes arbitrary queries on data points reconstructed from models
  - Interface: `Tid int, TS timestamp, Value float, <Dimensions>`
- Segment View:
  - Executes aggregate queries directly on models for better performance
  - Interface: `Tid int, StartTime timestamp, EndTime timestamp, SI int, Mid int, Parameters blob, Gaps blob, <Dimensions>`
  - UDAFs for aggregation on segments are suffixed with `_S`, e.g., `COUNT_S`
  - UDAFs for aggregation over time on segments are suffixed with a time interval, e.g., `COUNT_MINUTE`, `MIN_HOUR`, `MAX_MONTH`, and `SUM_YEAR`

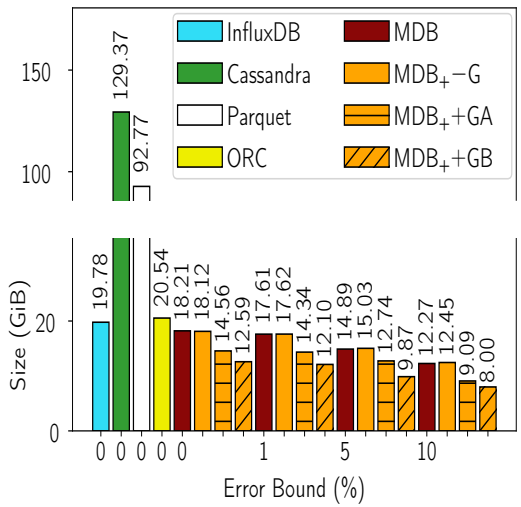
Evaluation Environment

- Evaluation primarily uses real-life data sets from the energy domain:
  - **EP** 45,353 time series collected from energy producers with a sampling interval of 60s and occupying 339 GiB as uncompressed CSV
  - **EF** 197 time series collected from wind turbines with a sampling interval of 200ms and occupying 372 GiB as uncompressed CSV
- Most experiments are performed on a small cluster of commodity PCs
- Scalability experiments are performed using Microsoft Azure
- ModelarDB$_+$ configurations: with no grouping (MDB$_+$-G), with `auto` (MDB$_+$+GA), and with the best primitives per data set (MDB$_+$+GB)

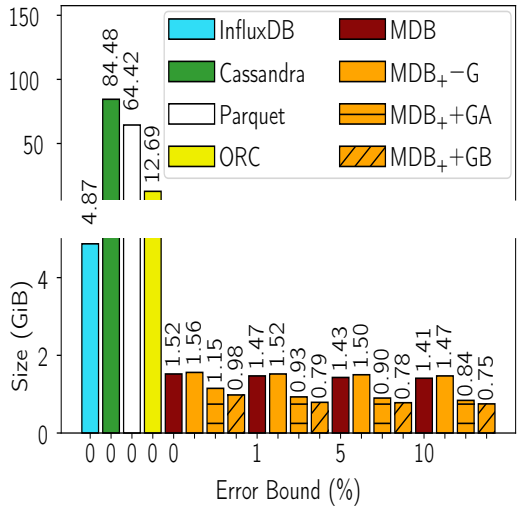
Ingestion and Storage



Ingestion Rate, EP



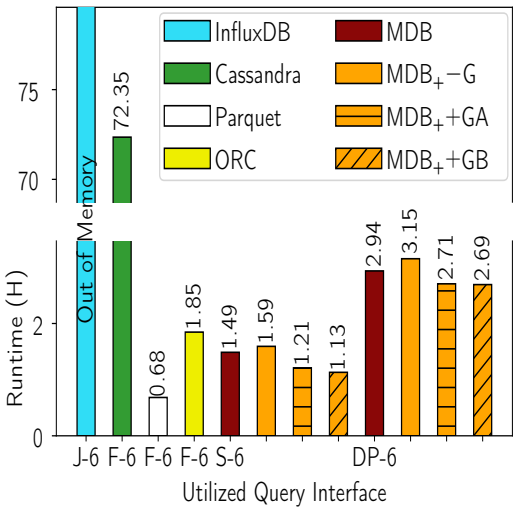
Storage Used, EP



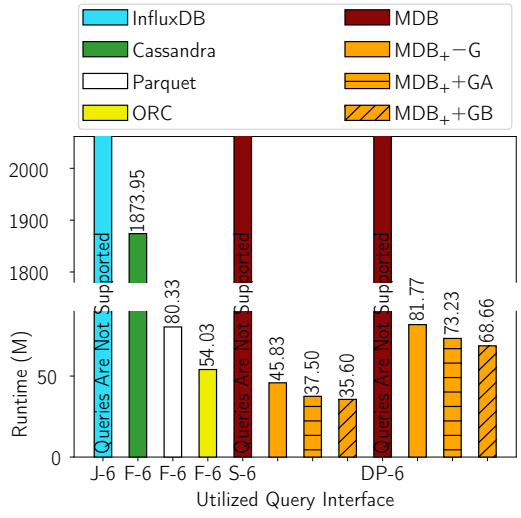
Storage Used, EH

- ModelarDB+ provides both a much higher ingestion rate and requires much less storage than InfluxDB, Cassandra, Parquet, and ORC
- Using multiple model type allows GOLEMM to automatically adapt
- Grouping improves the ingestion rate due to the higher compression
- Creating groups automatically from metadata improves the compression

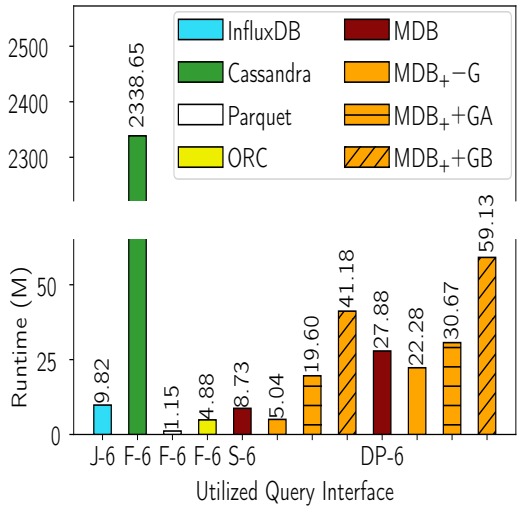
Aggregate Queries



Large Scale, EP



Month and Category, EP



Small Scale, EH

- ModelarDB+ is faster than the other formats when timestamps and values are used, and experiments on Azure show that it scales linearly
- Grouping time series decreases query time for queries that use most or all of the time series in each group (Large Scale / Month and Category)
- Grouping can also increase query time if the query only use a few time series from each group (Small Scale) or if the groups are on one worker

Summary and Future Work

Summary

- Storing sensor data as simple aggregates discards valuable information
- Grouping time series and storing them as models provides multiple benefits over storing them as simple aggregates or raw data points
- We proposed methods for creating (**Primitives**), compressing (**the MMGC method GOLEMM**), and querying time series groups
- The evaluation of ModelarDB+ showed that grouping can provide faster ingestion speed, reduced storage required, and faster aggregate queries

Future Work

- Indexing techniques exploiting that data is stored as models
- Query and cluster aware grouping and partitioning methods
- Support for high-level analytical queries directly on models