Finanziato dall'Unione europea
NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Ideas for benchmark metrics

Tommaso Diotalevi (UniBO)

WP2 Working Meeting, 23/01/2024

Introduction

The starting point for this discussion is based on two analysis applications, already ported on the high rate platform and previously discussed on WP2/WP5 general meetings:

- Vector Boson Scattering (VBS) of two same-sign W bosons (ssWW) in a hadronic tau and a light lepton. The analysis porting has been made by Tommaso Tedeschi on the INFN CMS Analysis Facility.
  - Latest WP meeting: link

- Heavy Neutral Lepton (HNL) analysis in the search of heavy neutrinos from D_S decays. The analysis porting has been made by Tommaso Diotalevi on the INFN CMS Analysis Facility.
  - Latest WP meeting: link

Sources of metrics

The metrics considered so far in these studies are coming mainly from different sources, with different purposes:

- **Timestamps** on the Jupyter Notebook running the analysis. "User" vision of the execution times, divided in the different phases of the analysis workflow. Using python standard modules for time keeping.

- **Task-wise**: metrics on the resource usage (CPU, memory, network), with a single Dask partition (task) granularity, adopting "*beta*" features of the **ROOT RDataFrame distributed** (v6.27):
  - Specific image for the Dask workers, built using a dev version of ROOT. The image for this platform is yet to be produced (?)
  - The metrics of each task are saved on the workers, in two separate *csv* files: one saving resource usage each second, the other is a one-line with the overall times and the number of processed events. At the end of the execution, it is required to *stage-out* them on a persistent storage.
  - Warning: If the analysis workflow triggers multiple event loops, such *csv* files are overwritten every time. Some workaround is required.

- **Node-wise**: metrics of the "physical" nodes, saved on some **InfluxDB** instance. Entire view of the cluster, summed on all the facility users in a certain time window.

Example 1: VBS ssWW in hadronic tau and a light lepton

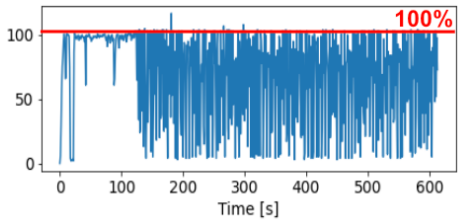
courtesy of Tommaso Tedeschi

[some example code]

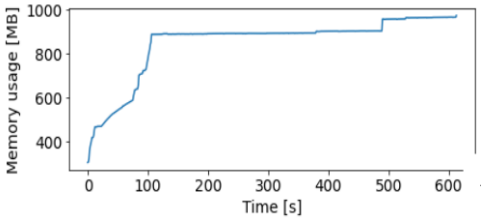
Metrics collected:

- Overall execution time:
  - Time elapsed from the start of the execution to the end of execution.
- Rate (events/s), overall (considering initialization time) and event-loop-only:
  - The ratio between the total number of events processed and sum of processing times obtained from single job logs.
- Network read:
  - Per node information about total bytes read from the network during the execution. This value is summed across all nodes.
- Absolute memory occupancy (RSS):
  - Per node information averaged across executions time and across all the available nodes.

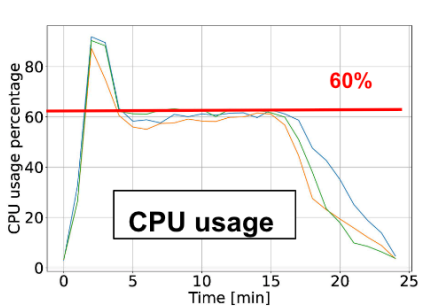
CPU usage

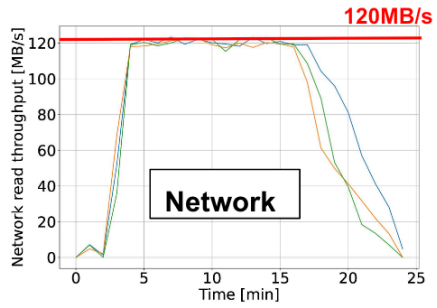


Memory usage



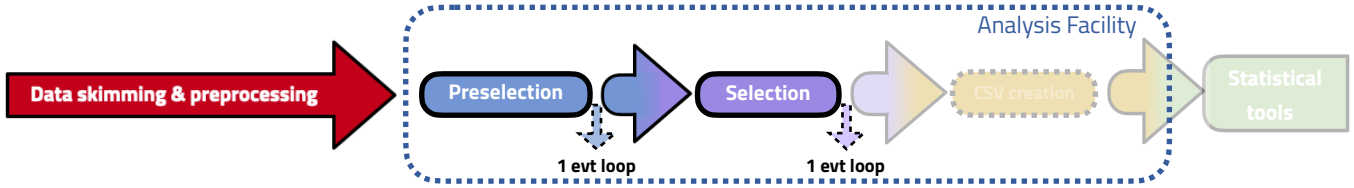
Task-wise RDF





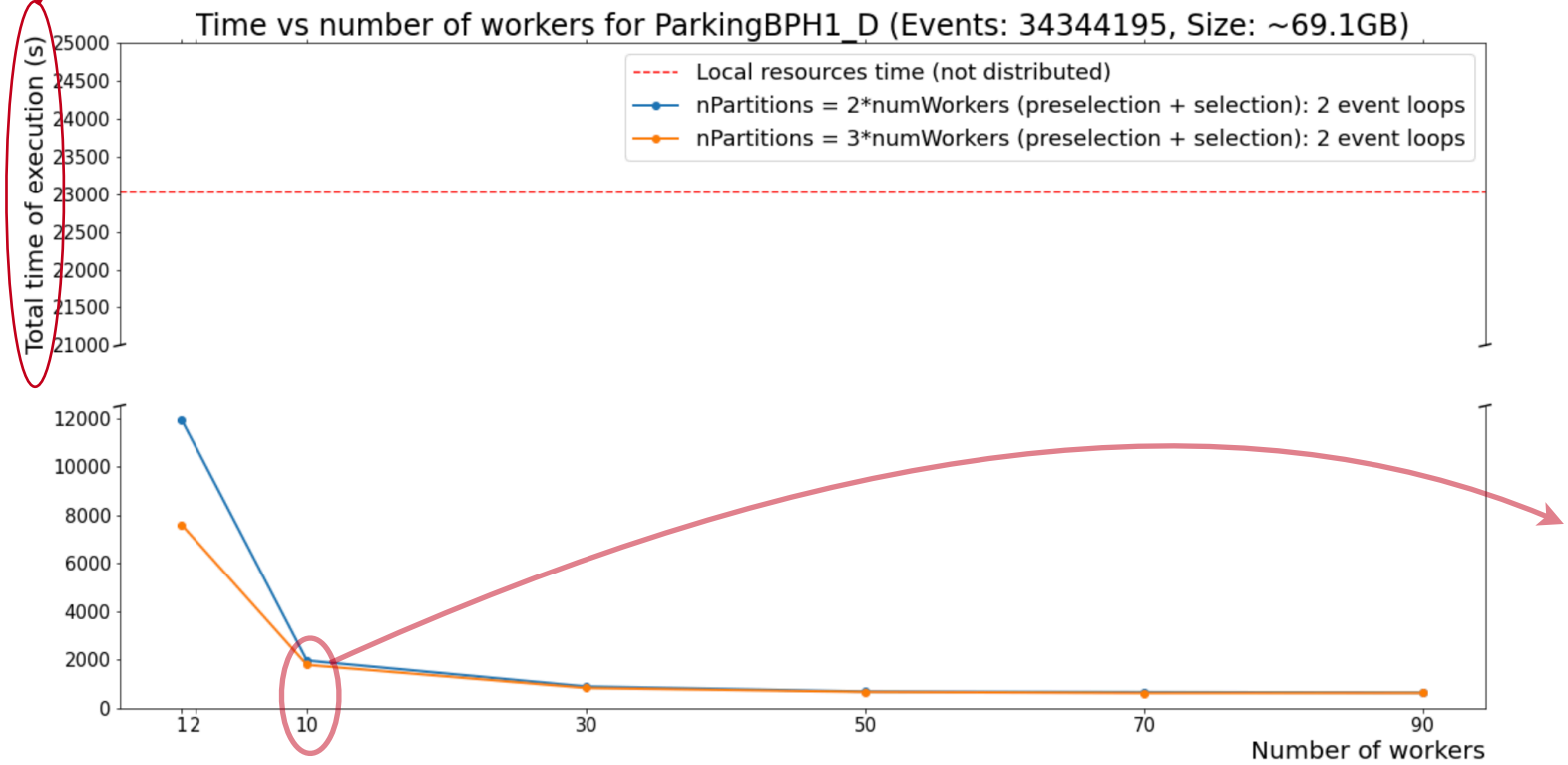
Node-wise InfluxDB

Example 2: HNL search for heavy neutrinos in D_S decays



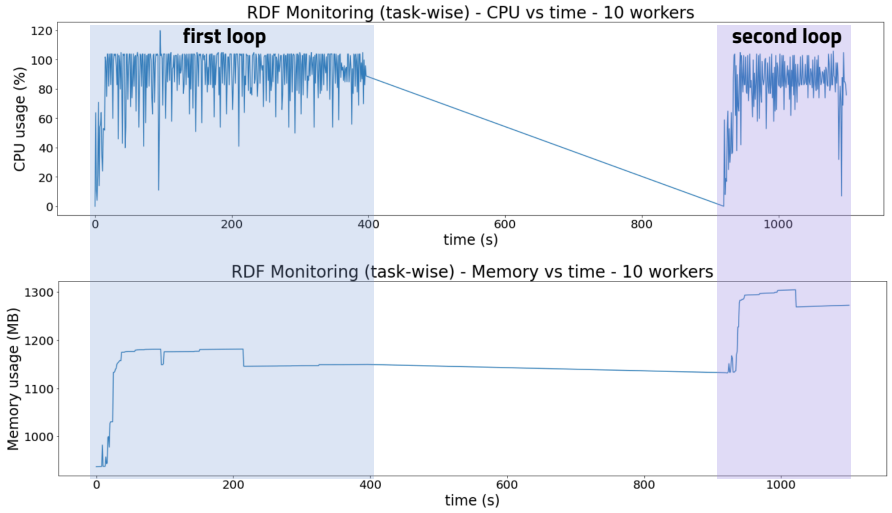
[some example code]

Timestamp based metric

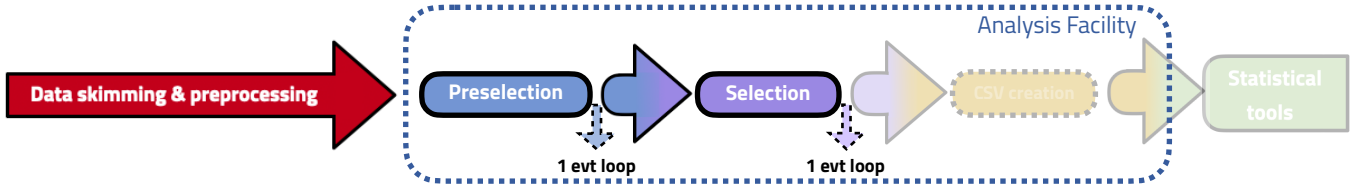


Number of workers = DASK workers sharing the computation.
nPartitions = granularity (tasks) in the RDF configuration.
In red, the execution time of the same workflow, on a single job, without Dask.

Zoom on task-wise RDF internal metrics

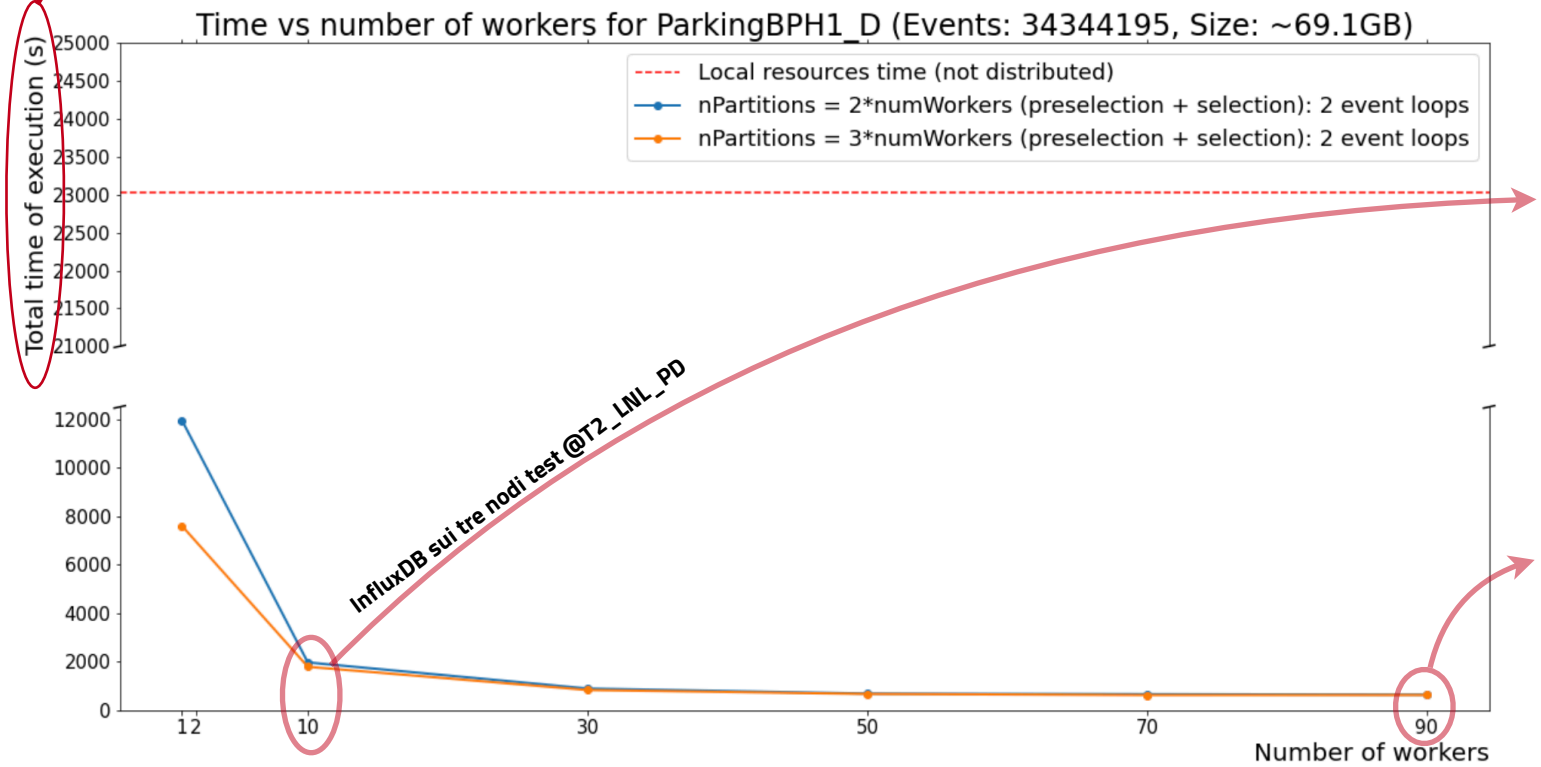


Example 2: HNL search for heavy neutrinos in D_S decays



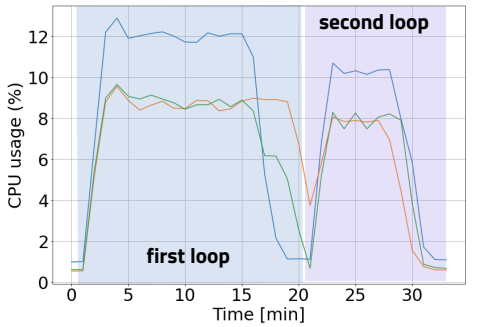
[some example code]

Timestamp based metric

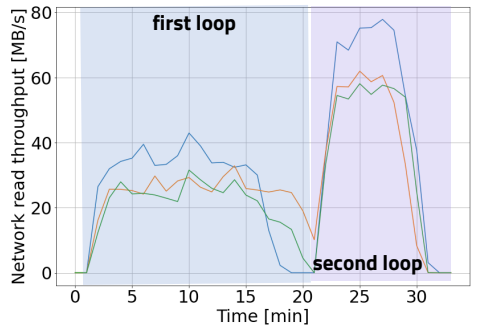


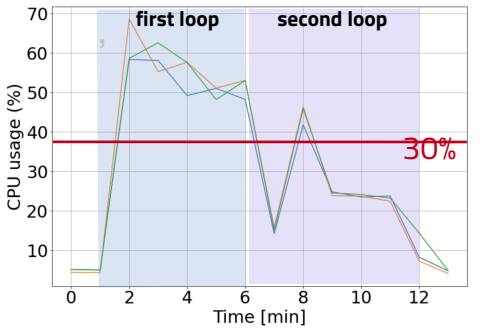
InfluxDB node wise metrics

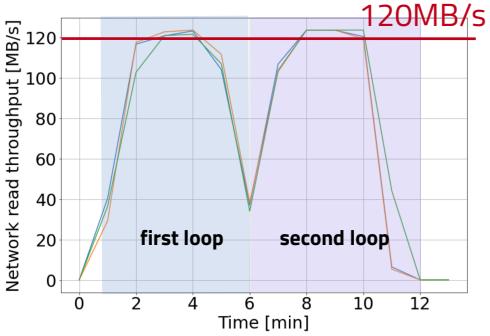
CPU



Network read







Conclusions

The metrics investigated so far are coming mainly from three different sources: timestamp based (on the analysis code), task-wise (on distributed RDF), node-wise (on-site metrics on InfluxDB).

Note: the task-wise metrics require a RDF approach due to the specific singularity image with ROOT 6.27. Other frameworks require different solutions.

Discussion:

- Unify metrics across use cases:
  - Overall execution time (s)
  - Rates (events/sec), considering (and not) initialisation time
  - Same rates, computed using task-wise metrics, to investigate a *distributed speed-up factor*
- The WP5 high rate platform has some on-site metrics? Monitoring node-wise information like CPU, memory, network…
- How should we handle user traffic during this benchmarking study?

Let's open the discussion!

BACKUP

ROOT RDataFrame

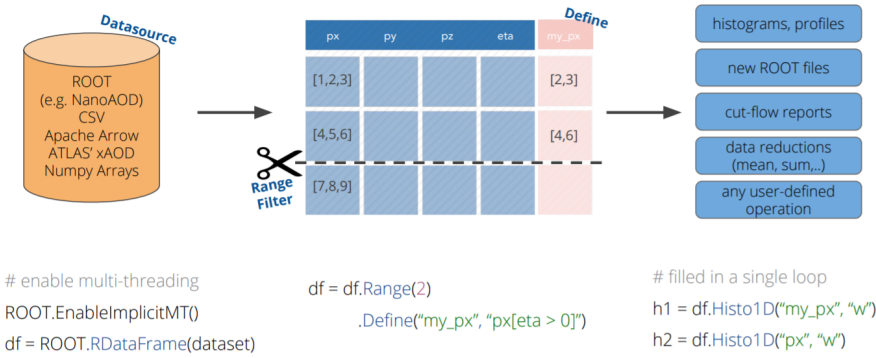
RDataFrame (RDF) is the high level ROOT interface for data analysis stored in TTree, csv and other data formats. It enables:

- multi-threading;
- low level optimisations (parallelisation and caching).

Computation is expressed in terms of a action and transformation chain, which together constitute a computational graph.

The execution of the graph can be done in a distributed fashion, adopting backends like Spark and Dask.

Grazie all'estensione "Distributed" di RDF, attiva in via sperimentale.




```
# enable multi-threading
ROOT.EnableImplicitMT()
df = ROOT.RDataFrame(dataset)
```

```
df = df.Range(2)
      .Define("my_px", "px[eta > 0]")
```

```
# filled in a single loop
h1 = df.Histo1D("my_px", "w")
h2 = df.Histo1D("px", "w")
```

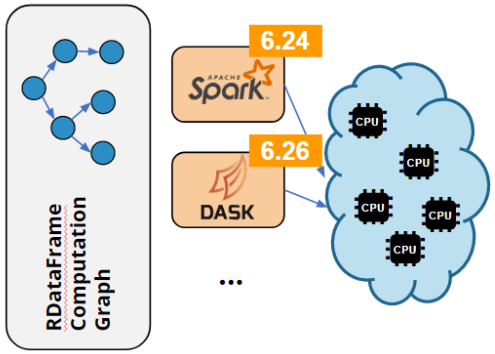
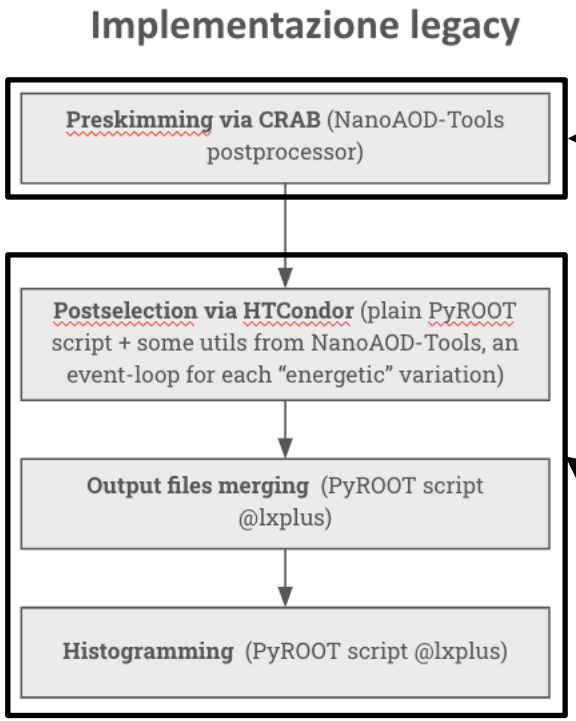


Image taken from "A Python package for distributed ROOT RDataFrame analysis", V. Padulano, PyHEP 2021

Workflow VBS analysis



Preselection:

Filtri basati su trigger e richieste "loose" sugli oggetti dello stato finale, definizione di quantità di correzione

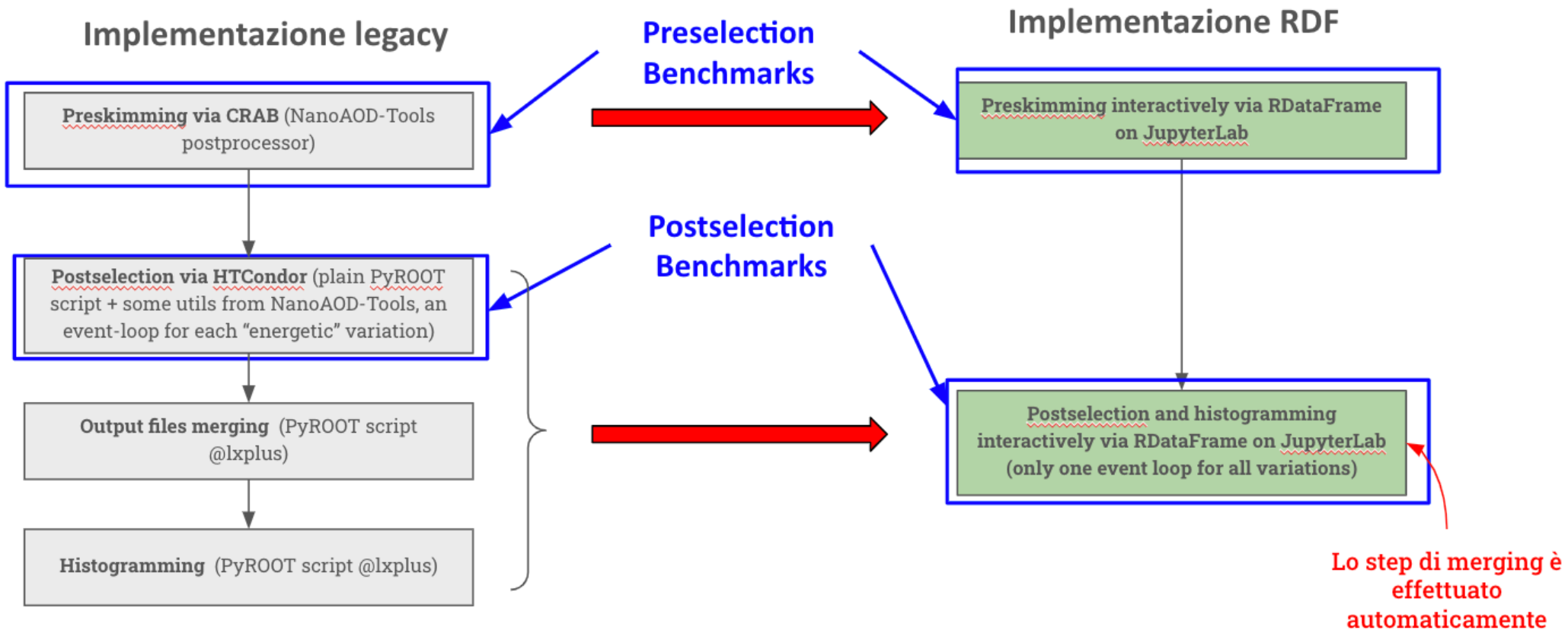
- Dimensione dataset da analizzare: 2 TB per anno di presa dati (1 TB dati, 1 TB MC di cui segnale 12 GB)
- Tempo di esecuzione: O(10h) su CRAB

Postselection:

Ricostruzione vera e propria dell'evento con la produzione degli istogrammi di variabili rilevanti per ciascuna variazione, sample, regione cinematica e stato finale

- Dimensione dataset da analizzare: O(10 GB) per anno
- Tempo di esecuzione: O(1h) sul CERN Batch Service

Porting VBS analysis



VBS Benchmark results

Considering the MC Run 2017 analysis (1.1 TB, 1274 files, about 700 mln events)
The two approaches have been tested on the same resources and conditions, for the pre- and post-selection steps.

- Legnaro Tier-2 resources (3 nodes, each with 32 logical CPU - 128 GB RAM - 1 Gb/s) - monitoring CPU, mem, net.

The benchmark metrics are:

- Overall execution time;
- Average rate of the single job/task (events/second), considering (total) or not (event loop) the initialisation time;
- Network read (amount of data read by remote);
- Average memory occupancy (RSS) per-node.

Preselection		
	Legacy	RDF
Overall time [min]	181 ± 1	23.8 ± 0.6
Overall rate [events/s]	786 ± 12	6915 ± 35
Event-loop rate [events/s]	858 ± 14	7632 ± 34
Overall network read [GB]	485 ± 1	362.5 ± 0.1
Average RSS per-node [GB]	23.3 ± 0.6	31.3 ± 0.4
Postselection - Main scenario		
	Legacy	RDF
Overall time [min]	48.3 ± 0.5	12.6 ± 0.3
Overall rate [events/s]	62.9 ± 0.1	288 ± 1
Event-loop rate [events/s]	65.69 ± 0.05	355 ± 3
Overall network read [GB]	84.46 ± 0.08	17.46 ± 0.08
Average RSS per-node [GB]	5.5 ± 0.2	26.7 ± 0.5

Workflow HNL analysis

Operazioni

- Data reduction
- File format change

- Tagli di preselezione
- Applicazione pesi (MC, SF trigger, SF muon, PU)

- Selezione ottimizzata per categoria;
- Selezione miglior candidato HNL
- Ridefinizione pesi totali

- *CSV temporaneo per incompatibilità con tool statistici (combine)*

- Fit e analisi statistica