Enabling distributed analysis for ALICE Run 3

Raluca Cruceru[1,*]

CERN,
Espl. des Particules 1, 1211 Meyrin, Switzerland

E-mail: raluca.cruceru@cern.ch

The ALICE Collaboration has just finished a major detector upgrade that increases the data-taking rate capability by two orders of magnitude and will allow to collect unprecedented data samples. For example, the analysis input for 1 month of Pb-Pb collisions amounts to about 5 PB. In order to enable analysis on such large data samples, the ALICE distributed infrastructure was revised and dedicated tools for Run 3 analysis were created. These are firstly the O^2 analysis framework that builds on a multi-process architecture exchanging a flat data format through shared memory implemented in C++. Secondly, the Hyperloop train system for distributed analysis on the Grid and on dedicated analysis facilities implemented in Java/Javascript/React. These systems have been commissioned with converted Run 2 data and with the recent LHC pilot beam and are ready for data analysis for the start of Run 3. This contribution discusses the requirements and the used concepts, providing details on the actual implementation. The status of the operation in particular with respect to the LHC pilot beam will also be discussed.

41st International Conference on High Energy physics - ICHEP2022
6-13 July, 2022
Bologna, Italy

[1]On behalf of the ALICE Collaboration
[*]Speaker

1. Introduction

The ALICE detector has undergone a number of significant upgrades that considerably increased the data-taking capability, as the main detector parts have been replaced and improved. Starting with the installation of a new Inner Tracking System and a new Fast-Interaction Trigger system, the Time Projection Chamber has also been upgraded in the process [1]. Furthermore, a new Online-Offline data taking and processing infrastructure has been developed.

Following these substantial developments, the ability to record data has increased by two orders of magnitude, allowing for 100 times more collisions recorded compared to Run 1 and Run 2. The resulting data throughput from the detector is estimated to be greater than 1 TB/s for Pb-Pb collisions. The reconstruction output, which is the analysis input, has an unprecedented size, too, for 1 month for Pb-Pb collisions it amounts to about 5 PB.

In order to efficiently enable analysis on such large data samples, the ALICE distributed infrastructure was revised and dedicated tools for Run 3 analysis were created. The first one is the O^2 analysis framework [2] implemented in C++, which builds on a multi-process architecture exchanging a flat data format through shared memory. The second tool is the Hyperloop train system for distributed analysis on the Grid and dedicated analysis facilities, which was implemented in Java/Javascript/React. These systems have been commissioned with converted Run 2 data and with the recent LHC pilot beam and are at present used for the analysis of Run 3 data.

2. The O^2 Framework

As dedicated Framework to tackle the challenges of LHC Run 3, O^2 is derived from the ALICE High-Level Trigger architecture – a message-passing multiprocess system used in Run 1 and 2, and allows for distributed and efficient processing of the new amount of data [3]. It involves two main processing stages: the synchronous processing (online), resulting in the CTF (Compressed Time Frame) - stored on the disk buffer, and the asynchronous stage (offline) that performs the reconstruction with the final calibration and produces the final AOD (Analysis Object Data). The CTF and AOD are then sent to permanent storage. O^2 hides the underlying distributed framework, which allows analyzers to focus on the physics. This is done through the three main components depicted in Figure 1.

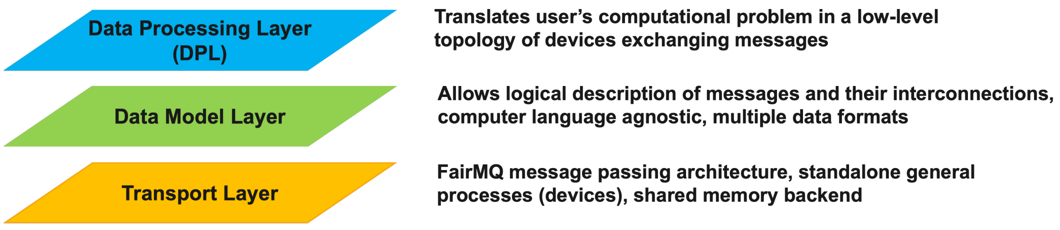

Data Processing Layer (DPL)
Translates user's computational problem in a low-level topology of devices exchanging messages
Data Model Layer
Allows logical description of messages and their interconnections, computer language agnostic, multiple data formats
Transport Layer
FairMQ message passing architecture, standalone general processes (devices), shared memory backend

Figure 1: O^2 Framework Layers

2.1 Data Model and table manipulation

The data model format is similar to relational databases, while the complexity of representation is shielded from the user. For example, as illustrated in Figure 2, collisions and tracks are represented in trees (flat tables) that are connected through indices passed through shared memory.

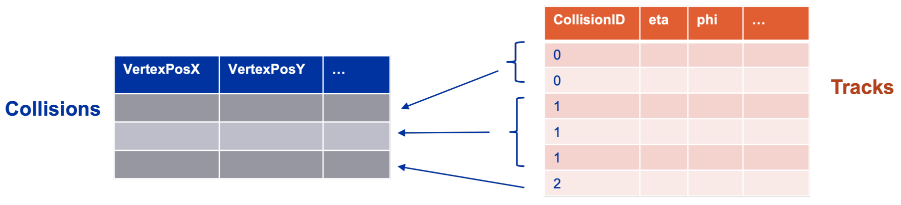


Figure 2: Interconnected tables

A table is defined as a unique C++ type, templated on columns. The columnar format is provided by Apache Arrow [4], which provides a contiguous in-memory layout for large amounts of data and ensures inter-process communication. This allows fast and efficient operations that, instead of being applied per collision or event, they work per large data blocks with thousands of collisions.

Joining, grouping, partitioning and filtering are requested by the analyzers and the operations can be combined when needed (Figure 3). Apache Arrow ensures zero-copy operations, which means that the underlying contiguous arrays in memory are immutable, therefore no data is removed or copied in common operations.

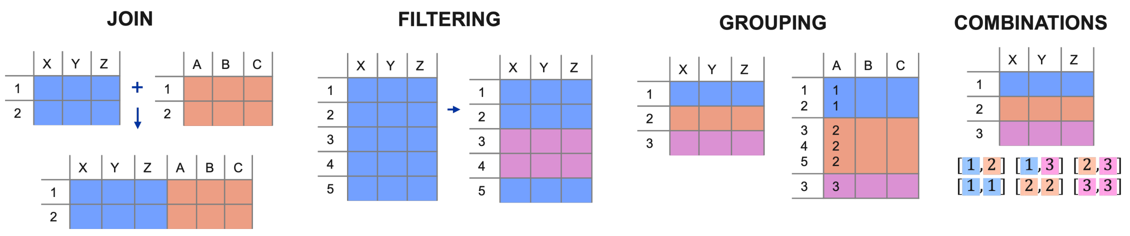


Figure 3: Table operations

2.2 Data Processing

The O^2 framework ensures multiprocess capability. The user defines tasks that include callbacks and declarations of input and outputs, thus subscribing to different tables. These tasks are then combined into workflows representing a particular analysis or group of analyses, and the final result is the ROOT serialized output. This is illustrated in Figure 4.

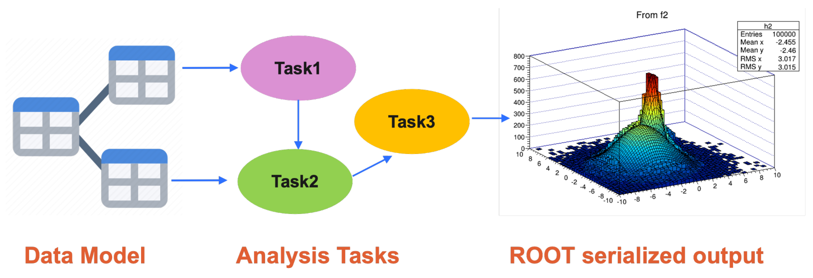


Figure 4: Data processing

The analysis model is both declarative and imperative. The declarative way provides clear and compact definition of analysis cuts and flow, while the imperative way is used to represent certain conditions that cannot be implemented in a declarative way.

The Data Processing Layer hides the complexity of the abstract transport layer and low-level data model, builds the workflow topology based on interdependencies between tasks and translates the defined workflows to an actual FairMQ topology of devices.

3. Hyperloop Train System

Hyperloop implements the concept of analysis trains used to optimize the usage of computing resources and was built upon tools used in Runs 1 and 2. It allows task configuration, enables fast and demanding analysis workflows on Grid and Analysis Facilities, which are specialized Grid sites with CPU and disk resources adjusted for analysis needs. A sketch of the analysis process in Hyperloop is shown in Figure 5.

In Run 2, 90% of the analysis activity used the so-called LEGO analysis trains [5], which utilized on average about 40 000 cores at any given time. This tool was re-written with a modern reactive front-end technology, React [6], and fully integrated with O^2, providing efficiency, user-friendly features, and dedicated views for users and operators.

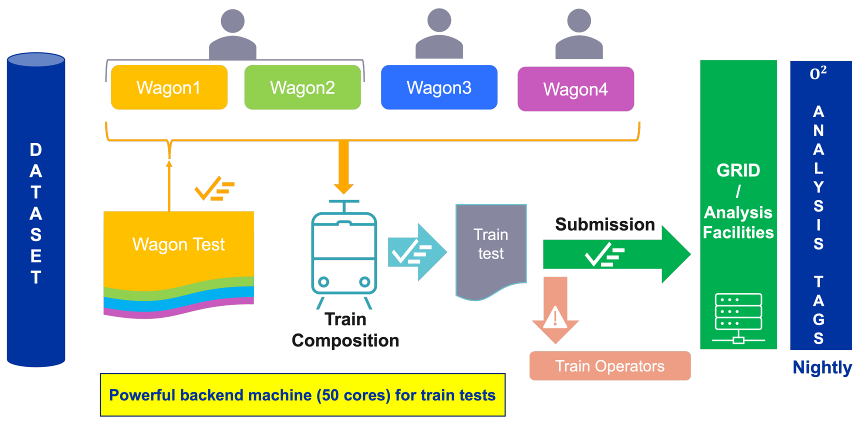
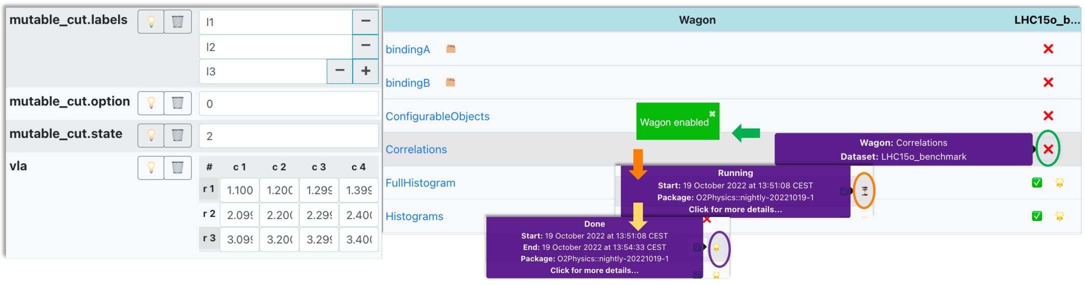


Figure 5: Analysis process in Hyperloop

3.1 Wagon tests and analysis process

The data available in Hyperloop is converted Run 2 data, Run 3 data and MC and derived data. The user enables datasets per analysis (an organizational unit shared by multiple users) and is then able to create individual workflows, known as wagons. As illustrated in Figure 6, the wagon configuration supports a variety of parameter types such as primitive types, arrays, matrices, labelled matrices, and histogram binning. Subwagons can also be added and will run the same task with different parameter values, for instance for the evaluation of systematic uncertainties. Furthermore, output can be stored as derived data which is then used by a subsequent train (so-called tuple creation or skimming).

Once a wagon is enabled, a functional test will start immediately and the resulting information will be available in the wagon test view. This displays per device (reader, workflows, writer) and total performance metrics, as well as expected resources in interactive graphs. If the wagon test is successful, it will be composed together with wagons of other users into a train (either automatically or manually by an operator). If the train run test is successful, it will be submitted to the Grid or to an analysis facility (Figure 5).



mutable_cut.labels	l1
	l2
	l3
mutable_cut.option	0
mutable_cut.state	2

vla

#	c 1	c 2	c 3	c 4
r 1	1.100	1.20(	1.29(	1.39(
r 2	2.09(	2.20(	2.29(	2.40(
r 3	3.09(	3.20(	3.29(	3.40(

Wagon	LHC15o_b...
bindingA	✗
bindingB	✗
ConfigurableObjects	✗
Correlations	✗
FullHistogram	✓
Histograms	✓

Wagon enabled

Wagon: Correlations
Dataset: LHC15o_benchmark

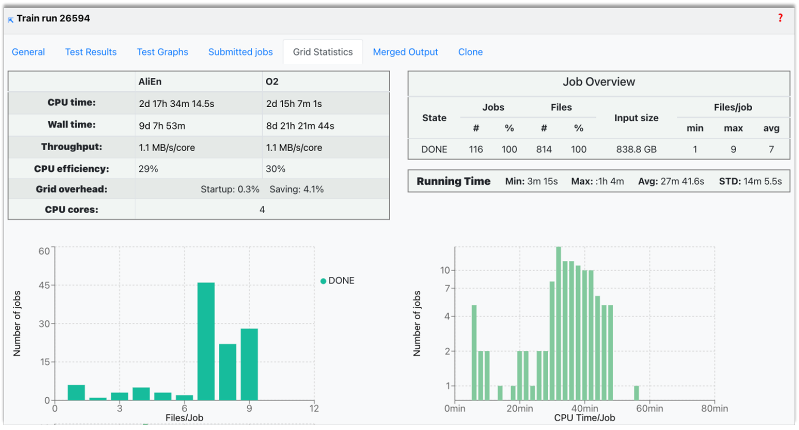
Running
Start: 19 October 2022 at 13:51:08 CEST
Package: O2Physics::nightly-20221019-1
Click for more details...

Done
Start: 19 October 2022 at 13:51:08 CEST
End: 19 October 2022 at 13:54:33 CEST
Package: O2Physics::nightly-20221019-1
Click for more details...

Figure 6: Wagon configuration (Left), Wagon test process (Right)

3.2 Train Runs

An automatic train composition schedule can be defined per dataset, and, if enabled, will compose trains at specified times based on target memory, wagon configuration and dependencies. After the train test is successful, it will be automatically submitted. The user can see the information about their train runs in a dedicated train result view. This displays the test results, the summary of submitted jobs and several statistics in interactive graphs (Figure 7).



Train run 26594

General | Test Results | Test Graphs | Submitted jobs | Grid Statistics | Merged Output | Clone

	AliEn	O2
CPU time:	2d 17h 34m 14.5s	2d 15h 7m 1s
Wall time:	9d 7h 53m	8d 21h 21m 44s
Throughput:	1.1 MB/s/core	1.1 MB/s/core
CPU efficiency:	29%	30%
Grid overhead:	Startup: 0.3% Saving: 4.1%	
CPU cores:	4	

Job Overview

State	Jobs		Files		Input size	Files/job		
	#	%	#	%		min	max	avg
DONE	116	100	814	100	838.8 GB	1	9	7

Running Time **Min:** 3m 15s **Max:** :1h 4m **Avg:** 27m 41.6s **STD:** 14m 5.5s

Figure 7: Grid statistics

If automatic composition is not enabled for a dataset, users can request train composition on the dedicated operation channel. This request will be taken care of by one of the operators who work on a 24/5 shift-type basis (provided by four ALICE institutes spread across time zones). Besides helping with train composition, the operators offer support for warnings, failed tests and trains which are stuck due to specific problems. There is a dedicated page for trains which show pathological behaviour, which are then followed up by the operators. This allows a timely escalation to, for example, Grid experts if needed.

3.3 Bookkeeping and monitoring

History views are available for wagons and datasets, which allow to track all changes made. The user is also able to compare train and wagon configurations. Updates regarding these or other crucial elements in Hyperloop are sent to the user through a system of automatic notifications. Each notification contains an explanation of the update and links to details and documentation.

The current status of the system is displayed in a dedicated dashboard. This is used by experts to track information such as number of wagons waiting to be tested and trains ready to be submitted. Moreover, it presents the summary of the previous week, showing the jobs' number and state per site.

4. Summary

The ALICE detector received several major upgrades, allowing for 100 times more data to be recorded in Run 3 and 4. Considering the challenge to handle this great amount of data, new tools were developed: the O^2 Framework and the Hyperloop train system. The O^2 framework allows for distributed and efficient processing of the unprecedented data. It will hide the underlying complexity from the users, allowing them to focus on the physics analysis.

The Hyperloop train system enables analysis workflows to be run on the Grid and Analysis Facilities. Comparing to previous tools, it involves significant levels of automatic activity (e.g. train composition and submission), which translates into less manual actions to be taken by the operators. The modern interactive user interface offers a responsive, reliable and user-friendly application, and offers easy access to status and updates.

User support is carried out by operators that work on a 24/5 shift basis schedule. A dedicated channel is used for requests, questions and general updates. Moreover, extensive documentation has been created, which is easily accessible from several sections in Hyperloop and explains in detail the components, functions and processes that are part of the new tools.

Hyperloop is in production and presently being used by 160 users in about 130 analyses. 133 datasets are available for processing and 4970 train runs have been submitted. Taking everything into consideration, the ALICE analysis infrastructure is ready for Run 3.

References

[1] Abelev, B and others, *Upgrade of the ALICE Experiment: Letter Of Intent*, J. Phys. G **41** (2014), 087001 doi:10.1088/0954-3899/41/8/087001

[2] A. Alkin, G. Eulisse, J. F. Grosse-Oetringhaus, P. Hristov and M. Kabus, *ALICE Run 3 Analysis Framework*, EPJ Web Conf. **251** (2021), 03063 doi:10.1051/epjconf/202125103063

[3] Buncic, P. and Krzewicki, M. and Vande Vyvre, P., *Technical Design Report for the Upgrade of the Online-Offline Computing System* (2015). Available at https://cds.cern.ch/record/2011297/files/ALICE-TDR-019.pdf

[4] Cross-language development platform. *Apache Arrow*. Available at https://arrow.apache.org/. Retrieved on 20.10.22

[5] Zimmermann, Markus, *The ALICE analysis train system*, J. Phys. Conf. Ser. **608** (2015), 012019 doi:10.1088/1742-6596/608/1/012019

[6] Facebook Open Source. *React.js*. Available at https://reactjs.org/. Retrieved on 20.10.22