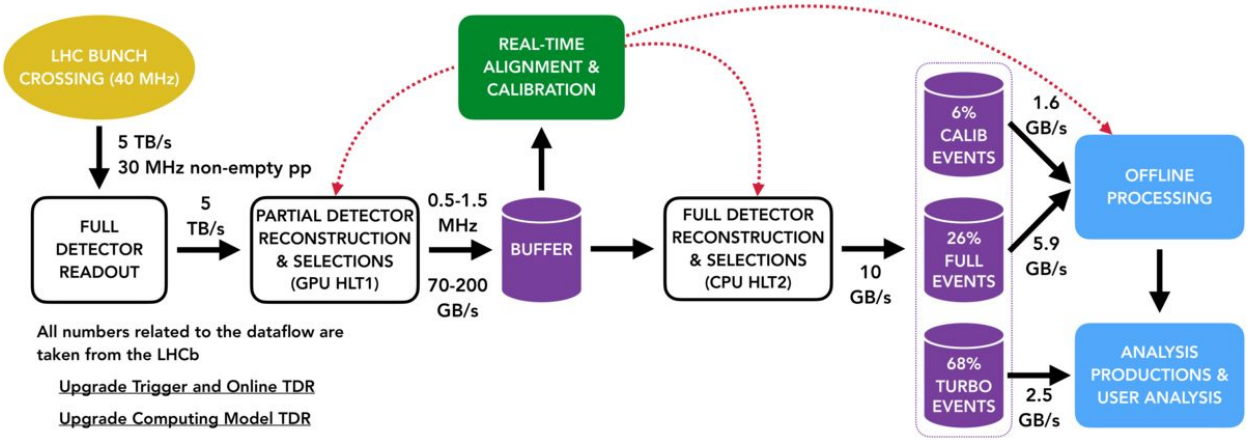
LHCb input

Eduardo Rodrigues, Nicole Skidmore
for the LHCb collaboration

HL-LHC Analysis Mini-Workshop, All@Home, May 4th 2021

“Real-Time Analysis” paradigm for triggering in Run 3

- Recall - Run 2 trigger:
  - Particle collisions at ~40 MHz, ~55 kB per collision,
    collected 9 fb^{-1}, > 10^{12} b hadrons produced in the acceptance
- From Run 3 onwards:
  - 0.6 GB/s -> 10 GB/s to storage, first-level trigger in GPUs, aim to collect ~10 fb^{-1} / year



https://lhcbproject.web.cern.ch/lhcbproject/Publications/fp/LHCb-FIGURE-2020-016.html

Real-Time Analysis - forward looking

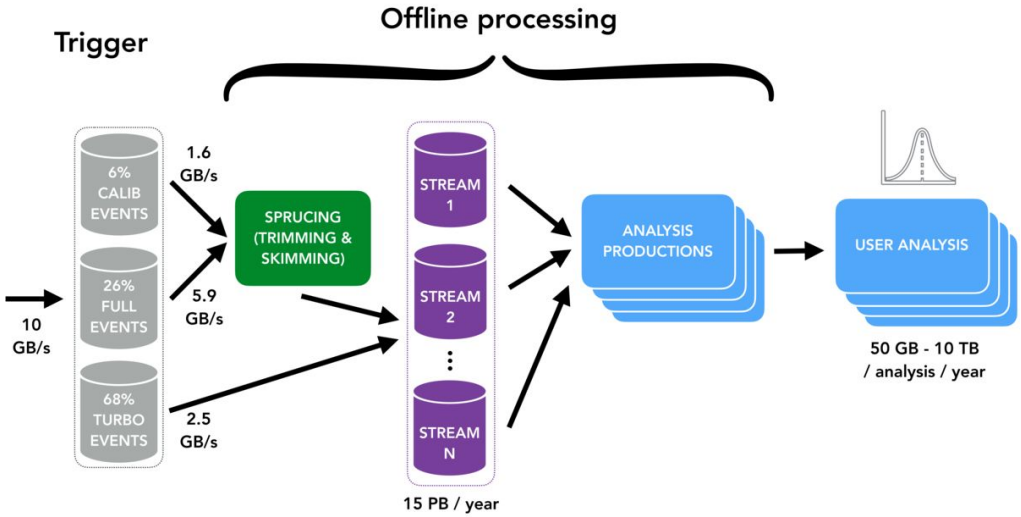
- A full software trigger is here to stay
- Likewise concerning GPUs for HLT(1)
  - “(1)” rather than “1” because it is not impossible to have a full HLT in GPUs on a timescale of a decade
    - “The sky is the limit”
    - FPGAs e.g. for clustering, are already being investigated, BTW. Even IPUs for ML ...

- More analysis will move “closer to the detector”
  - For Run 3, HLT2 is the terrain of selections but in the future …? Again, “the sky is the limit”
- Examples:
  - Run even more ML on the GPUs
  - Perform flavour tagging calibrations online (atop the alignment and calibrations for Run 3) if large enough samples become available to the buffer given the lumi increase?

Offline processing & analysis in Run 3 - two core ideas

- Centralised trimming and skimming (aka LHCb's "Sprucing")
  - A fraction of HLT2 outputs require extra processing before samples handed over to analysts
- Centralised analysis productions for physics WGs and users
  - Ensures e.g. better validation hence more efficient use of resources

- LHCb has a diverse range of analyses
  - Event sizes in Run 2: 18kB/evt -> 83kB/evt on disk (eg. CEP -> B to open charm event)
  - Throughput to **analysis** (kB/s) scales with luminosity



https://lhcbproject.web.cern.ch/lhcbproject/Publications/fp/LHCb-FIGURE-2020-016.html

Offline processing & analysis - forward looking

- Run-3 paradigm is here to stay,
  though it will evolve as our trigger strategy evolves
  - Unlikely to ever have all processing fully in HLT
  - And centralised production of ntuples seems anyway promising as the default
    - E.g. better validation
- If full HLT evolves to running on a GPU farm, Sprucing will likely follow (shared framework)
  - This would mean a full (central) data processing on GPUs!
    - Requirement to then be able to use GPUs efficiently also on the Grid?
- To set a scale: a typical Run 2 “Sprucing” campaign running over the full dataset takes ~2 months
  - This cannot scale as-is in the years to come

- Diversity of analyses will continue to be a reality
  - Throughput to **analysis** (kB/s) scales with luminosity

- Analysis preservation guidelines and “constraints” are also being included within offline processing & analysis. Expect these to be streamlined and become in many cases requirements rather than guidelines

Common Software

- Stating the obvious for completeness
  - Gaudi and ROOT are central pieces of our software stack
  - And that's not going to change
  - HLT1 on GPU farm with Allen software framework. Our baseline.
    To be seen if other experiments get interested, in which case it can become a common framework …
    TL;DR: invitation to get involved! E.g. open Gaudi-Allen workshop last year with LHC exps.
  - For Run 3 (and most likely beyond) several calibration tools (e.g. PID)
    will be using Data Science tools (several Scikit-HEP packages, SciPy packages, JAX, etc.)

- Stack development increasing involves only interacting with Python
  - All underlying C++ has Python wrappers
  - E.g. all application configuration and running of jobs done in Python

- We also now use Conda extensively

Analysis in Python & the Python ecosystem

- Already in 2018 an LHCb-wide survey showed that ~50% of analysts do their analyses in Python
  - ROOT and many Data Science tools such as SciPy, scikit-learn, TensorFlow, etc.
- The Data Science ecosystem is ever more used,
  and we assume this tendency will only get stronger in the next few years
- To this should be added our domain-specific tools:
  - Scikit-HEP project packages
  - zfit fitting package for CPUs as well as GPUs
- Hence we will rely on the "PyHEP ecosystem" (we already do for Run 3)
  - Several LHCb colleagues leaders or strongly involved in Scikit-HEP and zfit
- Already now, clear that we need e.g. to fit with GPUs
- Big question: how will the experiments help maintain these new tools, if?
  - Similar issue as for tools such as FastJet that we consume but do not maintain (?)

Training and development

- Students are no longer familiar with C++
  - Ever increasing trend since a handful of years
  - Serious implications for maintenance, especially that our HLT2 and HLT1 GPU are almost entirely written in C++ (configuration done in Python). We need more C++ experts!
  - How does this affect stack development going forward? Common issue

- Using GPUs for analysis needs to be incentivised by sufficient training/teaching material
  - Obviously an ever bigger opportunity for LHCb with its GPU farm for HLT1
  - Development of HLT1 for Run 3 involving Research Software Engineers was a challenge