# Task 1.7: Common software developments for heterogeneous architectures

Next Generation Triggers WP1 Intro

axel@cern.ch 2024-01-08

# Parties

- ALICE, ATLAS, CMS, LHCb, EP-SFT, IT innovation section
- All with different interests and use cases in mind
  - Example: IT & generators
- Arrived at collaborative setup with co-hosting, shared hires, etc

| | | 2024 | 2025 | 2026 | 2027 | 2028 |
|---|---|---|---|---|---|---|
| **efficient scheduling** | **IT** | **0.75 quest** | **0.75 quest** | **0.75 quest** | | |
| | **IT** | | | **0.1 origin** | origin | origin |
| | **ATLAS** | 0.5 quest | 0.5 quest | 0.35 quest | 0.35 quest | 0.35 quest |
| | **CMS** | 0.5 quest | 0.5 quest | 0.35 quest | 0.35 quest | 0.35 quest |
| | **LHCb** | | | 0.3 quest | 0.3 quest | 0.3 quest |
| | | | | | | |
| **efficient data structures** | **ALICE** | 0.75 quest | 0.75 quest | 0.75 quest | 0.75 quest | 0.75 quest |
| | **ATLAS** | 0.25 quest | 0.25 quest | 0.25 quest | 0.25 quest | 0.25 quest |
| | **LHCb** | 0.5 doct | 0.5 doct | 0.5 doct | 0.5 quest | 0.5 quest |
| | **SFT** | 0.5 doct | 0.5 doct | 0.5 doct | 0.5 quest | 0.5 quest |
| | | | | | | |
| **common libraries** | **CMS** | | **0.5 doct** | **0.5 doct** | **0.5 doct** | |
| | **SFT** | 0.5 origin | 0.5 origin | 0.5 origin | 0.5 origin | |
| | | | | | | |
| **programming languages** | **SFT** | 0.3 origin | 0.3 origin | 0.3 origin | 0.3 origin | |
| | **LHCb** | 0.2 origin | 0.2 origin | 0.2 origin | 0.2 origin | |
| | | | | | | |
| **ML inference interface** | **CMS** | **0.33 doct** | **0.67 doct** | | | |
| | **LHCb** | 0.3 doct | 0.3 doct | 0.3 doct | | |
| | **SFT** | 0.7 doct | 0.7 doct | 0.7 doct | quest | 0.5 quest |

# Multiple R&D Themes

- Efficient heterogeneous scheduling
- Efficient portable data structures
- Common accelerated libraries
- Alternative programming languages
- Efficient accelerator interfaces to ML inference

# Efficient heterogeneous scheduling

- Problem: schedule tasks that depend on other tasks, across CPU and accelerators (think e.g. "multiple GPUs"), with data transfer and conversion
- Use case: experiments' workflows, today and future; generators
- R&D aspects: optimized utilization / throughput; ease-of-use; overhead, etc
- Relevant milestones: provide stand-alone experiment workflows as demonstrators; implement, benchmark scheduling techniques
- Possible candidates: stdexec,... - compare to tbb as baseline

# Efficient portable data structures

- Problem: need vector<Track> as array-of-structs, struct-of-arrays (vector<pt>, vector<phi>,...), array-of-struct-of-array etc. Currently done by each experiment, separately, e.g. with C++ preprocessor macros.
- Use cases: experiments' frameworks; generators
- R&D aspects: Memory allocation: one region for all structs-of-arrays, alignment. Memory movement / layout conversion. C++ reflection, LLAMA,...
- Relevant milestones: EDM prototypes; benchmarking (compare to experiments' existing solutions); I/O

# Common accelerated libraries

- Problem: most experiments have collection of kernels used for their frameworks / online / ..., in *one* implementation; create central repository for synergy, visibility, maintenance / optimization; start with Lorentz vectors + medium-sized matrix ops

- Use cases: experiments' codes

- R&D aspects: provide, benchmark and optimize implementations for multiple backends (C++, CUDA, alpaka, SYCL,...)

- Relevant milestones: collect existing implementations, provide tests, benchmark, optimize, implement missing backends

# Alternative programming languages

- Problem: our compiled languages C++ / CUDA / SYCL /... are difficult, Python is slow; what else is suitable?
- Possible candidates: Julia, Mojo,...
- R&D aspects: define relevant metrics (beyond benchmarks) for HEP, including "use of accelerators", "interfacing C++", etc
- Relevant milestones: C++ interoperability, ease of accelerator use

# Efficient accelerator interfaces to ML inference

- Problem: given an accelerator kernel, call into ML inference kernel controlling scheduling. Use industry standard solutions where possible, try to work on industry standards where needed

- R&D aspects: batching / data layout; language interoperability; designing interfaces to existing inference libraries; consider inference library (SOFIE)

- Relevant milestones: implement demonstrators / prototypes; benchmarking + optimization on realistic workflows; agree with industry partner(s) on new / optimized ML inference interface(s).

Photo by Alora Griffiths on Unsplash

# Schedule

- Currently hiring
  - Preparing job descriptions
  - Waiting for upcoming DOCT round
- First people to start in July

# EP Software Seminar

- Considering to resume EP Software Seminar series
  - With help from NextGen WP 4
- 2024: share where experiments are today, including potential to improve
- 2025...: cover NextGen advances, esp as stage time for fellows

# Contact

- Coordination at ngt-1-7-coord@cern.ch
- Discussions at https://mattermost.web.cern.ch/nextgen-triggers/channels/task17
- Indico category https://indico.cern.ch/category/17798/
- Or the task leader team Axel.Naumann@cern.ch Andrea.Bocci@cern.ch Attila.Krasznahorkay@cern.ch

# "Contractual Milestones"

M1.1.4: Report on the preparatory work done in WP1.7 and the concrete work planned for year 2 of all of the sub-projects

M2.1.4: Workshop at CERN for discussing the status and plans for all of the sub-projects in WP1.7.

M3.1.4: Produce a report on the status of all of the sub-projects from WP1.7.

M4.1.4: Complete the development of the task scheduler prototypes. Produce a recommendation for the experiments on heterogeneous task scheduling.

M5.1.4: Present the results of all WP1.7 sub-projects at major computing conferences.