Impact of ROOT compression algorithms on DAOD_PHYS and DAOD_PHYSLITE data formats

Meeting annuale ATLAS Italia Computing - Milano

30.XI – 01.XII 2022

Caterina Marcon – INFN Milano

INFN MILANO
Istituto Nazionale di Fisica Nucleare
Sezione di Milano

Outline

- Introduction
- Previous results
- Investigations on Autoflush
- Work in Progress & Conclusions

INFN MILANO
Istituto Nazionale di Fisica Nucleare
Sezione di Milano

Introduction

Motivations

- In the coming runs, the LHC accelerator will provide higher luminosity of particle collisions to the ATLAS experiment:
  - more simultaneous collisions per event -> **higher demand of disk space to store the events**;
  - a **larger event rate** will require processing;
  - the need for data compression has grown significantly -> more interest in profiling the compression algorithms provided by ROOT;
  - in this presentation, the impact in terms of **file size and reading speed of different compression algorithms** provided by ROOT on DAOD_PHYS and DAOD_PHYSLITE ATLAS datasets will be presented.

ROOT Compression Algorithms

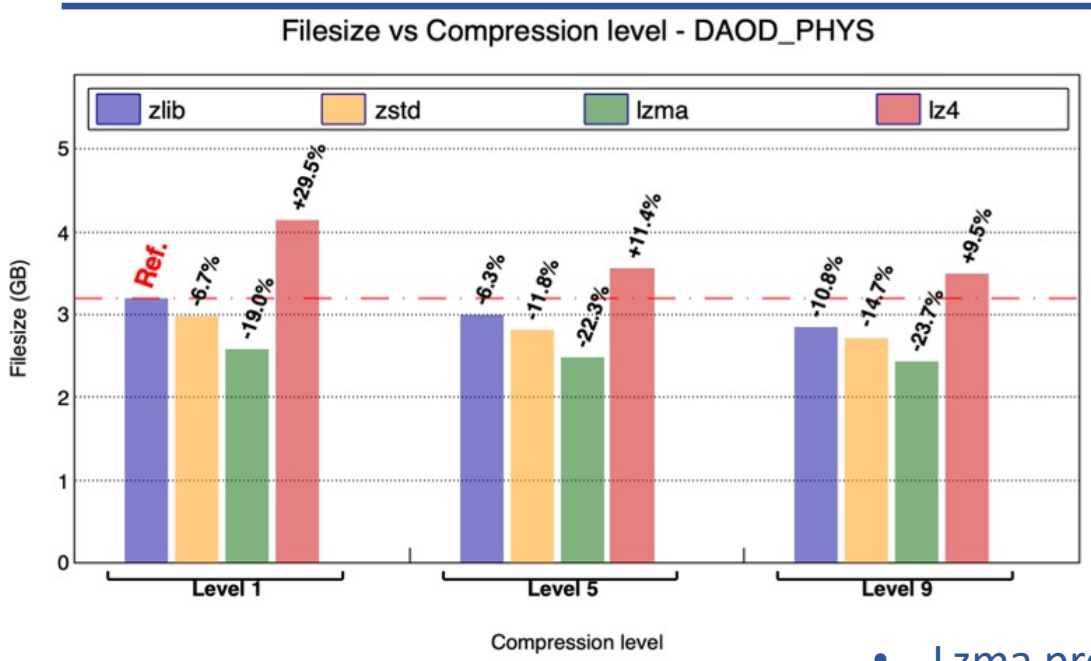
- ROOT provides four different compression algorithms:
  - Zlib;
  - Lzma;
  - Lz4;
  - Zstd.
- All these algorithms can be tuned via the **compression level** option ranging from 1 to 9;
- Higher compression levels offer stronger compression;
- All the algorithms apply **lossless compressions** -> no validation is needed;
- ROOT provides different mechanisms to control how data are written to ROOT files (e. g. AutoFlush and SplitLevel).

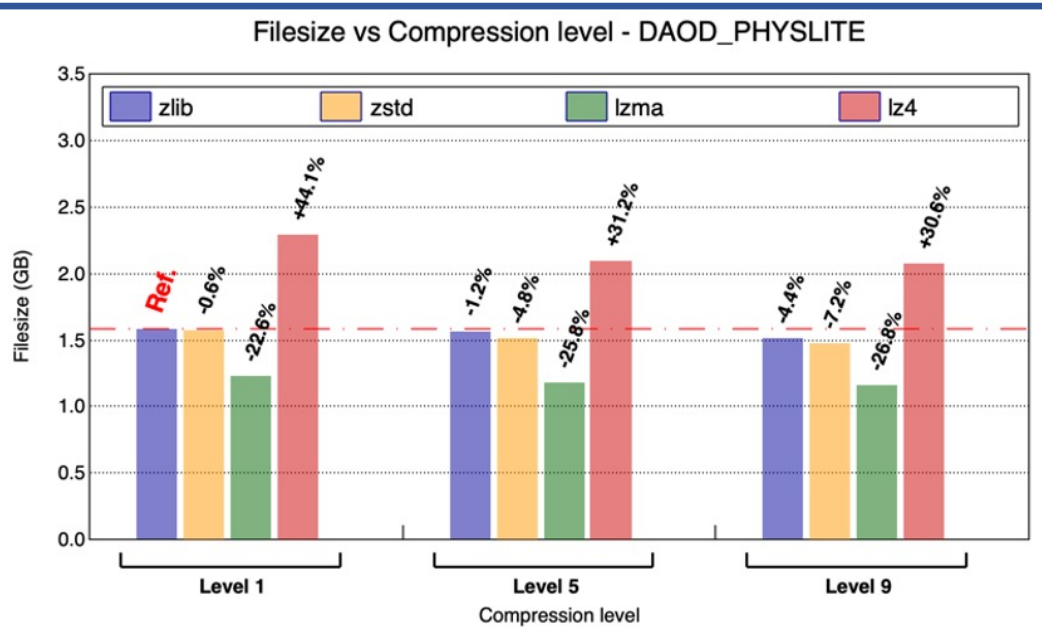
Methods

- Files compressed with a minimal Athena tool (compressionTool) instead of the existing repo;
- Disk-based reading tests allow to collect I/O performance metrics;
- I/O performance metrics are collected via PerfStats (tool provided by ROOT -> access to a range of performance statistics from within the process);
- Reading tests emulate the typical ATLAS data access by reading events from the CollectionTree TTree;
- The CollectionTree object accounts for ~90% of the total file size;
- For file access, the EventLoop backend has been used;
- All tests are carried out using ROOT 6.24, on a dedicated standalone machine;
- Each test has been repeated 5 times and standard deviations are of the order of 3%.

Previous results

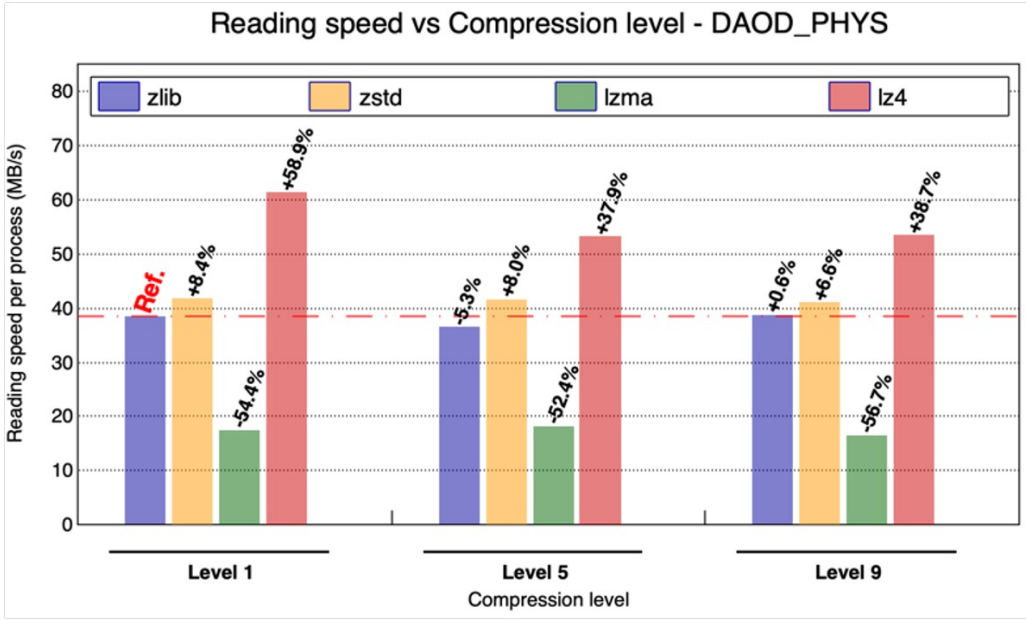
File size VS Compression Level

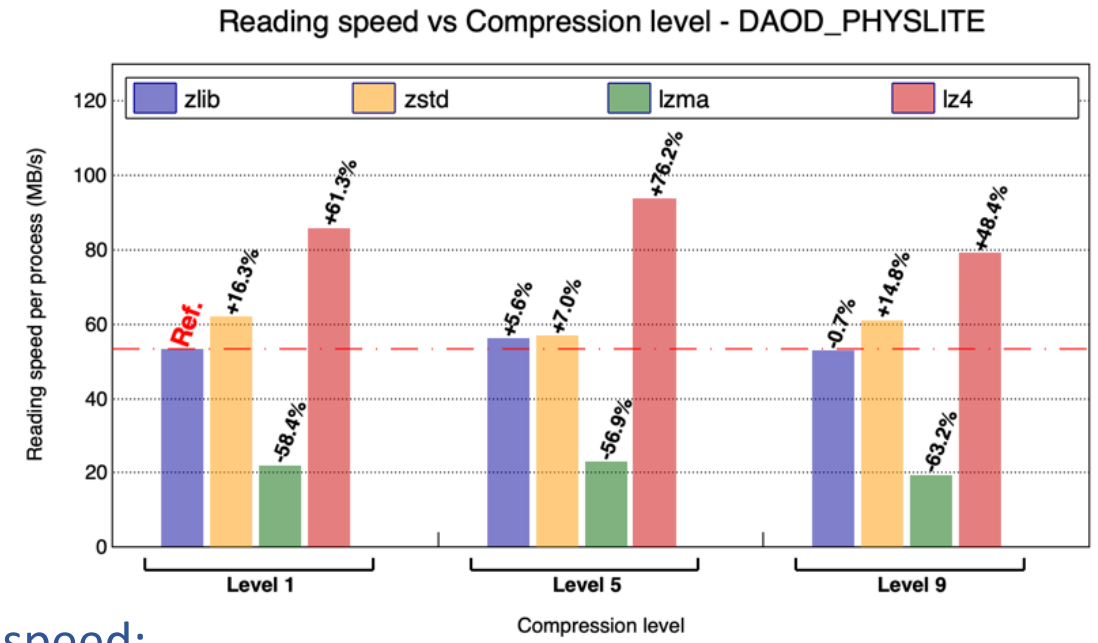




- Lzma provides the **best compression**;
- Lz4 results in the largest files;
- The file size depends primarily on the compression algorithm and not on the compression level.

Reading speed VS Compression Level



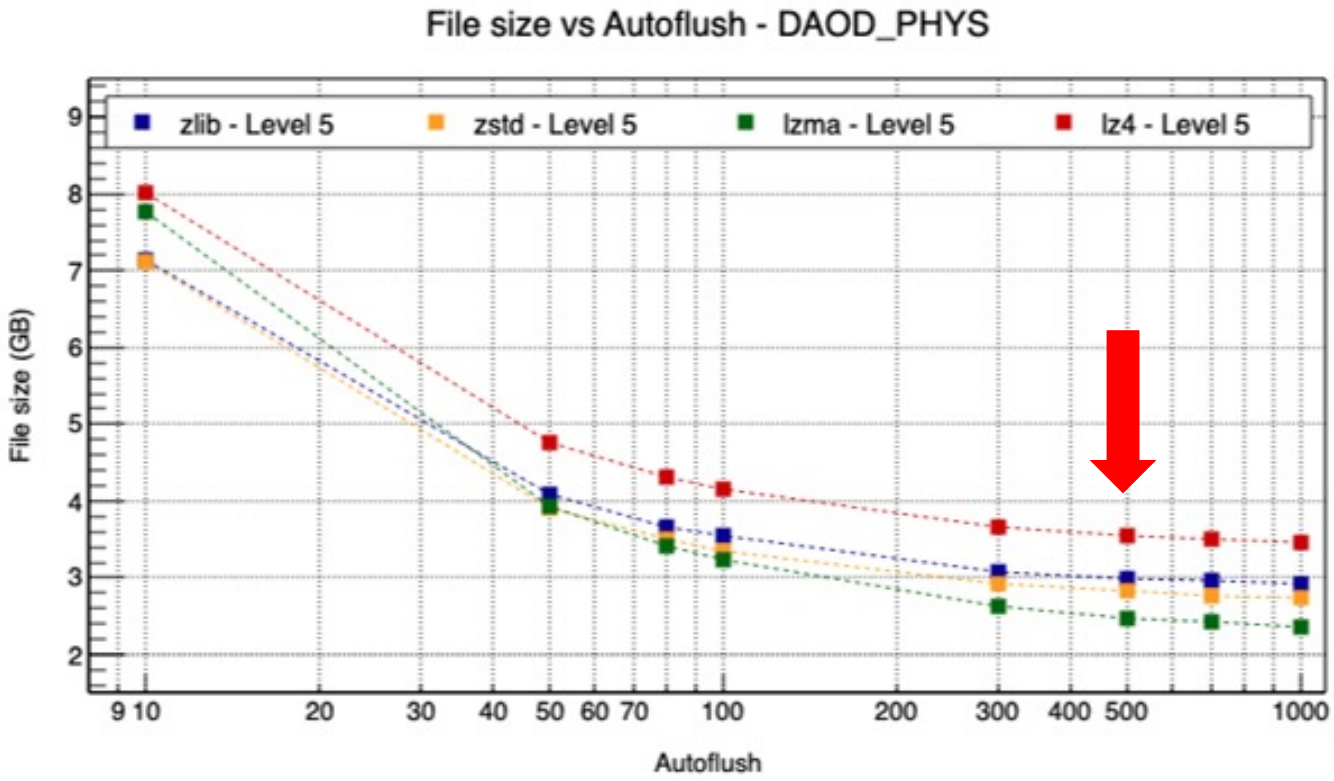


- Lzma has a low reading speed;
- Lz4 is the fastest in reading;
- The reading speed depends primarily on the compression algorithm and not on the compression level.

Autoflush

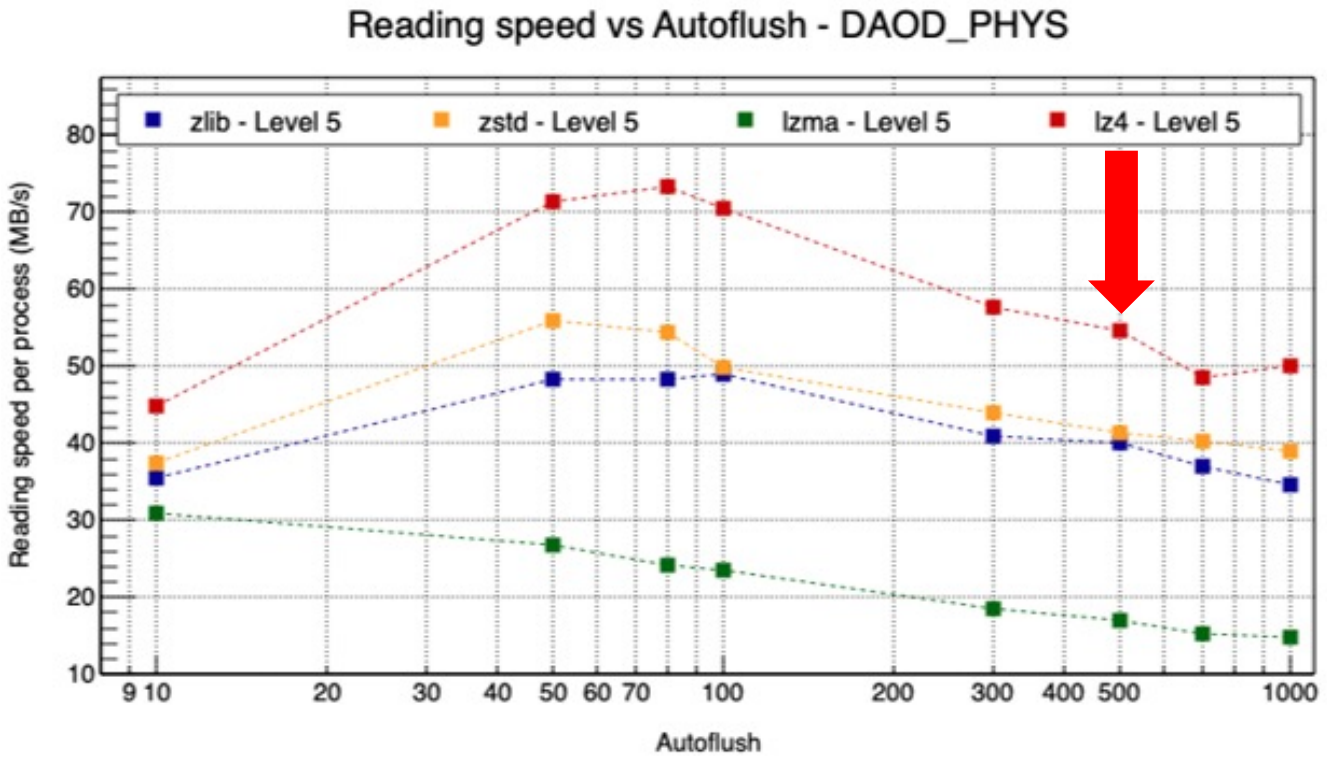
FileSize vs Autoflush DAOD_PHYS

- AutoFlush specifies **how large a single compression unit of a TTree** is in terms of number of events;
- The **original AutoFlush** value of the file is **500**;
- The tests are carried out for all the compression algorithms setting the level to 5.
- Compression algorithms are more efficient with more data to compress;
- **The original AutoFlush value (500), is reasonable.**



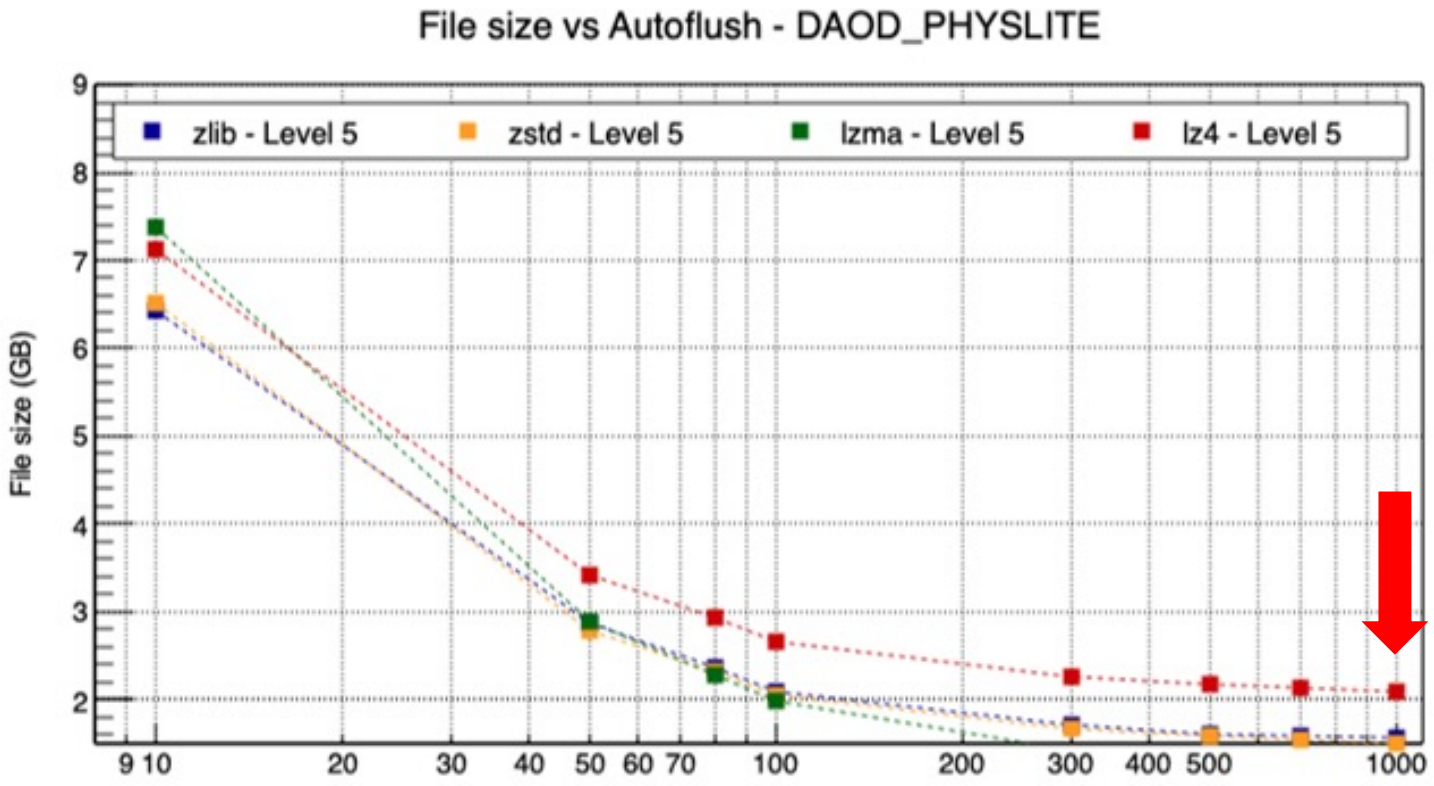
Reading Speed vs Autoflush DAOD_PHYS

- AutoFlush specifies how large a single compression unit of a TTree is in terms of number of events;
- The original AutoFlush value of the file is 500;
- The tests are carried out for all the compression algorithms setting the level to 5.
- Compression algorithms are more efficient with more data to compress;
- The original AutoFlush value (500), is reasonable: it shows a good performance both in terms of file size and reading speed;
- **The behaviour with the visible peak followed by a slow decay is not fully understood.**



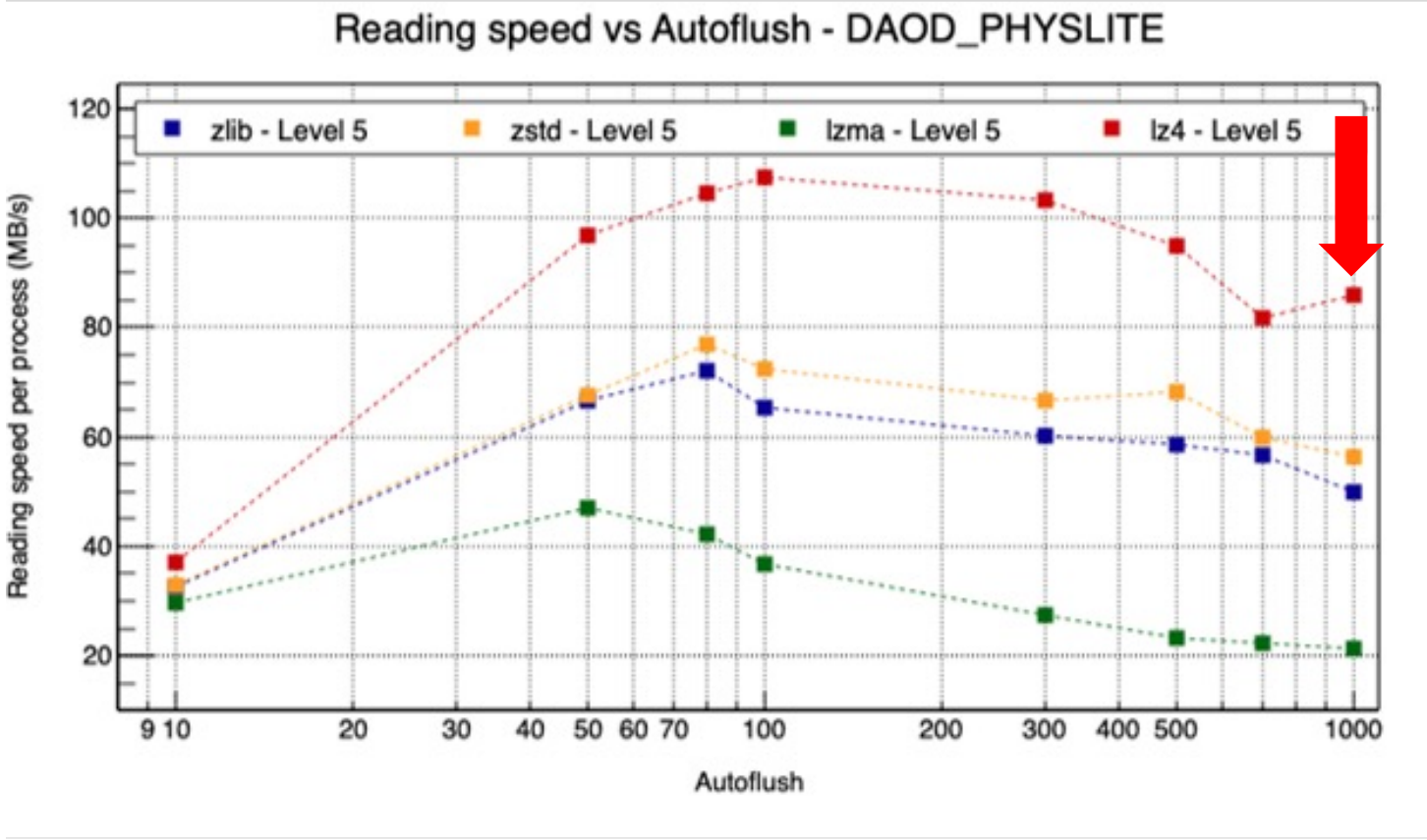
FileSize vs Autoflush DAOD_PHYSLITE

- **AutoFlush specifies how large a single compression unit of a TTree** is in terms of number of events;
- The original **AutoFlush value of the file is 1000**;
- The tests are carried out for all the compression algorithms setting the level to 5.
- Compression algorithms are more efficient with more data to compress;
- The original AutoFlush **value (1000)**, is reasonable;



Reading Speed vs Autoflush DAOD_PHYSLITE

- AutoFlush specifies **how large a single compression unit of a TTree** is in terms of number of events;
- The original AutoFlush value of the file is 1000;
- The tests are carried out for all the compression algorithms setting the level to 5.
- Compression algorithms are more efficient with more data to compress;
- The original AutoFlush value (1000), is reasonable; **although AutoFlush = 500 shows a slightly better performance in terms of reading speed;**
- **The behaviour with the visible peak followed by a slow decay is not fully understood.**



Reading speed vs Autoflush: investigating the behaviour

- Tests have been carried on considering: ttbar physlite sample, reading 20000 events with a ratio of 0.5;
- The results obtained with two different machines have been compared:

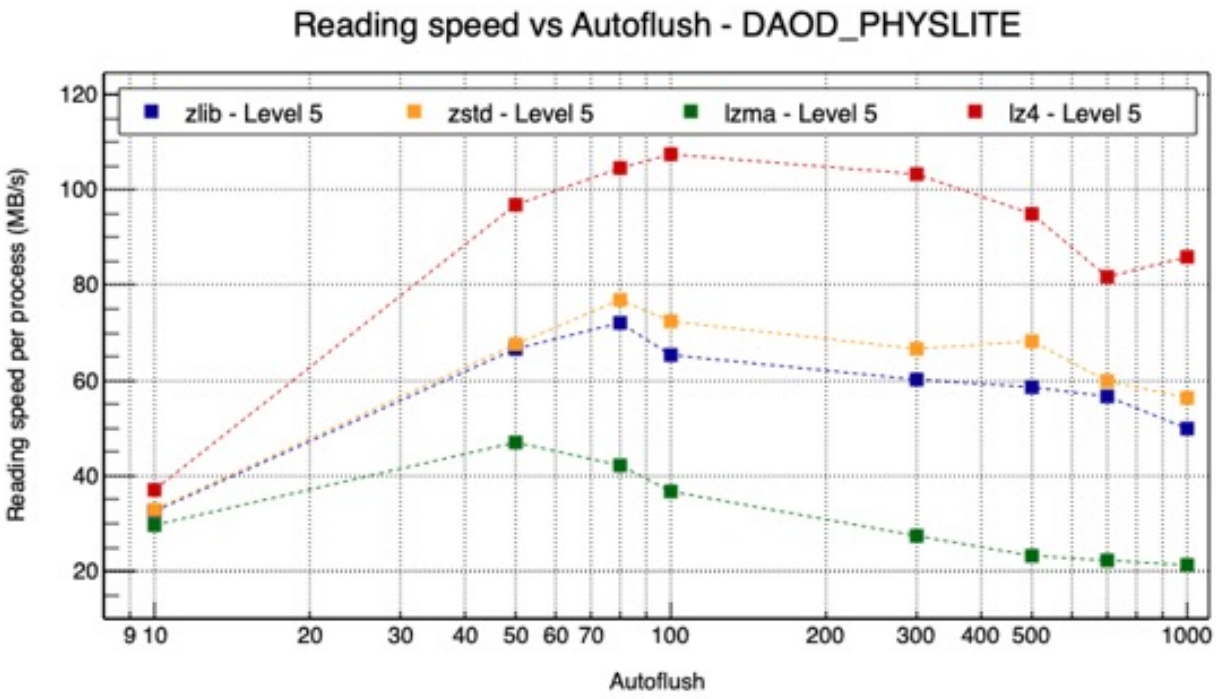
CERN Machine features:

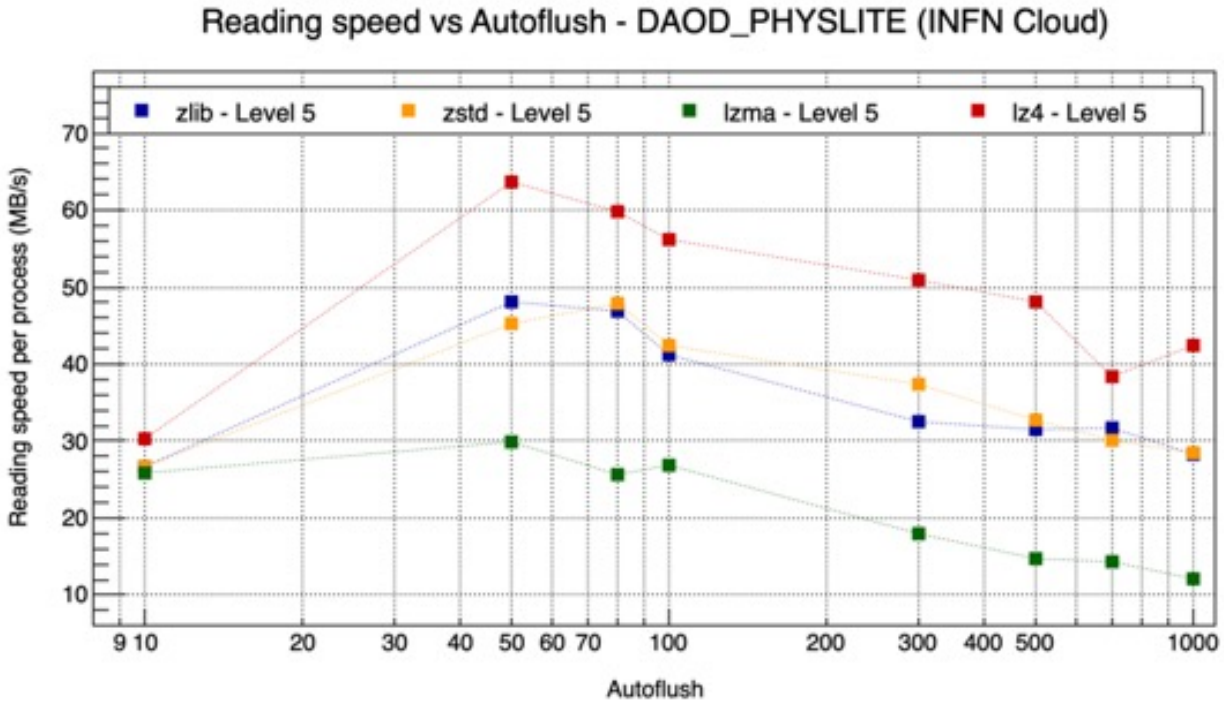
- cpu family : 23
- model : 49
- model name : AMD EPYC 7302 16-Core CPU
- stepping : 0
- microcode : 0x830104d
- cpu MHz : 3000.000
- cache size : 512 KB

Virtual Machine INFN Cloud - Racas Bari features:

- cpu family : 6
- model : 85
- model name : Intel Xeon Processor (Cascadelake)
- stepping : 6
- microcode : 0x1
- cpu MHz : 2199.998
- cache size : 16384 KB

Reading speed vs Autoflush: investigating the behaviour





In both cases, the behaviour is compatible but the position of the peak depends on the features of the machine used.

Work in progress

- **Memory consumption** during compression and reading is being investigated;
- In addition to different **event read ratios, the impact partial event reading (split level) is being studied;**
- **Documentation** will be reviewed and extended;
- Study the combined effect between lossless compression and lossy compression.

Conclusions

Conclusions

- For both types of derived files, **AutoFlush = 500 could be considered a good compromise considering both file size and reading performances.**
- **The behaviour of the Reading speed vs Autoflush seems to depend, in general, on several factors:**
    1) processor speed;
    2) processor cache;
    3) hard drive reading speed;
    4) speed of reading/writing from/to RAM.