Enabling continuous speedup of CMS Event Reconstruction through continuous benchmarking

C. Caputo[1], on behalf of the CMS Collaboration

[1]Université catholique de Louvain

Introduction

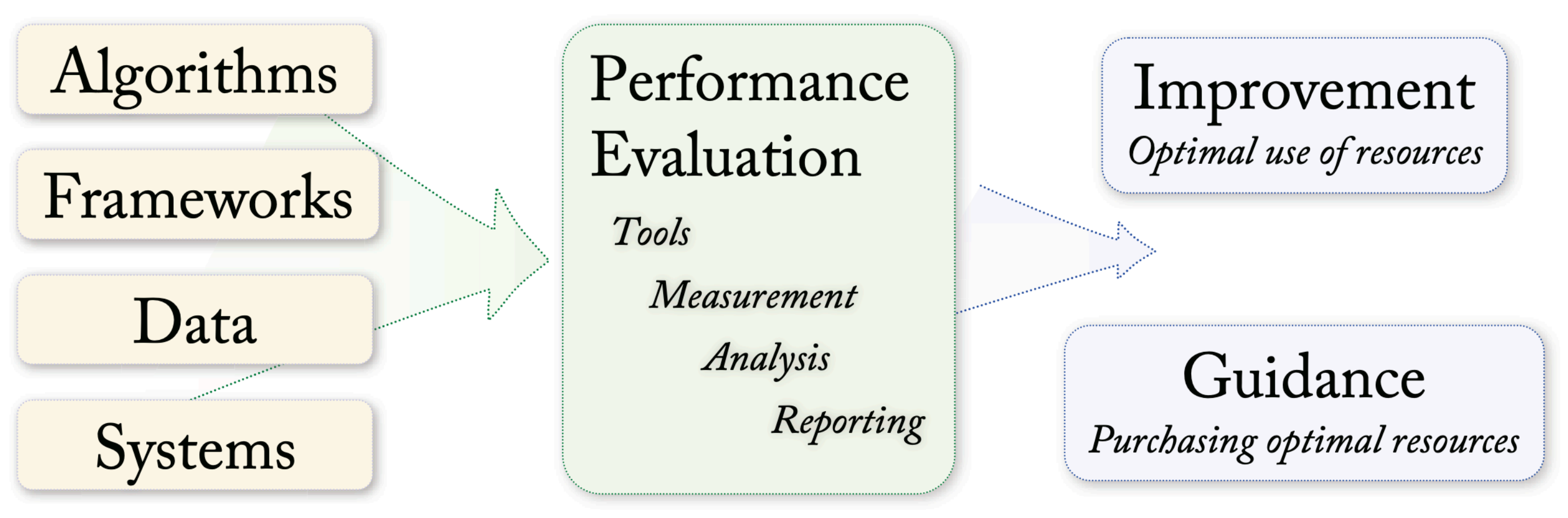
The outstanding performances obtained by the CMS experiment during Run1 and Run2 represent a great achievement of seamless hardware and software integration. Among the different software parts, the CMS offline reconstruction software is essential for translating the data acquired by the detectors into concrete objects that can be easily handled by the analyzers. The CMS offline reconstruction software needs to be reliable and fast. The long shutdown 2 (LS2) elapsed between LHC Run2 and Run3 has been instrumental in the optimization of the CMS offline reconstruction software and for the introduction of new algorithms reaching a continuous CPU speedup. In order to reach these goals, a continuous benchmarking pipeline has been implemented; CPU timing and memory profiling, using the igprof tool, are performed on a regular basis to monitor the footprint of the new developments and identify the possible areas of performance improvement.

CMSSW Releases

- The overall collection of software, referred to as **CMSSW** [1-2], is built around a **Framework**, an **Event Data Model** (EDM), and Services needed by the simulation, calibration and alignment, and reconstruction modules that process event data so that physicists can perform analysis.
  - The primary goal of the Framework and EDM is to facilitate the development and deployment of reconstruction and analysis software.
  - The CMSSW event processing model consists of one executable, called *cmsRun*, and many plug-in modules which are managed by the Framework. All the code needed in the event processing (calibration, reconstruction algorithms, etc.) is contained in the modules. The same executable is used for both detector and Monte Carlo (MC) data.
- CMSSW development proceeds in "cycles": each cycle is intended for a specific goal, i.e. MC production campaign, data taking campaign, etc.
- Before being deployed, each cycle is in a "pre" phase, that can last ~3 months, where development that change reconstruction algorithms can be integrated.
  - Integration, review and testing of the new developments is performed on GitHub [3]
  - On regular basis, a full build of the CMSSW release (pre-release) is lunched and validated

Guidances for the future

- Measuring the performances (CPU timing, MEM, output size) for each pre-realese, both globally and locally, allows us to identify criticality and define, in an objective and quantify way, the next steps to embark on
  - Improvements in the algorithms
  - Identification of potential area of performance improvements
  - guidance for future plans

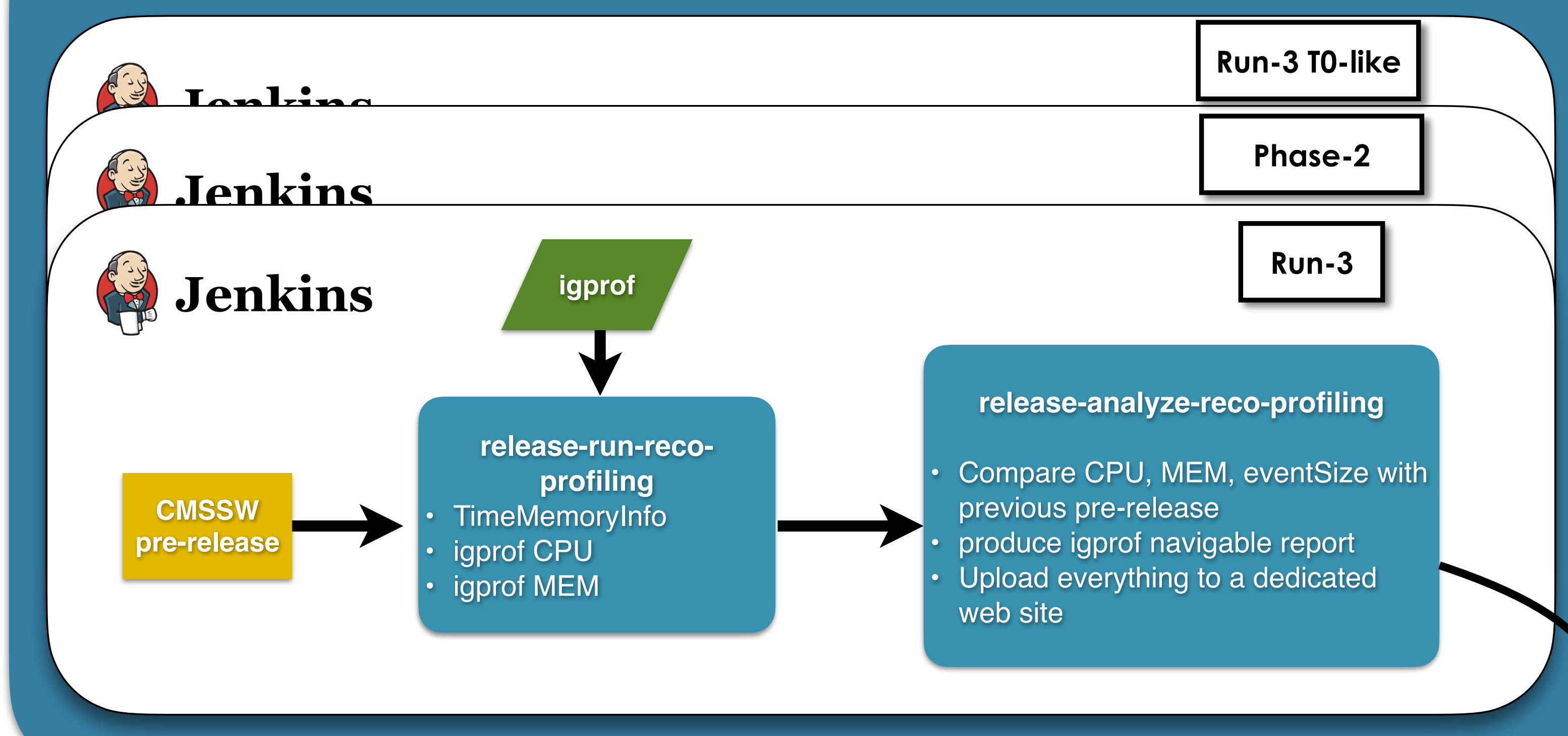


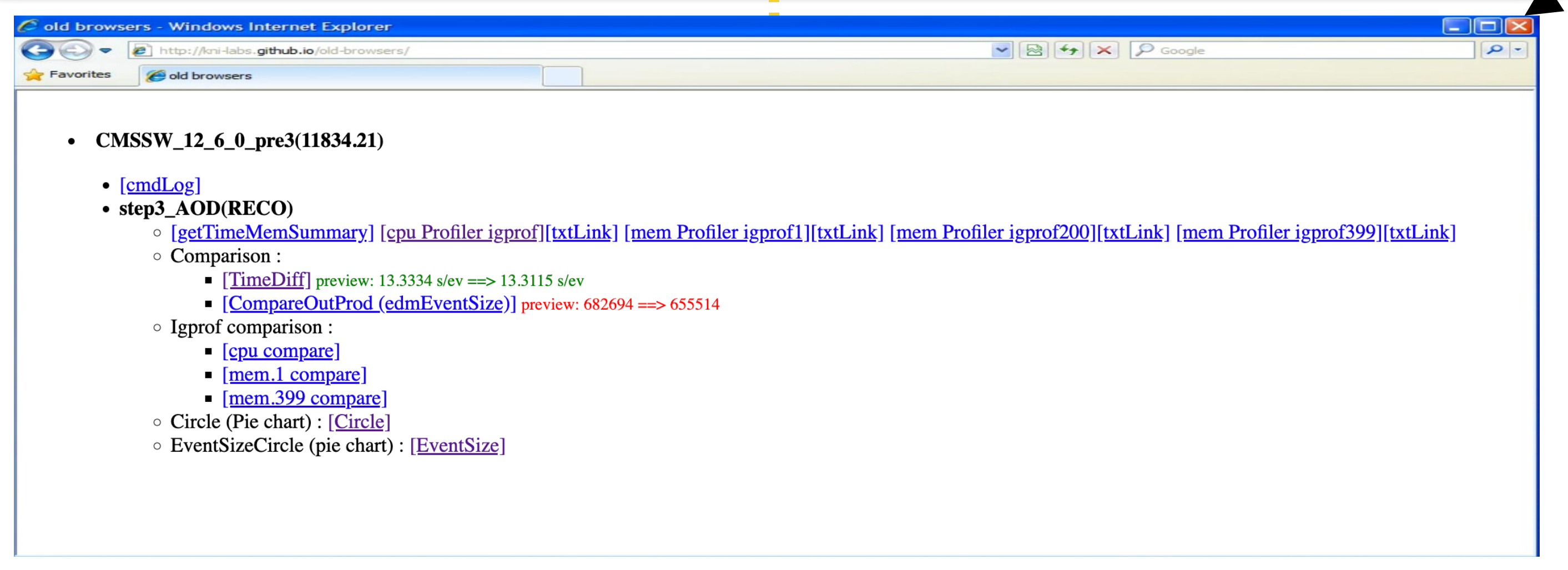
Benchmarking

Tools and flow

- Run detailed profiling jobs on a dedicated machine: Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz
- Reco, Mini, Nano steps with TimeMemoryInfo, igprof CPU, igprof MEM
  - IgProf [4-6], the Ignominous Profiler, is a simple nice tool for measuring and analysing application memory and performance characteristics.
  - IgProf requires no changes to the application or the build process.
  - web-navigable report: easy to navigate from one part of a call stack to another
- Profiling info analyzed and propagated to the web by a jenkins job

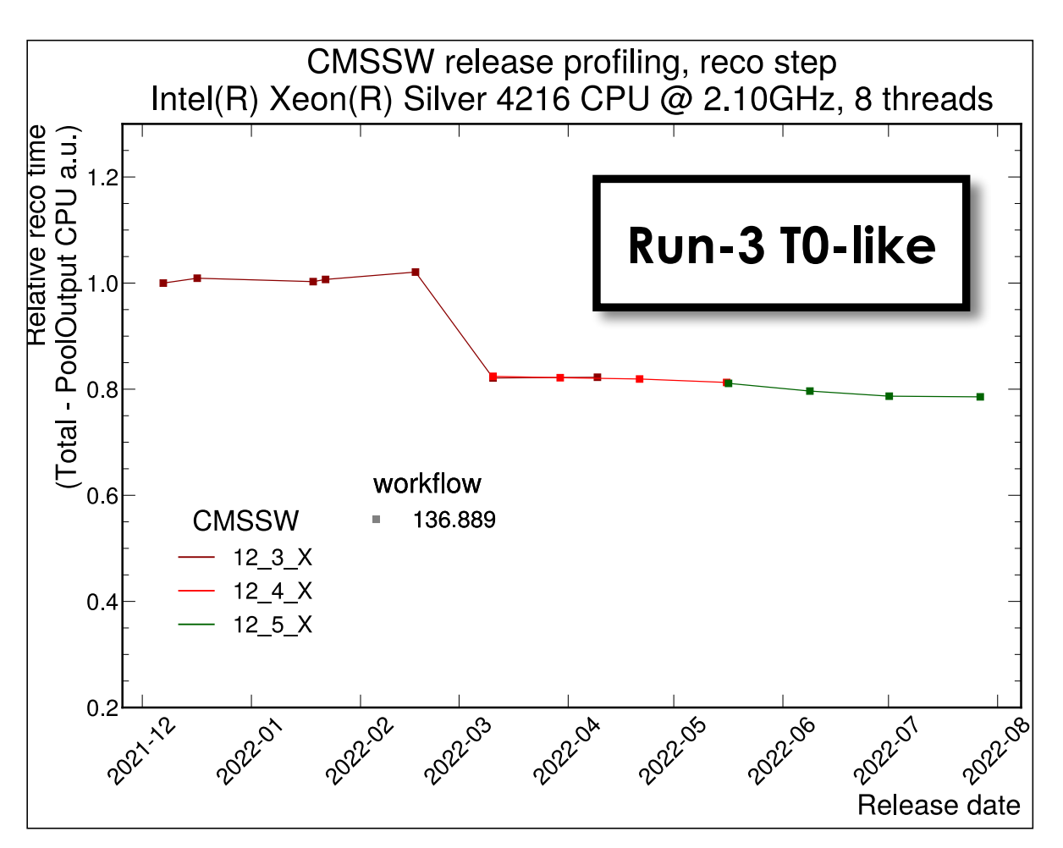
- 3 different conditions tested: Run3, Run3 T0-Like, Phase2

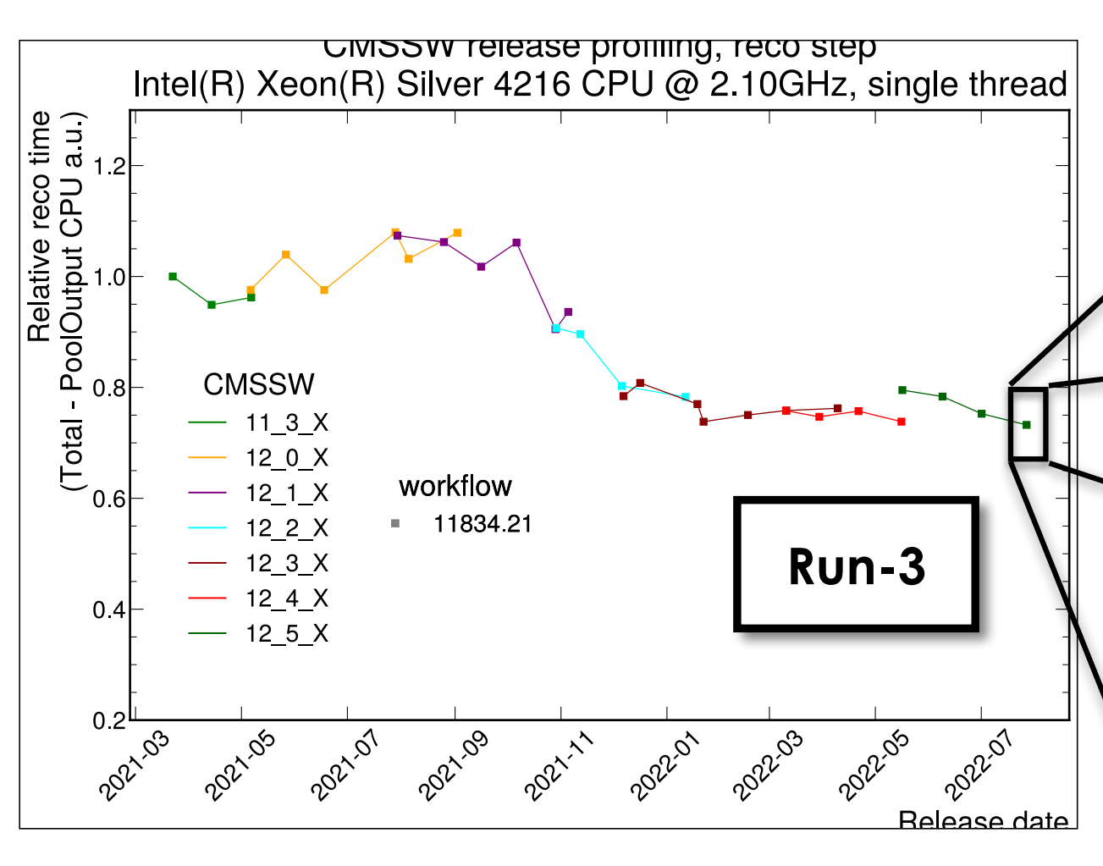

Run-3 T0-like
Phase-2
Run-3
Jenkins
Jenkins
Jenkins
igprof
CMSSW pre-release
release-run-reco-profiling
TimeMemoryInfo
igprof CPU
igprof MEM
release-analyze-reco-profiling
Compare CPU, MEM, eventSize with previous pre-release
produce igprof navigable report
Upload everything to a dedicated web site

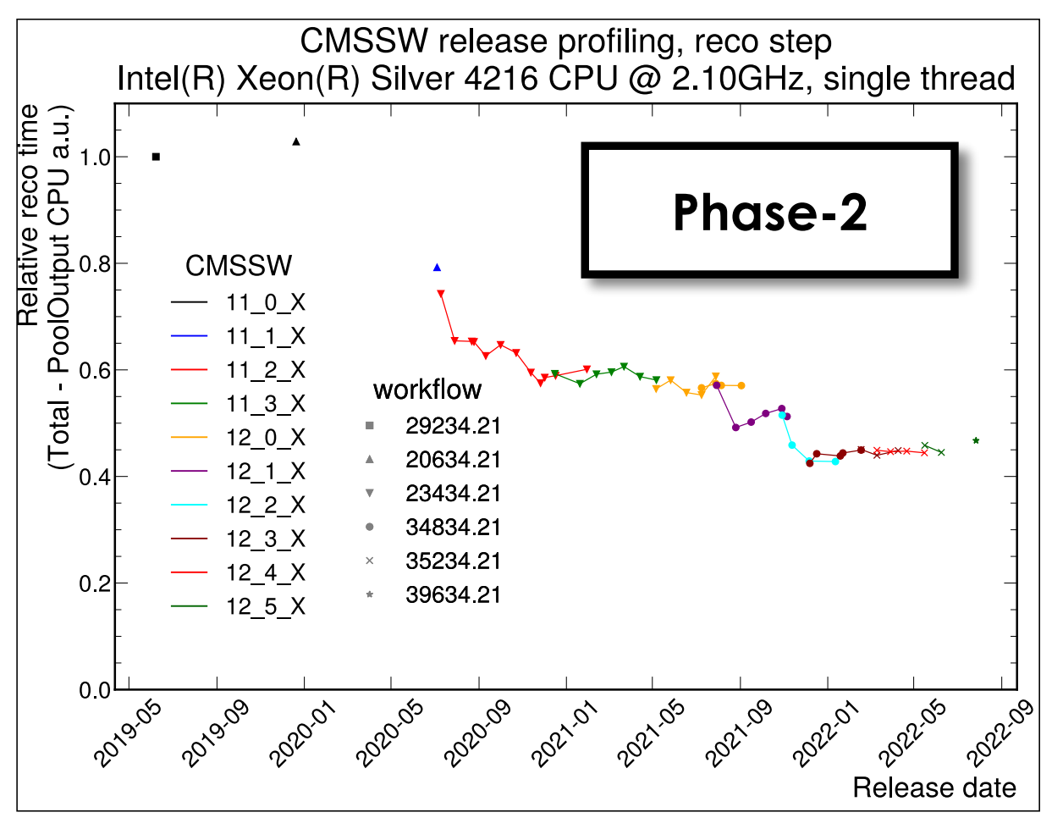

old browsers - Windows Internet Explorer
http://kni-labs.github.io/old-browsers/
Google
Favorites
old browsers

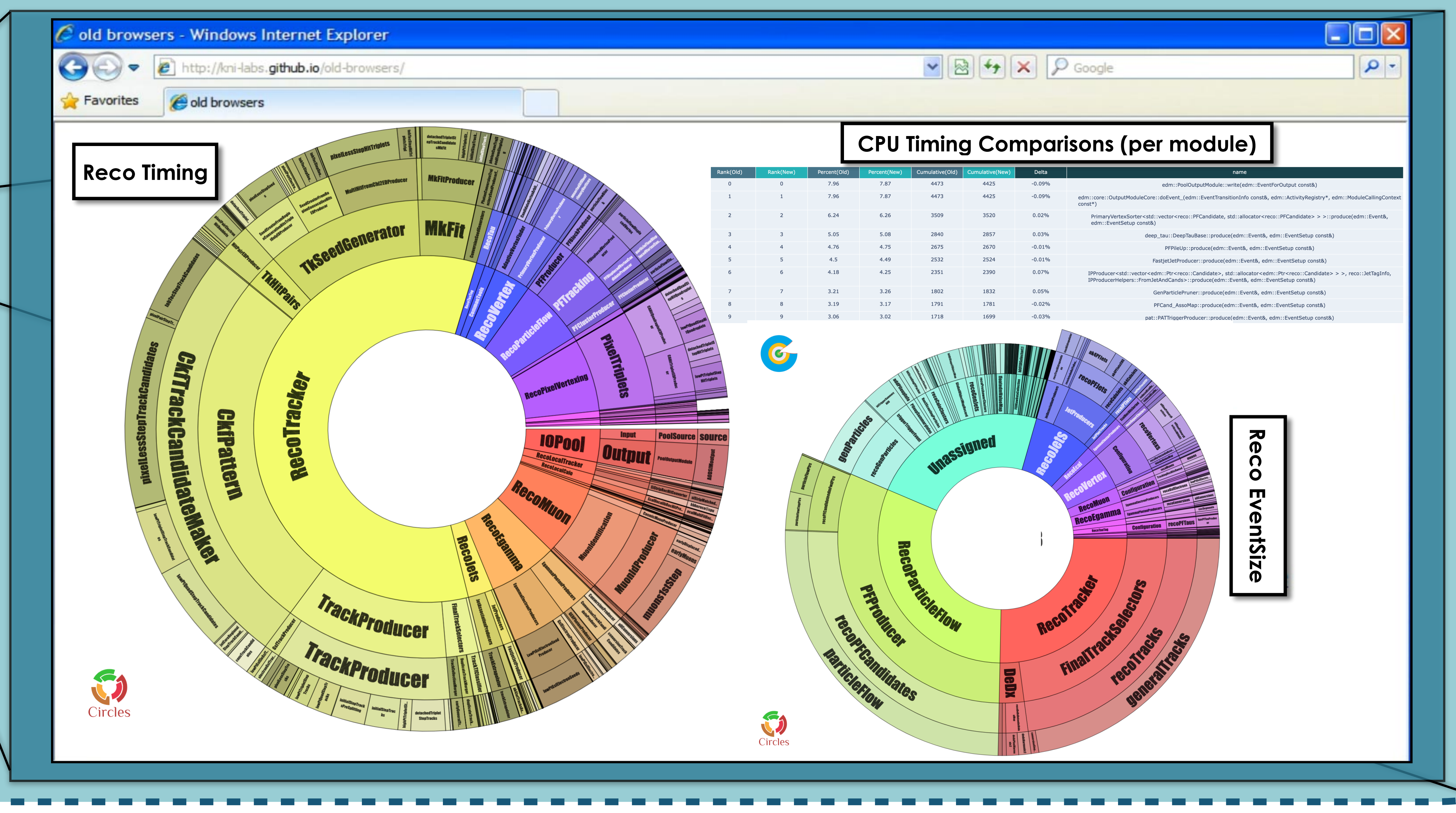
- CMSSW_12_6_0_pre3(11834.21)
  - [cmdLog]
  - step3_AOD(RECO)
    - [getTimeMemSummary] [cpu Profiler igprof][txtLink] [mem Profiler igprof1][txtLink] [mem Profiler igprof200][txtLink] [mem Profiler igprof399][txtLink]
    - Comparison :
      - [TimeDiff] preview: 13.3334 s/ev ==> 13.3115 s/ev
      - [CompareOutProd (edmEventSize)] preview: 682694 ==> 655514
    - Igprof comparison :
      - [cpu compare]
      - [mem.1 compare]
      - [mem.399 compare]
    - Circle (Pie chart) : [Circle]
    - EventSizeCircle (pie chart) : [EventSize]

CPU Reconstruction Timing Summary Plots









References

[1] https://cms-sw.github.io/
[2] G. Benelli: The CMS software performance at the start of data taking, Nuclear Science Symposium Conference Record (NSS '08), 2008, doi:10.1109/NSSMIC.2008.4774926 (paper, slides).
[3] https://github.com/cms-sw/cmssw
[4] L. Tuura, V. Innocente, G. Eulisse: Analysing CMS software performance using IgProf, OProfile and callgrind, Proc. Computing In High Energy And Nuclear Physics (CHEP07), Victoria, Canada, 2007, doi:10.1088/1742-6596/119/4/042030 (paper, slides).
[5] https://igprof.org/
[6] https://github.com/igprof/igprof