PERFORMANCE EVALUATION OF MODERN TIME-SERIES DATABASE TECHNOLOGIES FOR THE ATLAS OPERATIONAL MONITORING DATA ARCHIVING SERVICE

Introduction

Since **P-BEAST**[1] was developed and put into production, new and promising database technologies able to efficiently handle time series data have become available. An evaluation of those technologies was performed in order to assess the possibility of using them in **P-BEAST**.

The **main requirements** for any **potential candidate** were:

- support data types supported by the ATLAS operational monitoring:
  - integers, floats, strings and arrays;
- sustain data injection rates observed during real data taking runs.

A **preliminary survey** was done among time-series database technologies, columnar database technologies and key-value stores. As a result of the preliminary survey, two technologies were **selected**:

- ***InfluxDB***[3] – a time-series database
- ***ClickHouse***[4] – a columnar database

ATLAS operational monitoring data are stored using a *class.attribute* data model. The smallest piece of data stored is a time series data point that represents a value of an attribute. The attributes whose values are stored, always belong to objects. The most important dimension is the number of objects that need to be stored. The class and attribute counts are less variable compared to the object counts.

Implementation

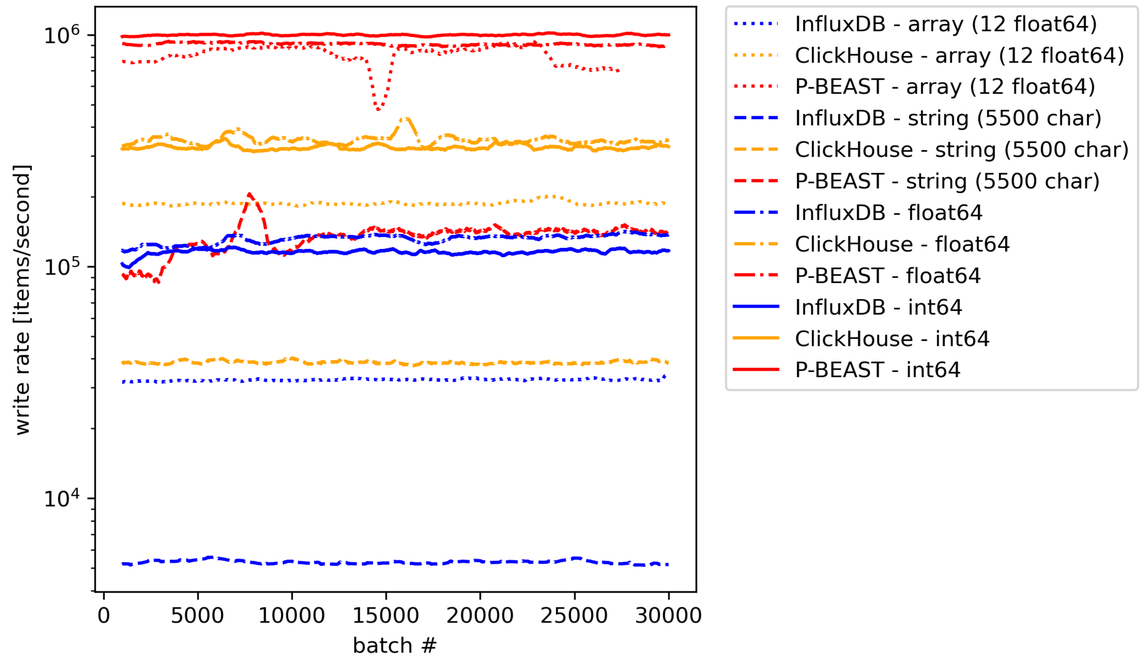
Software

- Performed preliminary tests using different data models for both **InfluxDB** and **ClickHouse** and selected the best one available on both, where, in this case, *best* means the *best write performance*:
  - a single table per attribute, storing all objects of that attribute;
- Ran tests using various real *ATLAS* data archived during *ATLAS Run 2* operations and selected four data types that are the most representative ones:
  - arrays of 12 float64, strings of approximately 5500 characters, float64s and int64s;
- The tests have been implemented in the Go! programming language which is:
  - natively supported by **InfluxDB**
  - supported via third-party libraries by **ClickHouse**

Hardware

- All the tests have been run on a dual-CPU machine with:
  - 2 Intel Xeon E5-2630 v2 @ 2.60GHz CPUs (each with 6 cores and Hypethreading, for a total of **24 threads**) and
  - **32 GB of RAM** and
  - An 18 TB **RAID0** array using hard disk drives

Write speed testing (synthetic)



For each technology, a separate test was implemented and each one was run on four types of ATLAS operational monitoring data. The monitoring data types have been selected in order to have an array of data types as wide as possible.

Writes are done in batches of 10000 records of the respective data types.

P-BEAST stores data using an in-house format based on *Google Protocol Buffers*. Every data file stores time-series data for many objects of a single attribute. Inside a file the data are indexed by *object name*. The data files are compacted and compressed weekly.

InfluxDB: each object is stored in a measurement (InfluxDB table) - the timestamp and a tag (InfluxDB indexed column) containing the object name make up the primary key in each measurement.

ClickHouse: each object is stored in a single table - the columns containing the timestamp and the object name make up the primary key in each table.

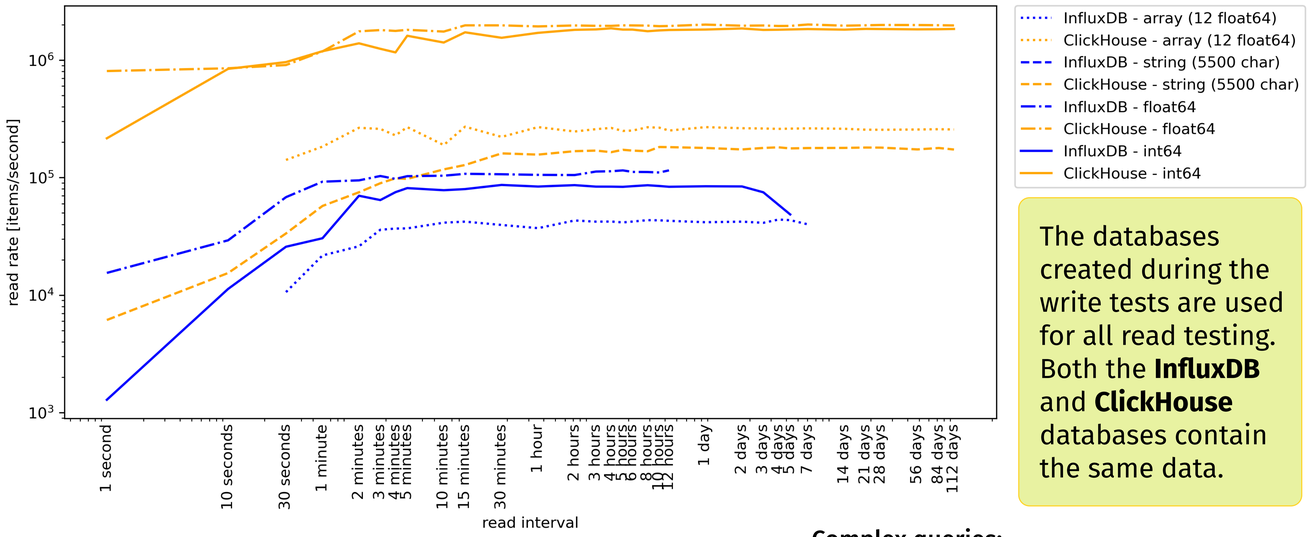
MATEI-EUGEN VASILE[1], GIUSEPPE AVOLIO[2], IGOR SOLOVIEV[3]
[1] IFIN-HH, [2] CERN, [3] UCI

ATLAS EXPERIMENT

PERFORMANCE EVALUATION OF MODERN TIME-SERIES DATABASE TECHNOLOGIES FOR THE ATLAS OPERATIONAL MONITORING DATA ARCHIVING SERVICE

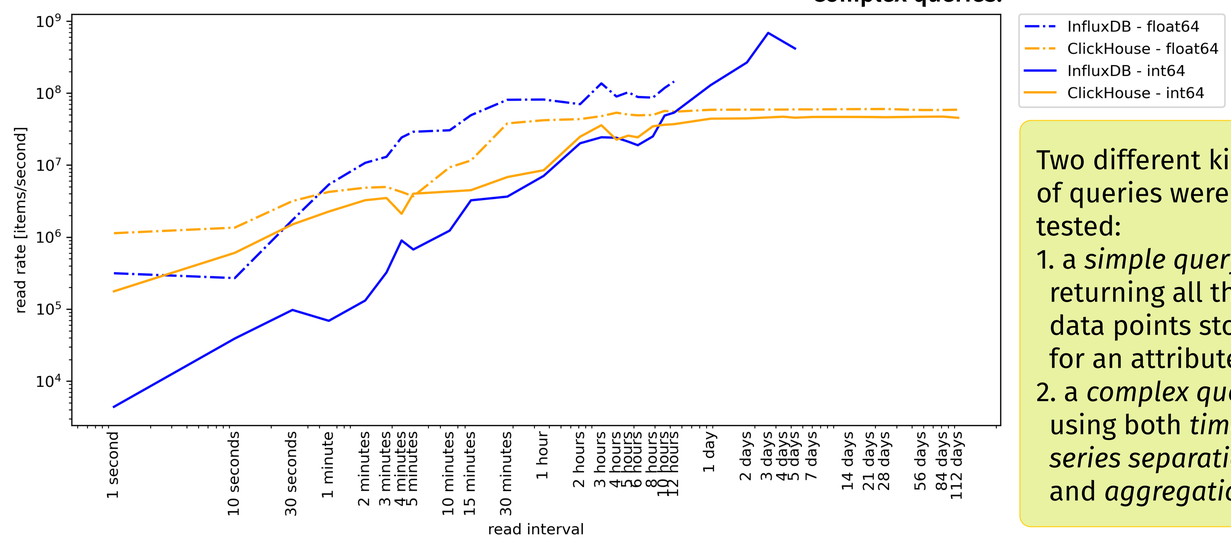
Read speed testing (synthetic)

Simple queries:



The databases created during the write tests are used for all read testing. Both the **InfluxDB** and **ClickHouse** databases contain the same data.

Complex queries:

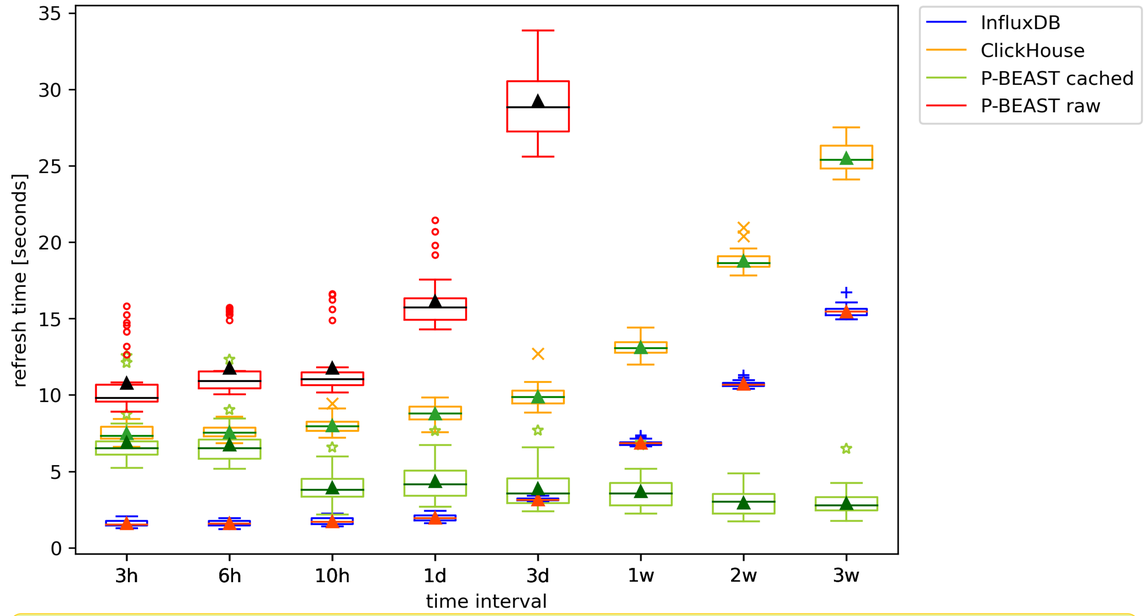


Two different kinds of queries were tested:
1. a *simple query* returning all the data points stored for an attribute;
2. a *complex query* using both *time series separation* and *aggregation*.

- The *simple query* was selected to get an idea of the raw performance of the technology;
- The *complex query* was selected because that is the type of query which will be most commonly used in **Grafana** dashboards.

Complex queries can be run only on basic data types because:
- *Aggregation*, while theoretically possible for strings and arrays, is a more complicated topic which was outside the scope of this work;
- *Time series separation*, although conceptually possible, was impossible to implement for *ClickHouse* because of a limitation of the *ClickHouse* library used for developing the tests.

Read speed testing (Grafana)



A "close-to-real-life" test setup using **Grafana** was developed so that the performance of the database technologies in a production-like setup could be assessed.

A P-BEAST dashboard was used for reference, and InfluxDB and ClickHouse versions were created. The load times of the entire dashboards for time intervals from 3 hours up to 21 days have been measured.

Measurements on both *InfluxDB* and *ClickHouse*, as well as for P-BEAST with caching *enabled* and *disabled* have been done so that a comparison as realistic as possible could be made.

Conclusions

Write performance:

- Both **InfluxDB** and **Clickhouse** are **slower** than **P-BEAST**;
- **ClickHouse** has been consistently **faster** than **InfluxDB**;
- **ClickHouse** comes with an **free and open source clustering support**; **InfluxDB** has **only commercial clustering support**;
  - Clustering support can increase the write speed linearly with the number of computers in the cluster.

Read performance:

- For **simple queries**, **ClickHouse** is **faster** than **InfluxDB;**
- For **complex queries**:
  - **InfluxDB** is **faster** the more data is being processed by the query
  - The *Grafana* read speed testing also show **InfluxDB** to be faster and more consistent
- Both **InfluxDB** and **ClickHouse** are **faster** than **raw P-BEAST**, so **caching** should **increase speed** for both technologies.

MATEI-EUGEN VASILE[1], GIUSEPPE AVOLIO[2], IGOR SOLOVIEV[3]
[1] IFIN-HH, [2] CERN, [3] UCI

ATLAS EXPERIMENT