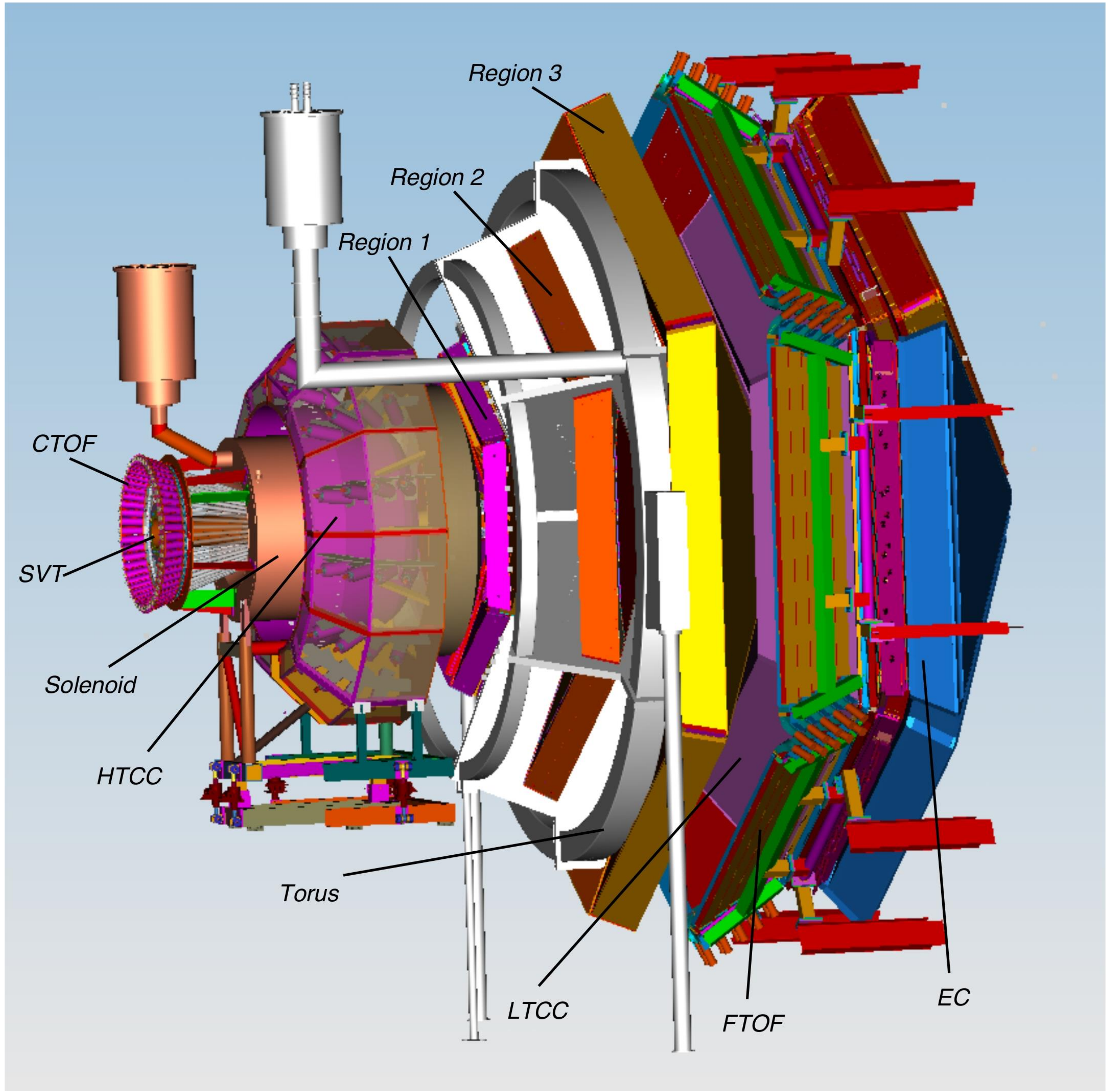
Data Format for CLAS12 Experiment

Gagik Gavalian (Jlab)
CHEP 2019 (November)

Introduction

- Past
  - Hall-B at Jefferson Laboratory was running experiments using Cebaf Large Acceptance Spectrometer (CLAS) with 6 GeV electron beam.
    - On hydrogen target
    - On nuclear target
    - Using tagged photon beam
- Present
  - With the upgrade of Jlab accelerator to 12 GeV CLAS detector was upgraded to run experiments with higher beam energy and higher luminosities.
    - Introduced many new detectors
    - Increased data volume (about 50x)
- Future
  - Larger data sizes demand new approach to data formats

CLAS12 Detector



DETECTOR COMPOSITION:

- Drift Chamber inside Toroidal field for forward tacks.
- Electromagnetic Calorimeter for electron identification and neutral particle detector.
- Time of Flight system for particle identification.
- High Threshold Cherenkov Detector for electron pion rejection.
- Silicon tracker for central detector charged particle tracking in Solenoidal Filed.
- Central Neutron Detector for neutron identification.

DATA ACQUISITION:

- >100K Channels
- DAQ data rate 12 kHz,
- Data rate 400 Mb/sec
- Up-to-Date collected ~1.2 Pb

Data Flow

DAQ

- data acquisition rate 12 kHz
- data format EVIO.
- flush ADC pulses.

DECODE

- apply translation table
- fit ADC pulses
- write beam conditions banks
- write output in HiPO

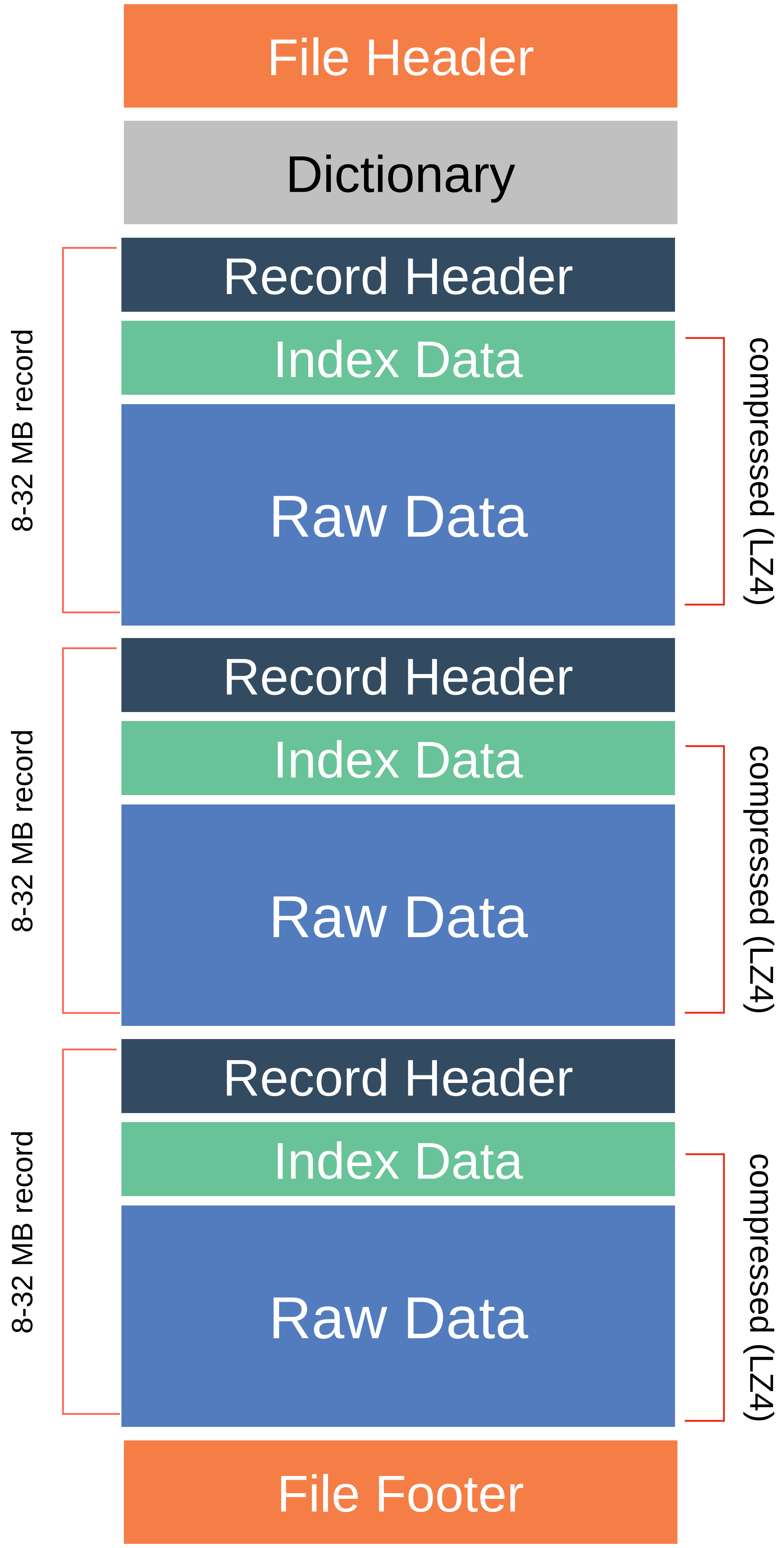
RECONSTRUCTION

- written in JAVA with SOA architecture.
- each detector component is a multithreaded micro-service.
- services interact with data in HiPO format
- output is DSTs in HiPO

- Early in development limitations of DAQ format were noticed:
  - no compression
  - no random access
  - highly inefficient in IOPS
- New Data format was developed (High Performance Output):
  - highly indexed file format
  - compression enabled
  - separated records for different types of data
- Requirement:
  - JAVA interface
  - C++ interface

Stages	Data Size TB
DAQ	2000
DECODE	500
RECONSTRUCTION	200

File Structure



File Header:

- File metadata - version, compression etc.
- Dictionary for banks stored in the data
- Location of File Footer

Record Header:

- record metadata - version, compression and tags
- number of events and record length, index array length

Index Data:

- relative position of each event in the record

Raw Data:

- collection of events of any type

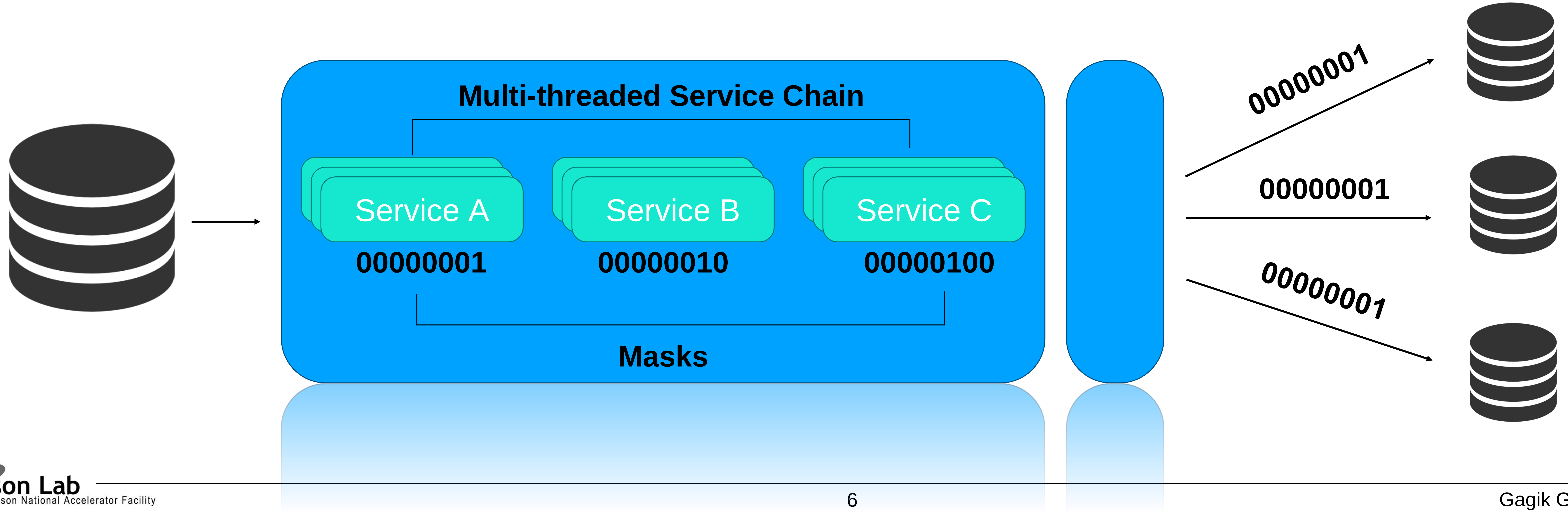
File Footer:

- location of each record and their tags
- number of events in each record

Data Trains

DATA TRAINS

- Data trains are used to select data for different physics analysis
- Output is written in separate files depending on physics final state
- Also used for selecting useful events for calibration of different detectors
- Has to be done several times while calibrating

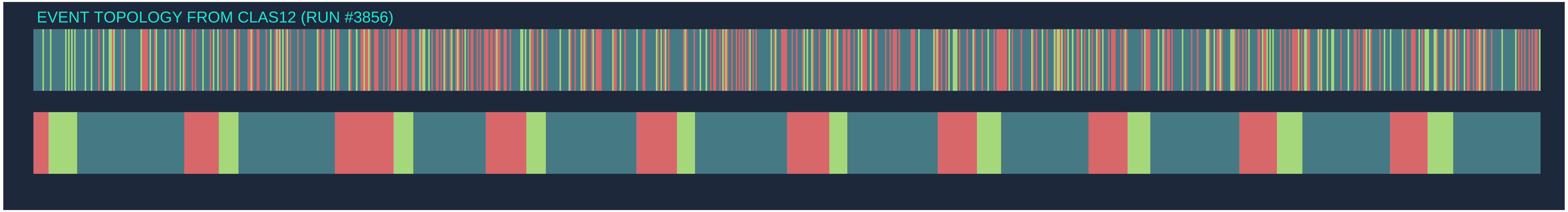


File Structure (Event Tagging)

Event Tagging:

- Event are tagged in reconstruction stage.
- Each tag is written in separate records
- Record reading sequence is initialized by user request.
- Detector diagnostics data is kept in separate records for checks.

- Analysis groups can receive files containing several final states for analysis
- The data for each analysis can be read separately.
- Experimental conditions, such as beam helicity and beam charge are common for all analysis, and are present in the file.

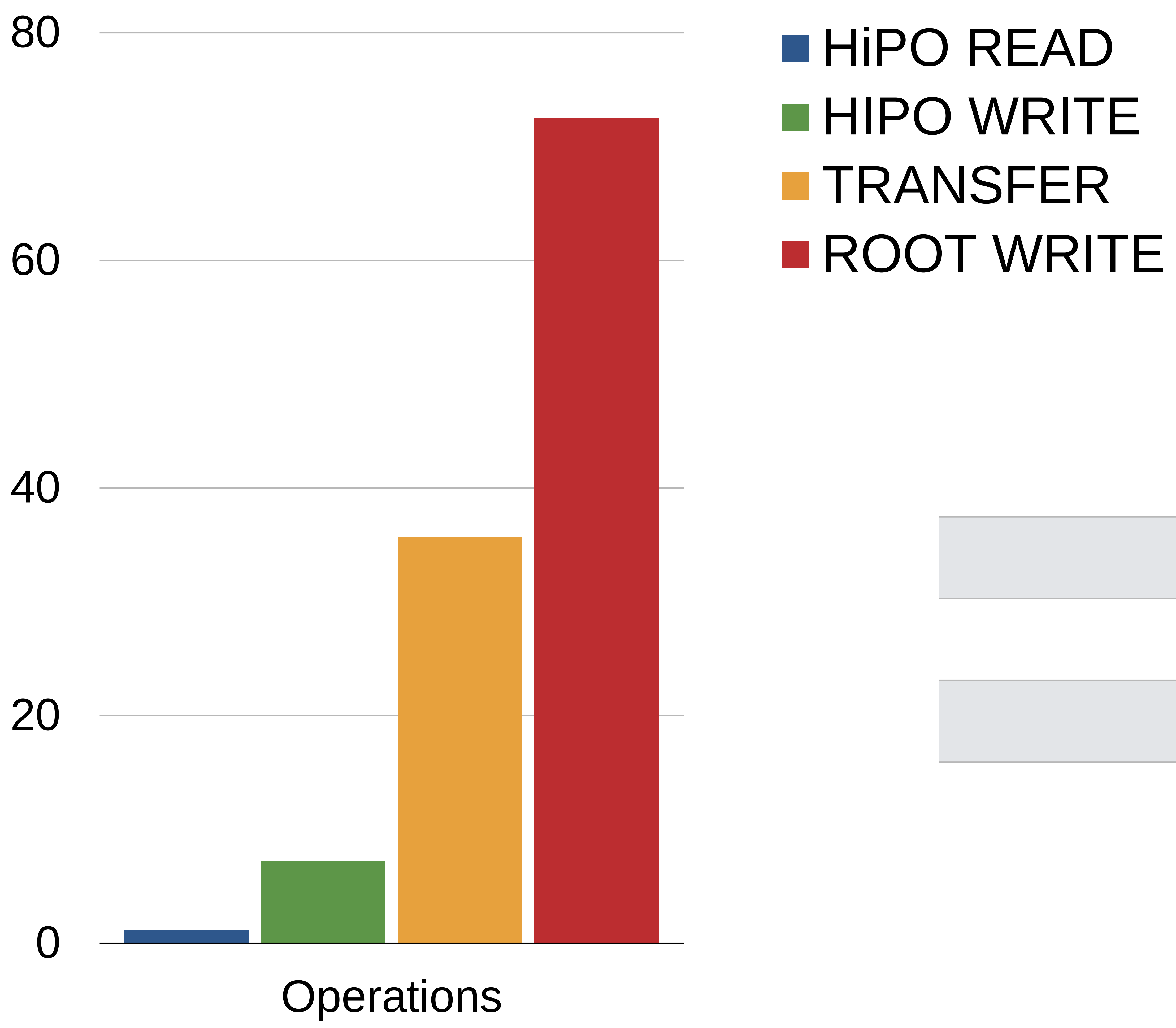


59.7% Trigger particle is not an electron. No electron Forward Tagger.

25.6% Electron trigger. Forward Detector

14.7% Forward Tagger No Electron in ECAL

HiPO 2 ROOT conversion

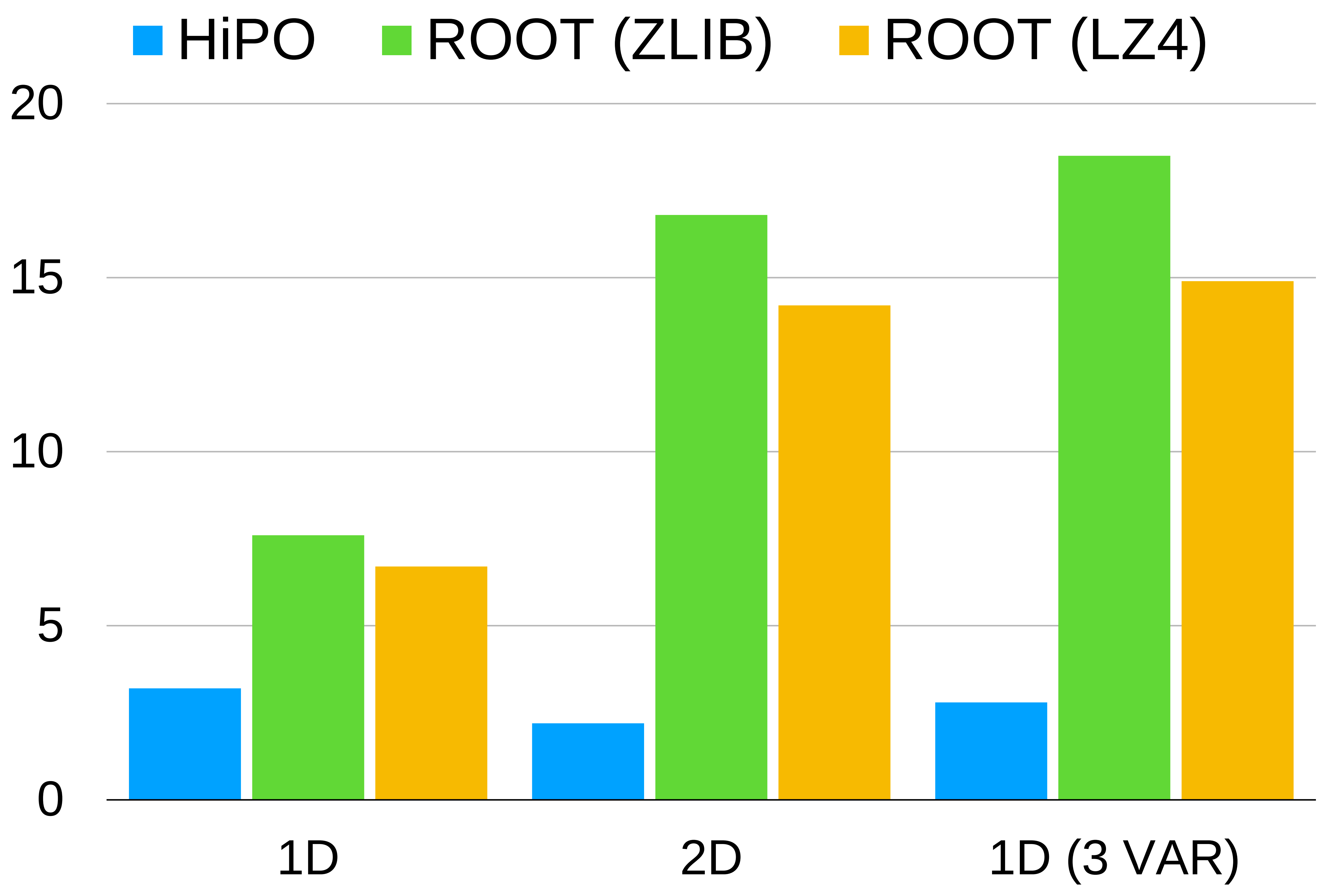


- Converting HiPO to ROOT
  - Read all branches in HiPO file
  - Transfer all columns and rows into std::vector
  - write ROOT file with branches as vectors

Operation	Time (sec)
HiPO Read	1.5
HiPO Write	7.2
Transfer Structures	35.5
ROOT Write	72.5

ROOT vs HiPO Benchmarks

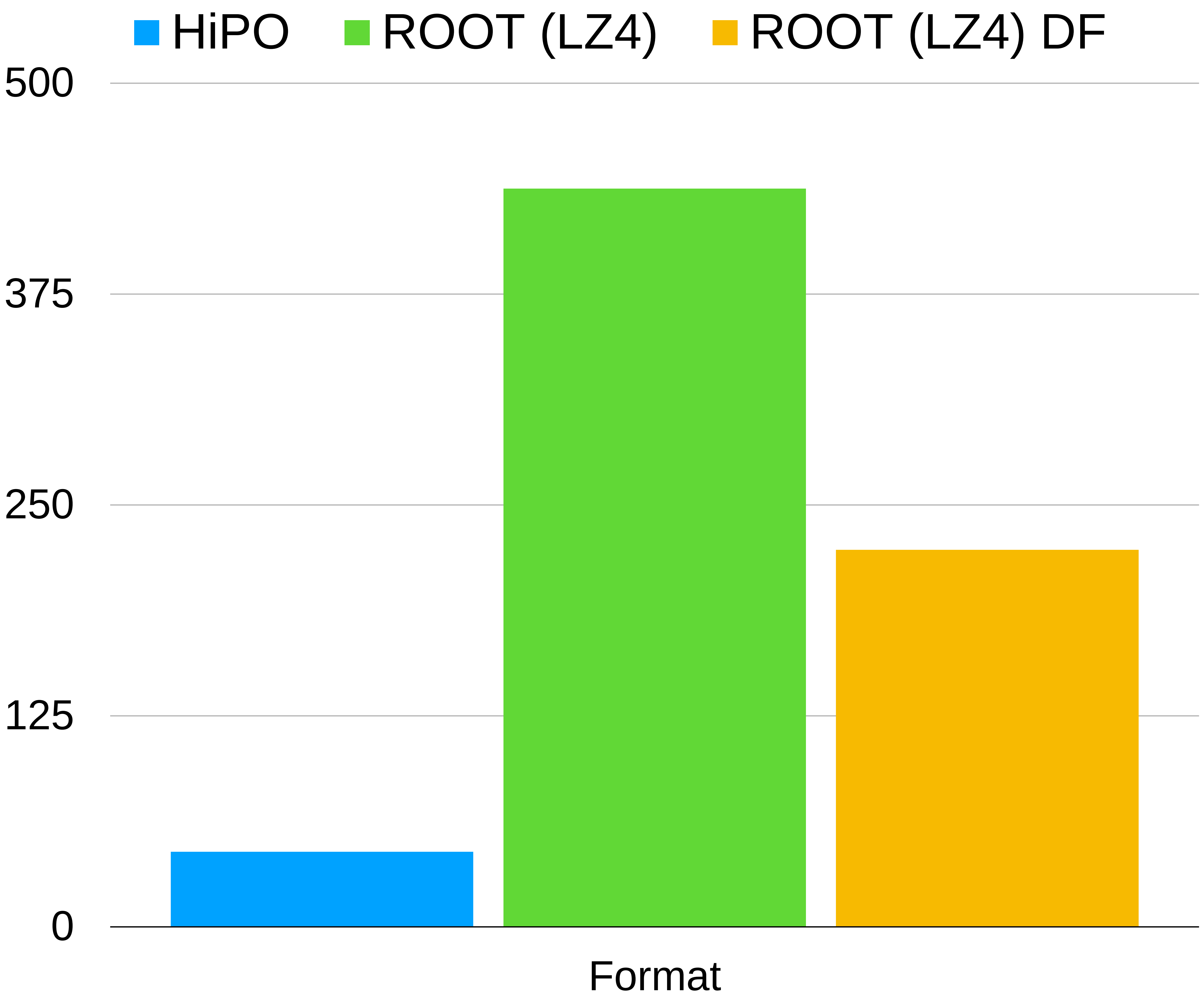
- **Interface**
  - C++ interface was developed extending ROOT classes to read HiPO files from ROOT
  - Expression parsing interface for plotting directly from HiPO.



Format	Compression	File Size	Events
HiPO	LZ4	1.48 GB	6.3 M
ROOT	LZ4	1.95 GB	6.3 M
ROOT	ZLIB	1.60 GB	6.3 M

- **1D - plotting 1 variables:**
  - HiPO reads all branches
  - ROOT reads 1 branch (1/10 of data)
- **2D - plotting 2 variables:**
  - HiPO reads all branches
  - ROOT reads 2 branches
- **1D (3VAR) - plotting 1d histogram calculated from 3 variables:**
  - HiPO reads all branches
  - ROOT reads 3 branches

ROOT vs HiPO Benchmarks (Data Frames)



Format	Compression	File Size	Events
HiPO	LZ4	7.42 GB	32.4 M
ROOT	LZ4	8.00 GB	32.4 M

- 1D - plotting 1d histogram from 8 variables:
  - HiPO reads all branches
  - ROOT reads 8 (out of 12) branch

Summary

- Data format
  - new data format is developed for transient data for CLAS12 detector, features:
    - full random access
    - compression (LZ4)
    - record tagging and event type separation
- Performance is better than ROOT:
  - data sorting and skimming is done using HiPO format
  - a ROOT interface is developed for plotting data
  - analysis can be done in ROOT using C++ interface.
  - final DSTs are stored in HiPO
- ROOT as Analysis File Format
  - is good for small files to do plotting
  - not very efficient to store large data sets and run through them

Backup Slides

ROOT Benchmarks

```
----------------------------------------
**** reader:: header version   : 6
**** reader:: header length    : 56
**** reader:: first record pos : 1224
**** reader:: trailer position : 7427376676
**** reader:: file size        : 7427394804
----------------------------------------
processed events = 32464165, benchmark (WRITE) : time =     433.10 sec , count = 32464165
processed events = 32464165, benchmark (READ)  : time =       7.79 sec , count = 32464165
processed events = 32464165, benchmark (COPY)  : time =     258.73 sec , count = 32464165
processed events = 32464165, benchmark (REST)  : time =       5.58 sec , count = 32464165

 total time =     705.21
```

ROOT Benchmarks

```
treeLZ4->Draw(“sqrt(px*px+py*py+pz*pz)>>LZ3(200,0,10)","pid==11","hist");
```

Elapsed time Root LZ4 calculate momentum of e- : 18.4177

```
treeLZ4->Draw(“pid*charge*sqrt(px*px+py*py+pz*pz)/(vx+vy+vz)>>LZ3(200,0,10)","pid==11","hist");
```

Elapsed time Root LZ4 calculate momentum of e- : 29.4539

```
treeLZ4->Draw(“beta*charge*sqrt(px*px+py*py+pz*pz)*(vx+vy+vz)*status*chi2pid>>LZ3(200,0,10)","pid==11","hist");
```

Elapsed time Root LZ4 calculate momentum of e- : 36.5032

ROOT Benchmarks

ifarm1801

```
----------------------------------------
**** reader:: header version   : 6
**** reader:: header length    : 56
**** reader:: first record pos : 1224
**** reader:: trailer position : 7427376676
**** reader:: file size        : 7427394804
----------------------------------------
```

```
processed events = 6492833, benchmark (WRITE) : time =      77.01 sec , count = 6492833
processed events = 6492833, benchmark (READ)  : time =       1.25 sec , count = 6492833
processed events = 6492833, benchmark (COPY)  : time =      37.52 sec , count = 6492833
processed events = 6492833, benchmark (REST)  : time =       0.91 sec , count = 6492833
```