

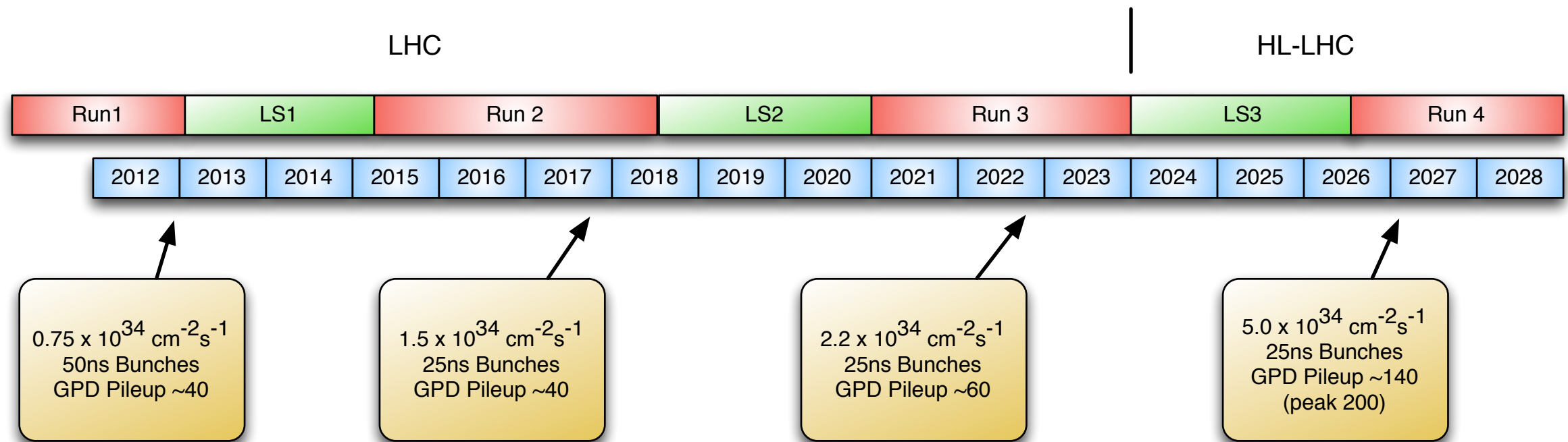


Software for the LHC Upgrade(s)

Graeme Stewart

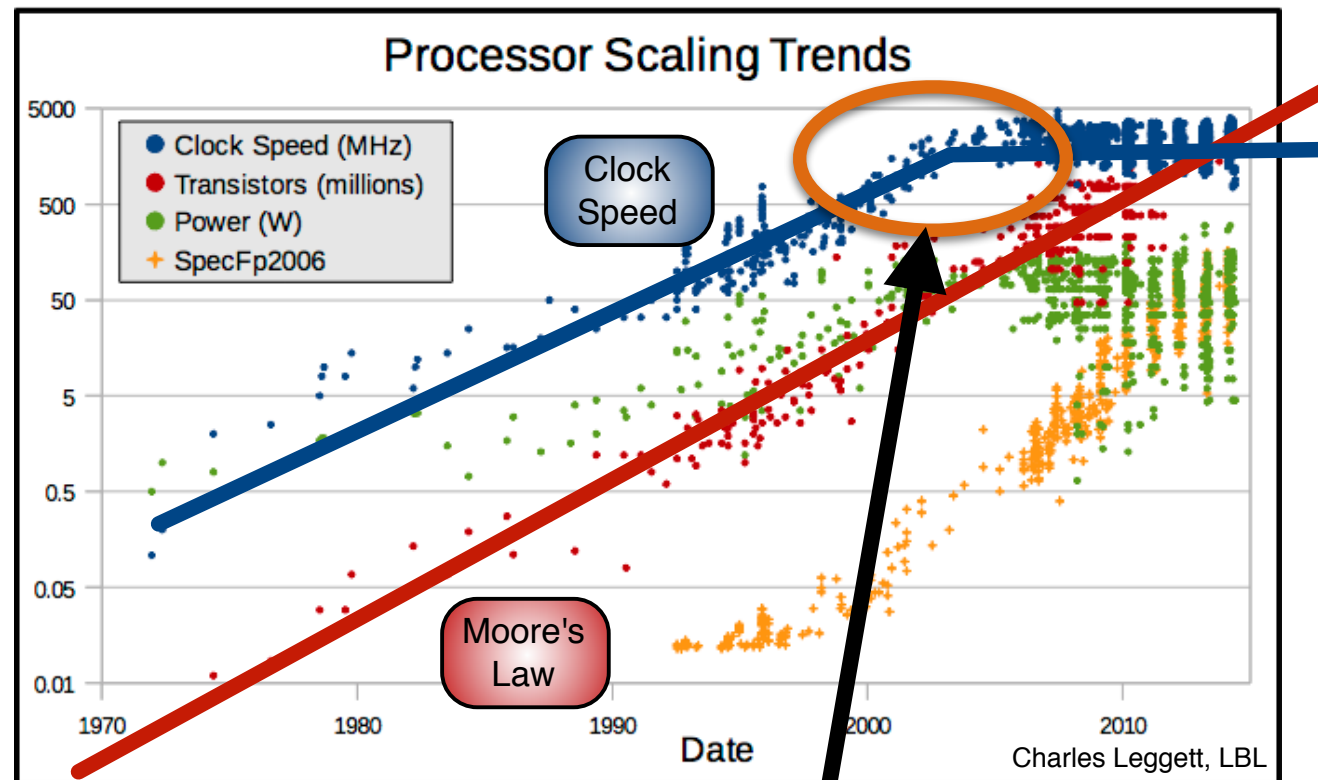
(with the help of many colleagues in ATLAS, CMS and LHCb)

LHC to HL-LHC



- Major upgrades for LHCb and ALICE come for Run 3 (2021)
 - LHCb: 40MHz readout with upgraded electronics and *full software trigger*
 - ALICE: Upgrades to inner tracking and time projection chamber, 50kHz trigger less readout for Pb-Pb; *new online/offline data reduction framework O2*
- Major upgrades for ATLAS and CMS come for Run 4 (2026)
 - New silicon inner tracker detectors with track triggers at L0/1 rates
 - Must *adapt to very high pileup* (140-200) and *much higher trigger rates*
 - Recall that *tracking is combinatoric*: factorial with pileup

Processor Evolution



Our software
developed here

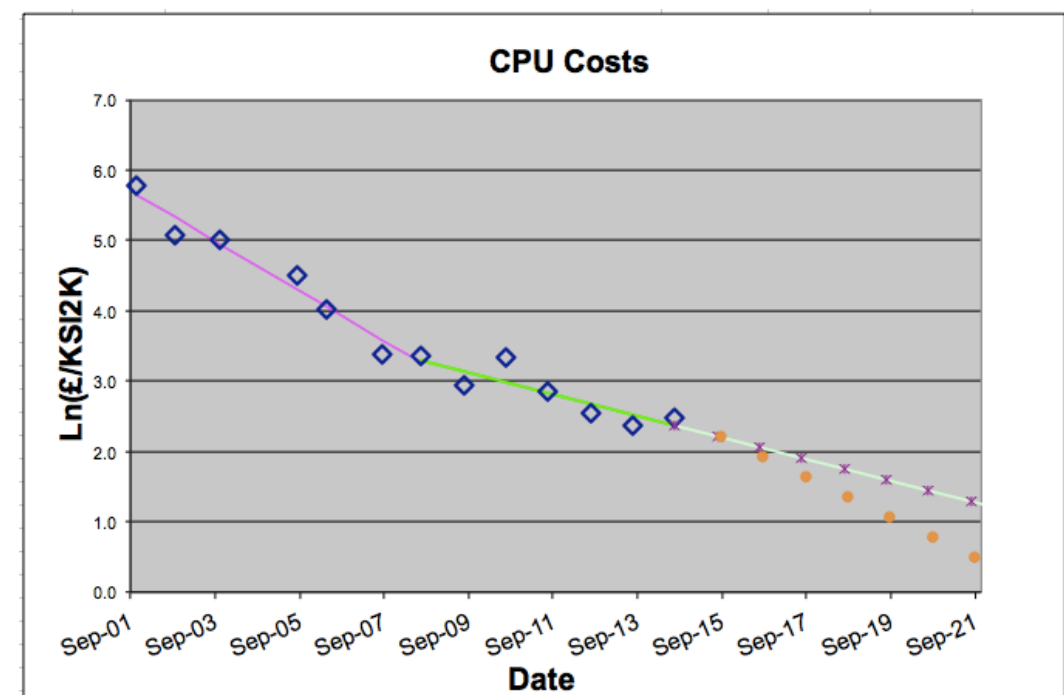
- Moore's Law continues
 - Doubling transistor density every ~24 months
 - Exact doubling time has a significant effect when integrated out to LHC Run 4
 - CPUs could be between x10 and x30 denser
- Clock speed stalled ~2005
 - Single core performance is essentially also stalled
- Driven now by energy performance
 - Figure of merit is nJ per instruction
 - Mobile devices and data centres are the key volume markets
- Memory consumption is a huge driver now

Silicon Diversity

- Doubling transistor density does not double our computing throughput
 - On the die we have more and more cores
 - Lower memory per core
 - Larger caches, but with decreasing payoffs
 - Wide vector registers, ever harder to fully utilise
 - Built in ‘specialist’ features, e.g., integrated GPUs in Intel Skylake
 - Integrated network controllers — more *System on a Chip* (SoC)
- GPGPU type architectures throw away a lot of the assumptions of CPUs
 - Banks of cores executing the same computational kernels
 - Very fast, but very local memory (forget cache coherence)
- None of these features and architectures are trivial to take advantage of in our code
 - Our frameworks and algorithms written for an earlier era of hardware and are hard to adapt

Modern Computing Hardware

- Away from the detector itself we are firmly *Commodity Off The Shelf* (COTS)
- Increased transistor density does not reduce all the other costs involved in actually building a real server
- Or allow us to simply build computing systems with higher throughput per £
 - Disk: capacity still going up, but i/o rates are basically the same as ever
 - At the moment SSDs are not an affordable replacement for 500PB of spinning disks
 - Tape: Healthy progress (less technical challenge than on disks), as ever slow to read back
 - Network: Capacity keeps rising, allowing cross site boundaries to become less important, but data needs to be read from a physical device *somewhere*



RAL Tier1 CPU Costs with projection

David Britton, Andrew Sansum, GridPP



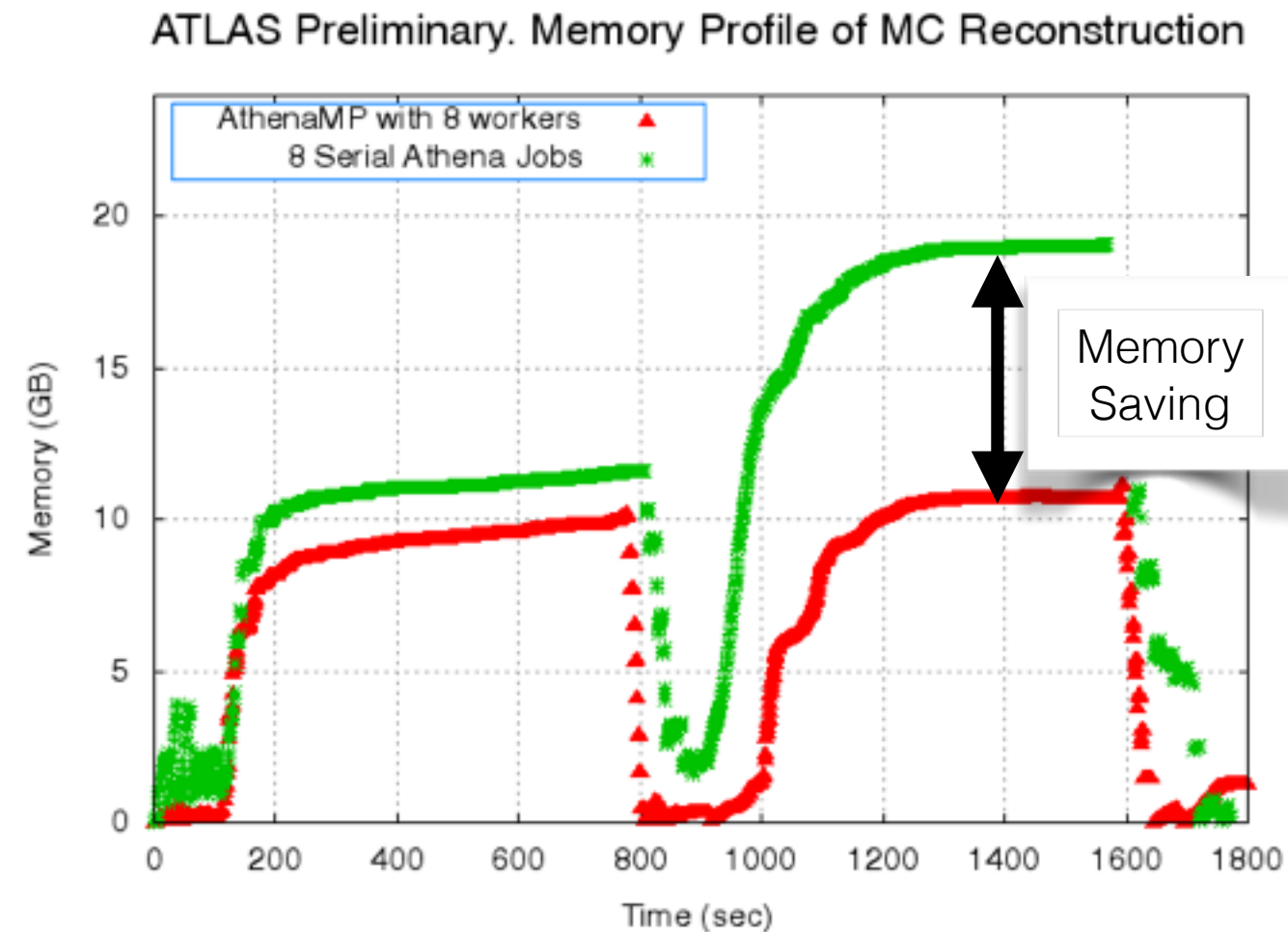
Software Challenge



- Current computing hardware evolution is not providing any more free lunches
- Demands on software to support our physics goals are increasing
 - Higher trigger rates
 - Higher pileup
 - Need *faster* and *better* software
- Our software was written for a different era
 - By people who have often now moved on to new areas
- We need to train a new generation of physicists in modern programming methods
 - C++98 → C++11 → C++14
 - Modern tools and development methods
- This is to say nothing of evolving our computing infrastructure on top of whatever software we have to meet these challenges

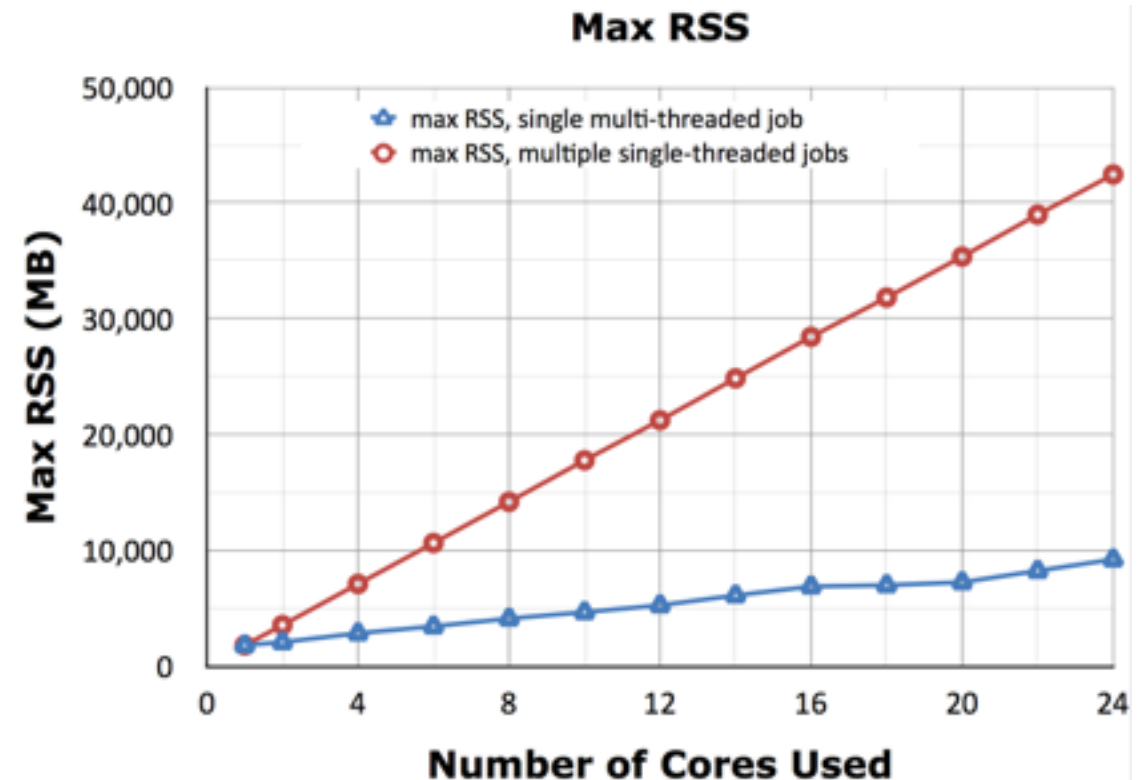
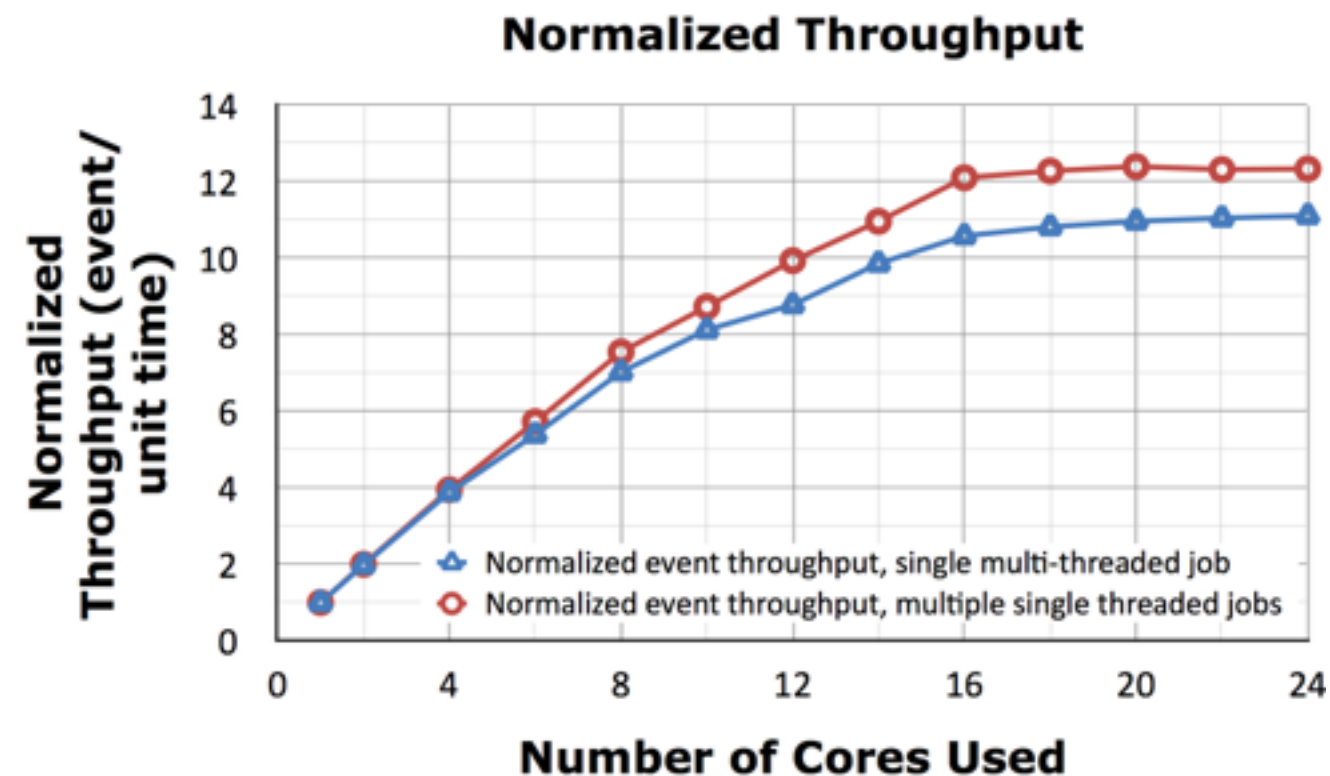
Framework Upgrades

- LHC experiment software frameworks were developed in the serial processing era
- As multi-core CPUs became more common trivial multi-processing was used
 - On N core launch N jobs
- First attempt to make this better was ATLAS's AthenaMP
 - Start Athena in serial mode, then fork worker processes after initialisation
 - Large memory structures (geometry, magnetic field) are shared by Linux kernel
- However
 - Unlikely to scale, even to Run3 parameters
 - Use of opportunistic resources and Xeon Phi demands even better memory/core performance than today
- Multi-threading is the future
 - True heap level memory sharing between all threads
 - (Far greater opportunities for making mistakes)

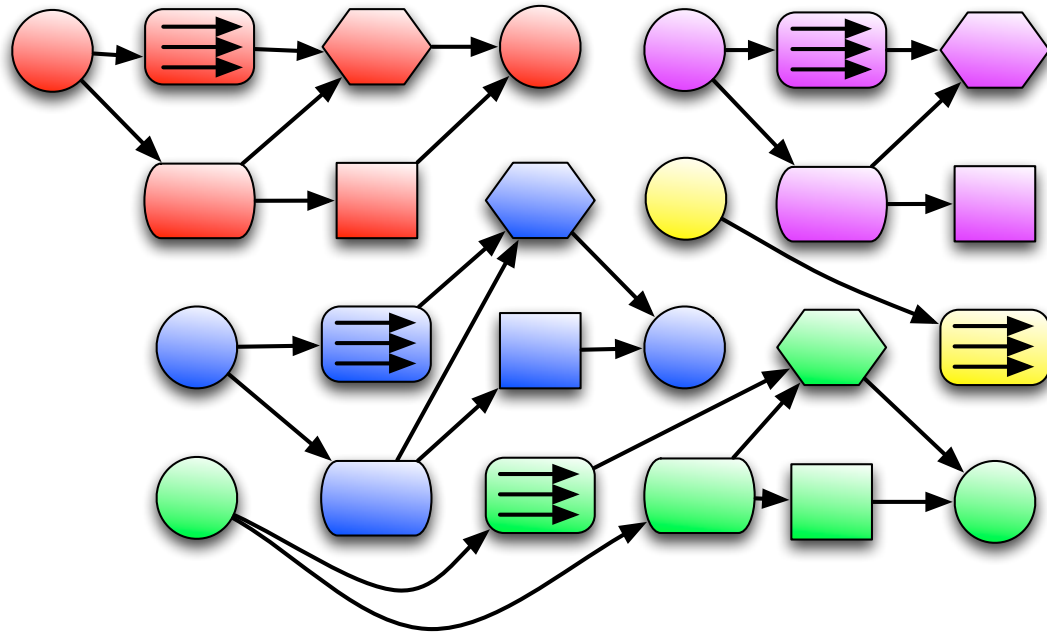


CMSSW Multi-threaded

- Split the concept of event processing into global and stream
 - Global sees the whole event and all transitions
 - Stream sees some events, in a defined sequence
 - Like an AthenaMP worker, on a thread
- Thread-safety is vital at the global level, less important at the stream level
 - Allows for a factorisation of the problem for framework transition
 - Good use made of static code checkers



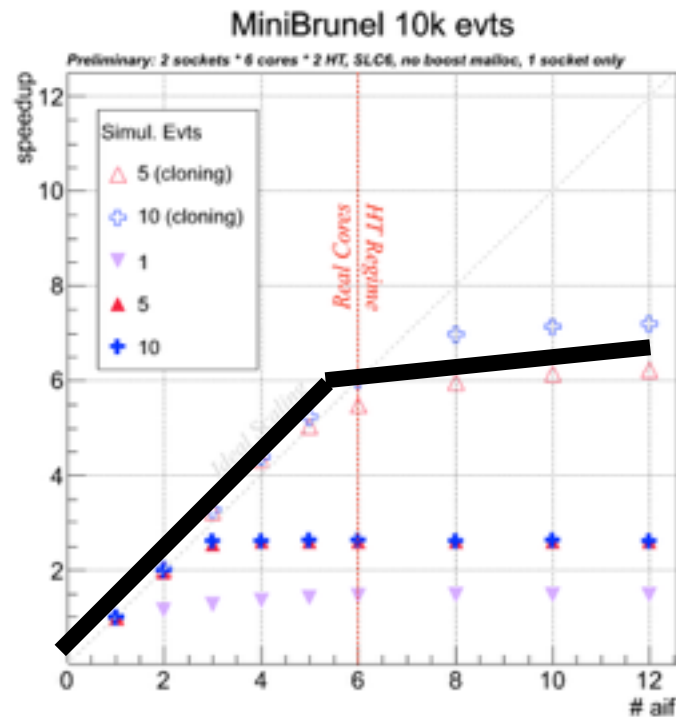
Gaudi



Multi-threaded processing
cartoon: each colour is a different
event, each shape a different
algorithm

- Gaudi framework was initially developed by LHCb and later adopted by ATLAS
- Now being adapted to multi-threading
 - Designed to exploit concurrency at many levels
 - Event level concurrency — multiple events in flight
 - Algorithm concurrency — independent algorithms can run in parallel
 - In-algorithm parallelism — heavy cpu consumers can exploit parallelism themselves (e.g., clustering, jet algorithms)

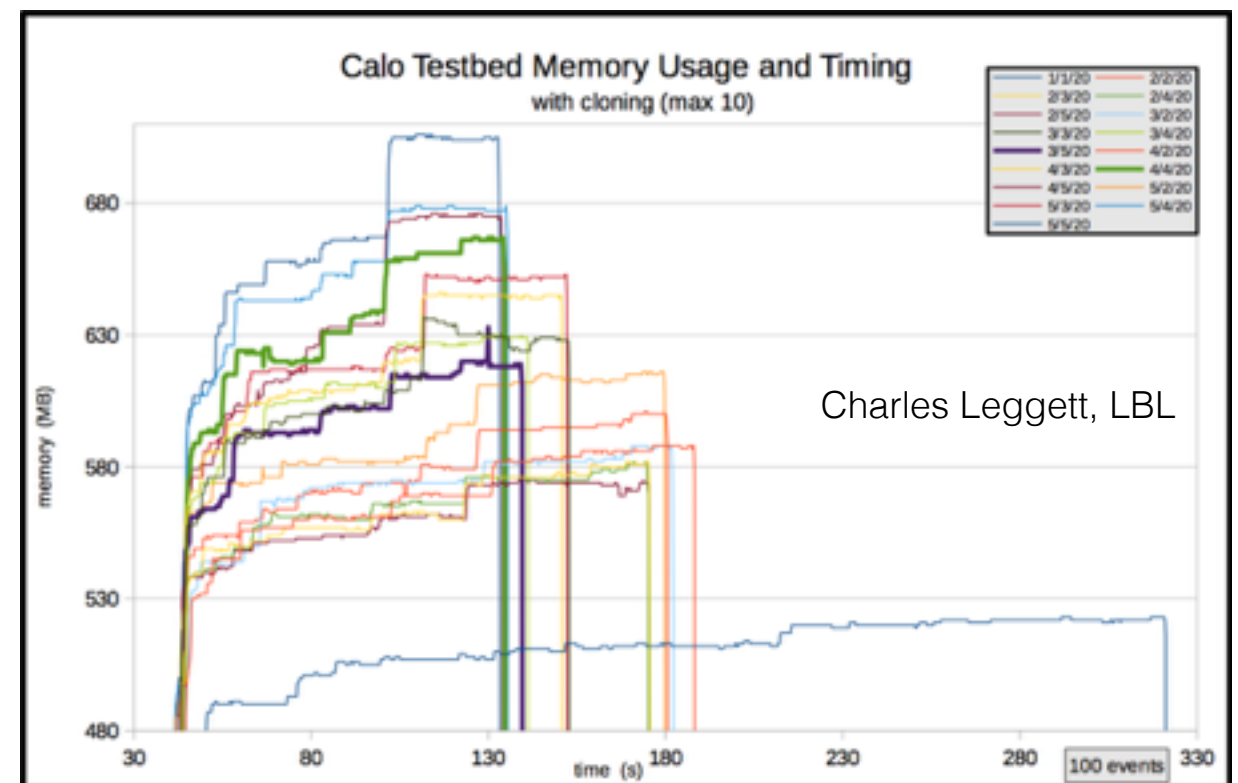
Mini-Brunel and AtlasHive



- Scaling of GaudiHive running LHCb mini-Brunel reconstruction
- Linear scaling up to CPU core count
- Expected boost from hyperthreading with only 10 events in flight
- Memory consumption only rises by 7% (limited reconstruction however)

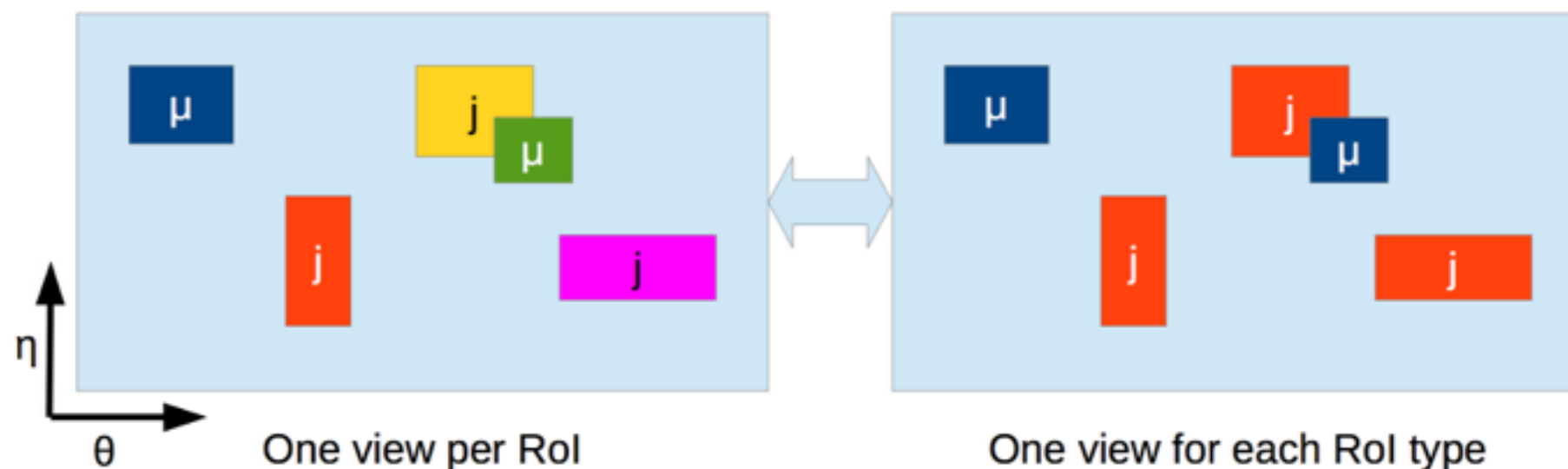
Benedikt Hegner, Danilo Piparo, CERN

- ATLAS Calorimeter testbed
- Best scaling x3.3 for 28% memory increase
 - Concurrency was limited here due to some serial components — expected improvements seen
- Multi-threaded simulation close to working



Gaudi and ATLAS HLT

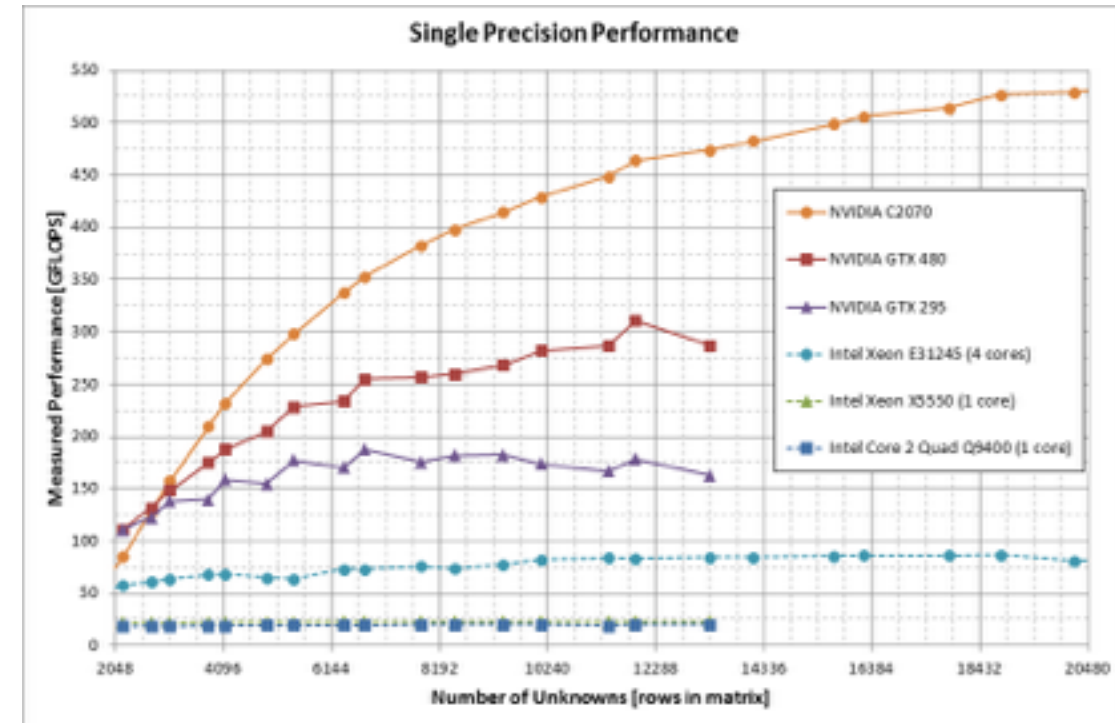
- The efficiency of the ATLAS HLT requires partial event processing in a region of interest (RoI)
 - ATLAS want to manage this with unmodified offline code (currently not possible)
- This was not a concept in Gaudi originally
 - The extension to manage this is called an *Event View*
- Currently assessing the impact of a design with one view per RoI...
 - Closer to the current HLT design, easier migration; harder scheduling problem
- ...or one view per RoI type
 - Larger issue to migrate current code; easier for the scheduler



Ben Wynne,
Edinburgh



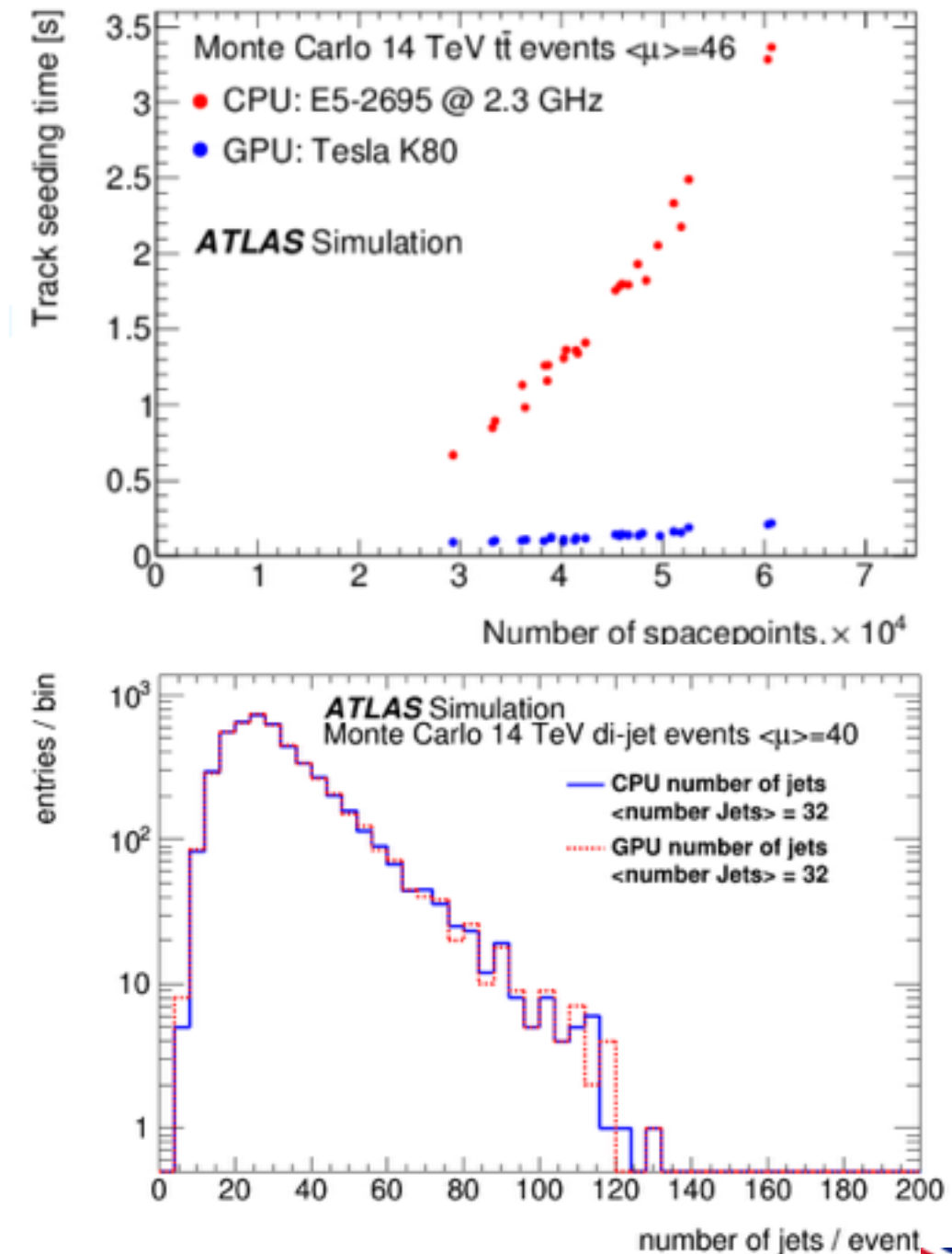
HLT and GPUs



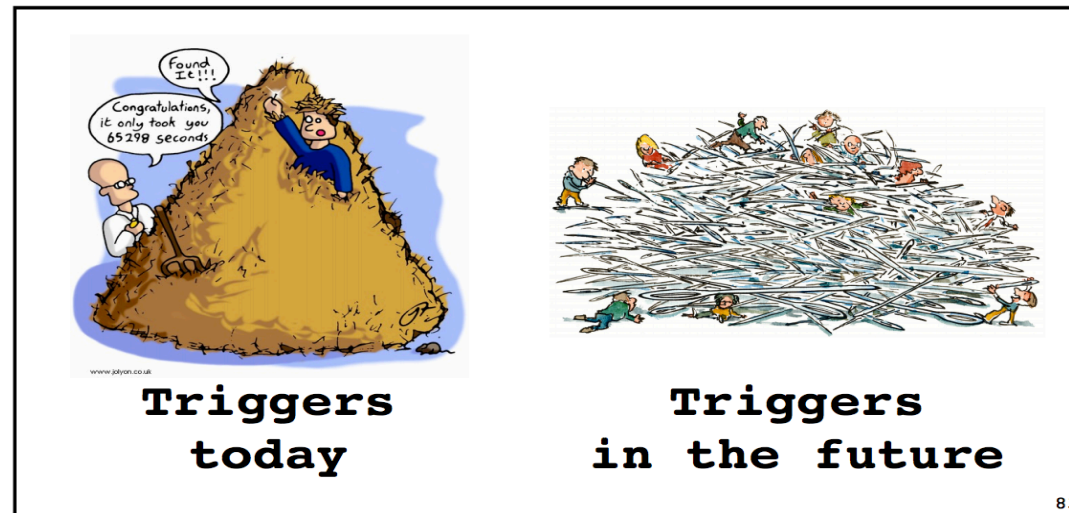
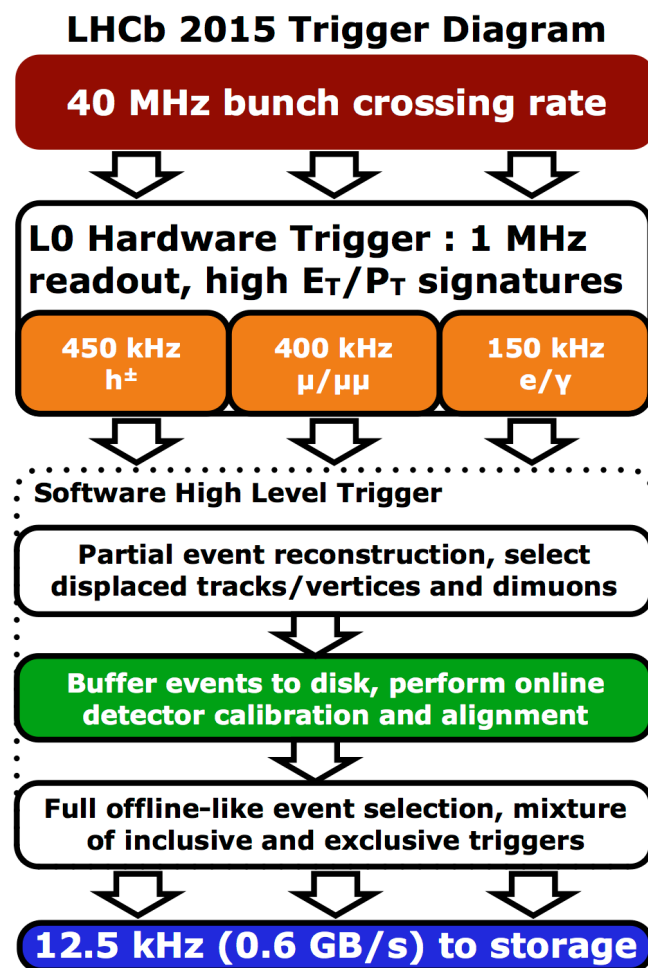
- Generally GPUs are much harder to use than alternative CPU architectures such as OpenPower, ARM64
 - Limited C++ support
 - Compilers still developing quite fast here — beware of vendor lock-in
 - Very different memory model
- However, online processing is a far more controlled environment than any generic grid site and has a more restricted workflow
 - Opportunities for GPUs are greater
 - e.g., ALICE have already deployed 50/50 split between GPU and CPU for their Run 2 trigger farm
- Raw FLOP power of GPUs does make them an option which we much take seriously

ATLAS HLT GPU demonstrator project

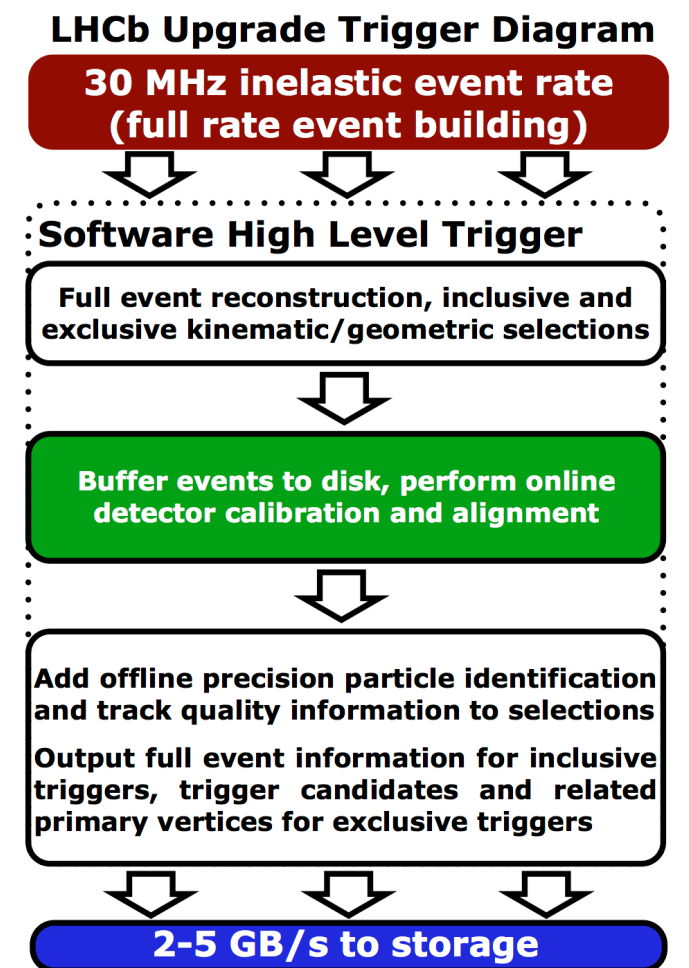
- RAL leading HLT track finding in Inner Detector porting efforts
 - Track seeding is 50% of CPU spent on full scanned events
 - Also work to port of topo clustering, muon segment finding and jet reconstruction
- Careful validation of results — generally not bit for bit identical, so need to check physics level validity
- Early results show a x1.8 overall speed-up when running with a Tesla K80 GPU
- However, further speed up requires porting the ‘long tail’ of algorithms to GPU
 - Diminishing returns in lines of code to port vs. eventual speed up
 - Need to also factor in costs of code maintenance and costs of physical infrastructure
- Still under investigation — no decision for Run 3 yet, with LHCb doing similar work, similarly with no firm conclusion yet



LHCb Run3 — Software Trigger



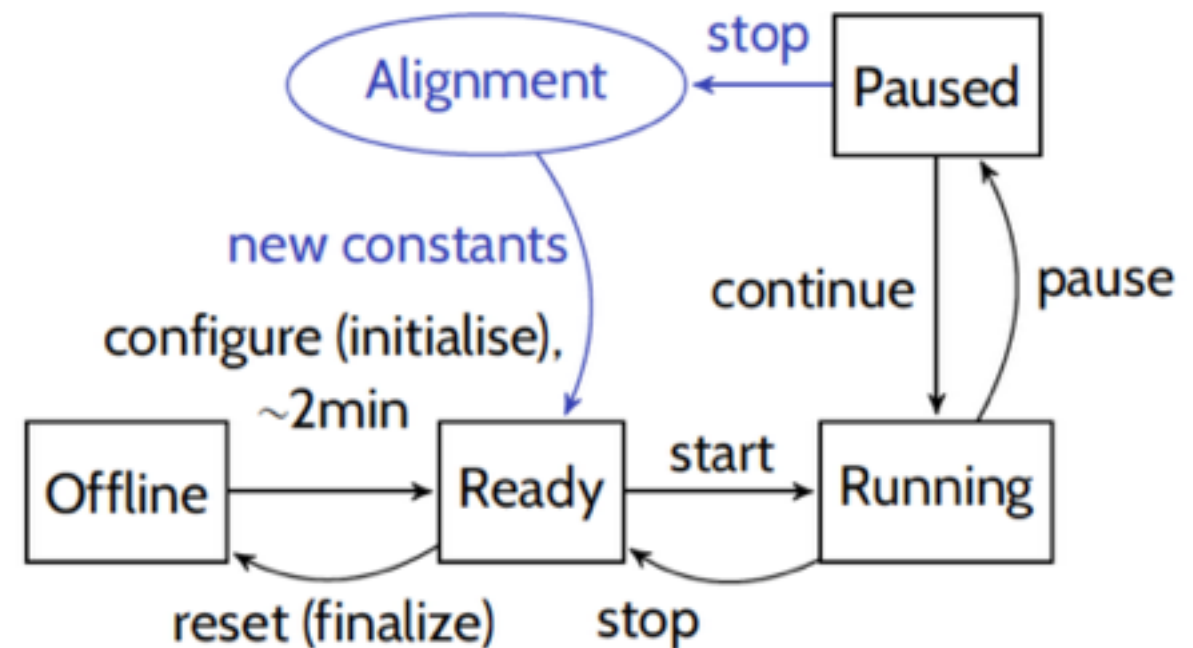
Vava Gligorov, CERN



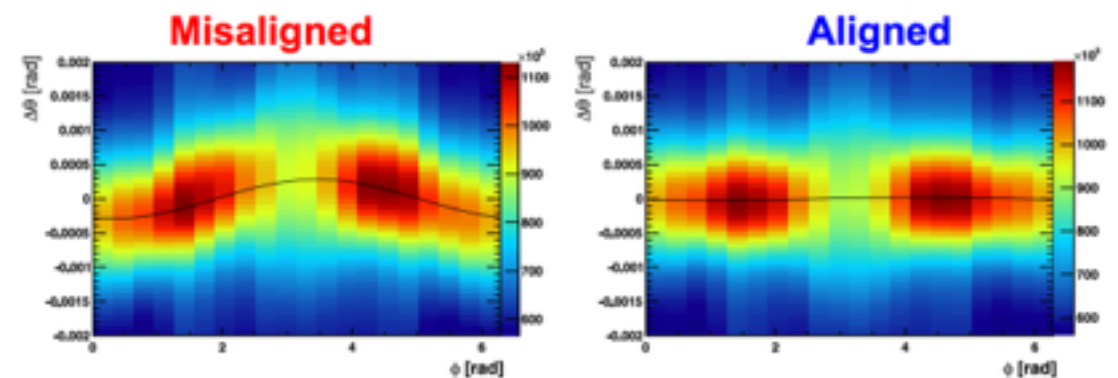
- 30MHz events to be triggered in software
 - 100kB event size → software trigger processes 3TB/s
- Significant strain on CPU efficiency of the trigger software
 - *Need considerable software improvements*

Real Time Calibration

- Don't want to discard information unless trigger produces “offline” quality reconstruction directly
 - Requires very fast and accurate calibration in “real time” == a few hours
- HLT1 runs — first pass reconstruction
- All events then buffered to disks on the HLT farm
- Real time calibration runs to assess calibration and alignment
- HLT2 runs with updated alignment to produce final outputs
- Most of farm occupied with HLT1 during data taking
 - Needs to be tuned to LHC data taking efficiency



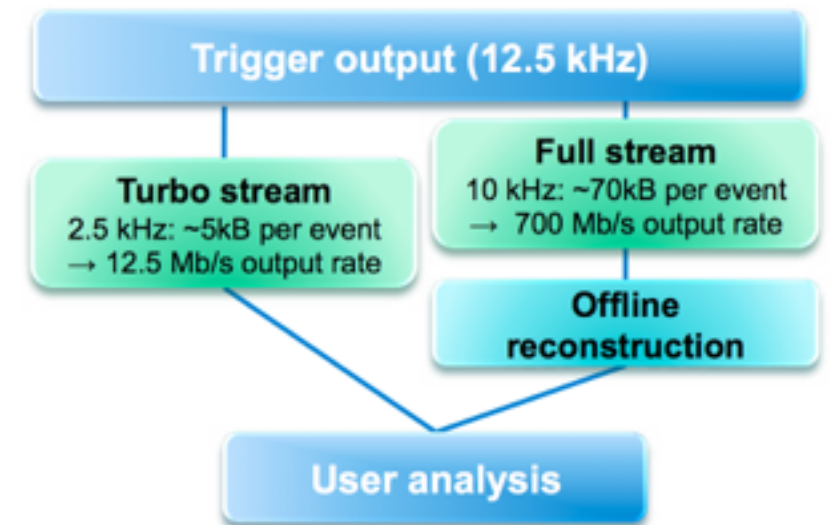
Alignment Workflow



RICH Alignment

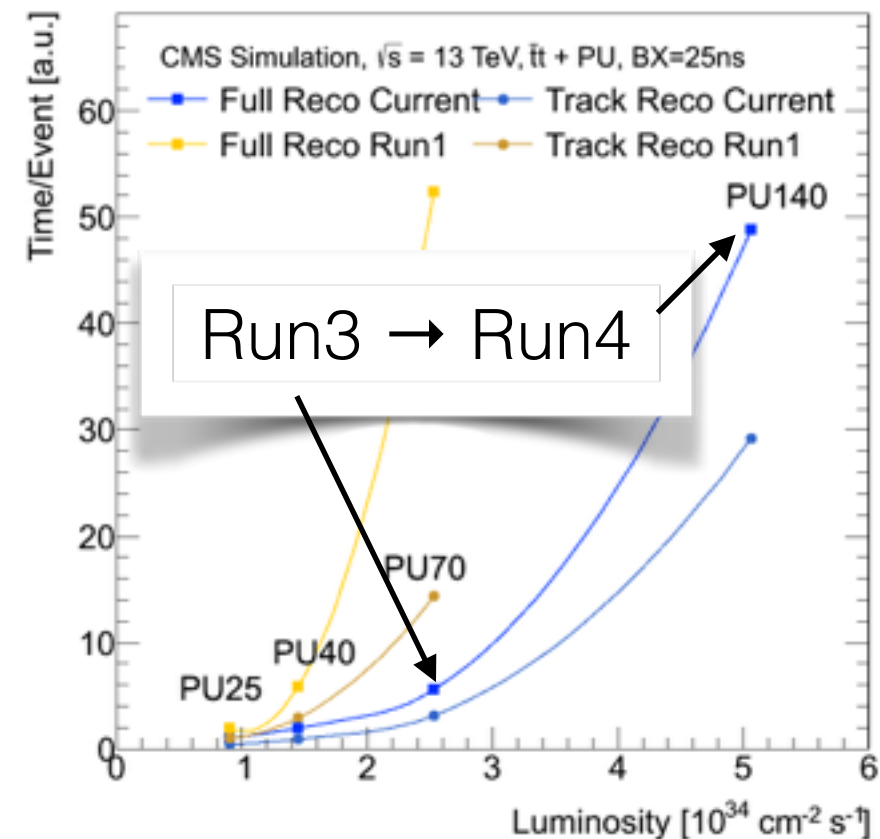
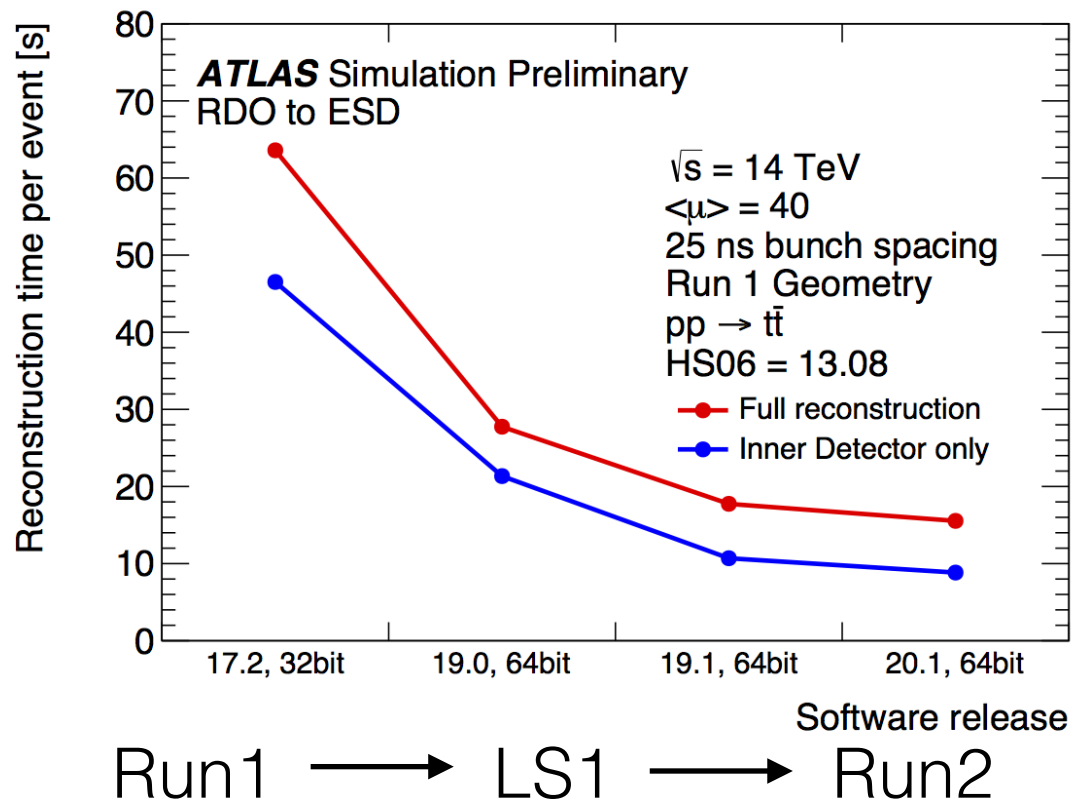


Turbo Stream



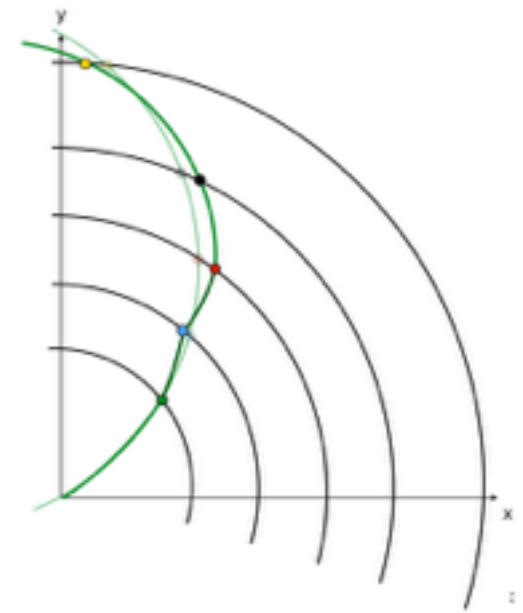
- Write analysis ready outputs DSTs directly online
 - Calculate luminosity information and resurrect trigger candidates
 - 70kB/event → 5kB/event
- Ideal for analyses with very high signal yields
- Already running and producing physics during Run 2
- Full deployment in Run 3
- Trigger stops being binary, but discriminates different signals and selects information to keep based on ultimate constraint of rate to offline
 - $N_{\text{events}} \times N_{\text{event_size}} = \text{data rate}$
 - Can play with these parameters based on analysis needs
- Reduce offline resources, e.g., less need for any reprocessing

Tracking — On the road...



- HL LHC means high pileup ATLAS and CMS
 - Track reconstruction rates go from 1M tracks/s to 60M tracks/s
 - Combinatorics of charged particle tracking become extremely challenging for GPDs
 - Even smart approaches scale a worse than linear
- Impressive improvements for Run 2
 - Option to throw more and more events into flight for Run 3, but memory is not free
 - Current strategies will actually just not work for Run 4

Future Tracking

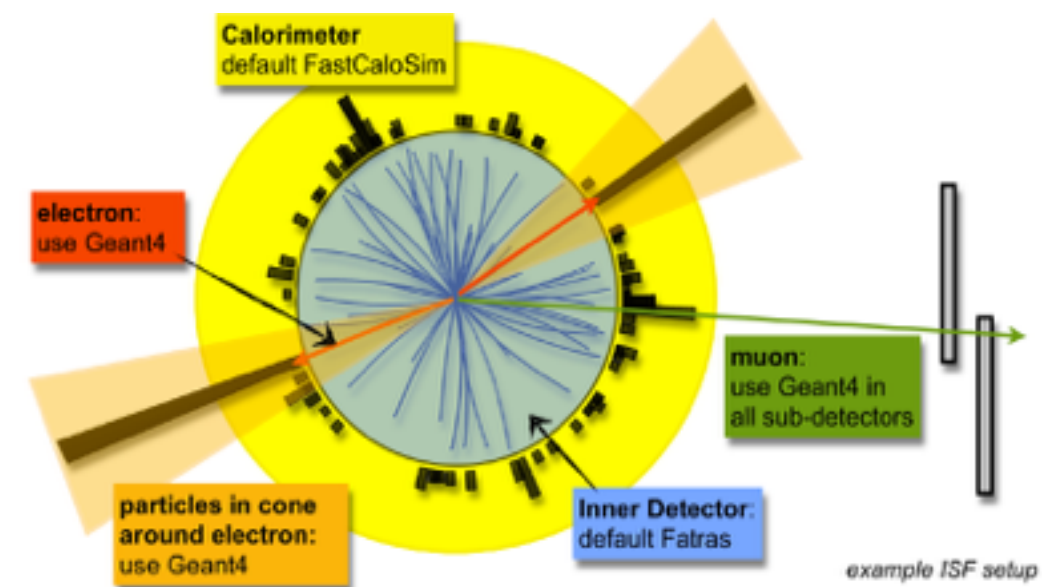
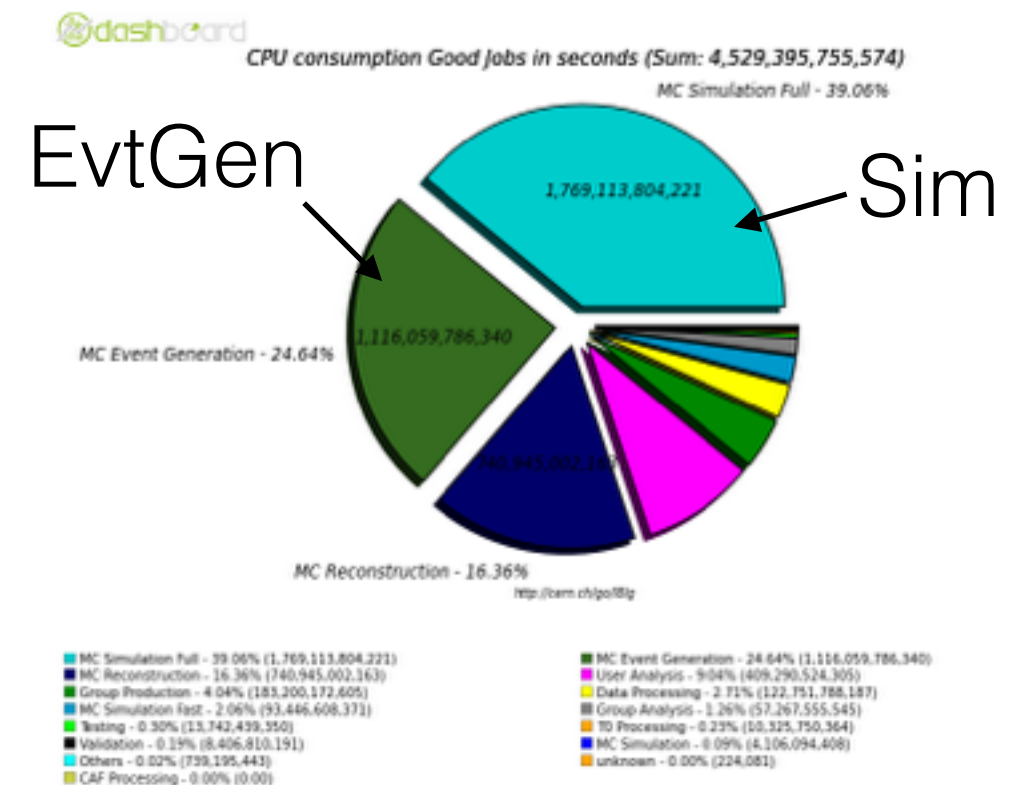


- Tricks for better scaling
 - Truth tracking for MC
 - Data could seed tracks from track triggers (but track triggers are extremely difficult to simulate)
- Tracking from cellular automata parallelises much better
 - Local problem, instead of global
 - However, can the physics performance be maintained for ATLAS and CMS?
- Conformal mapping techniques (Hough transform) can parallelise much better
 - But they don't cope so well with material scattering
 - Need recovery strategies for kinked tracks
 - Maybe it's machine learning
- Also want to vectorise
 - Could give x8 speed up on 512bit registers
 - Difficult and technically tricky work
- ACTS project factorises ATLAS tracking for FCC and other studies

Common Efforts: HEP
Software Foundation
Reconstruction Algorithm
Forum

Event Generation, Simulation, etc.

- Simulation eats up much of our distributed computing resources
 - With event generation becoming also important (NLO and beyond generators are expensive)
- Pileup scaling does not really apply to simulation
- Digitisation scaling is ~linear with pileup
 - But mixing in pileup events is a nasty problem — either it's memory hungry or i/o intensive
- Fast simulation is increasingly important (ATLAS ISF pioneered mixing fast and full simulation)
 - Supported by fast tracking and reconstruction
- Some more radical approaches may bear fruit for HL-LHC
 - GeantV prototype attempts to bring vectorisable code to simulation
 - Transport many particles in each vector register
 - But it will require *huge work* to get the physics right in GeantV and for experiments to manage to adopt it



ATLAS ISF

Conclusions and Prospects

- Experiments are all progressing on significant upgrades for Run 3
 - Frameworks are evolving to multi-threading
 - Migration of millions of lines of existing C++ will not be easy at all
 - Need to train a new generation of coders in modern methods
- LHCb planning a software only trigger for Run 3
 - Real time alignment system and Turbo Stream
 - Still needs a large boost in software performance
- Study of GPGPUs might improve event selection at reduced costs
- Tracking for ATLAS and CMS must adapt to HL-LHC
 - Very likely to bring revolutionary changes in software
- Event generation, simulation and digitisation all need to be improved as well
- Effort in these areas is *critical to LHC experiments' success*
 - Great software will increase physics return; poor software will hamper it
- UK funding and involvement here is poor to middling — more support would pay off well