Elements of Parallelism, Performance and Programming Model

Danilo Piparo
With input of many, in particular: Xavi, Enric, Lorenzo, Guilherme, Enrico, Gerri, Pere, Jakob

ROOT
Data Analysis Framework
https://root.cern

High Level Goals for 2018

More speed

- **Demonstrate** accelerator aided HEP data analysis at O(100) threads.
- **Consolidate** accelerator aided HEP Monte Carlo studies at O(100) threads.

More usability

- **Focus on ROOT analysis tools**, both in C++ and Python
- Generate value for HEP specific tasks - make our product the most attractive.

More throughput

- **Engage with the experiments** with the goal of maximising throughput of parallel RunIII (and beyond) data processing.

Meeting Infrastructure

- Room 4-S-030 for the PPP Meeting: Thursdays 16:00 to 17:30 (available for the entire year but 3 weeks!)

Keep the different meeting formats:

- Informative (round table, contributions about work progress, benchmarks)
- Decision making: e.g. where to invest, how to react to benchmark results
- Topical - spotlight on an area/sprint: e.g. status of Jira items for a release, PyROOT, Vectorisation, “Though issue XYZ”

Allocation of Person Power

Summary of Person Power allocation: **12+ FTMEs**

- **Programming model: 6 FTMEs**
- **Parallelism (task, distributed): 6+ FTMEs** (4+ FTMEs, 2 FTMEs)

Data parallelism treated by Guilherme: 7 FTMEs, not counted here

PyROOT treated by Enric: 4.5 FTMEs, not counted here

Programming Model – 6 Months

For PyROOT – 4.5 FTMEs, not counted here – See Enric's slides!

PyDataFrame - 1 FTME

Develop the PyDataframe wrapper in order to allow even smoother usage of TDataFrame from Python. Example functionalities we are after

- Passing Python functions/callables as cuts/defines
- Experiment with Numba to compile them

1) Because the majority of people analysing HEP data use PyROOT and at the moment the bare bindings are not always enough to guarantee a smooth experience

2) All analysers

3) Because we have a PyROOT TDF community already now and we'll be asked about this as soon as we present to experiment the declarative analysis approach in ROOT

4) 1 FTME

5) RN, presentations to the experiments, pinned post on the forum, tutorials

Helpers for “Common Operations”- 2 FTMEs

[TDF] We need a way to simplify the operations on collections. For example calculate DeltaR between 4-vectors, Cartesian products, create a collection of numbers starting from a collection of objects and a method.

1) Because we want to reduce the time needed to achieve high quality plots coming from sophisticated analyses

2) All analysers

3) Because these are routine operations for which we do not have a clear set of helpers for easing simple operations.

4) 2 FTMEs

5) RN, presentations to the experiments, pinned post on the forum, tutorials

TDataSource (Bulk I/O, Binary, …) - 1+ FTMEs

Assess the need of new data sources, for example, to accommodate the BulkIO functionality if the C++ interfaces which exist now, e.g. TTreeReader, appear to be not adequate. Potentially delegate to (Summer) students or young new comers bolder ideas about new data sources.

1) Because we believe that the TDF + TDS combination to express analyses is approachable by Physicists with a range of computer skills.

2) All analysers

3) Because we need speed. A "Fast Path", if the columns allow that, can be desirable.

4) 1 FTMEs (depending on the actual final form of the BulkIO?)

5) RN, presentations to the experiments, pinned post on the forum, tutorials

DataFrame Consolidation - 2 FTMEs

Consolidate the TDF to make it the ROOT's standard way to interact with columnar datasets. Implement even sturdier error recovery procedures, even more expressive warning/exception/error messages, further improve interactive usage

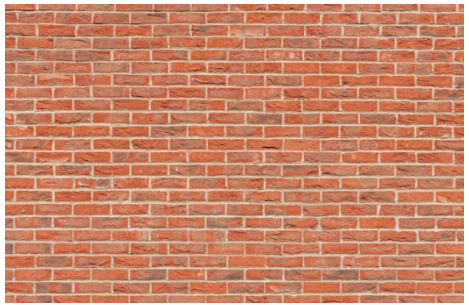
1) Because of its rapid adoption in the community

2) All analysers

3) Because it took off already in 2017!

4) 2 FTMEs

5) RN, presentations to the experiments, pinned post on the forum, tutorials



Expressing Parallelism – 6+ FTMEs

Data Parallelism – 7 FTMEs, not counted here – See Guilherme's slides!

Expressing Parallelism – 6+ FTMEs

Task Parallelism – 4+ FTMEs

The New Interfaces- ? FTMEs

Support the development and adoption of new interfaces, in particular considering the new memory management model we have in mind for ROOT7.

1) Because we can continue to parallelise without problems for 6 months, easily for 1 year but we need a long term perspective which would free us from the workarounds and helpers we need to cope with the ROOT memory model.

2) All analysers

3) Because we spent a considerable amount of time already in marrying the current memory management and the parallelisation. This was necessary since an evolution is systematically better than a revolution which implies the presence of too many variables (and potentially unforeseen costs).

4) Depends on how fast the new interfaces are developed and adopted

5) RN, presentations to the experiments, pinned post on the forum, tutorials

“Is this Thread/Task Safe?” - ?+ FTMEs

Identify a way to advertise, document and test the features we provide in terms of thread safety and parallelism

- What is thread safe and what is not?
- What is implicitly parallelised and what is not?

1) Because **the documentation should be as shiny as our implementations and interfaces.**

2) All ROOT users

3) No. There are signs everywhere about the need of better documentation.

4) ? FTMEs (hard to estimate)

5) Pinned post on the forum, topical manual (?)

TDF: Decouple Slots & N workers - 1 FTMEs

Decouple the number of TDF slots and the number of workers in the pool

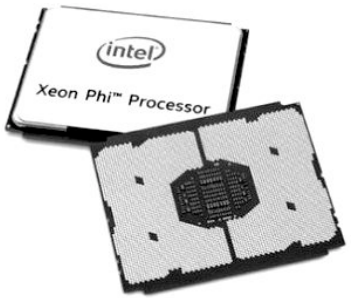
1) Because we cannot scale up to O(100) threads opening one file per slot as we do now. We need to exploit other opportunities for parallelism once the throughput of the storage device is reached

2) All analysers

3) No. We need to deliver something usable on big servers/accelerators. Incidentally, a 12 cores desktop may already saturate the disk throughput

4) 1 FTMEs

5) RN, presentations to the experiments, pinned post on the forum



TDF: Role of New Interfaces - 1 FTMEs

Clarify the role of the new interfaces (histograms mainly), i.e. interactions with the memory model. It is crucial to be able to support the v7 interfaces in TDF with the least effort possible.

1) Because the new interfaces are potentially the spine of the future of the project

2) All ROOT users

3) Because LHC Season III will start in 2021

4) It really depends from the state of the interfaces. ~1 or 2 FTMEs?

5) RN, presentations to the experiments, pinned post on the forum, tutorials

New Executors - 2 FTMEs

Augment the executors infrastructure in order to comprise:

- **A NUMA aware executor**, perhaps hidden from the user who continues to use the TThreadedExecutor
- **A Sequential executor** to accommodate sequential execution if it is required to do so
- **An improved version of the chunking** shall be identified and transparently introduced, or, if that is not possible, an improved interface provided.

1) Because executors are the right way to propose explicit parallelism to users

2) All analysers, we the ROOT team in the implementation of implicit parallelism

3) No, given that we want to support explicit parallelism expressed by users and increase as much as possible implicit parallelism within ROOT

4) 2 FTMEs

5) RN, presentations to the experiments, pinned post on the forum

Expressing Parallelism – 6+ FTMEs

Distributed Analysis – 2 FTMEs

PyROOTSpark - 1 FTME

Make sure PyROOTSpark provides all the features which are adequate for launching map-reduce workflows on a Spark cluster in order to be ready for the connection of SWAN to the IT-DB Spark resources.

1) Because presently the only distributed approach to data analysis consists in the consolidated and widely adopted job submissions. A complement to this approach based on a bleeding edge technology which is a standard in industry and supported at CERN and elsewhere is desirable.

2) Physicists in the experiments

3) No, because the link to the Spark clusters will happen in 2017. After that Physicsts will need a way to take advantage from it.

4) 1 FTME

5) RN, presentations to the experiments, pinned post on the forum, SWAN galleries

Distributed Executor- 1 FTME

Assess the possibility to extend the executors to a distributed approach, potentially based on Spark.

1) It's desirable to complement PROOF with an interface we think has a potential given its simplicity

2) Physicists in the experiments

3) We need to accumulate as soon as possible experience about the potential backends for distributed analysis as well as their potential acceptance in the experiments.

4) 1 FTME

5) RN, presentations to the experiments, pinned post on the forum