Software trigger/ Event reconstruction WG

Charged with considering new approaches to the software trigger and event reconstruction (WG charge in detail)

Challenges include:

- How to cope with increased event complexity/instantaneous luminosity
- How to adapt to commodity CPUs moving towards many-core,
- Which architectures (CPU+coprocessor hybrids, GPUs and/or FPGAs) will provide the best throughput per cost in the future?
- Can we exploit novel techniques to improve physics output or improve throughput (e.g., real-time analysis, machine learning algorithms)

WG organized by Vava Gligorov and David Lange. 3 WG meetings (SDSC, CTD, vidyo in April)

CWP Chapter Status and Evolution during week

Draft CWP chapter (Google doc)

Basic structure evolved based on feedback received. Also got comments on the content in live doc, being implemented.

- Challenges posed by future facilities, experiments, technologies
- Current implementations (CPU hungry algorithms, data structures, calibration requirements)
- ~~Community projects (essentially a skeleton at this point)~~
- ~~R&D topics identified by the WG (still rounding these up into the document)~~
- R&D roadmap goals

R&D roadmap goals

R&D roadmap goals section structure

Roadmap area 5: Robust techniques for merging spatial and precision timing measurements

Motivation: Proposed precision timing detectors for charged particles add a new dimension to tracking and vertexing capabilities and algorithms. This capability implies new ways to improve physics measurements as well as their throughput performance.

Overall goal: Implement and demonstrate tools needed to facilitate implementation of concepts including timing information into tracking algorithms.

Short-term goals:
Medium-term goals:
Long-term goals:

Example projects:

CWP Chapter Next steps

We have received feedback on content and structure, working on it

1. Mid July : complete but not final draft which should be distributed widely to community and stakeholders
2. Mid-End July : receive & implement feedback
3. End of July : ready for arXiv

BACKUPS

How can software developments in trigger/reco help physics output of future experiments?

From a purely technical standpoint, either:

- Going faster (eg, faster algorithm implementations),
- Doing things right the first time (eg, solid validation/verification procedures, deriving “good enough” conditions derived quickly)

Clearly reduces computing cost per physics output of any experiment

Given planned increases of 10x in event rate and 4x in event complexity (200 PU at HL-LHC) software improvements in this area *must be made* to live within current budgets

Software developments in trigger/reco can benefit physics output of future experiments

Doing more with less?

Constraint	Implication	Opportunity
Fixed CPU budget	CHF = (Time/event) * (# of events)	Keep more events (higher trigger rate) or do more per event (lower pt thresholds) with faster algorithms
Fixed storage budget	CHF = (Size/event) * (# of events)	Keep more events (higher trigger rate) with less information

Evolutionary vs Revolutionary approaches

- The steady improvement of the performance our hungriest algorithms (eg, tracking implementations) may be sufficient to keep up with demands of future facilities (eg, event rate, pileup)?
  - In which areas are new approaches needed to complement current algorithms?
- New detector concepts require R&D (eg, high-granularity calorimetry)
- Explore breaking the “event” paradigm in places to facilitate the use of GPUs or other “new” architectures
- Determine the “right” mix of “online” vs “offline” processing capabilities for physics. When does reusing online derived quantities help?
- The evolution of analysis facilities will impact data structure requirements
- Reducing the raw data saved per event requires big changes to calibration procedures (likely not possible in all cases)

How do we see the road ahead?

For ~2025 timescale, we need to encourage diversity, then understand what is best as projects mature. For CMS/Atlas, Run 3 is an important testing ground for new approaches (esp. real-time analysis ideas)

Meanwhile, keep improving codebases we have. Depending on the evolution of computing architectures this may be the right approach for many problems. (side benefit: risk reduction)

“Long term” Sustainability

Many codes are written by short term developers for short term needs. Nevertheless they become long term dependencies

- We do not see this changing even as code complexity increases. No one suggested that LHC experiments should start with a new codebase for HL-LHC
- Long term experts / professionals will continue to be needed to provide the most sensitive algorithms (and fix others)
- We should do better in reusing existing toolkits and making “toolkits” developed in house available to others
- We should consider operations/maintenance manpower into our decision making process

Commonality and Leveraging S&C beyond HEP domain

Opportunities for common tools across experiments in this WG area

- Math libraries (eg, matrices, clustering, KDTrees)
- Data structures, data compression libraries (loss-y compression)
- Infrastructures as the basis for heavyweight algorithms (eg, tracking, particle flow)
- Q/A Q/C infrastructures

Commonality and Leveraging S&C beyond HEP domain

Challenges and barriers to commonality

- Development community in experiments must know about toolkits and understand their long term support model (eg, commit to a co-development model for important codes)
- Toolkit developers must prioritize needs beyond their own experiment (possibly via co-development)
- Sometimes there is a tradeoff between commonality and performance for a specific problem [be sure to design for that]

Cross-cutting Elements

- Efficient use of new architectures: Frameworks, simulation
- Efficient use of large conditions products
  (eg, ML data structures)
- Data structures: Analysis