Efficient Handling of Large Scale Sensor Data

Valerie Bartel

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Universität Hamburg

valerie.bartel@uni-hamburg.de
Seminar Supercomputer: Forschung und Innovation

14.06.2022

Overview

Sensor Data

Reasons

- Monitor production
- Evaluate cause of errors

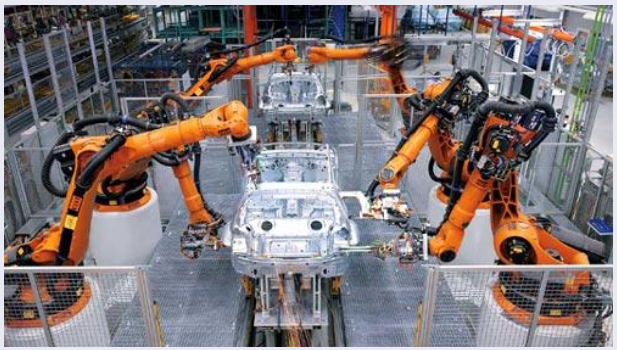

Figure: [3] Manufacturing Robots

Difficulties

- Analyzing data
- Limited bandwidth
- Limited storage
- Limited computing resources

Handling Data

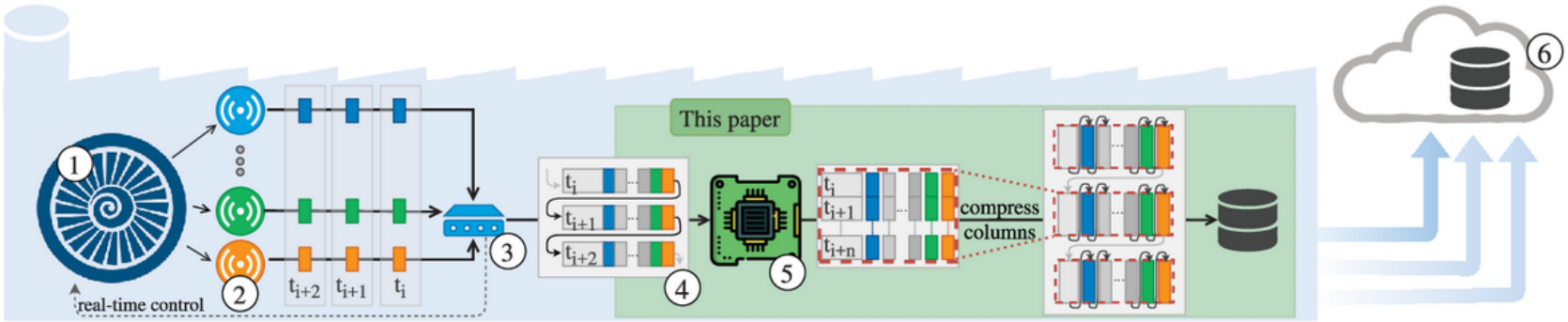

Figure: [1] Pipeline for handling sensor data

4 Layout transformation

5 Byte stream split and compression

Layout Transformation

- Data arrives per time instant
- Sensor orientation yields better analysis
- Easy to compute

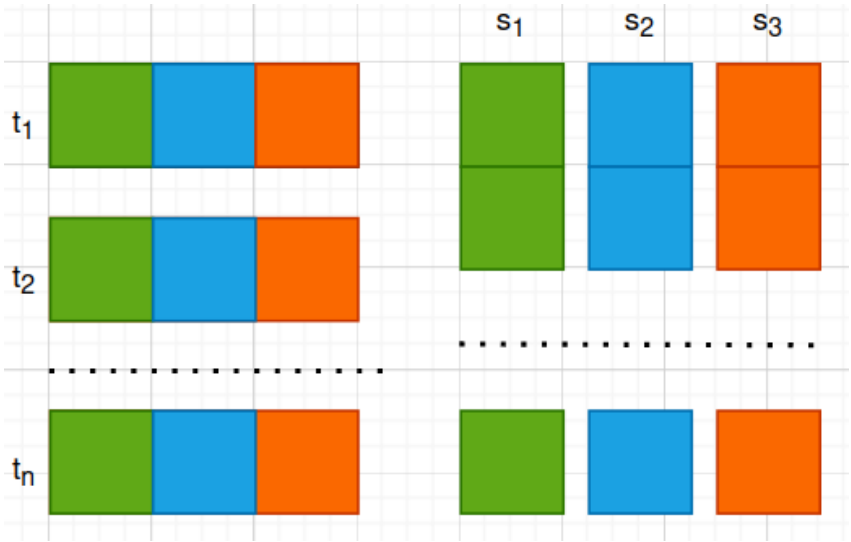

Figure: Row oriented and columnar oriented data

Layout Transformation

Two buffers

- Static number of sensors
- Row group size known
- $buffersize = rgs \times \sum sizeof(type(c))$

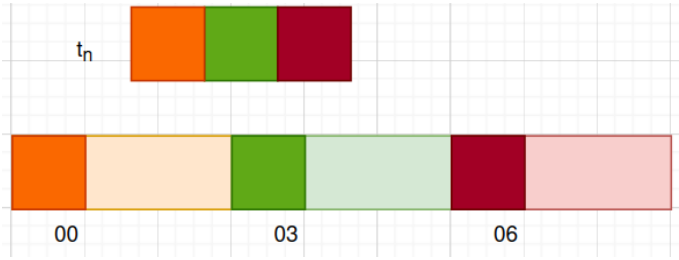

Figure: Reorganizing data of sensors to offsets

1. Mode

Fill buffer with data until full

2. Mode

Prepare buffer for serialization and write to file

Compression Step

1) Byte Stream Split

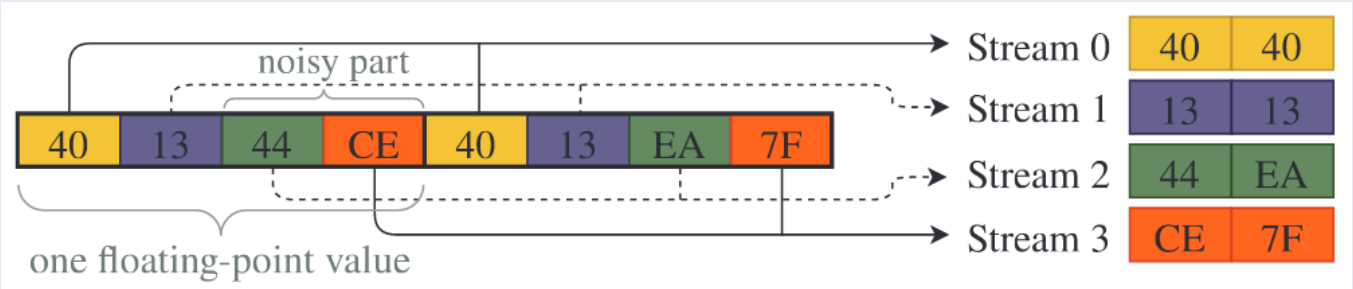

Figure: [1] Byte stream split

2) General purpose compressor

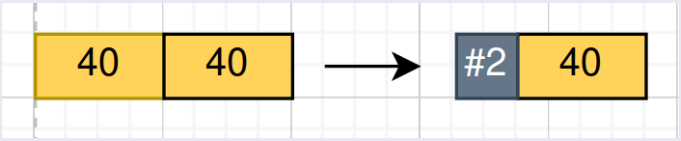

Figure: Compression

Apache Parquet

- Column oriented storage format
- Integrated in several data analytic frameworks
- Build in compressions and encodings
- Page is smallest unit of compression

Implementation of Approach

Layout Transformation

Memory is allocated

1. Fill buffer
2. Create new row group
3. Write columns to row group

Byte Stream Split and Compression

1. Divide row group into pages
2. Use Byte stream split as encoding
3. Use general purpose compressor for compression

Evaluation

Questions

1. Layout Transformation Performance
2. Compression Performance
3. Sustained Throughput Performance

Layout Transformation Performance

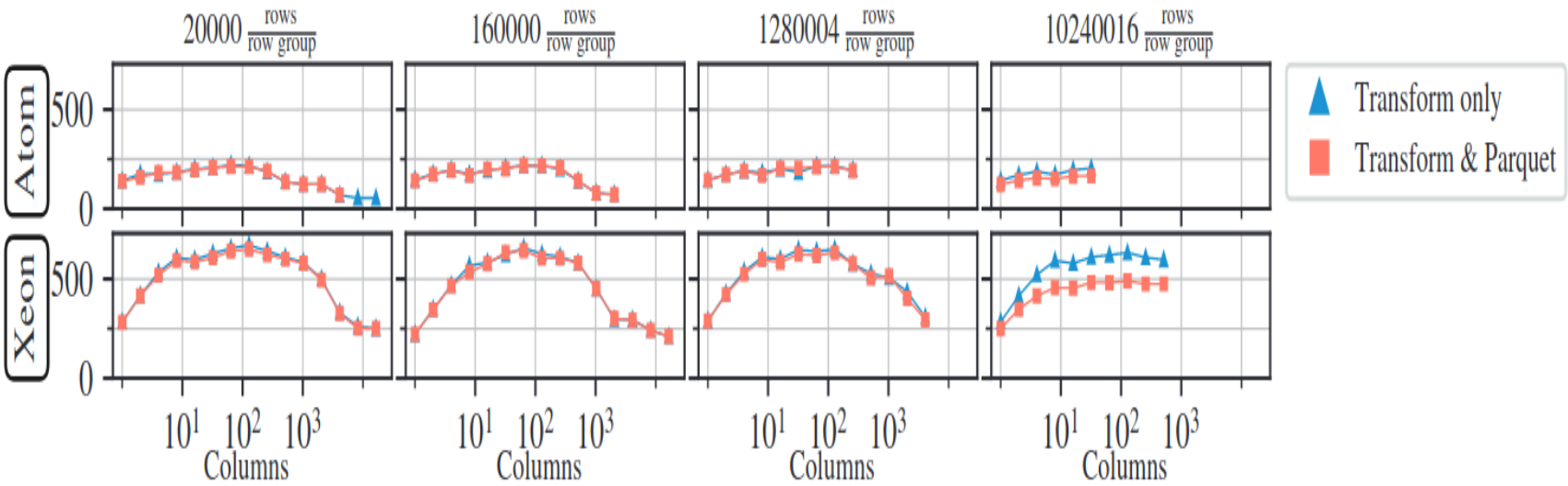

Figure: [1] Layout transformation throughput in MiB/s for different rows/row group ratios

Layout Transformation Performance

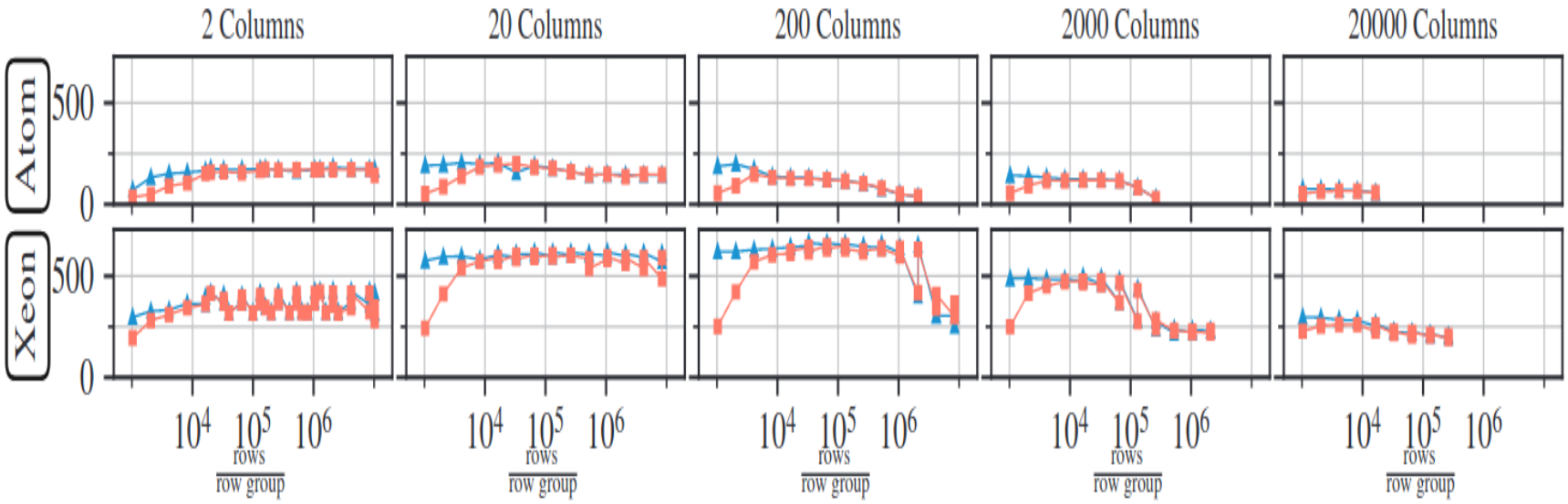

Figure: [1] Layout transformation throughput in MiB/s for different number of columns

Compression Performance

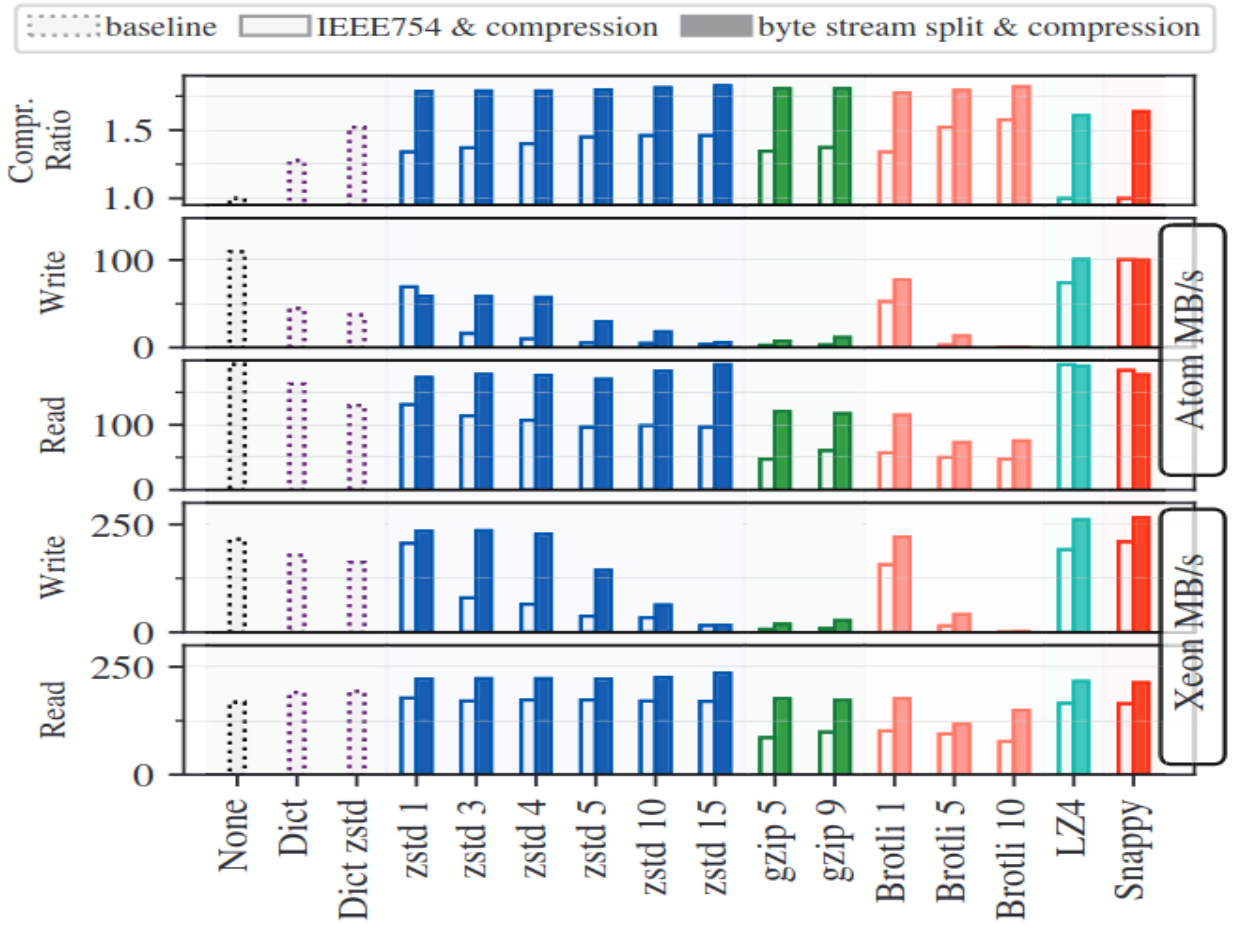

Figure: [1] Compression ratio, write and read throughput for Atom and Xeon of encoding and compression combinations

Sustained Throughput Performance

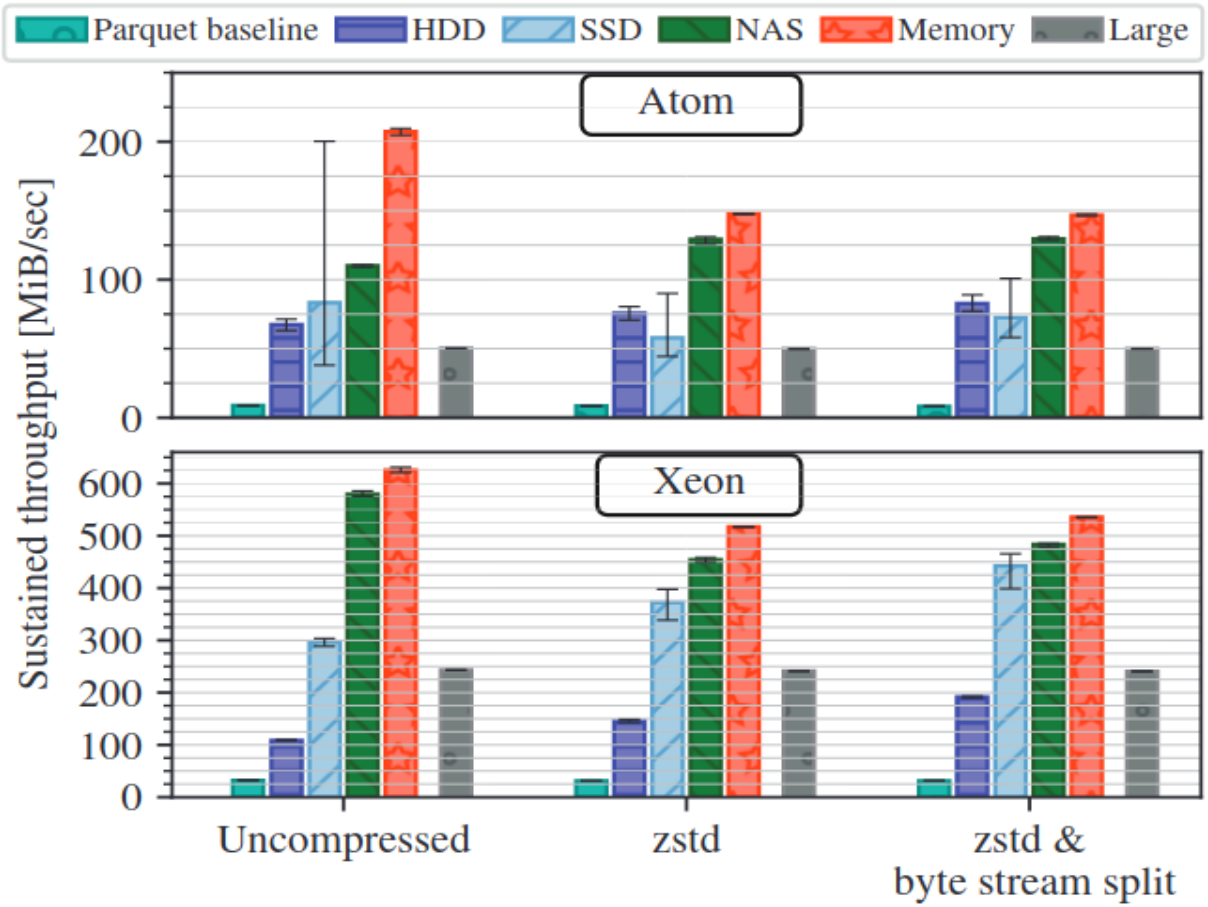

Figure: [1] Average throughput for 200 columns and a row group size of 500.000

Conclusion

- Better compression performance
- Better streaming performance
- Static row group sizes needed

Case Study

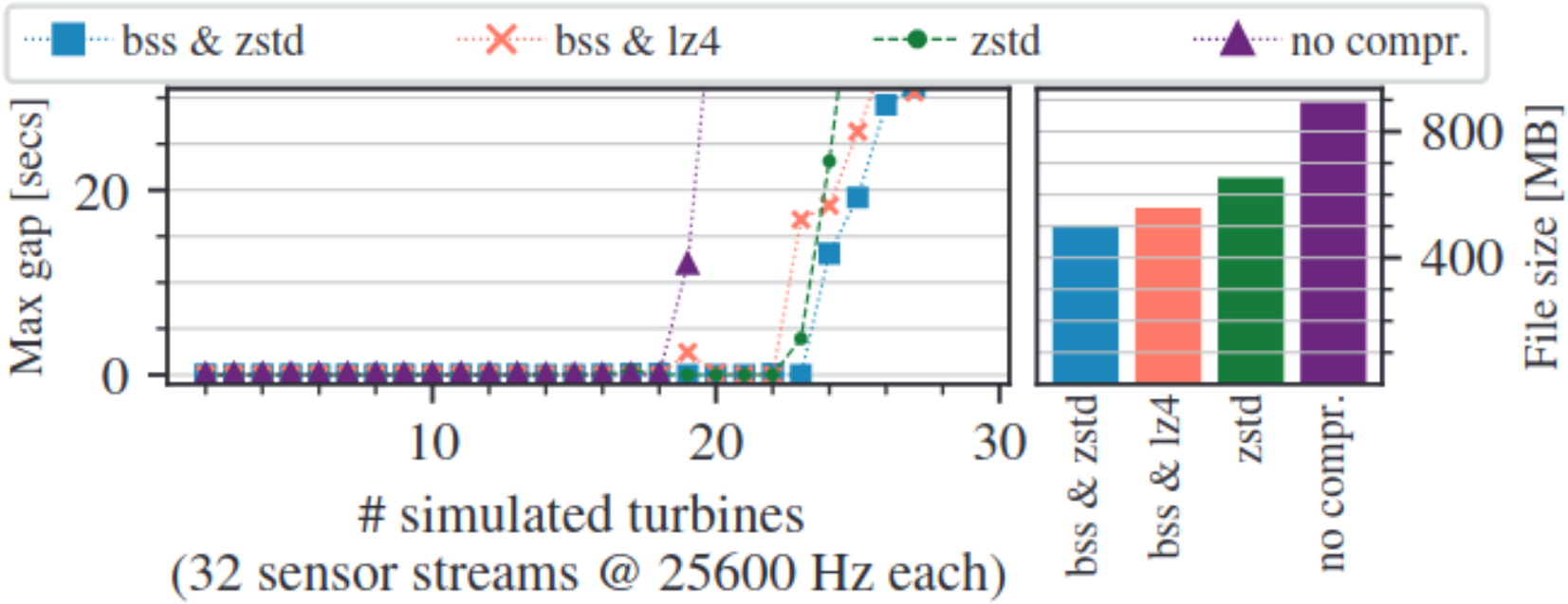

Figure: [1] Maximum gap of turbines

References

Roman Karlstetter, Amir Raoofy, Martin Radev, Carsten Trinitis, Jakob Hermann, Martin Schulz
Living on the Edge: Efficient Handling of Large Scale Sensor Data

Apache Parquet Docs
https://parquet.apache.org/docs/concepts/

Manufacturing robots
https://upload.wikimedia.org/wikipedia/commons/7/7d/Industrial-robots.jpg