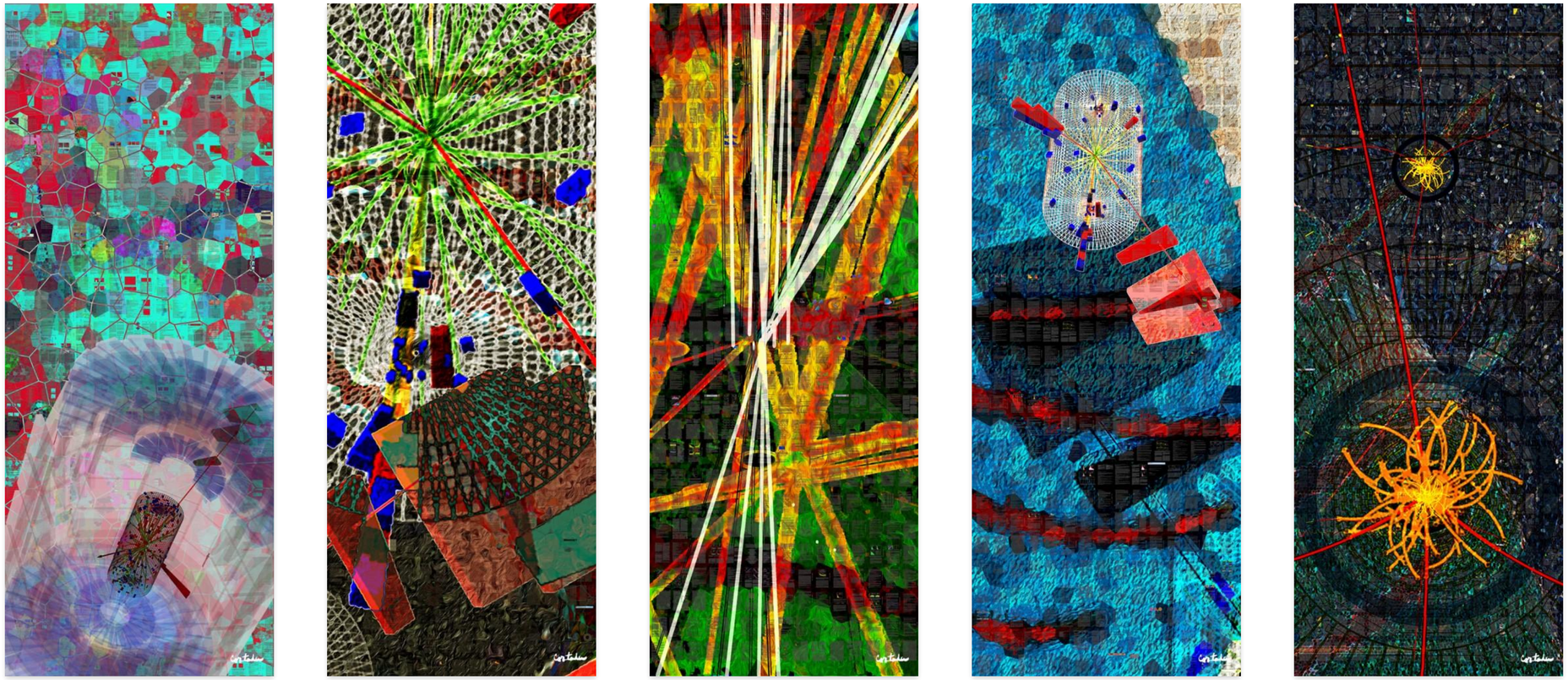
U.S. CMS Operations Program

CMS Feedback

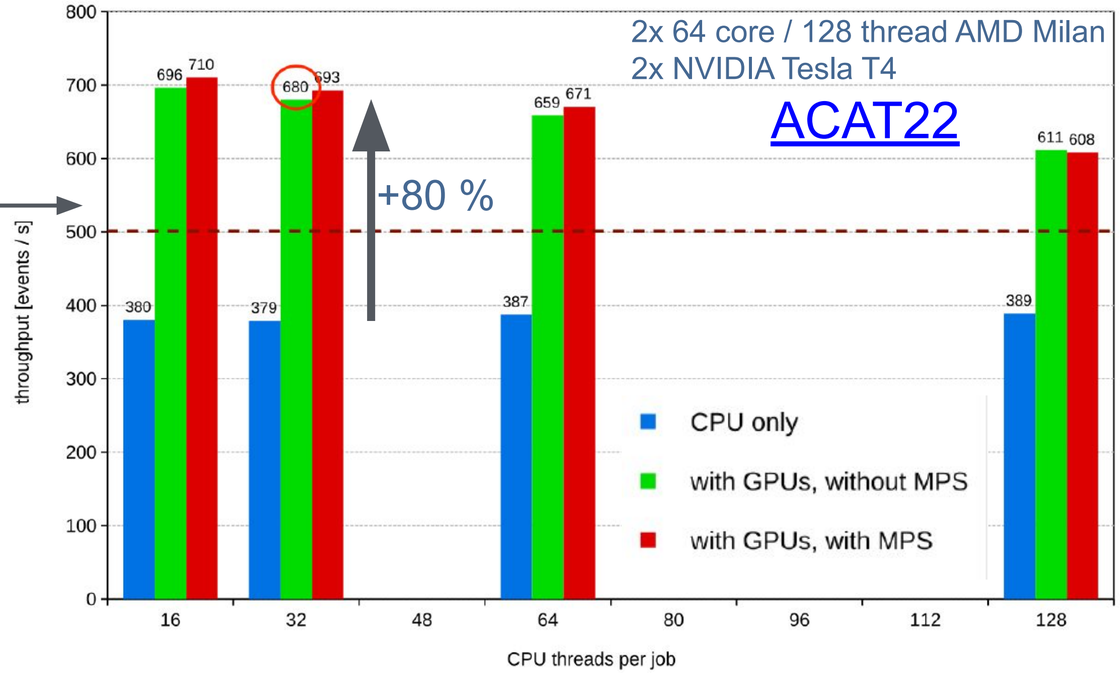
Matti Kortelainen, with the help of many people

HEP-CCE All Hands Meeting April 12 2023



Introduction

- Strategic goals
  - Ensure that there are appropriate levels of computing resources available to enable the physics program of the CMS experiment now and in the future, while using those resources efficiently with highly-performant software applications and minimizing operational effort, and completing computing requests in short, predictable amounts of time.
- Software
  - Multithreaded framework in production since 2015
  - GPUs in production at High Level Trigger since 2022
    - Achieved with CUDA, code being migrated to Alpaka
  - Next step: establish GPU-based offline workflows
  - I/O is all ROOT based

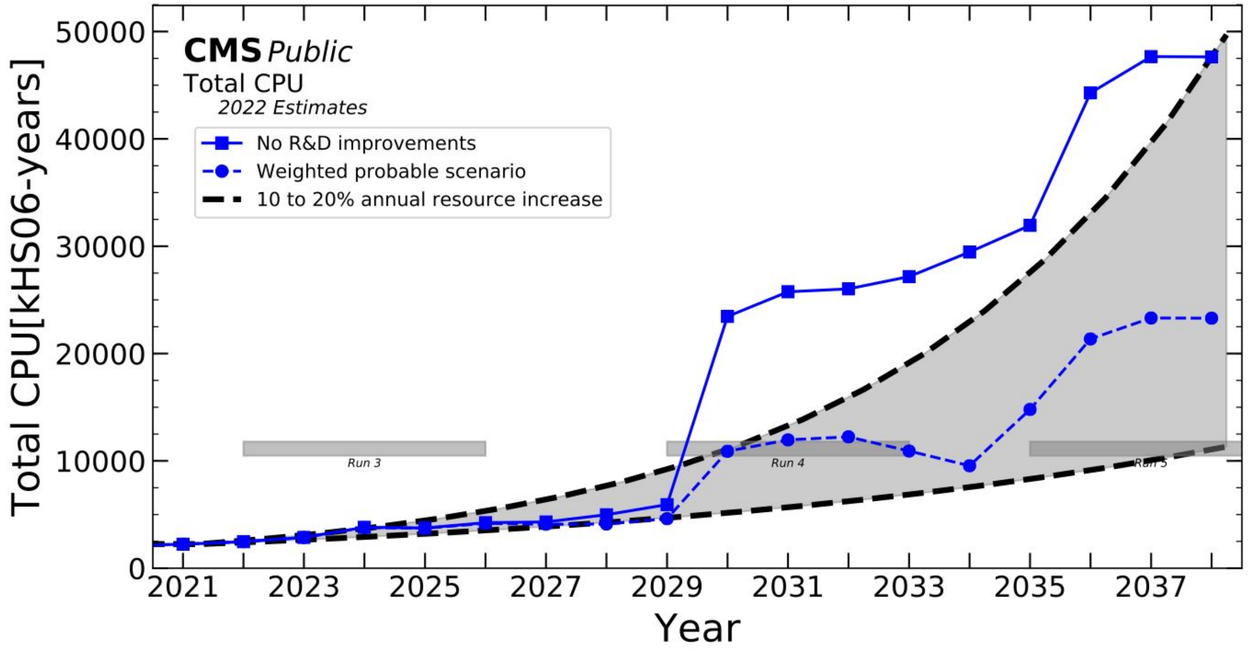


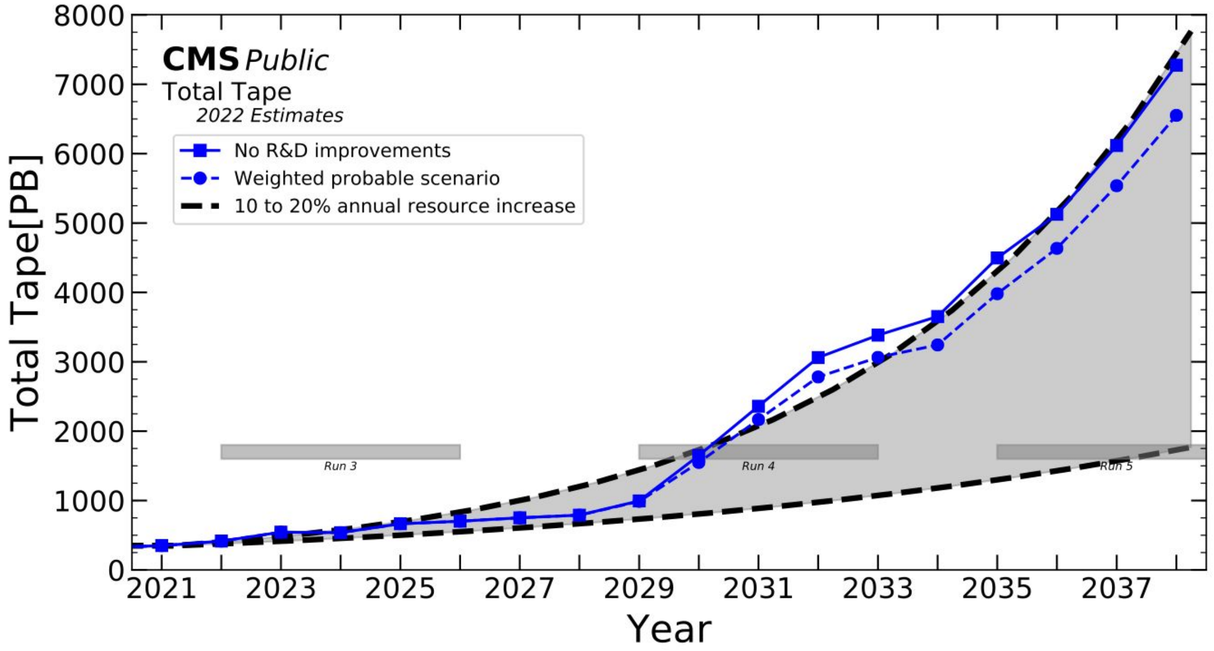
- Computing system
  - Global pool of HTCondor jobs that are executed at ~70 sites on ~400k CPU cores by glideinWMS pilots
  - Rely on remote reads (xrootd) for e.g. overlaying “pre-mixed” pileup-only events in simulation workflows

CMS' Computing

Resource Projections for HL-LHC

- CMS updates resource projections for HL-LHC regularly
- Latest public projections (July 2022) are available at https://twiki.cern.ch/twiki/bin/view/CMSPublic/CMSOfflineComputingResults
- Process ongoing for Conceptual Design Report for HL-LHC, within a timescale of a year





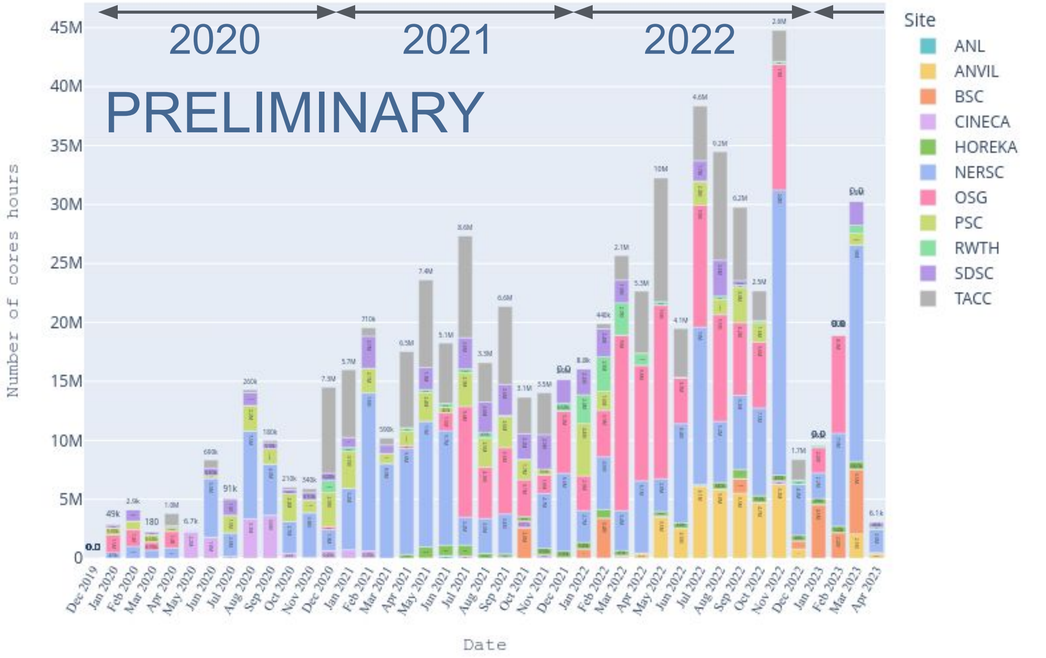
General areas of R&D needs (not ordered)

- Latest public document from LHCC review in 2022: https://cds.cern.ch/record/2815292

- Fast Monte Carlo chain, including fast simulation and reconstruction, with simplified geometry
- Event generators
- Re-engineering and redesigning algorithms and data structures for accelerators
- Full simulation running on accelerators
- Integration of accelerators into the application framework and workflows
- Integration of ML inference frameworks
  - Most ML inference frameworks are not really designed to be used as a part of large application
- Training of very large ML models

Role of HPC

- Currently HPC systems contribute about 5-10 % of CMS' total offline computing power
- CMS foresees a need to increase the compute capacity used at HPCs
  - Actively being planned and worked on
- Joint US-ATLAS / US-CMS HPC/Cloud blueprint process looked into exactly this question last year
  - Report passed to agencies in 2023, publication to arXiv in progress
- From the blueprint report
  - Potential growth on HPC with combination of CPU and GPU/accelerators
    - No significant capacity increases in CPU-only resources expected
  - Significantly expanding HPC use depends on applications getting better on utilizing accelerators
    - CMS demonstrated an effective way to utilize same-node GPUs in its High Level Trigger in 2022
    - Need motivated manpower to re-engineer the algorithms
  - Independently of software, we are pushing ahead with work on the integration of the LCF into the CMS computing infrastructure



Strategies for heterogeneity

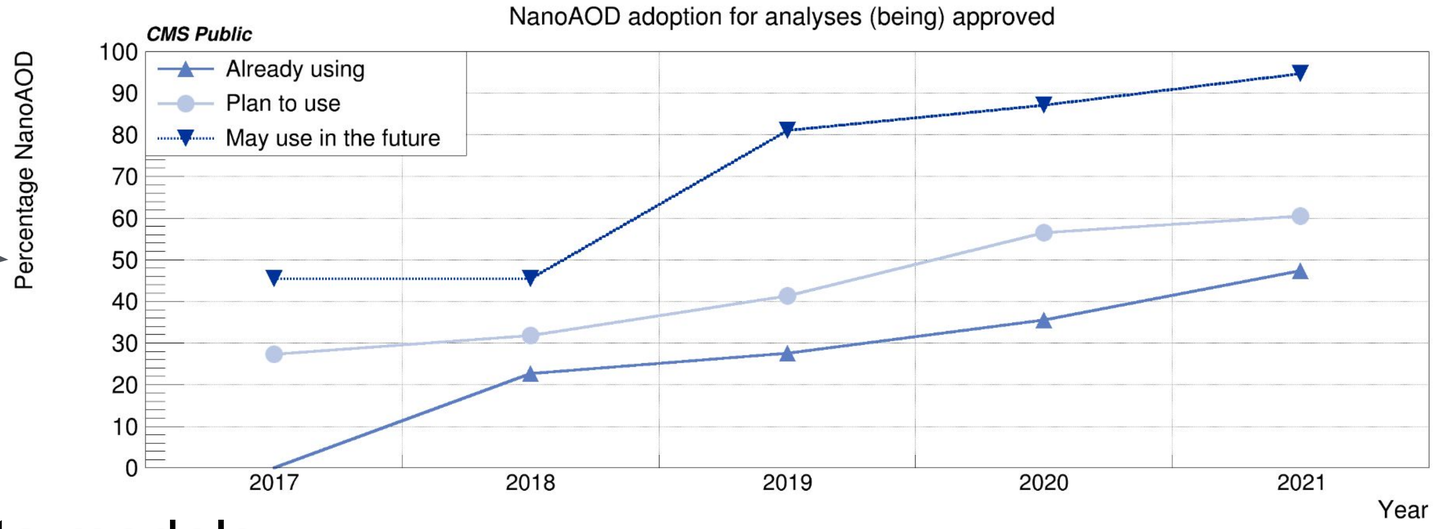
- CMS wants to be able to take advantage of modern architectures
- Across CPU architectures
  - CMS has different builds for x86, PowerPC (since 2016), and ARM (since 2014)
    - x86 and PowerPC validated for physics, ARM validation ongoing
    - Eventually thinking to add RISC-V
- GPUs
  - Portability between CPU and NVIDIA and AMD GPUs achieved with Alpaka
    - Extending to Intel GPUs is being worked on
  - Alpaka algorithms will be able to use same-node GPUs. Use of remote-node GPUs is being investigated
    - E.g. SONIC, MPI
- ML algorithm inference
  - Acceleration on same-node GPUs being worked on via TensorFlow and ONNX Runtime
    - Their integration into CMSSW generally tricky (especially TensorFlow): e.g. conflicting dependencies
  - R&D on remote-node GPU access via SONIC continues
- “Single-use” accelerators (e.g. FPGAs); unclear if want or need to utilize
  - Different usage model compared to general-purpose or “multi-use” hardware, little-to-no experience
  - May need a process-isolation or remote server approach (like SONIC)

Algorithmic R&D priorities

- Reduction of stored data on both disk and tape
  - More on next slide
- Speeding up reconstruction, especially pattern recognition in tracker and high-granularity calorimeter; strategies:
  - Massive parallelization on GPUs
  - Machine learning
    - Have to evaluate the potential gains against the cost (effort, increased complexity)
- Event generators: speeding up, addressing negative weights, good software engineering practices (e.g. invocation in multithreaded environment)
  - Strategy so far: rely on event generator developers
    - Not very good for technical software engineering part
- Speeding up simulation; strategies:
  - With the full simulation expertise within CMS work together with Geant 4 developers, and with AdePT and Celeritas projects
  - Investigate fast simulation and ML methods

Strategies for I/O and storage optimization

- Optimization of data formats
- Push even more analyses to use NanoAOD format
- Adopt ROOT RNTuple
- Investigate and deploy lossy compression



- Object stores as locally extensible data models
  - For analysis: aim to get rid of ntuple files
  - For production: try to reduce tape usage by avoiding duplication of products among data tiers
- Regional data lakes, caches in the network, managed network
- AI for infrastructure optimization
  - E.g. real-time automation/intelligence in dataset placement and compute

Other notes on computing needs

- CMS aims for flexibility and interoperability
- CMS wants to achieve more with fewer people
- CMS needs to focus more on tails of workflows
  - Need to automate and optimize
- CMS needs efficient, reliable, and monitorable reading of remote files
- Changes need to be analyzed from a cost-benefit point of view
  - E.g. if remote file reads would not be possible, would need more disk

Feedback on CCE projects

Portable parallelization

- CMS finds the CCE's portability layer evaluation extremely valuable
  - The work allowed CMS to understand peculiarities of several portability layers, and form a short and a long term strategy to converge on a single codebase for CPU and accelerators
  - We can't think of any portability layer to add to the evaluation
- The Patatrack use case is a sufficiently representative example of CMS reconstruction on GPU that includes the interaction with the framework and the build system. It is complemented by the simpler compute-heavy propagation-to-r benchmark
- CMS does not see much value in adding more applications for the evaluation
  - CMS would benefit a lot more from re-engineering its reconstruction algorithms and data structures for GPUs through Alpaka than investing heavily on continuing the portability layer evaluation
    - Possibility of moving Alpaka to something better is in CMS' plan, e.g. HL-LHC provides a window of opportunity
- In the longer term CMS needs to work on enabling unified memory support in CMSSW
  - Benchmarking unified memory systems could be useful

I/O and storage

- CMS finds the IOS studies extremely useful
- The “testing framework” studies have brought CMS tangible improvements, e.g.
  - Simplification of some very complex data format classes
  - Change of default compression algorithm from ZLIB-4 to ZSTD-4
  - Improvements of bottlenecks found in ROOT
  - Lossy compression (integration is still in progress)
- Darshan studies for CMS workflow (ROOT I/O) did not reveal any new information
- Serial HDF5 as intermediate storage did not seem to provide performance benefits
  - CMS does not have a use case that would benefit from parallel I/O from many processes
- Currently the most pressing I/O issue for CMS is the scalability of ROOT I/O for high thread counts
- CMS is interested in further studies in object stores, and in industry-standard access and transfer protocols
- Any means to reduce the disk and tape storage while keeping necessary physics precision are relevant for CMS

Accelerator-friendly data models

- CMS developed in-house Structure-of-Array data structure
  - In the leading order should be efficient on GPUs and vectorizing CPUs
  - Can be persisted with ROOT in a columnar fashion
- Another product with similar level of functionality would not be useful for CMS
- CMS would benefit from a better way to store the SoAs into a file
  - The interfacing with TTree has some quirks
  - It could be interesting to study the storage in ROOT's RNTuple, as well as e.g. Parquet/Arrow/Feather from the industry
  - Eventually could be useful to be able to
    - read from storage and deserialize directly in the device memory
    - to serialize and write to storage directly from the device device

Event generators

- CMS sees the re-engineering of event generators to use GPUs (and other accelerators), as well as to follow modern software development practices, very important
- The work on Madgraph4GPU in the current CCE project is a very good start
  - More effort and more organized work plan would benefit CMS
- CMS uses NLO generators heavily already now
  - E.g. for Standard model measurements > 50 % of the events are already generated at NLO
  - Only some exotic or SUSY searches “can live” with LO samples
    - And even they are generated with high jet multiplicity to “mimic” NLO samples
  - CMS foresees the future direction to be towards higher precision

Complex workflows

- CMS is not very familiar with Parsl or FuncX
  - CMS already has a capability to abstract away resources within a workflow via SONIC
    - Any added value of another FaaS or similar approach should be evaluated and weighed against the additional complexity
- CMS could benefit from a complex workflow execution engine that could be somewhat specific to HPCs and orchestrates the tasks with varying resource requirements within the single HPC

Other feedback

Detector simulation

- GPU accelerated full detector simulation is very important for CMS
- CMS follows the development of AdePT and Celeritas, and is in frequent contact with their developers
- CMS wants to be in direct contact with the AdePT and Celeritas developers
- Concerning Celeritas specifically, funding Celeritas is a high priority
  - CMS thinks Celeritas should be funded (by DOE) either directly, or as part of HEP-CCE as a new project within CCE
  - Celeritas integration into CMSSW needs also CMS experts to work with the Celeritas team

Useful things (in somewhat decreasing order of priority)

- Efficient geometry and magnetic field data structures for GPUs would be very useful
  - Something along “efficient search structures by detector element ID”
  - Should understand ongoing efforts in Celeritas etc and avoid overlap/redundancy
- R&D on system / framework level for making accelerator use more efficient
  - E.g. reducing overheads by batching data from many collision events
- Generic parallel algorithms that are used in HEP reconstruction
  - No concrete examples beyond things like “prefix scan” or “sort”, but expect to come up more
- Missing parts in large-scale ML training that are more or less specific to HEP
  - Resource discovery
  - Handing dynamics of multi-user resource
  - Smooth interoperation of storage (EOS etc) and data formats (jagged arrays) in large ML training setup
- Making Awkward Arrays portable across GPU vendors etc would be useful for analysis
- Matrix operation library with parallelization along “matrix per (half) warp”
  - E.g. CMS pixel reconstruction and mkfit operate one matrix per GPU thread or vector lane
  - Vendor large matrix libraries operate one matrix on the full GPU
  - Could there be a useful operation point in between for small matrices?