Model-based storage and management of massive sensor time series

Christian Thomsen chr@cs.aau.dk

Work done with Søren Kejser Jensen and Torben Bach Pedersen

Center for Data-intensive Systems

The challenge

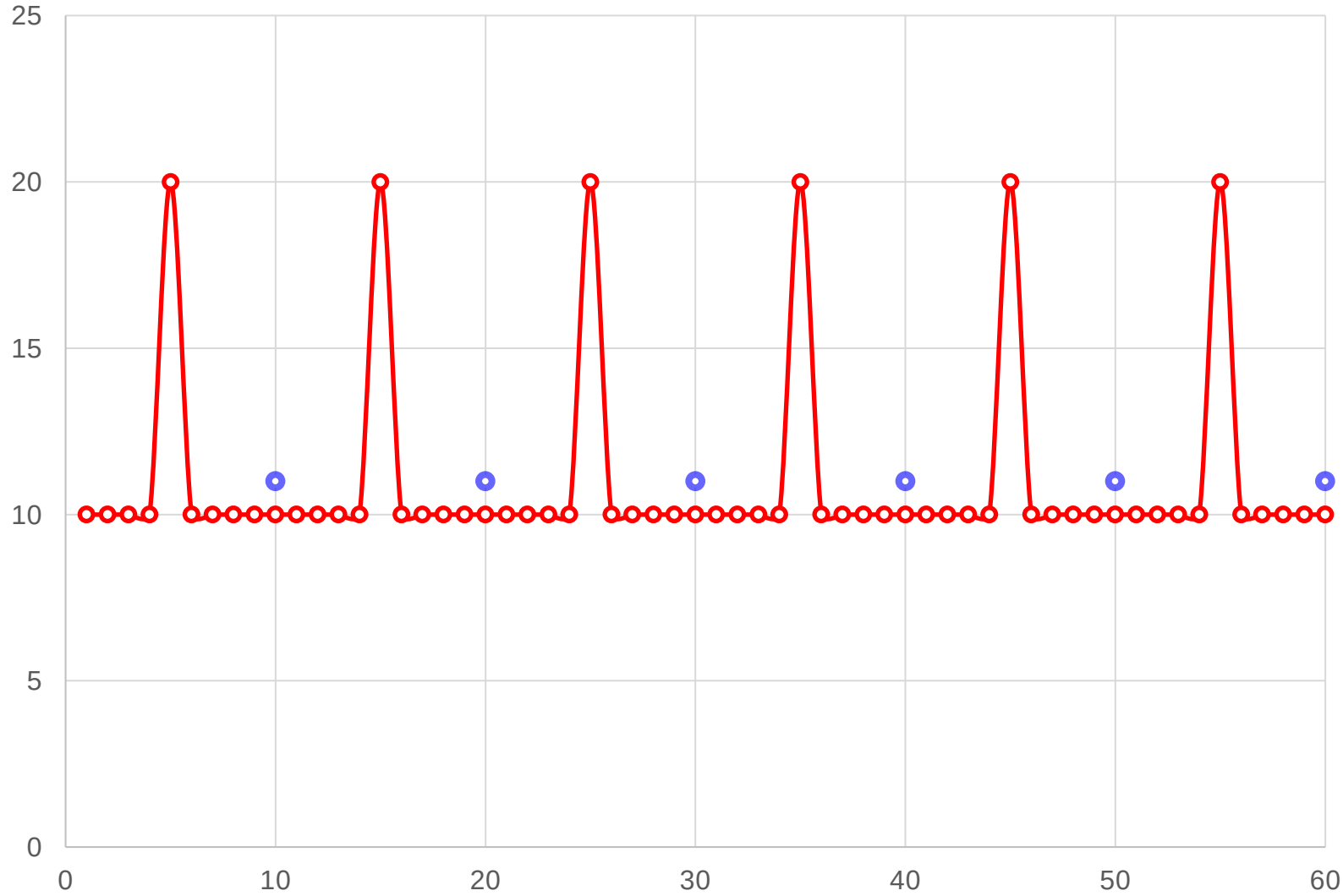
- Wind turbines and solar panels have a lot of sensors that can deliver data values several times per second
- A modern wind turbine has up to 6,500 streams

- This generates a lot of data
  - 10 reads/second, 4 bytes, 6,500 streams ➔ ~20 GiB per day from one wind turbine
  - In practice, some sensors are sampled less often, but a single wind turbine still produces around 1GiB data per day
  - The sampling frequencies and amounts of data to store are increasing

The current situation

- The available information is currently not fully exploited or stored
- Many monitoring solutions consider few (~100) sensor streams and store only a single value (e.g., the average) for every *x* minutes (*x* typically ½, 1, 2, 5, or 10)
- Important things (e.g., failing equipment) might not be seen since outliers and fluctuations can be lost

Example of ”missing the point” :-)



The vision

- Store and use **all** available sensor data
- Support efficient aggregate queries on large amounts of historical data without additional storage costs
- Support analysis of data while it is being ingested
- ➔ Enable detection of underperformance and other problems immediately

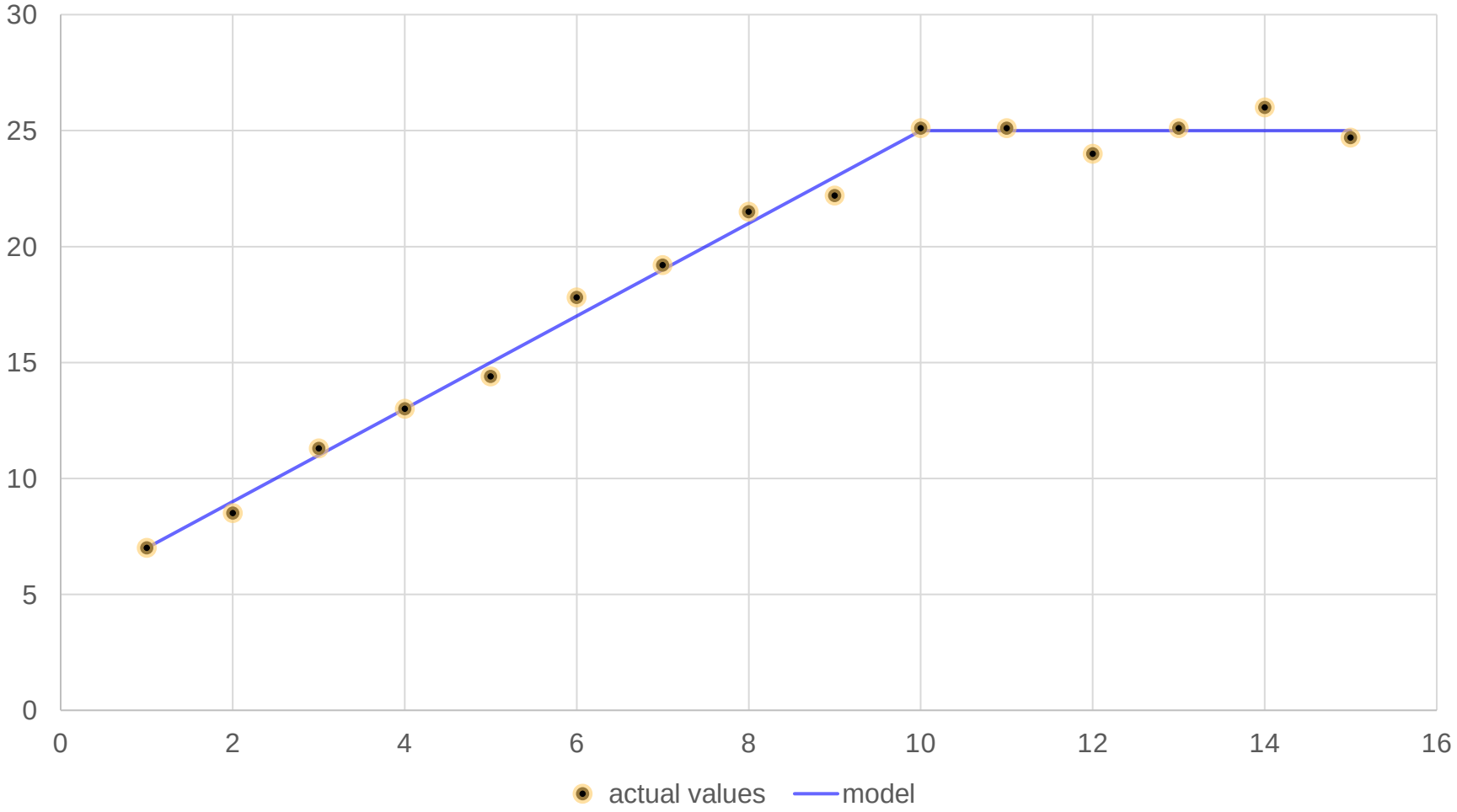
Why is that good? €€€ and less CO2

- More insight from the available data
- Enable predictive maintenance: Detect and fix a problem before the wind turbine breaks down
- Big savings
- Less downtime → more production

How to do it

- Time-series can contain millions of points
- An efficient way to store and process them is to represent them by ***models***, e.g., a linear function
- We use a ***model-based*** approach for the time-series data
- A user-defined error-bound can be set
  - For example, 5%, 1%, **or even 0%**
- Allowing an error in the representation leads to better compression and performance

Simple example of models



ModelarDB

- We have developed the time series management system ***ModelarDB*** which uses models to store time series data

- We have implemented a set of ***model types*** and the user can *optionally* add more
- ModelarDB automatically picks the best model type to use for a given part of a time series, i.e., adapts to the dataset

- Query processing and storage from existing systems
  - Apache Spark and Apache Cassandra
  - H2 RDBMS

Storage requirements for a real-world data

Storage Method	Size in GiB
CSV files	582.68
PostgreSQL 10.1	782.87
RDBMS-X (row)	367.89
RDBMS-X (column)	166.83
InfluxDB 1.4.2	4.33 – 4.44
Apache Parquet files	106.94
Apache ORC files	13.50
Apache Cassandra 3.9	111.89
Model-based ModelarDB	***2.41 – 2.84***

When the error bound is 10%, the actual average error is only 0.005% here!

Summary

- ModelarDB provides model-based compression within an error bound
- ModelarDB adapts to the dataset and compresses well by dynamically choosing among multiple models
- Significant compression and good query performance

- ModelarDB continues to be developed by AAU and the spin-out ModelarData in the H2020 MORE project (AAU, InAccess, ModelarData, and more partners)

More information

- S.K. Jensen, T.B. Pedersen, and C. Thomsen: ”ModelarDB: Modular Model-Based Time Series Management with Spark and Cassandra”, PVLDB 11(11), is available from http://www.vldb.org/pvldb/vol11/p1688-jensen.pdf

- S.K. Jensen, T.B. Pedersen, and C. Thomsen: ”Scalable Model-Based Management of Correlated Dimensional Time Series in ModelarDB+” is available from https://arxiv.org/abs/1903.10269 (preprint) or https://ieeexplore.ieee.org/iel7/9458599/9458600/09458830.pdf

- https://more2020.eu/