



**CHEP2015**  
OKINAWA, japan

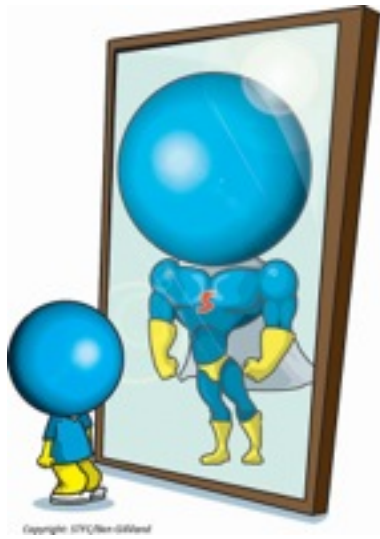
21st International Conference on Computing in High Energy and Nuclear Physics **CHEP2015** Okinawa Japan: April 13 - 17, 2015

# Evolution of Computing and Software at LHC: from Run 2 to HL-LHC

Graeme Stewart

# Overview

- LHC Evolution
  - Physics motivations for Run3 and High Luminosity
  - Detector upgrade plans
- Future Computing Hardware
  - What does the future hold for commodity computing?
  - Guess at scaling out to 2025
- Software for Upgrades
  - From hardware to software design patterns
  - Challenges: Tracking, Simulation, Generation, Event Generation
  - New models and new frameworks
  - Algorithmic and Analysis Code Evolution
- Scaling to 2025
  - The data and computing challenge at High Luminosity
  - Future resource provisioning
- Conclusions



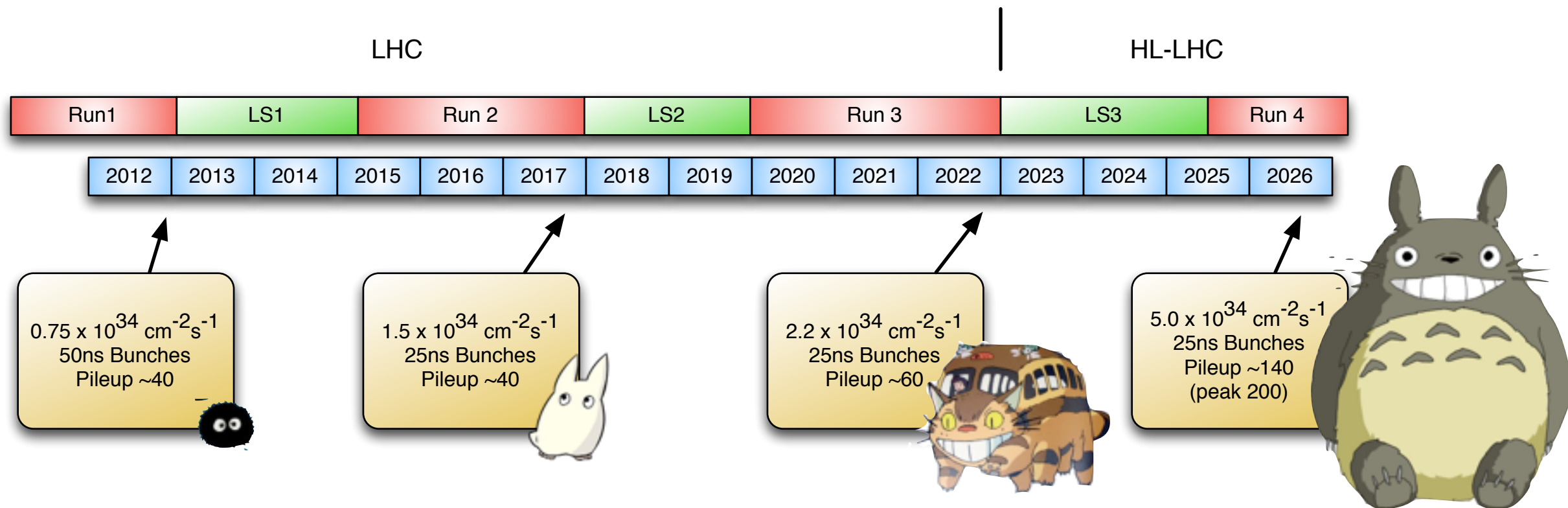
# Physics Goals of HL-LHC



From [PhD Comics](#)

- Didn't we find the Higgs already?
  - Yes, but now we need precision Higgs measurements from ATLAS and CMS (General Purpose Detectors — GPDs)
- Requires high statistics of very rare processes
  - Measure Higgs properties, e.g.  $t\bar{t} \rightarrow H(\rightarrow \gamma\gamma)$  to measure coupling, try to assess its self-coupling
  - Vector boson scattering cross section
    - Consistent with standard model?
- High luminosity extends the reach into possible physics beyond the standard model (BSM)
  - Super-symmetry, dark matter candidates, something entirely new...?
- LHCb is very sensitive to rare processes that indicate beyond the standard model physics
  - CKM angle  $\gamma$  to 1 degree precision
  - $3\sigma$ -from-0 measurement of  $\phi_s$  (also new physics sensitive)
- ALICE specialises in quark-gluon plasma measurements - upgrade goals are precise measurements
  - Heavy flavour hadrons; Low-momentum quarkonia; Low mass di-leptons

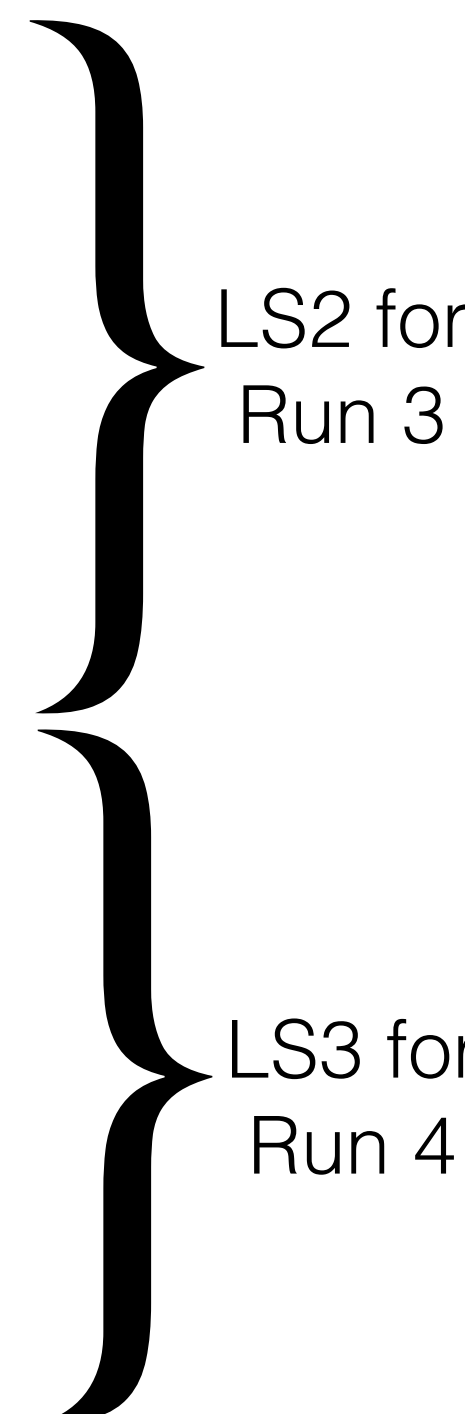
# LHC Machine Evolution



- Steady increase in machine luminosity both within runs and between runs
- Ultimate goal of  $3000\text{fb}^{-1}$  in 10 years of HL-LHC running
  - 📌 pp Collision rate of 5.6GHz
- Pileup is the most important metric of event complexity for reconstruction software

	Integrated Lumi (fb)	Pileup for GPDs
Run 1	25	25
Run 2	100	40
Run 3	300	60
HL-LHC	+300 per year	140

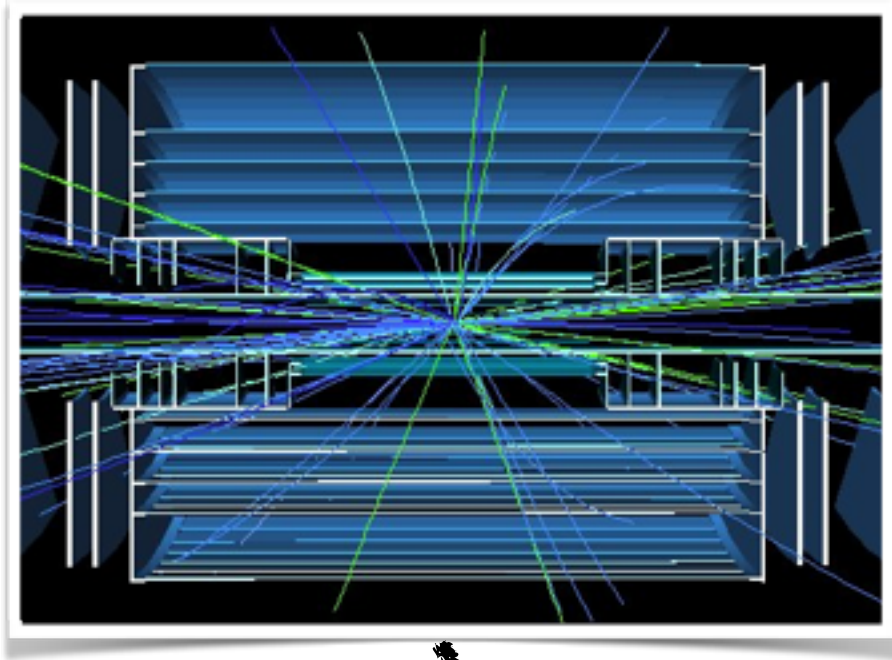
# Experiment Upgrades

- LHCb:
    - 1MHz→40MHz readout with a full software based trigger
    - Upgraded front end electronics and sub-detector readouts
  - ALICE:
    - Significant sub-detector upgrades to inner tracking and time projection chamber
    - Triggerless readout for Pb-Pb at 50kHz into new integrated DAQ-online system with smart data compression
  - CMS:
    - New silicon tracker with trigger capabilities (at interaction rate)
    - Other sub-detector and readout improvements (L1 latency and new calo end caps)
  - ATLAS:
    - New inner tracker (ITk) and readout improvements in calorimeters
    - Track trigger operating at Level 0 rates
- 
- LS2 for Run 3
- LS3 for Run 4

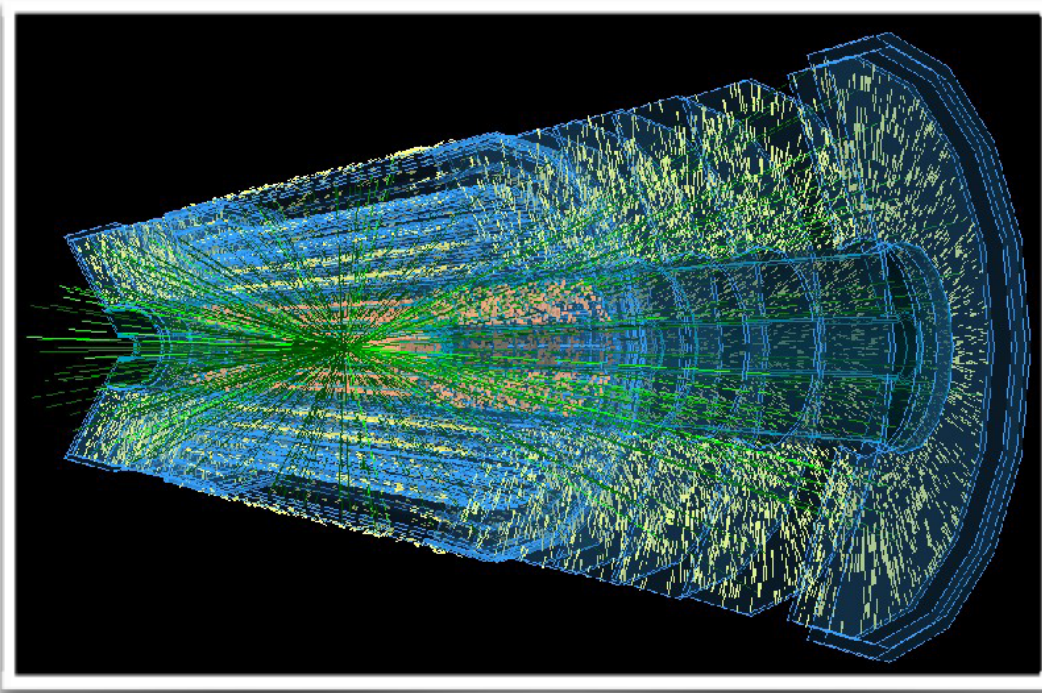
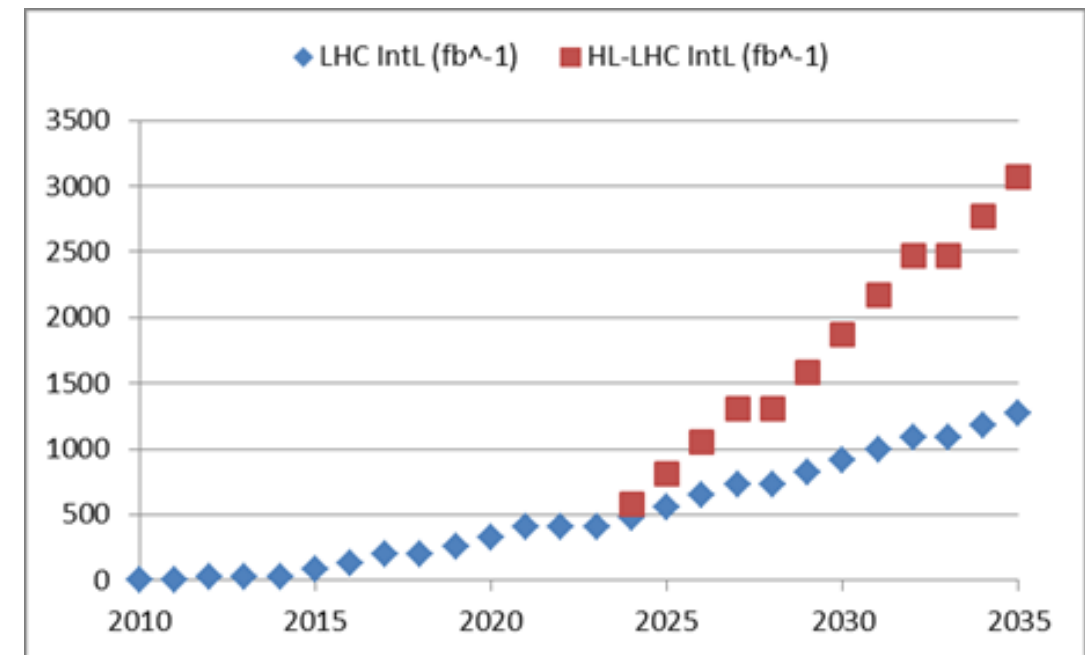


# The Challenge

Rende Steerenberg, CERN



X



## Event Complexity x Rate = Computing Challenge

- Reconstruction event complexity is naively  $\mu!$  (factorial)
- Rate increases
  - 40MHz LHCb Run3
  - 50kHz PbPb Alice Run3
  - 1kHz  $\rightarrow$  5-7.5kHz GPDs Run4 (hides the huge challenge of an efficient trigger with this level of pileup)

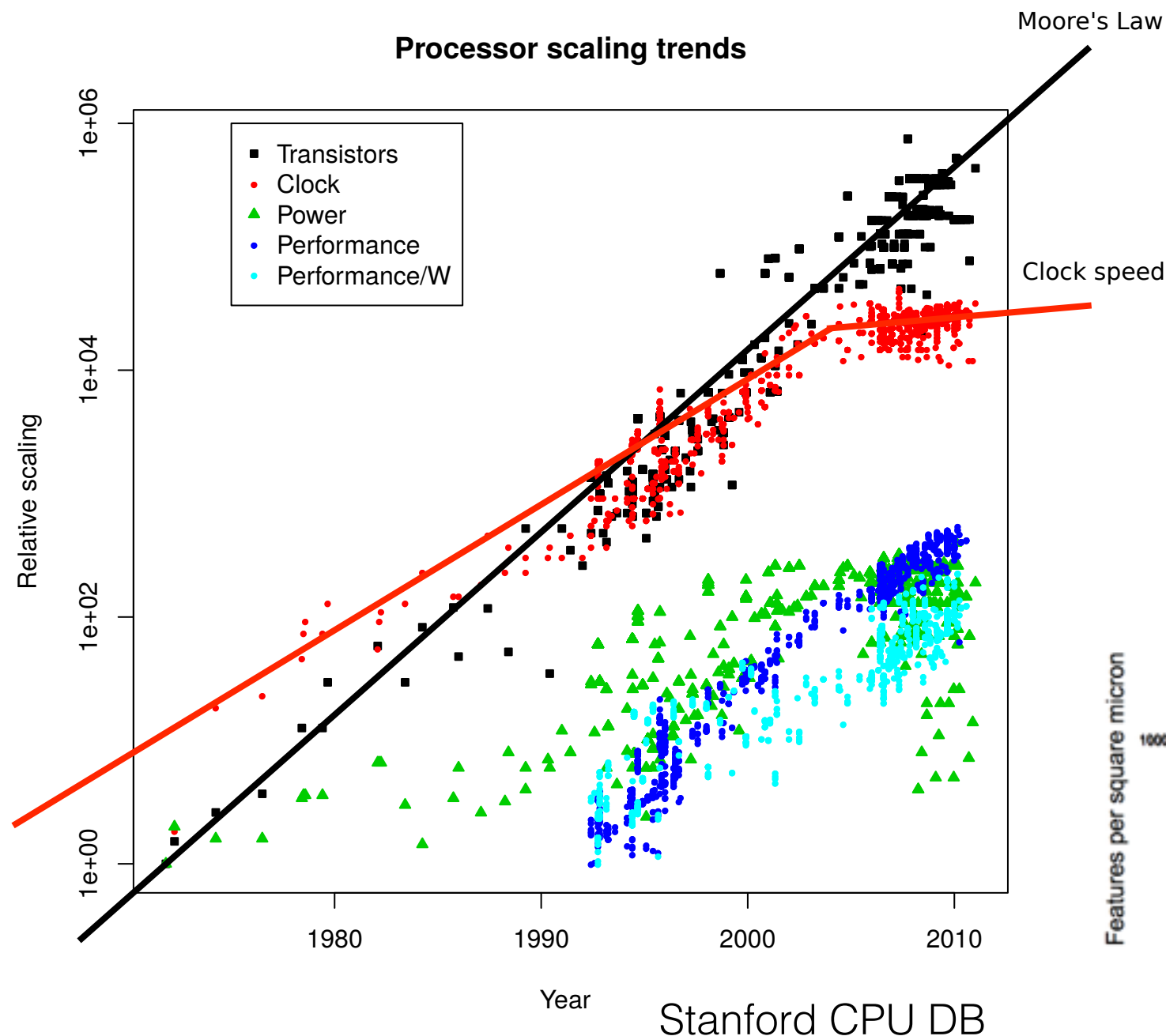
# Nuts and Bolts: Future Hardware Evolution



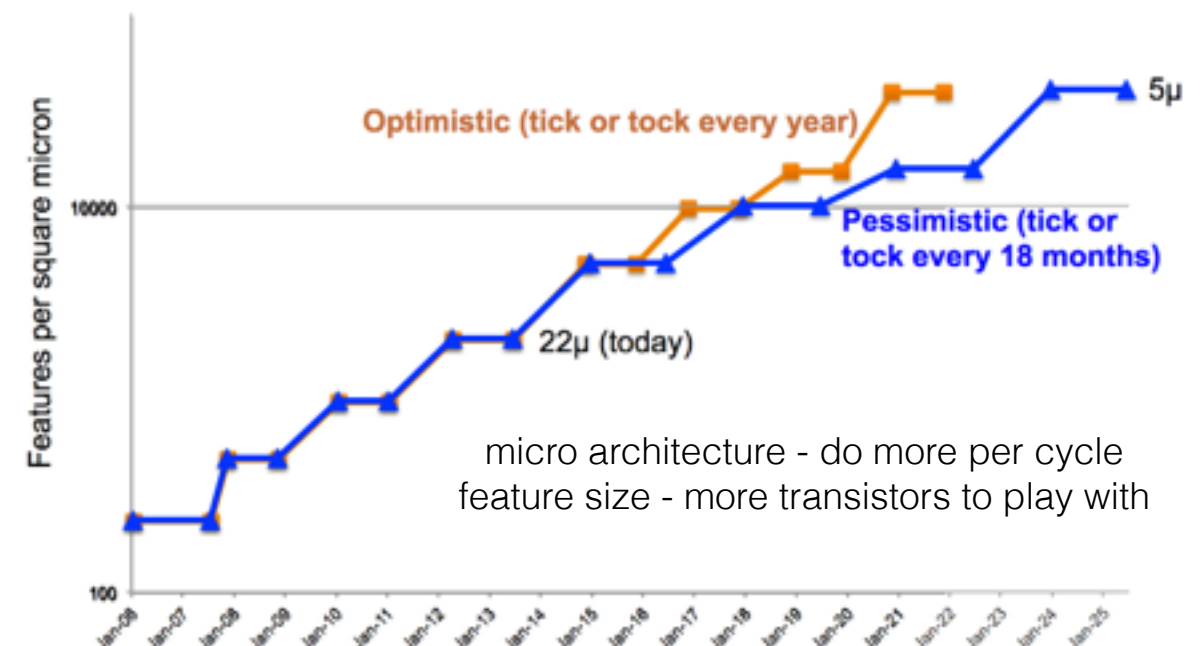
- From the high level trigger onwards we firmly use Commodity Off The Shelf hardware (COTS)
  - We can't use anything different — we are too small
- Here we have enjoyed huge and steady advances over the years (x1000 in CPUs over 20 years)
  - Advances that have enabled the computing that we now have in WLCG
- However, many technologies are reaching a state of maturity that means that future gains may not be so rapid
- What's our best sense about how things will look by 2025?
- And how does this impact on how we should evolve software and computing in the next 10 years?



# Processor Evolution

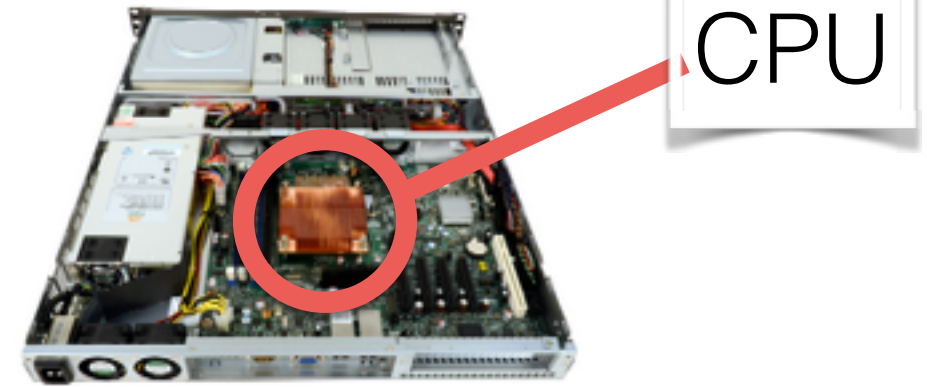


- Moore's law for processors is not dead
  - But more uncertainty than the past
  - Will not hit brick wall
  - But feature density doubling time uncertain: 24 or 30 or 36 months?
  - Over 10 years this makes a significant difference:
    - x32 vs x16 vs x10
- Clock speeds stalled ~2005 and no breakthroughs expected here
  - Baseline serial job performance stalled
- Main drivers for architecture:
  - Low power consumer devices
  - Data centre backends

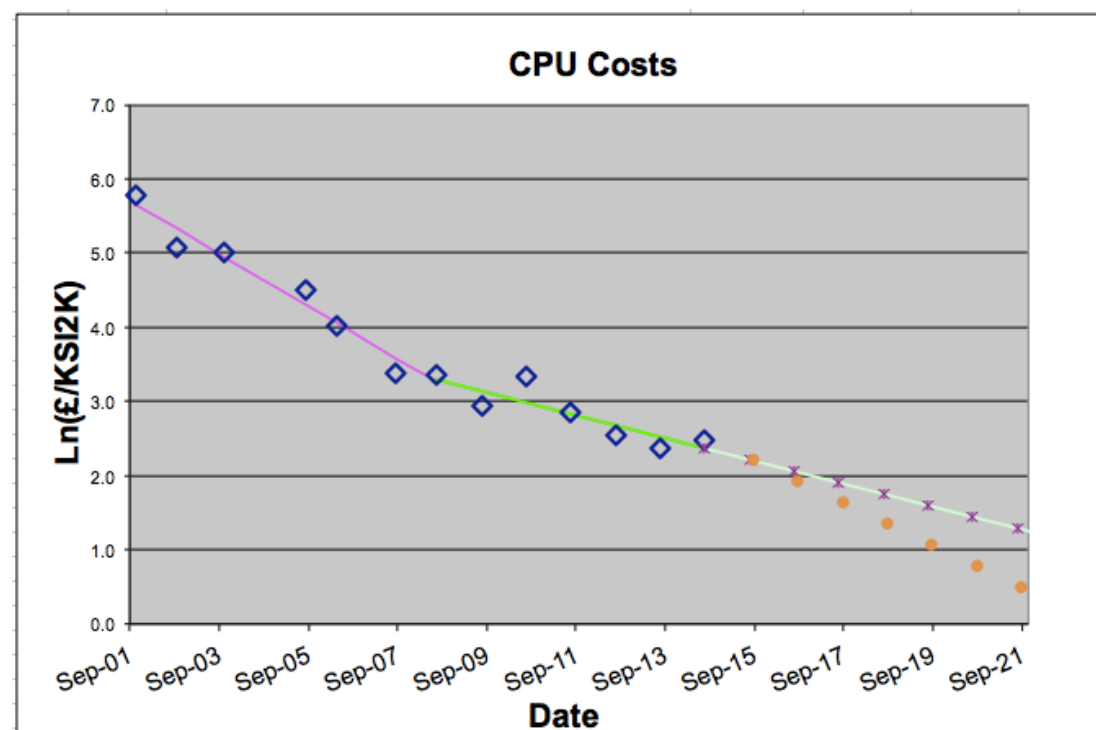
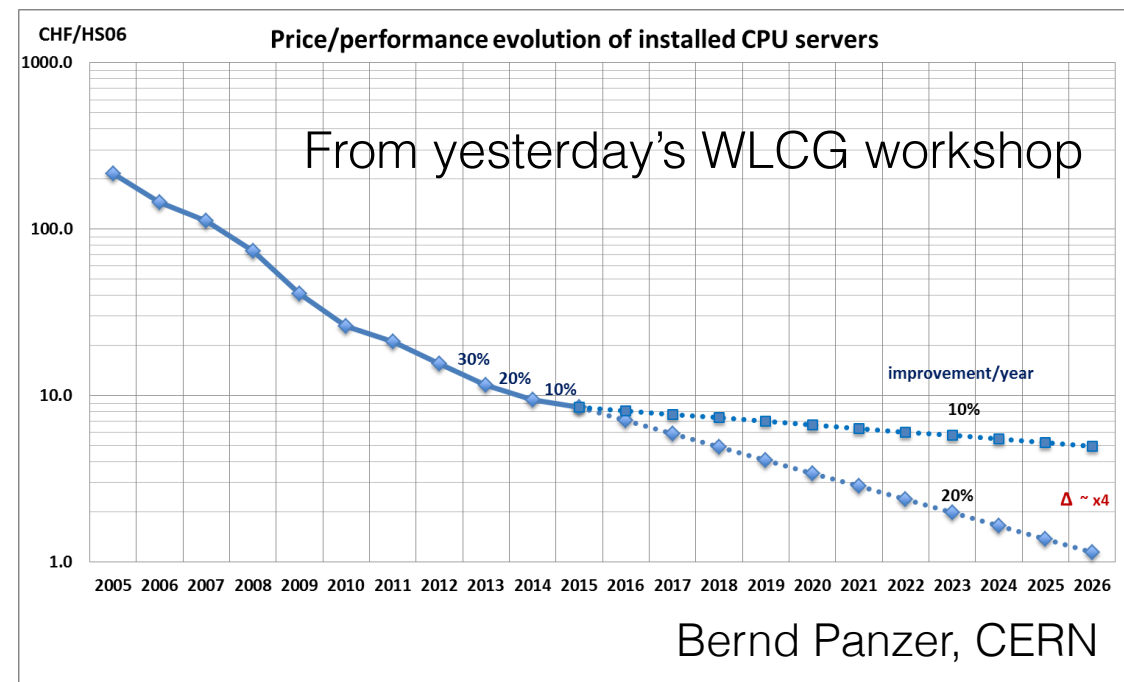




# CPU Servers



- Areal density of features on silicon does not translate to price/performance in a data centre
- Actual improvements are more like 20%/year
- Also here  $\pm 5\%$  will make differences of  $\pm 60\%$  over 10 years
- Perhaps key question on HL-LHC timescale is will this still be the architecture we base our processing on?

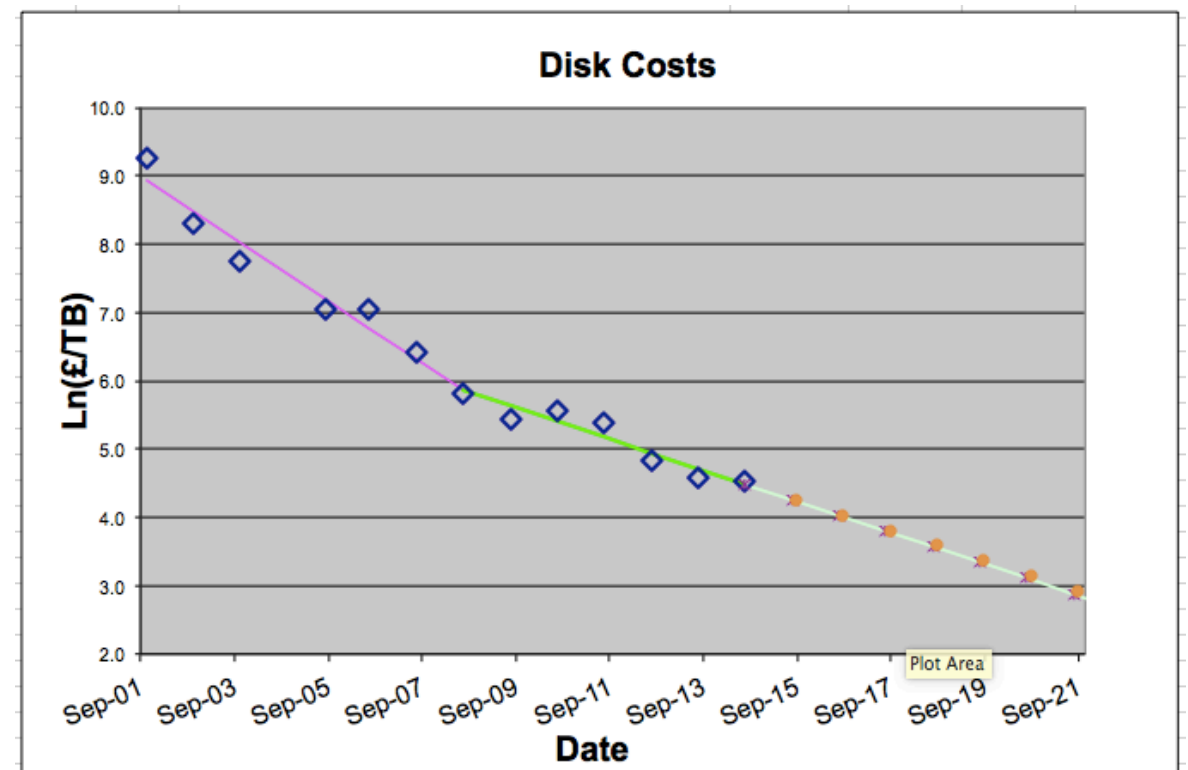
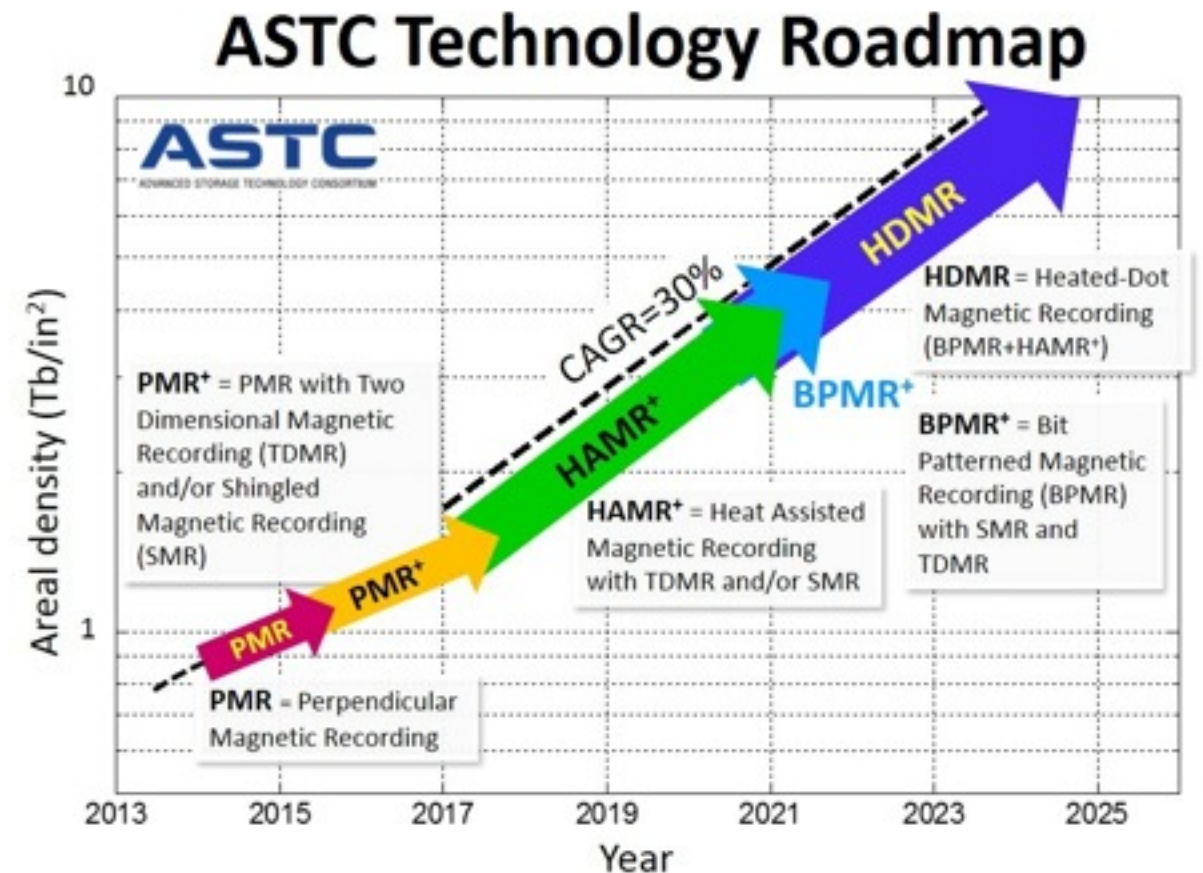


RAL Tier1  
CPU Costs  
with  
projection

David Britton, Andrew  
Sansum, GridPP

# Disk Evolution

- First drives with shingled magnetic recording available
  - 8TB capacity at reasonable price (c.f. helium filled drives)
  - Affects write performance
- Many later technology options might take us to 100TB drives by HL-LHC
  - No knockout blows
- Capacity going up but IOPS/\$ pretty much stalled
  - Definite problem for IO bound workflows



RAL Tier1 Disk Costs  
with projection

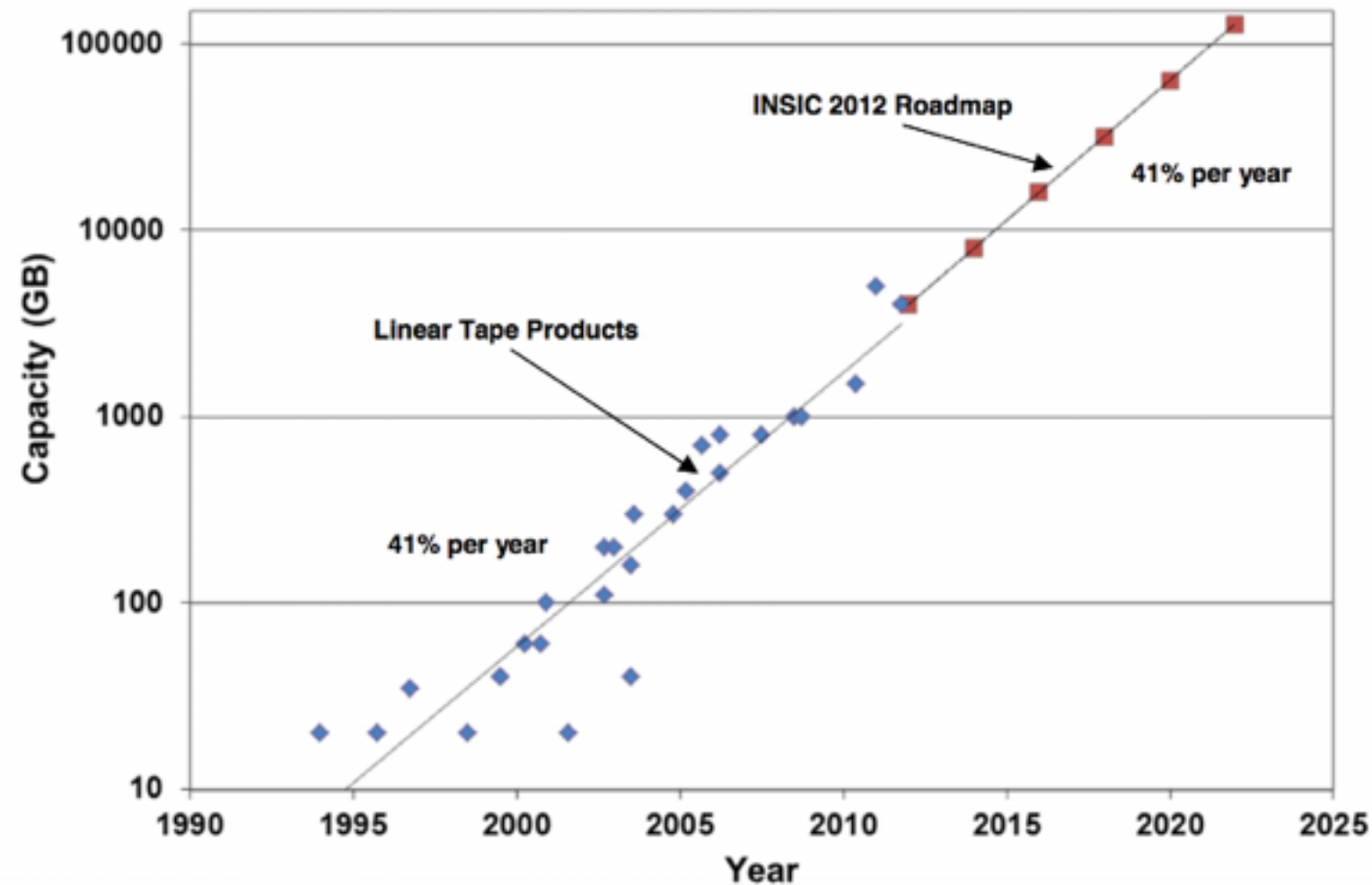
David Britton, Andrew  
Sansum, GridPP

# Tape Evolution

*Reports of my death have been greatly exaggerated*

- Feature densities way below that of HDD
  - Less problems with technology evolution
- Future of tape in the 'big data' era seems fairly secure
  - e.g., Amazon Glacier

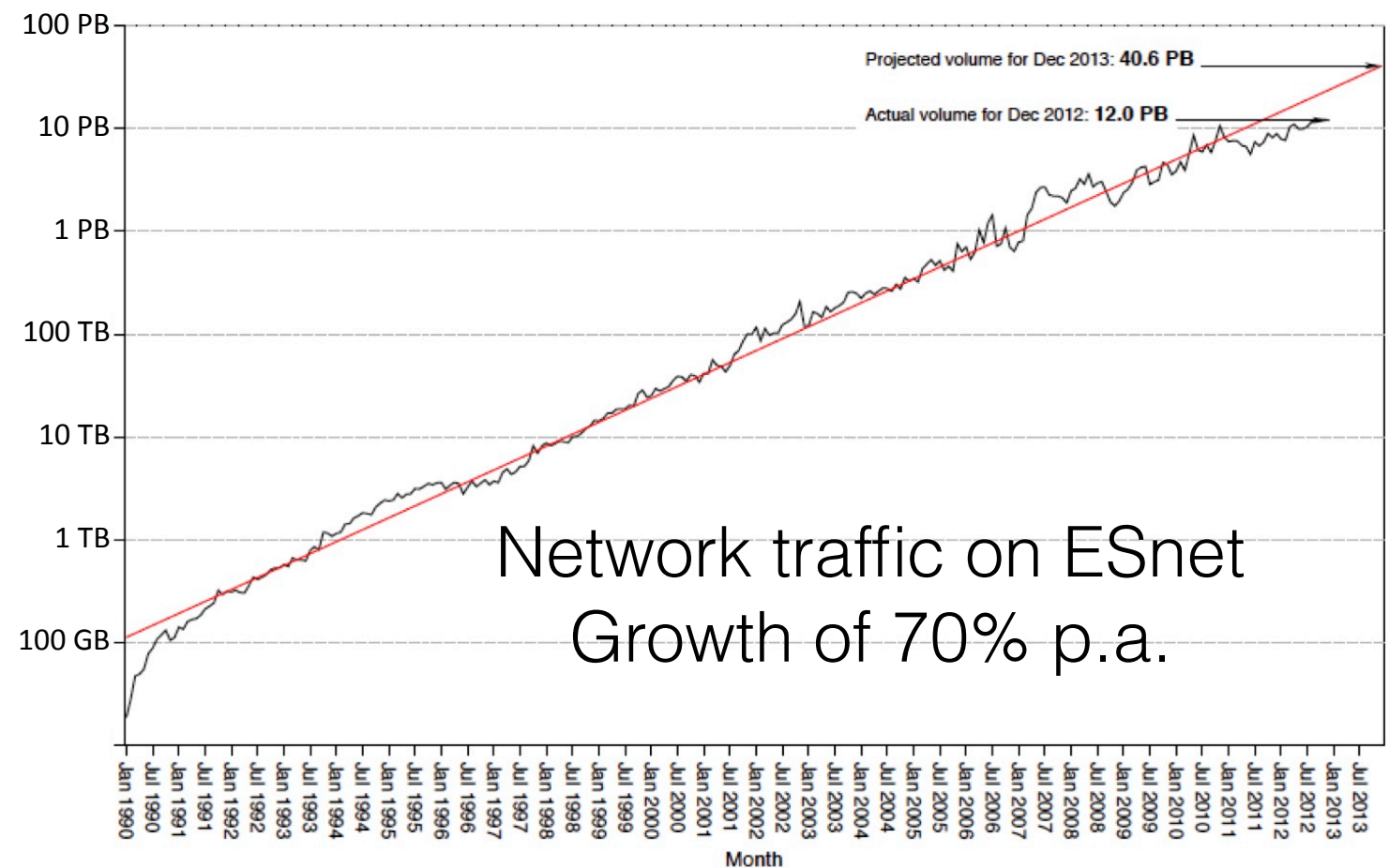
Google Nearline provides an interesting challenge to this model, but current pricing may be below cost to acquire market share



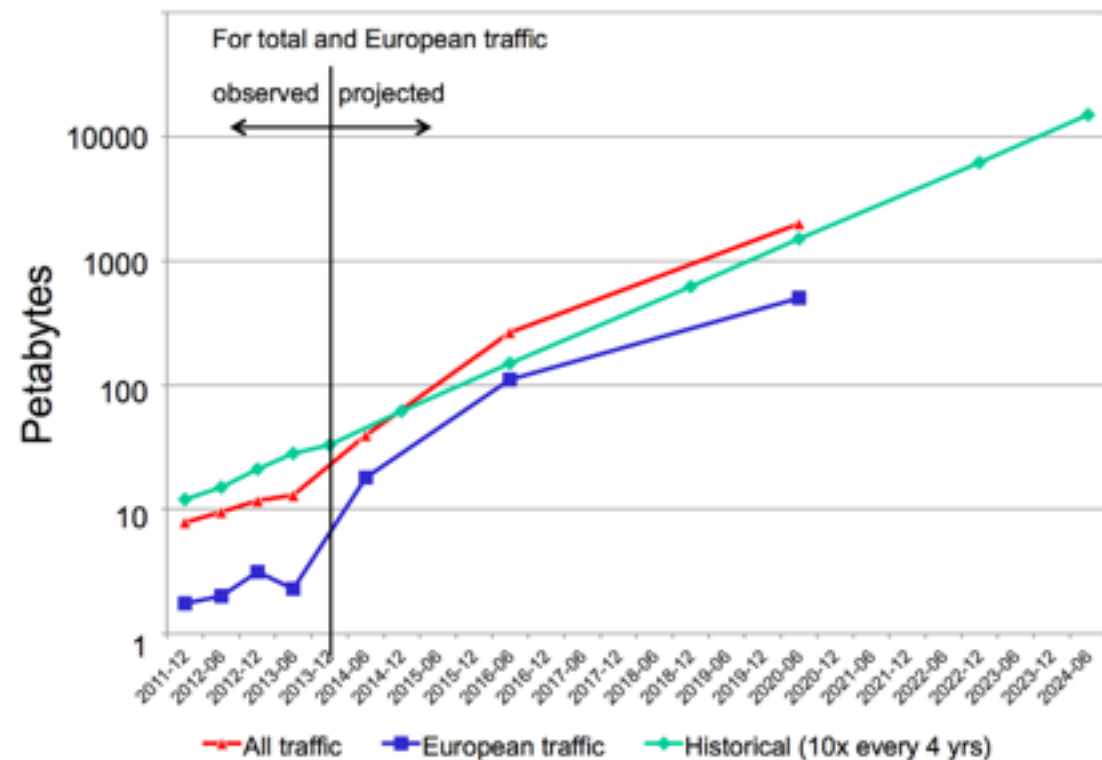
Information Storage Industry Consortium Roadmap

# Networks

- Network usage and capacity growing exponentially
- No large technology hurdles foreseen
- Demands likely to remain high and growing
  - Fuels capacity that HEP can take advantage of
- On demand network routing and protocols interesting, but yet to be really seen if HEP can take advantage of these



Total volume in bytes of traffic handled by ESnet per month



# Technology: Baseline Boundary Conditions in 2025

Technology	Growth in 10 years
CPU Servers	x4 - 14
Disk Capacity	x4 - 10
Tape Capacity	x10 - 30
Network Capacity	x30 - 200

- Per unit cost
- Assumes no truly radical change in what we do and no massively disruptive technological advances



# Back to the heart of the beast: What is all the silicon doing?



- Increasing silicon feature density (AKA Moore's Law) does not allow for higher clock speeds anymore
  - Shrinking size played off against CPU frequency in power consumption in the past
- Now silicon area used for:
  - Improved micro architecture (gains are diminishing)
  - Higher numbers of cores (linear in time)
  - Larger caches (decreasing payback)
  - Wide vector registers (reached the limit already...?)
- Most efficient use of a CPU requires exploiting all of these features

# Memory Hierarchy and Bandwidth

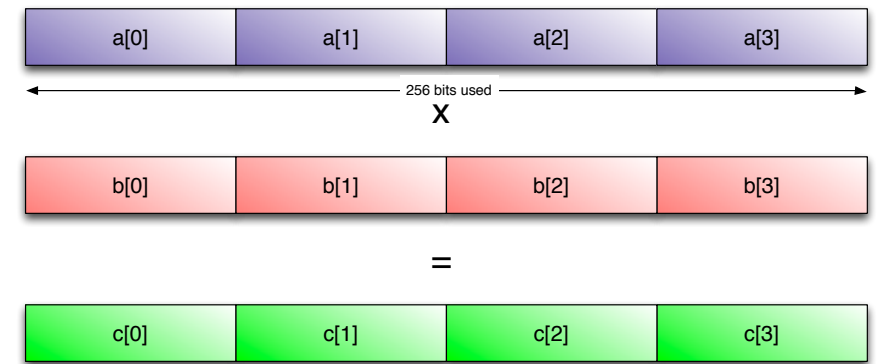
DRAM		Multi-GB Main Memory	200 cycles
Level 3 Cache		8MB Cache (shared)	40-70 cycles
Level 2 Cache		1MB Cache	10 cycles
Level 1 Cache		32kB (Data and Instruction)	4 cycles
Core	Core	... 32 x 64bit registers	1 cycle

Typical Cache Hierarchy on current Intel Xeon processors

- Costs of waiting for data on modern CPUs is very high
- Need to pay attention to alignment issues as well for SIMD
- More sophisticated architectures can introduce further issues
  - Non-uniform memory access
  - Cache coherence costs (of *having* or of *not having*)
- Energy costs of memory access are starting to be greater than those of the CPU
  - Poses a fundamental problem for exascale computing

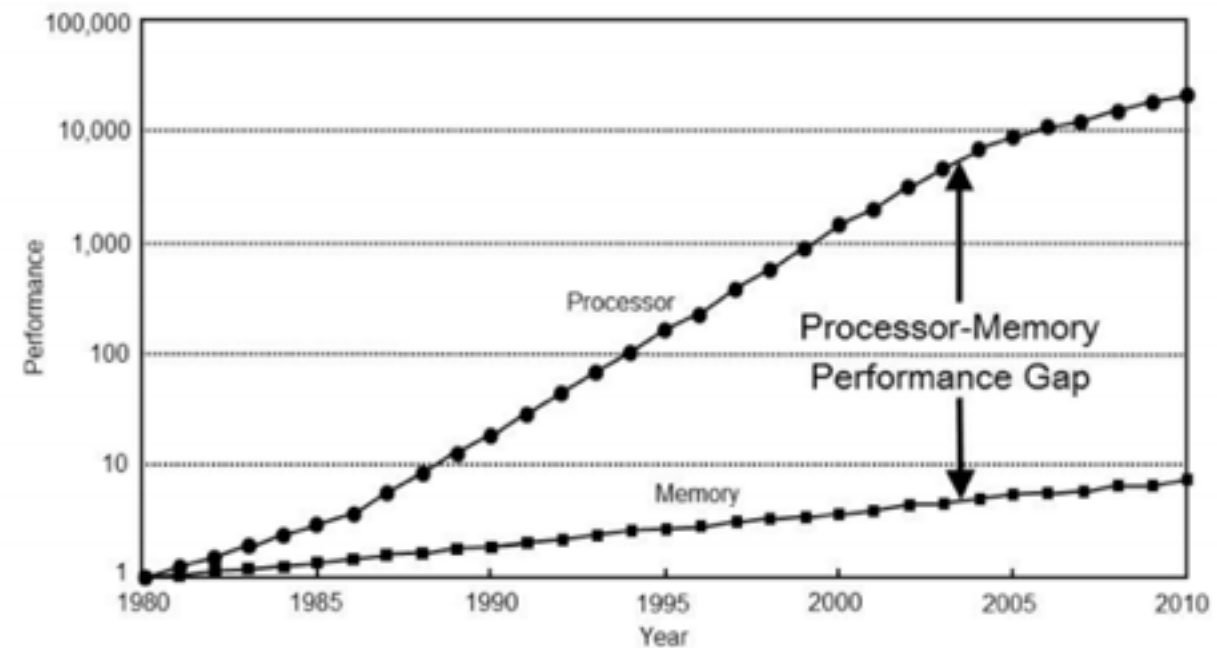
# SIMD and Dark Silicon

AVX 256 Bit Registers:  $c[] = a[] \times b[]$



- Efficient use of SIMD can bring benefits:
  - SIMD fused multiply add **saves 16** cycles over serial operations
- However, now need to load 16 floats and store 8
  - In the worst case this could **cost few 100** cycles in data transfer times
- Similar issues apply to serialisation and de-serialisation of data for accelerators
  - Data needs accessed, rewritten, shipped, recovered and then rewritten again
  - Data transport costs often outweigh processing time
- Gains of advanced CPU features, like vectorisation, are dwarfed by the loss of efficiency if memory layout is poor
- Patchy use of vectorisation can harm an application's throughput due to the thermal cost of enabling the wide vectors
  - Increased heat load from more lit silicon leads to reduced clock speeds
  - This phenomenon of *dark silicon* seems likely to grow, alongside core specialisation (e.g. ARM's big.LITTLE)

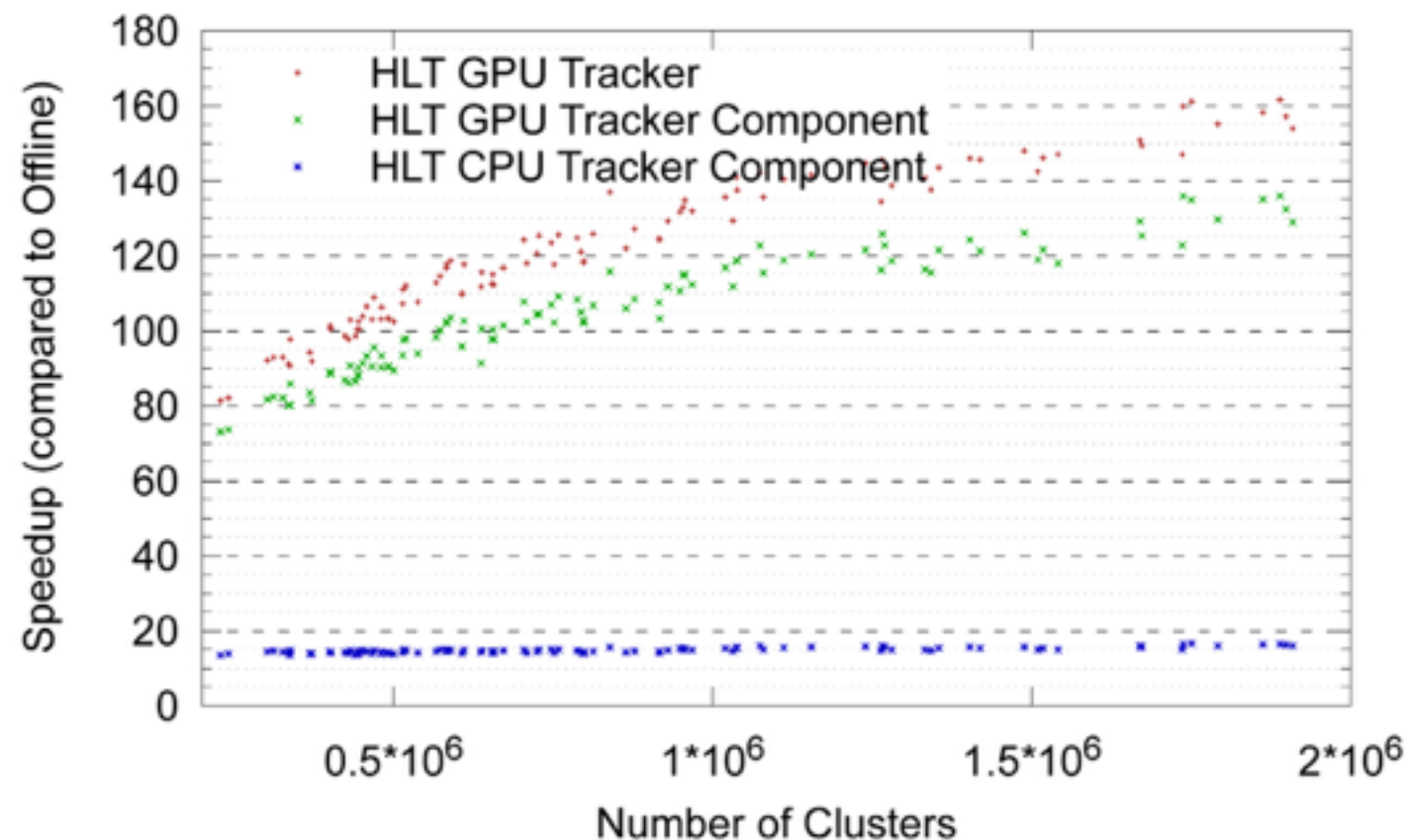
# Data Oriented Design



- Trend in high throughput computing to re-orient designs around data rather than 'objects'
- The point of the program is to transform data from one form into another
  - If you don't understand the data, you don't understand the problem
- Have to base reasoning about costs on understanding the data and understanding the hardware
  - There is no abstract solution that works with all data on all hardware
  - Have to maximise the use of all data that is fetched from main memory (think cache lines!)
    - Do not pollute caches with data you know will not be used again
- This trend counters the early HEP C++ data model of multiple/deep inheritance
- Internal data model can be drastically different from the global one
  - e.g., Run2 improvements in the ATLAS ID layout

# Change the Architecture? (or how easy is it to utilise GPGPUs?)

- ALICE have actually managed to do this (needed for Run2)
  - Code is not generally portable between architectures (common tools can lack performance)
- ALICE scheme involves a build time declaration that switches in the correct code
  - Keeps code as common as possible, but switches between CPU and GPU as needed
  - Very good for targeting a compute cluster known at compile time



Tracking clusterisation  
performance using GPU vs.  
multi-core CPU

ALICE Collaboration



# For everyone else?



- More generally ATLAS have a demonstrator project (APE) that will utilise runtime switching between CPU and accelerator versions of algorithms
- However, the prospects for widespread conscious adoption of current GPGPUs on the grid seem very weak
  - Not enough code within LHC experiments has been (or even can be) ported to GPUs effectively
  - Cost effectiveness of a solution needs to be integrated across the lifetime of the hardware (total cost of ownership) and across all users of the facility
  - Validation needs to be folded in to the experiments software maintenance costs!
- Use of GPGPUs in trigger farms is a more serious proposition (e.g., ALICE Run2) as
  - Problem domain is more focused
  - Direct control of farm by the experiment
  - Physics criticality of event selection justifies a possible lower duty cycle for the hardware

# Future Processor Directions: General vs Specialist Trends

- The more specialist a piece of hardware is
  - The more precisely it solves problems for which it is suited (lower energy per unit of computation), but the fewer problems it can actually solve (or solve easily)
- Hardware vendors want to sell as many units as possible
  - Push to expand the problem space in which a particular piece of hardware can be used
    - Can be done in hardware itself: make the device more general purpose
    - Or by providing enhanced software support that translates a large class of problems into machine code that runs on a piece of hardware
- We see both trends in action with co-processors:
  - Successive generations of Nvidia's CUDA are introducing more C++ functionality and shared memory space
  - Next generations of Intel Phi offer superior micro-architecture with out of order execution more suited to branchy code; available as main board processor
- So no surprise to see convergent evolution at work and next generations of coprocessors might well be easier of HEP to use
  - Agile code will evolve best, given this unclear future

# Future Processor Directions: General vs Specialist Trends

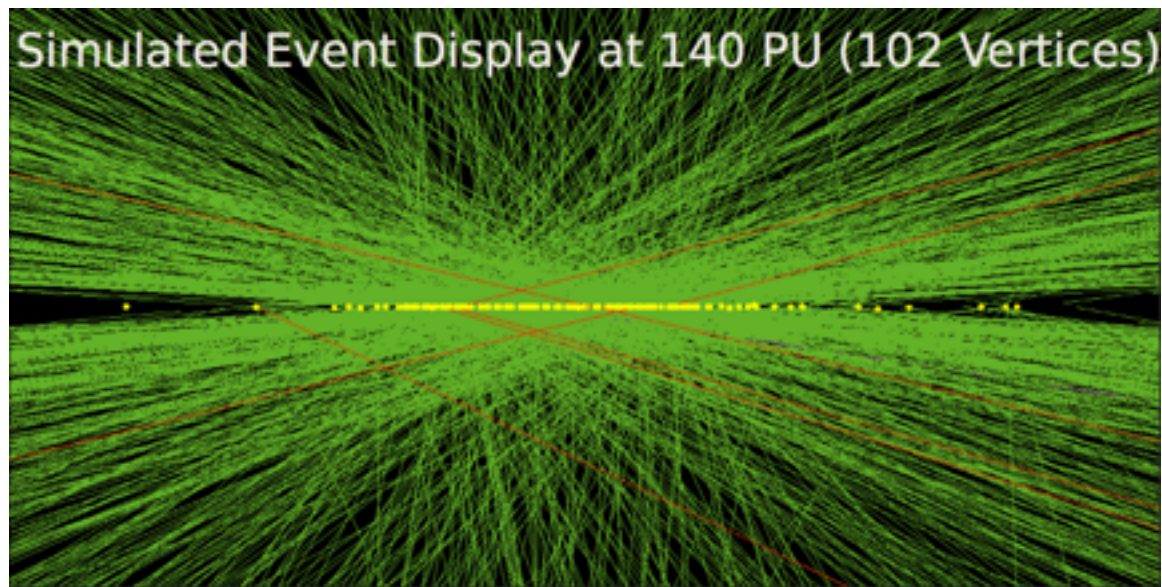
- At the same time applications which can be considered ‘bulk-specialist’ are cost effective to tackle with a custom architecture (lower power is the ‘killer feature’)
  - FPGAs could be used to run cloud apps such as voice recognition and search (with OpenCL as a programming interface)
  - See Intel and Altera’s collaboration on HARP
    - 12-core Intel microprocessor with an Altera Stratix V FPGA module
  - Specifically targeting the development of tools to broaden the base of suitable applications
- N.B. we do not have to migrate the entire workload — just enough to fill the available hardware resource (evgen, sim)
  - Note that attached to HPCs we often have exotic hardware that might sit idle

Summary: processor futures look very heterogenous and very interesting - lots of opportunities for smart work.

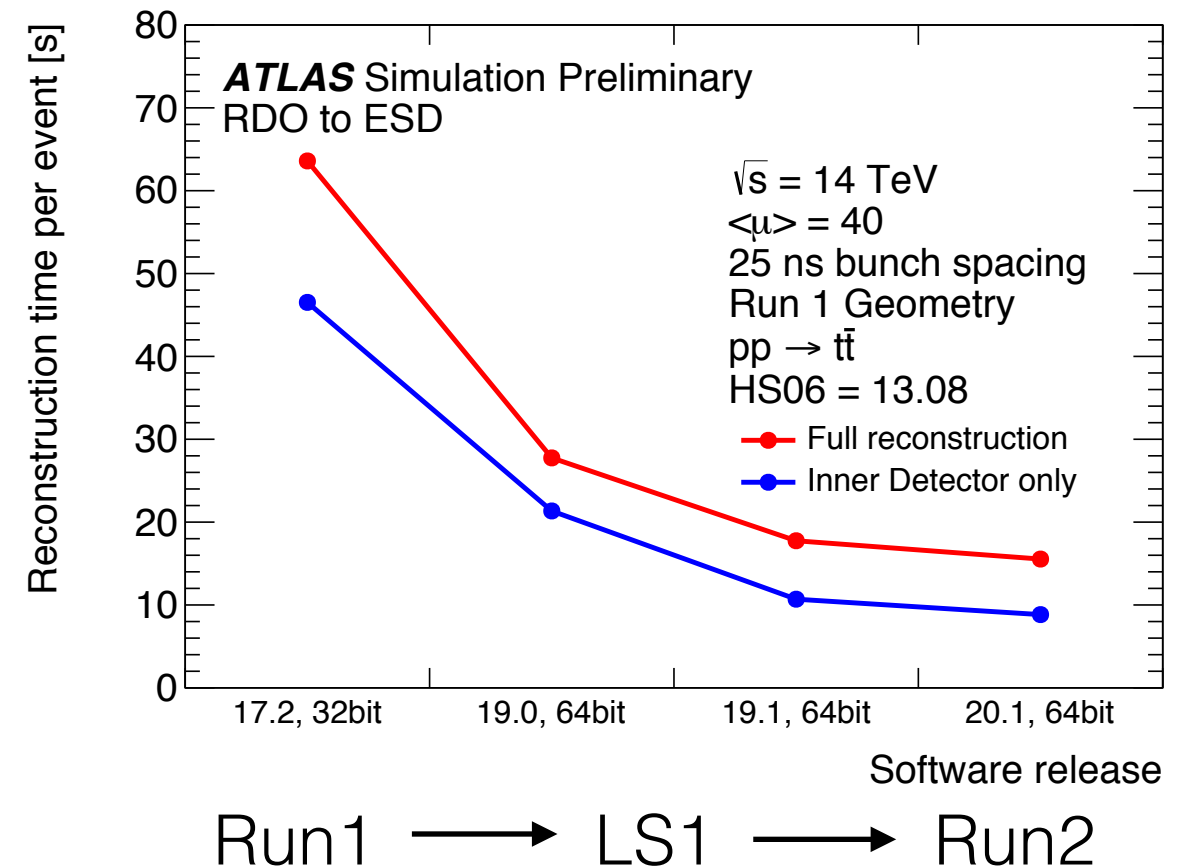
*(Did someone mention validation of the results and physics performance...?)*



# Application Code: Tracking

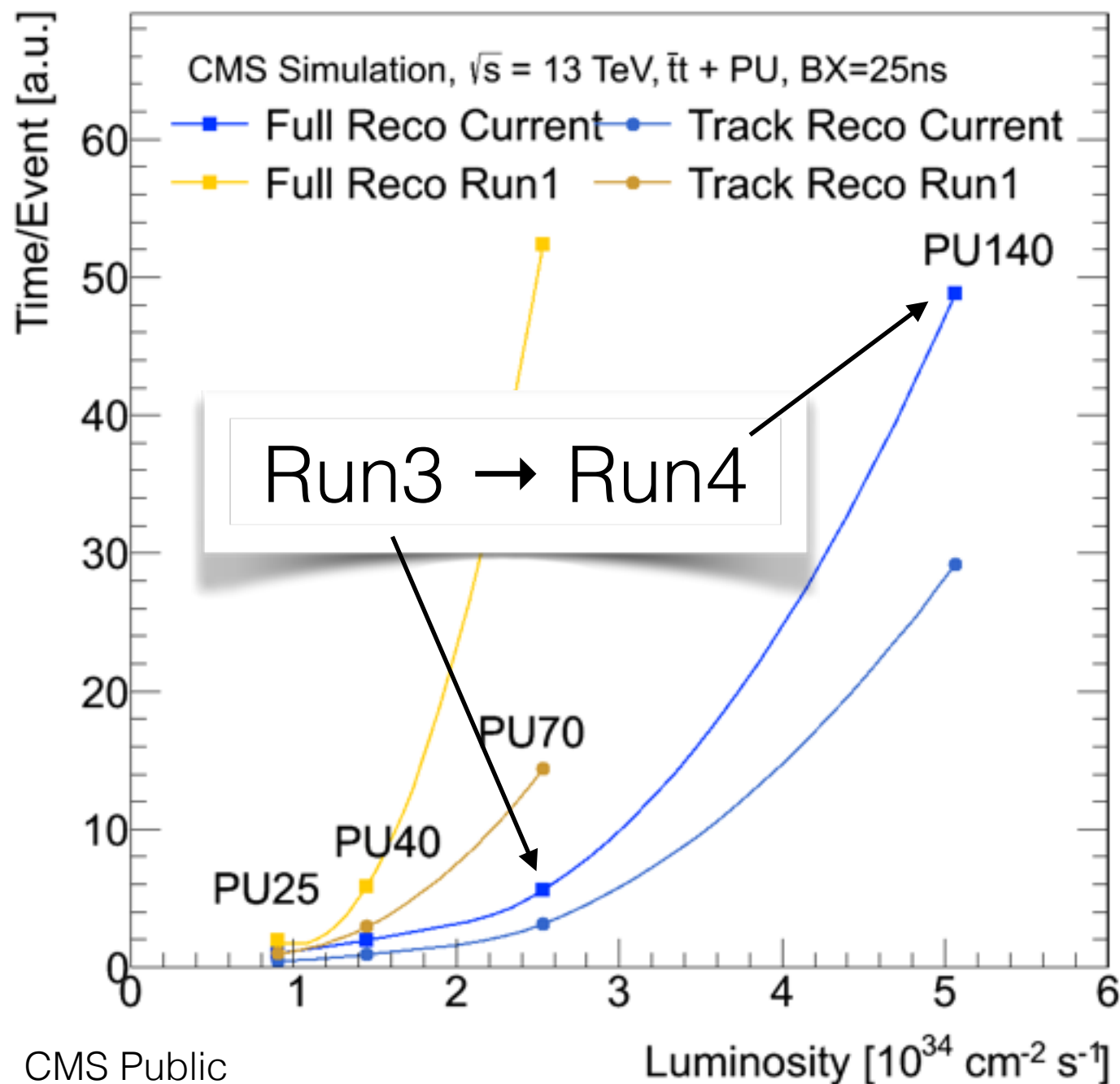


CMS Simulation  
Erica Brondolin, HEPHY



- High luminosity means high pileup
  - 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, but we need to go much further

# Tracking to HL-LHC



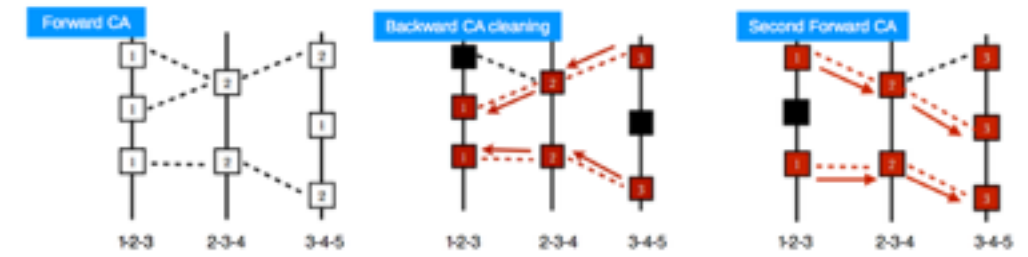
CMS Public

- CMS similar improvement to Run2
- Extending this strategy to Run3 looks ok
- To Run4 CPU time goes up by x10
- *ATLAS and CMS need to investigate new approaches by 2025 for HL-LHC conditions*

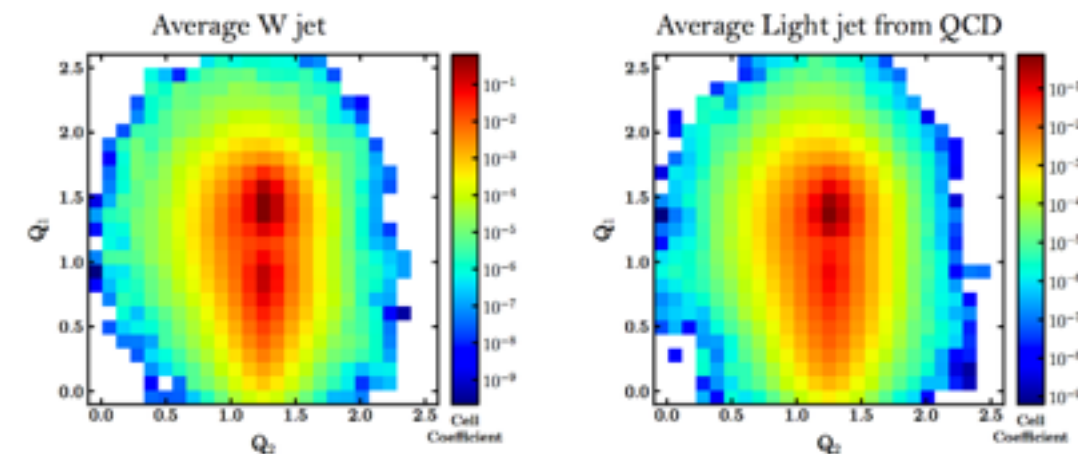


# Current and Future Strategies

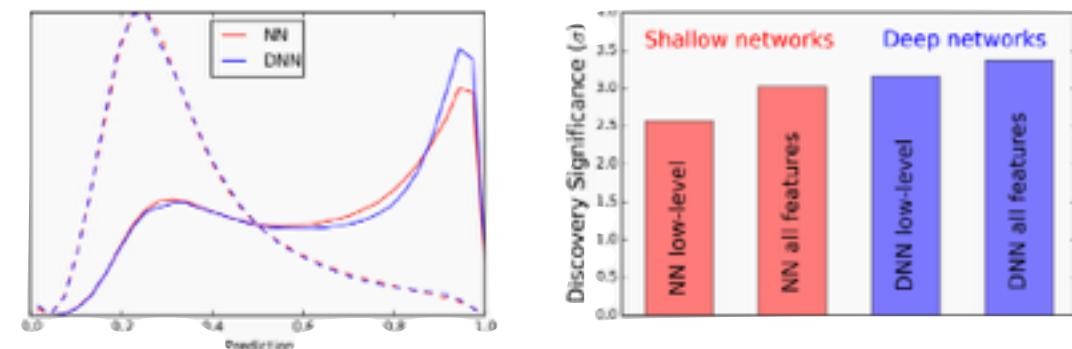
- Current strategies are based on early rejection
  - Serial chain of algorithms designed to beat down combinatorics
- However, serial dependencies between algorithms limit concurrency opportunities
  - Can try throwing more events at the problem
    - More events consume more memory and we may run out of memory before filling all cores
  - Alternatively, seek algorithms that cost more CPU, but allow for a higher number of cores to be brought to bear on the problem
    - More CPU cycles, but improve throughput
- Clearly a lot more studies are needed, but physics performance is paramount
  - Fast algorithms with poor results is not what we want
- Deep learning algorithms could take us beyond current boosted decision trees and neural networks



Iterative track finders are inherently more parallelisable, e.g., cellular automata (Erica Brondolin, HEPHY)



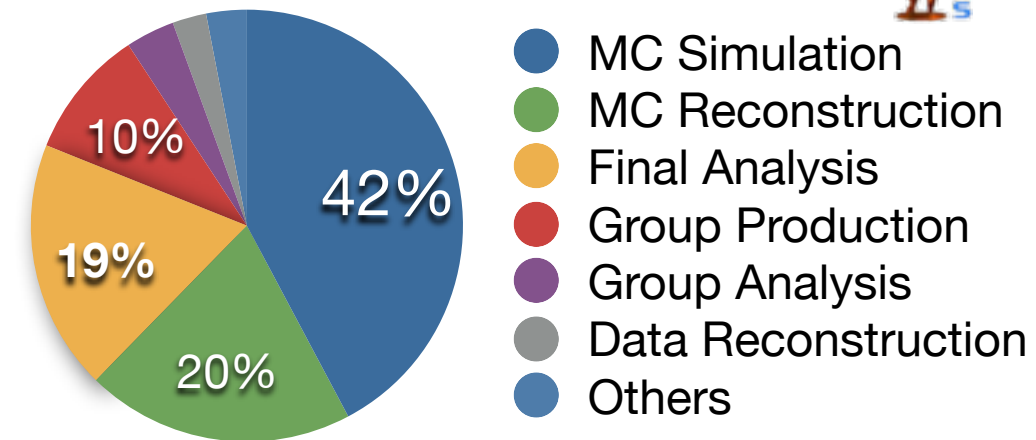
W jets vs. light jets characterised via linear discriminant analysis (Michael Kagan, SLAC)



Deep Neural Network improvements in  $H \rightarrow \tau^+ \tau^-$  +significance (Peter Sadowski, UC Irvine)

# Simulation

## GRID CPU Consumption

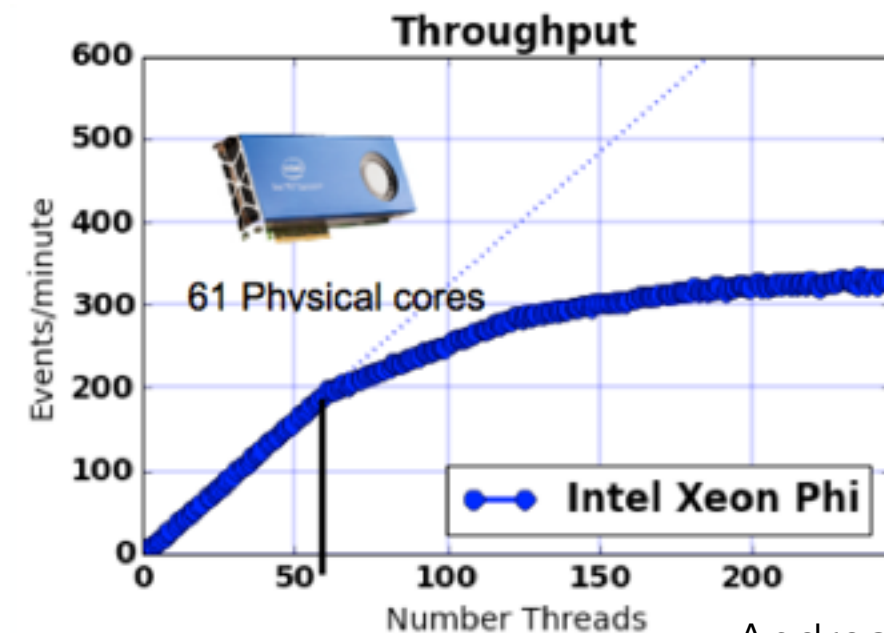


ATLAS, Run1

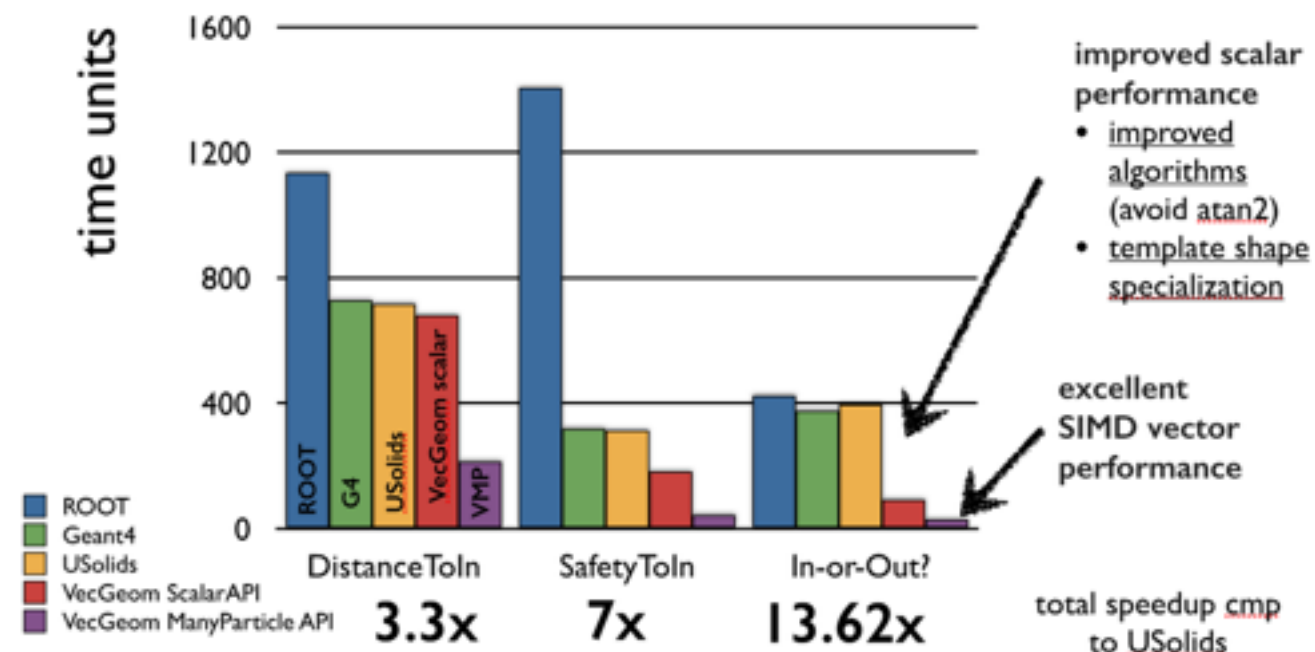
- Simulation is a very large consumer of offline computing resources
  - GEANT 4 full simulation of general purpose detectors is expensive: up to ~1000 seconds per event
    - Not to mention difficulties of simulating track triggers (c.f. ATLAS FTK simulation)
  - Simulation subject to the same general software issues as any other piece of software, with optimisation opportunities and continued development:
- Simulation does not suffer the same scaling issues with  $\mu$  seen in reconstruction
  - Signal events simulated separately
  - Geant4 simulation is underlying event complexity (energy into the calorimeters) plus detector complexity (geometry and complexity of physics interactions)
  - Pileup digitisation is naively linear (but i/o worries?)

# Geant Progress

- Geant4.10 split data into static and dynamic parts
  - Static heap shared, dynamic in thread local storage
  - Spawn individual events on each thread
  - Massive memory savings and thus impressive scaling
- GeantV vector prototypes use new VecGeom classes
  - Vectorise for specific targets using templates
  - i.e., plugins for different backends
  - Aim is to have performance gains with few interface changes for experiments
- Need to concentrate improvements in areas where experiments spend the most cycles
  - Real detectors spend a lot of time in messier volumes - harder to vectorise
  - Physics processes might go in parallel (as long as memory doesn't blow up)

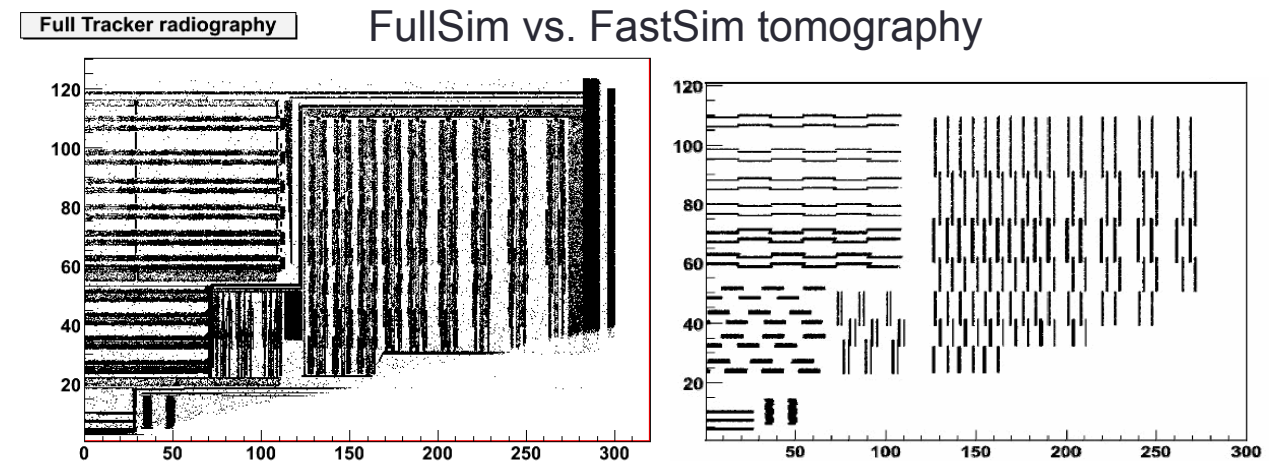


Andrea Dotti, SLAC

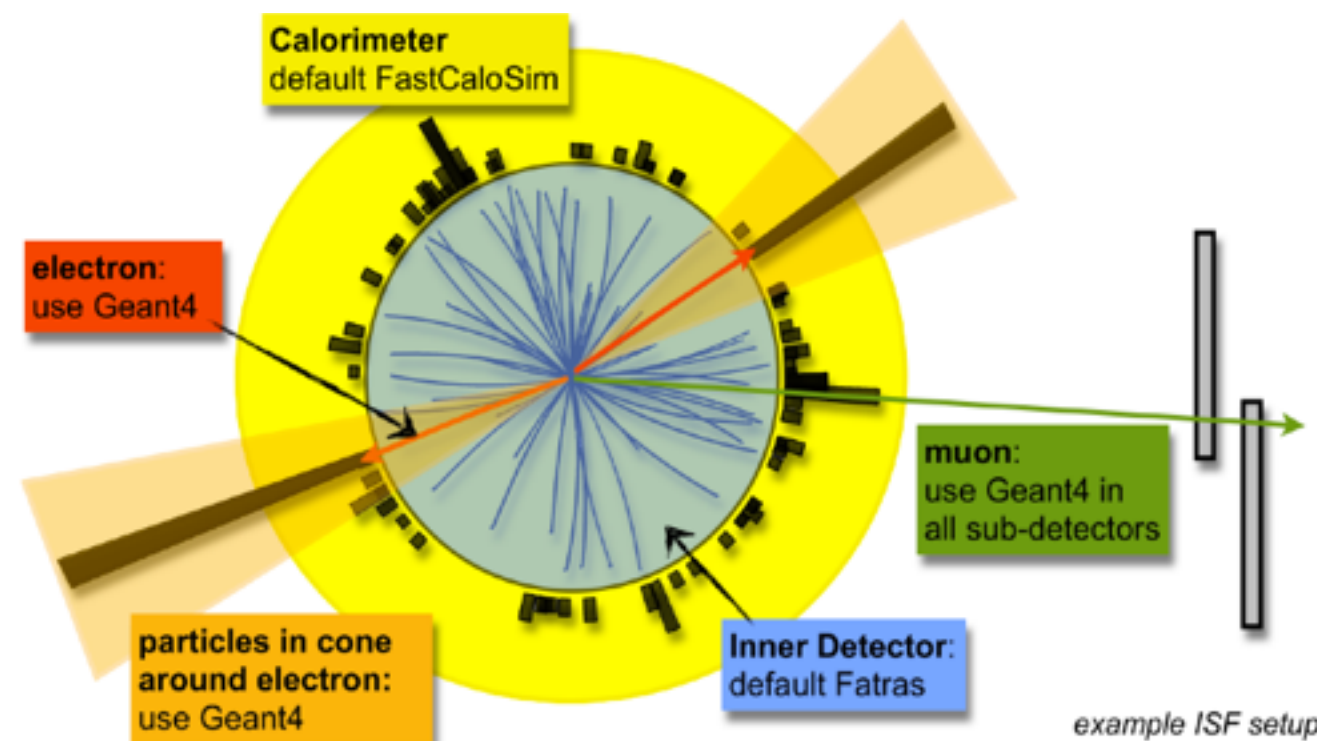


# Fast Simulation

- Need to take advantage of fast simulation where appropriate
  - Tradeoff accuracy for speed
    - Smearing
    - Frozen Showers (precomputed)
    - Parametric techniques
  - *Trade off people time for CPU time (which is hard as we are short of both)*
- ATLAS's Integrated Simulation Framework allows clever mixing of fast and full simulation within the same event
  - Keep high precision for some particles and regions
  - Use fast simulation in areas that are not so important
  - x100 speed ups possible, with much better results than normal fast simulation
- LHCb have a similar scheme
- Feedback ATLAS ISF experience into GEANT4 (e.g. FCC simulations)

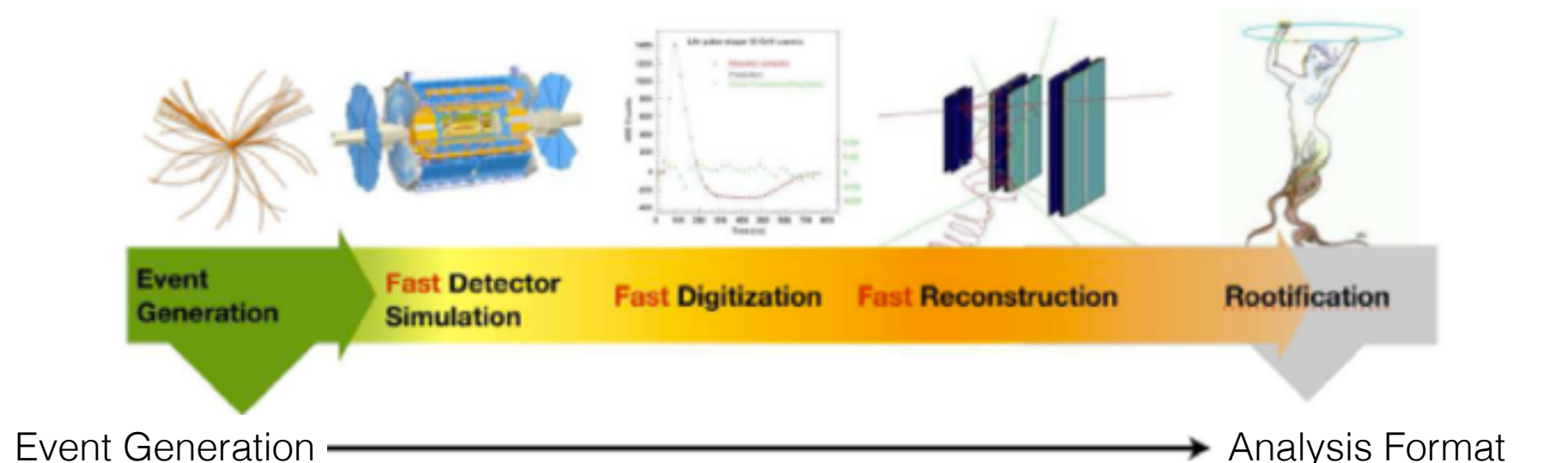


CMS FastSim Geometry



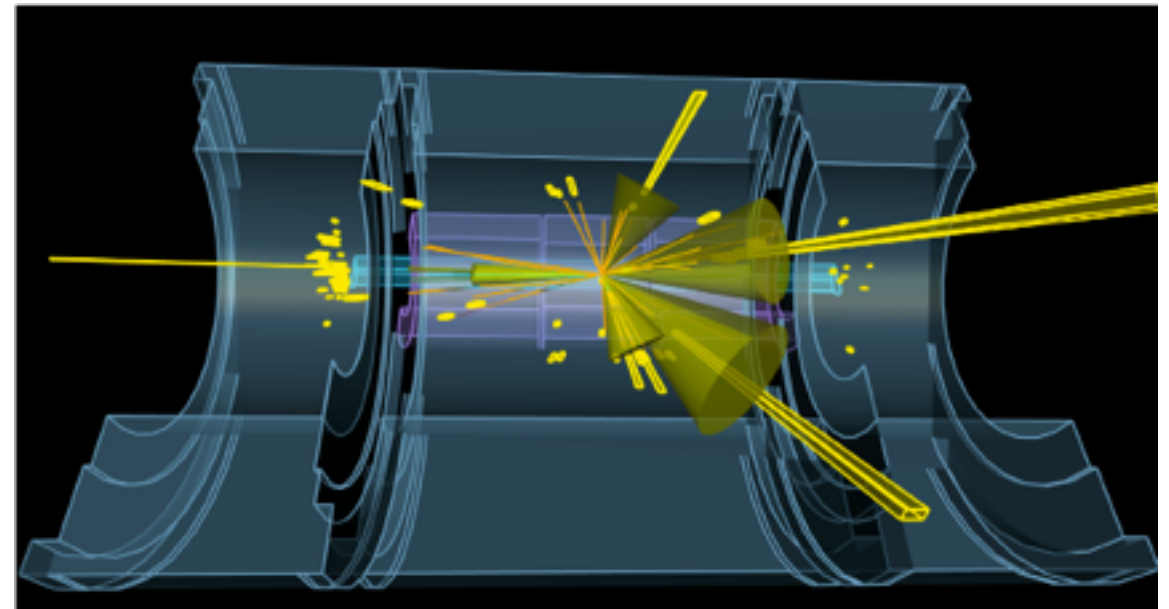
# Fast Everything!

- As simulation speeds up other steps in the MC chain become bottlenecks
  - Digitisation
  - Reconstruction
- So there is a need to support other faster techniques to improve the MC chain from event generation to analysis outputs
  - e.g. fast tracking in MC using truth tracks
- Aside: note that super quick means we can fill all of our storage up far more quickly if we are not careful
  - Produce final skimmed, thinned analysis outputs in one job





# Event Generation



Alpgen+Pythia8 Z+5jet  
event produced on Mira

Taylor Childers, Tom LeCompte,  
Tom Uram, Argonne

- Historically a fairly low consumer of overall grid cycles
- But this is rising, especially as precision physics requires lower theoretical errors
  - NNNLO generators
  - Madgraph-like matrix element calculations for SUSY
  - Pre-filtering generator outputs is leading to a lot of inefficiency (10B pythia events with 5k accepted in ATLAS evgen)
- Many optimisation opportunities and code is often is very parallelisable
- Good candidate also for more unusual architectures and HPC clusters
  - Alpgen ported to PowerPC to run on Mira at Argonne: 260,000 parallel threads producing 30M W+5jet events per hour

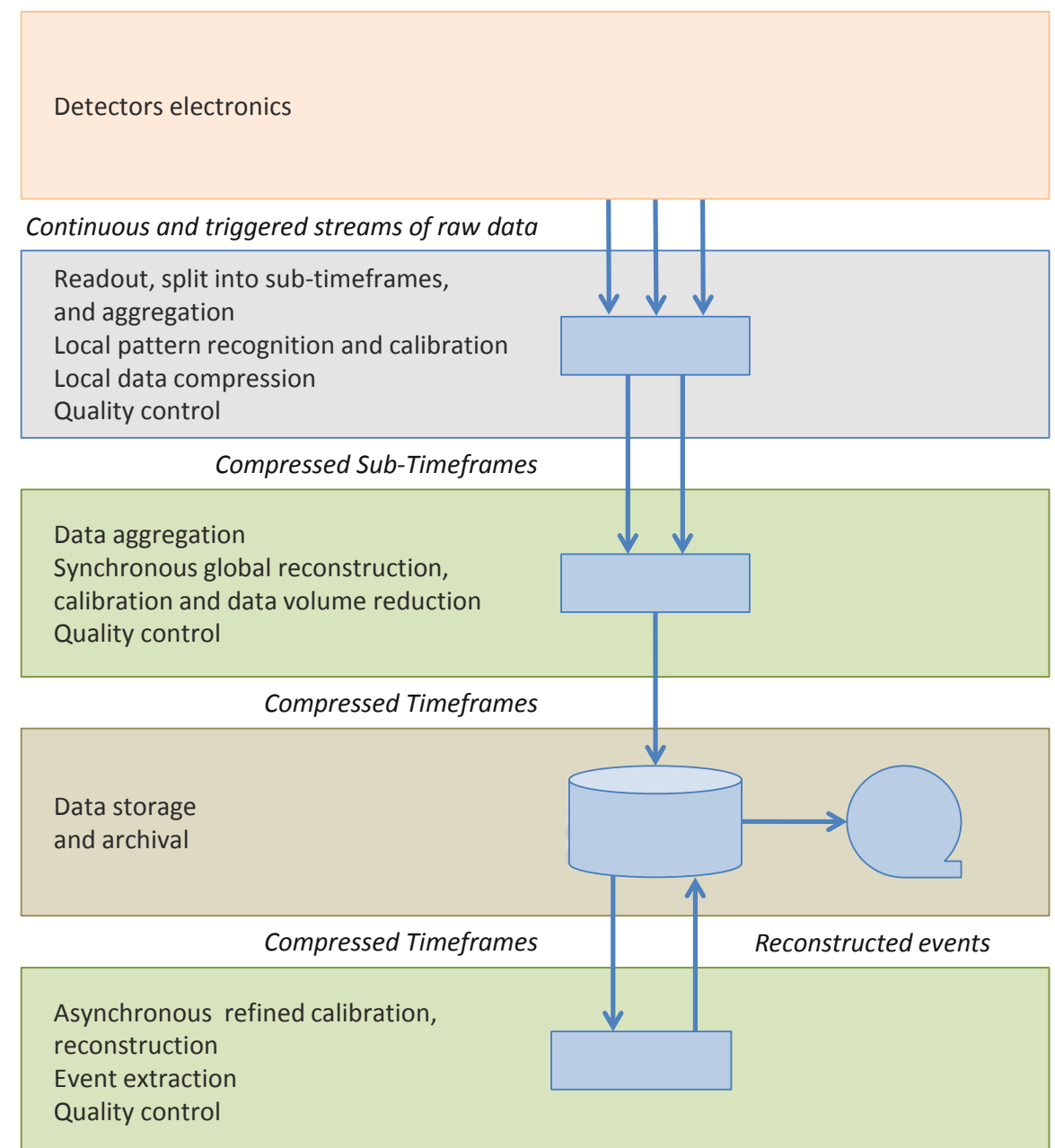
# Overall Server costs and a Memory Wall

- Continued march of number of transistors leading to increasingly multi/many cores
- Gap between CPU cores and *affordable* memory is increasing
  - Current Xeon grid servers have 2-4GB of memory per core
    - WLCG Requirement is only 2GB, of course
    - Per hyper-threaded core this is halved: 1-2GB (throwing away 20-25%)
  - Current Xeon Phi has 60 cores and 16GB of memory
    - ~256MB per core (64MB if running 4 threads/core)
      - Knights Landing has more (a lot more), but still needs to be paid for and cores are weaker than Xeon servers
  - Tesla K40 has 2880 cores and 12GB of memory
    - ~4MB per core
- Undoubtedly HEP needs to make efforts to lower its memory footprint, even to make best use of current hardware
  - Only way to go now is multi-threaded — new frameworks



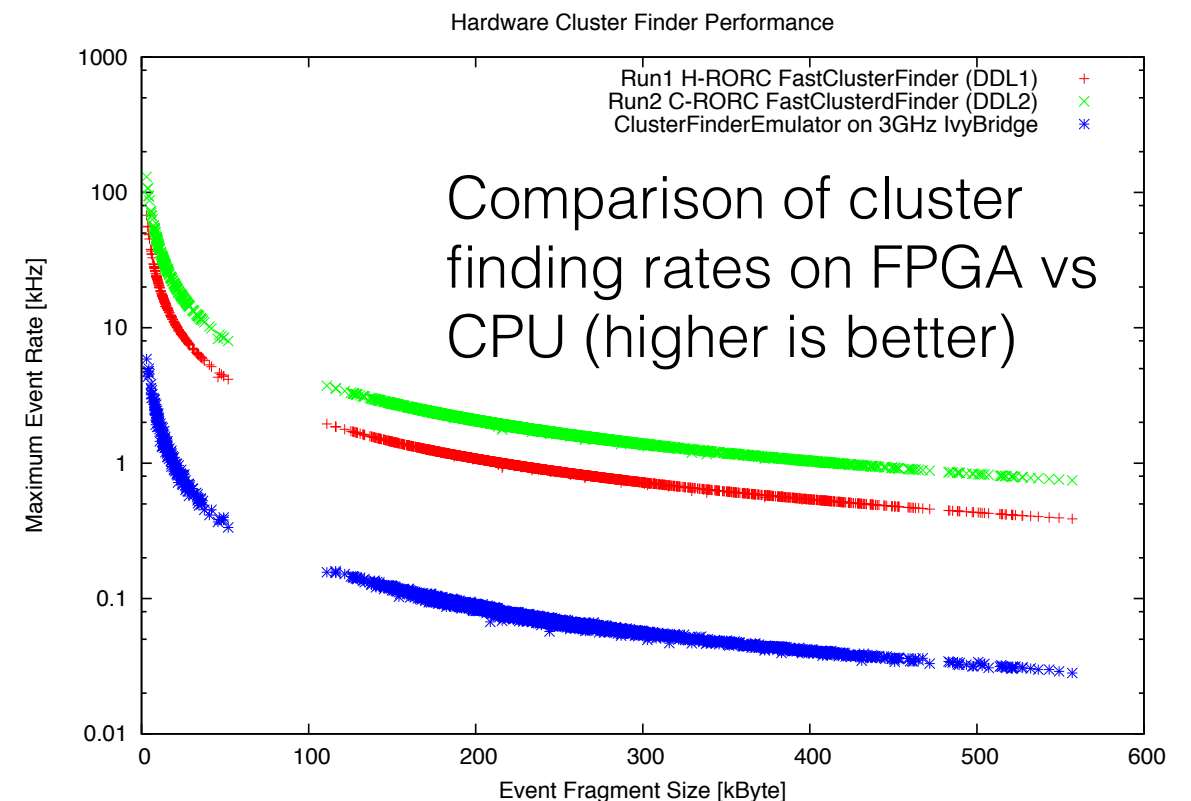
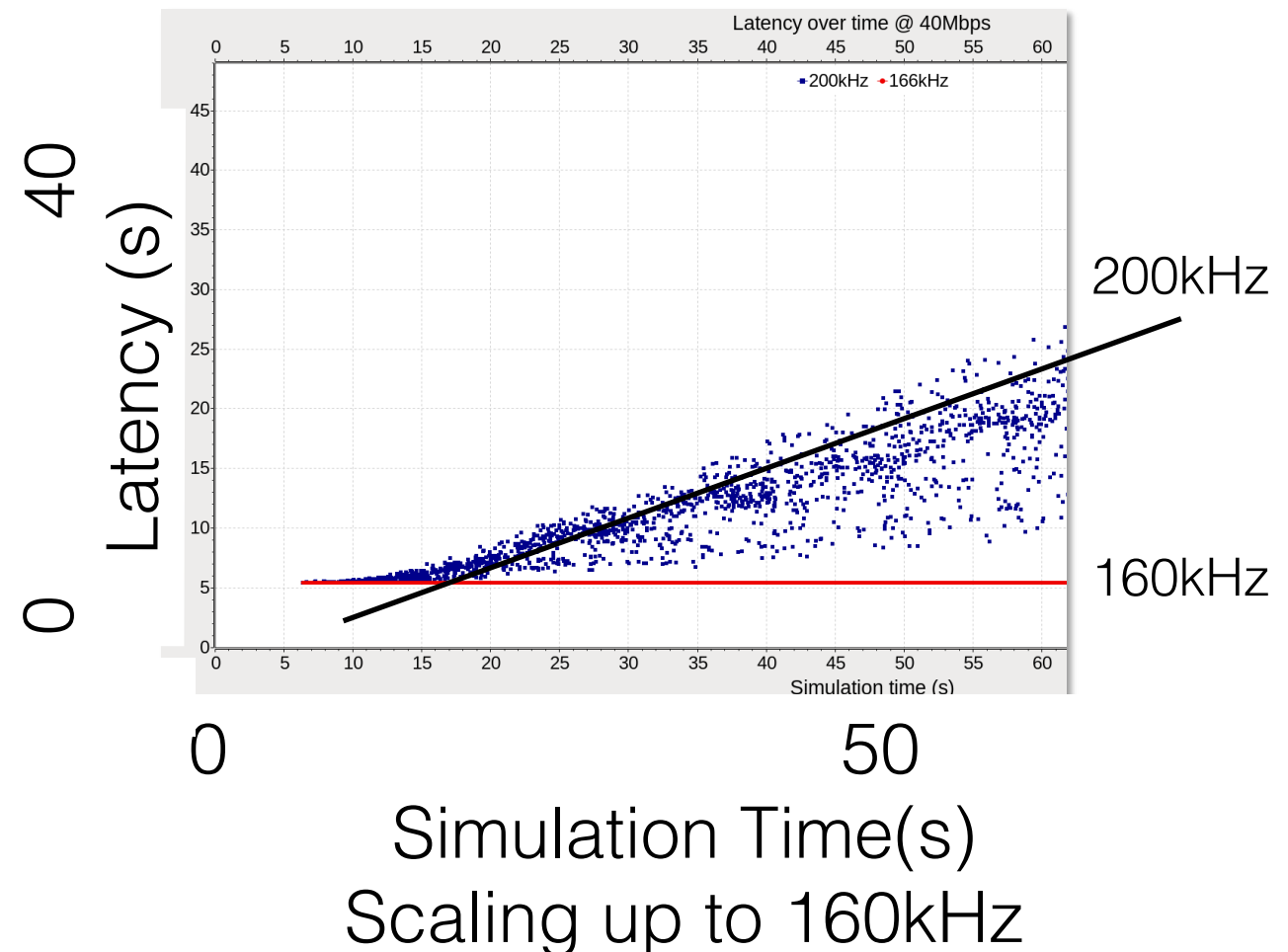
# ALICE: O<sup>2</sup>

- Physics signals have low signal/noise and large backgrounds
- Design an online and offline software framework for ALICE that supports data flows and processing
  - Reflects a trend to blur the distinction between online and offline
  - All collisions are interesting, but not all tracks are equally interesting
    - Flow vs exclusive channels to study properties of QGP
    - Aim at online compression of events to maximise physics data per byte
- Handle 50kHz of PbPb and >1TB/s detector input and continuous readout from TPC
  - Reduce data to acceptable rates for offline storage, 20GB/s average

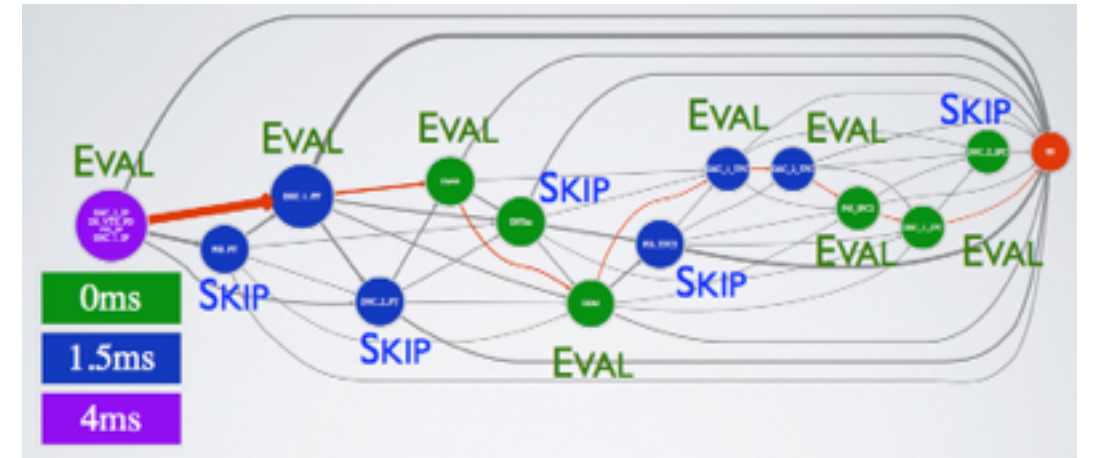


# ALICE: O<sup>2</sup>

- Optimal use of compute nodes
- Take full advantage of CPU resources, including hyperthreading, and make use of GPUs and FPGAs
  - ZeroMQ allows resources to be connected with very low cost IPC
- Project promotes use of common ALFA development framework
  - Sharing core software infrastructure is a good model!
- TDR due very soon



# LHCb Software Trigger



Balazs Kegl, CNRS/IN2P3

- Processing of 'offline quality' at 40MHz
- Smart trigger utilising boosted decision trees
  - Distinguishing between different signals, more than signal/background
    - e.g., b-hadrons over c-hadrons, BDT rejects charm events 3x more effectively for the same b efficiency
  - Essential for LHCb to maximise physics where luminosity is traded for event quality
- Pushes onwards a trend to see integrated processing from the DAQ onwards where possible



# LHCb Data in Run3

- For Run3 and beyond may, for some events, keep only derived data, dropping or compressing RAW
  - MDSTs produced in the online farm
  - Reduce overall storage costs (considerable for RAW on tape) and maximise physics utility of data
- Will be trialled during Run2 (TurboDST) to see if this plan works
  - Proof of concept: *can a complete physics analysis be done based on a MDST produced in the HLT?*
- Interesting to compare with the ALICE strategy of keeping lossy compressed RAW
  - LHCb plan to keep full RAW for the most interesting events; compressed RAW for others and no RAW at all for others
  - No longer limited to binary decisions about RAW data

# ATLAS and CMS Trigger & Offline Strategy

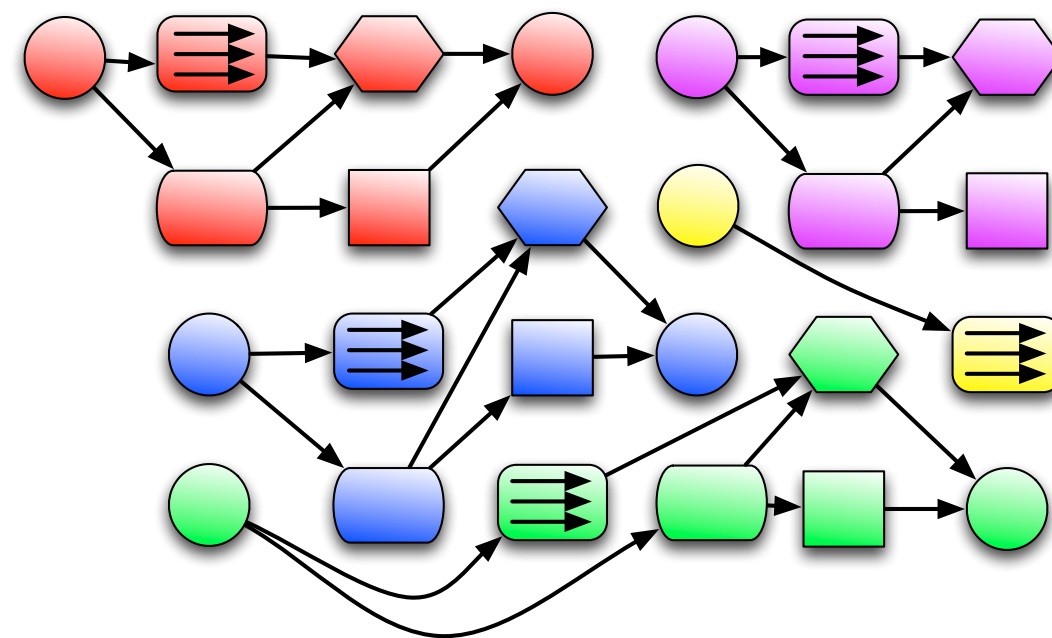


- In contrast to LHCb and ALICE, ATLAS and CMS will maintain a hardware trigger and an HLT trigger farm in Run4
  - Not feasible to extract data from the detector at full L1 rate ( $\sim 120\text{TB/s}$ ) at reasonable material cost (c.f., ALICE Run3 rate of  $1\text{TB/s}$ )
- Track trigger will help maintain trigger efficiencies by greatly improving selection
  - Search billions of track patterns in a few cycles using associative memory
  - Pre-filter track seeds with dual layer pixels
- Processing after L1 will proceed in ‘traditional way’ with high level trigger running in software, followed by offline quality reconstruction
  - In this, more flexible use of HLT/Tier-0 is only to be expected, but this will not radically change the design of the event processing framework
- This provides the context for upgrades to the event processing frameworks

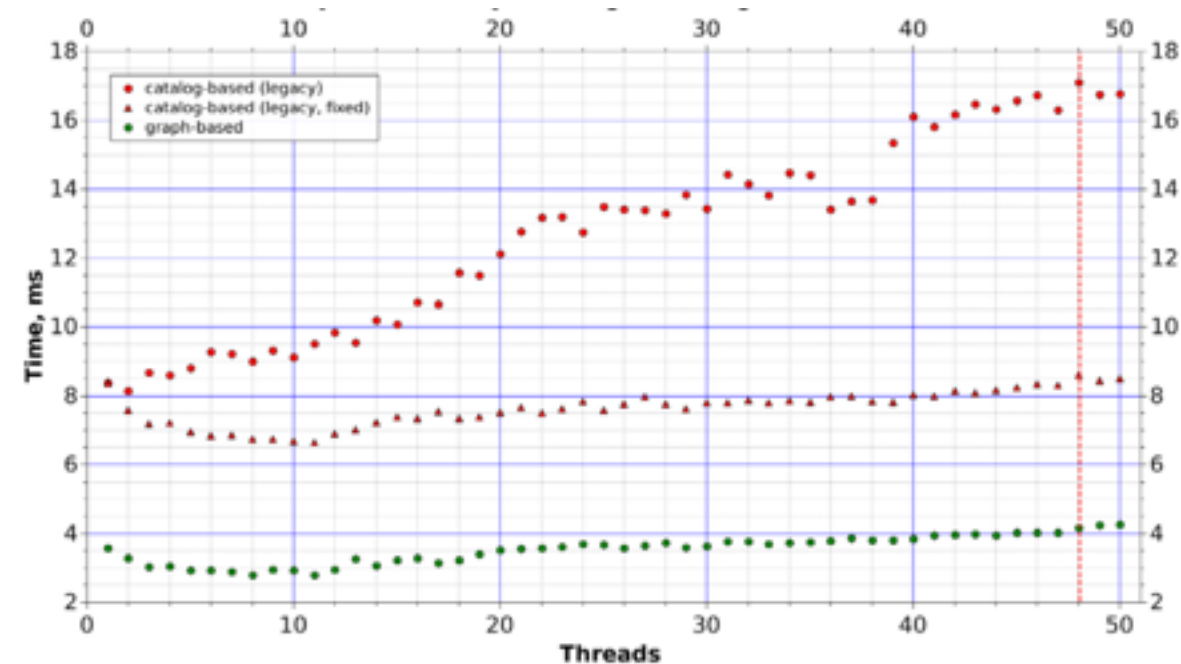


# GaudiHive

- Development to introduce parallelism into the Gaudi framework used by ATLAS and LHCb
- Take advantage of parallelism between algorithms and across multiple events
- Scheduler is data flow driver, but control flows can also be given (important for online)
  - Recent improvements to unify this decision making process into a single graph
- Considerable experience now with design patterns which are good for threading and anti-patterns which are not

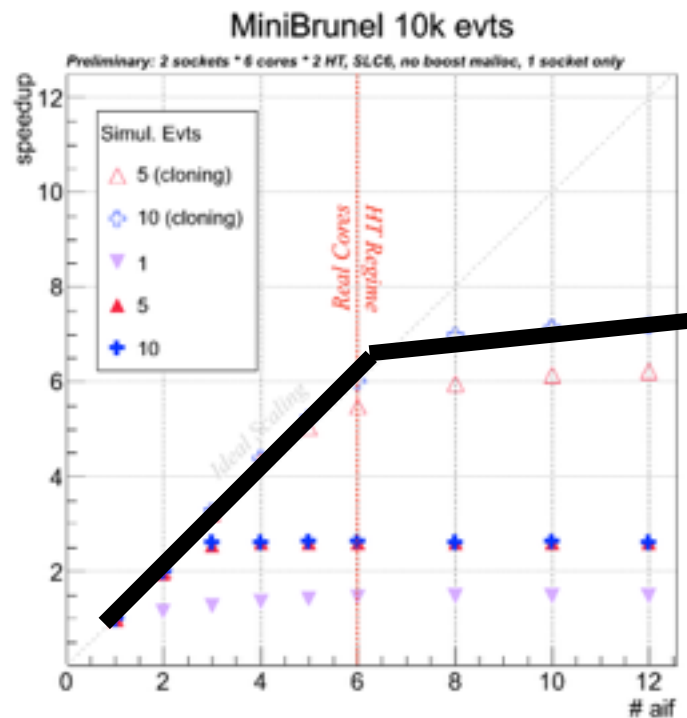


Colours represent different events,  
shapes different algorithms



Recent technical improvements in GaudiHive  
scheduling (Iliya Shapoval, Marco Clemencic)

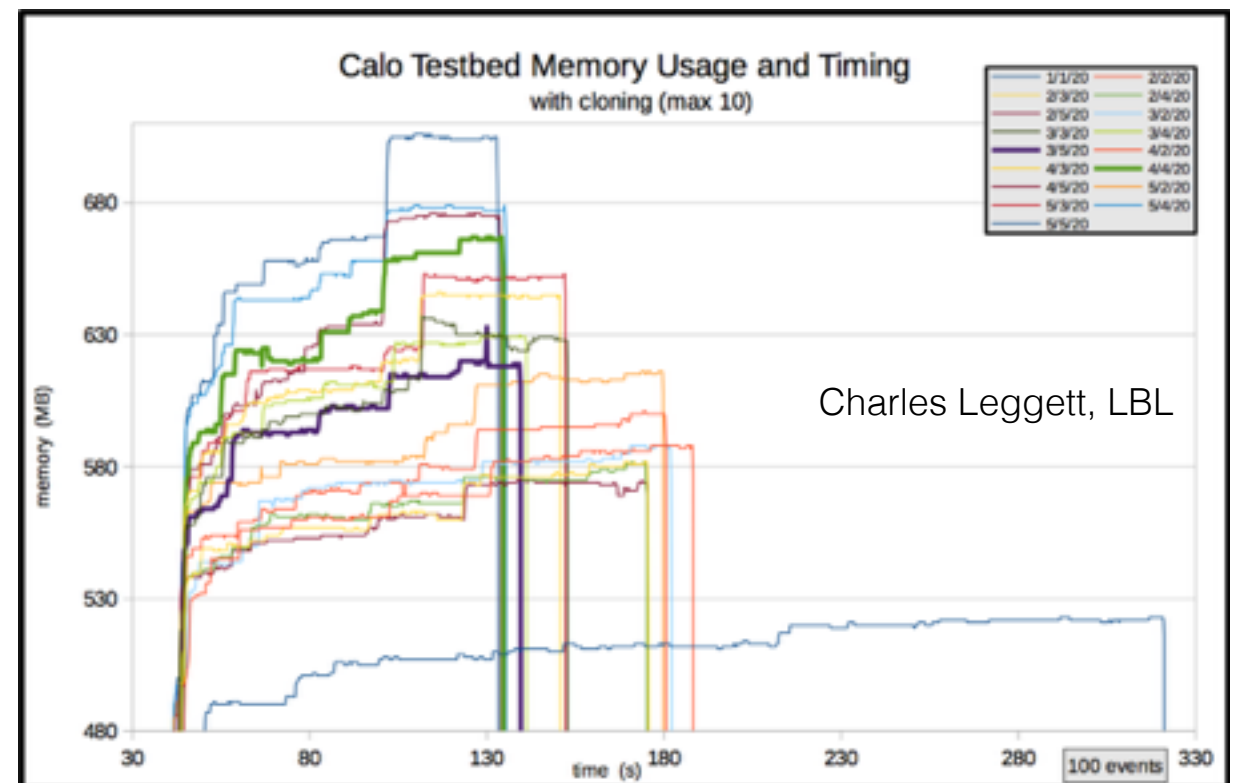
# GaudiHive Scaling Tests



Benedikt Hegner, Danilo Piparo, CERN

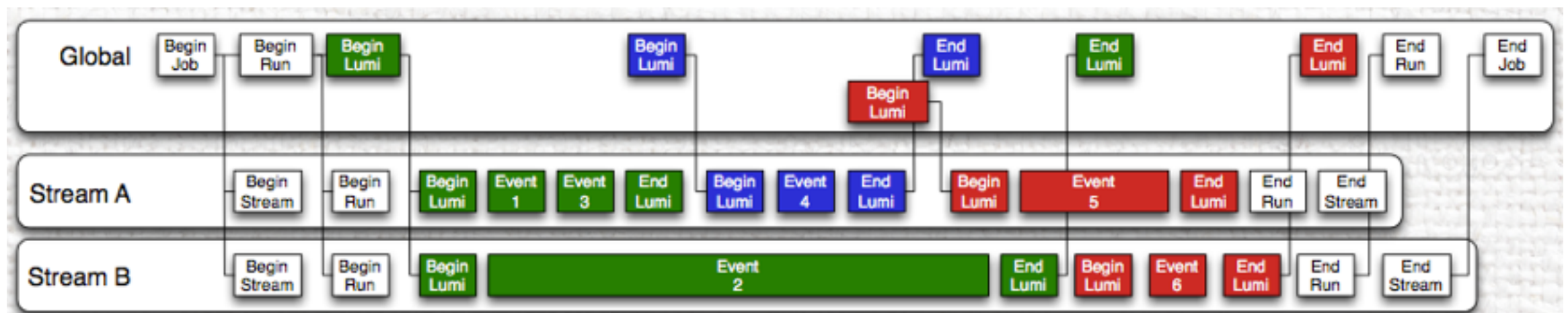
- Scaling of GaudiHive running LHCb mini-Brunel reconstruction
- Linear scaling up to CPU core count
- Expected boost from hyperthreading with only 10 events in flight
- Memory consumption only rises by 7% (limited reconstruction however)

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



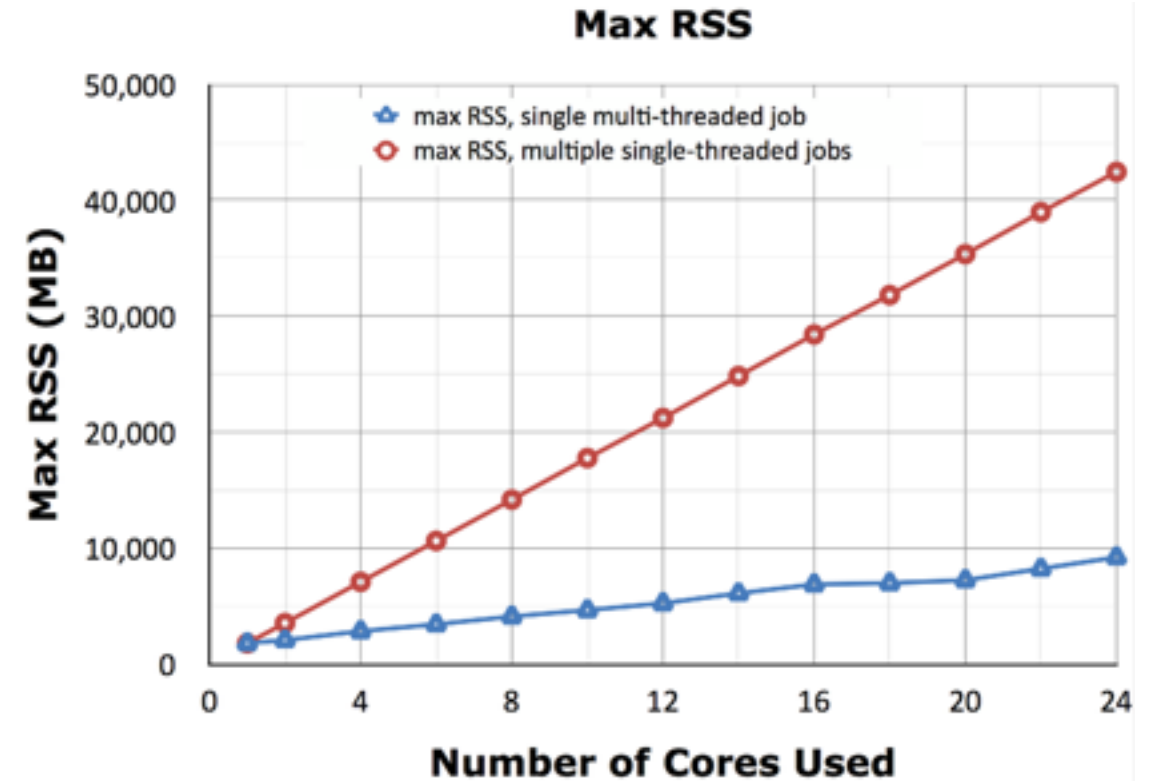
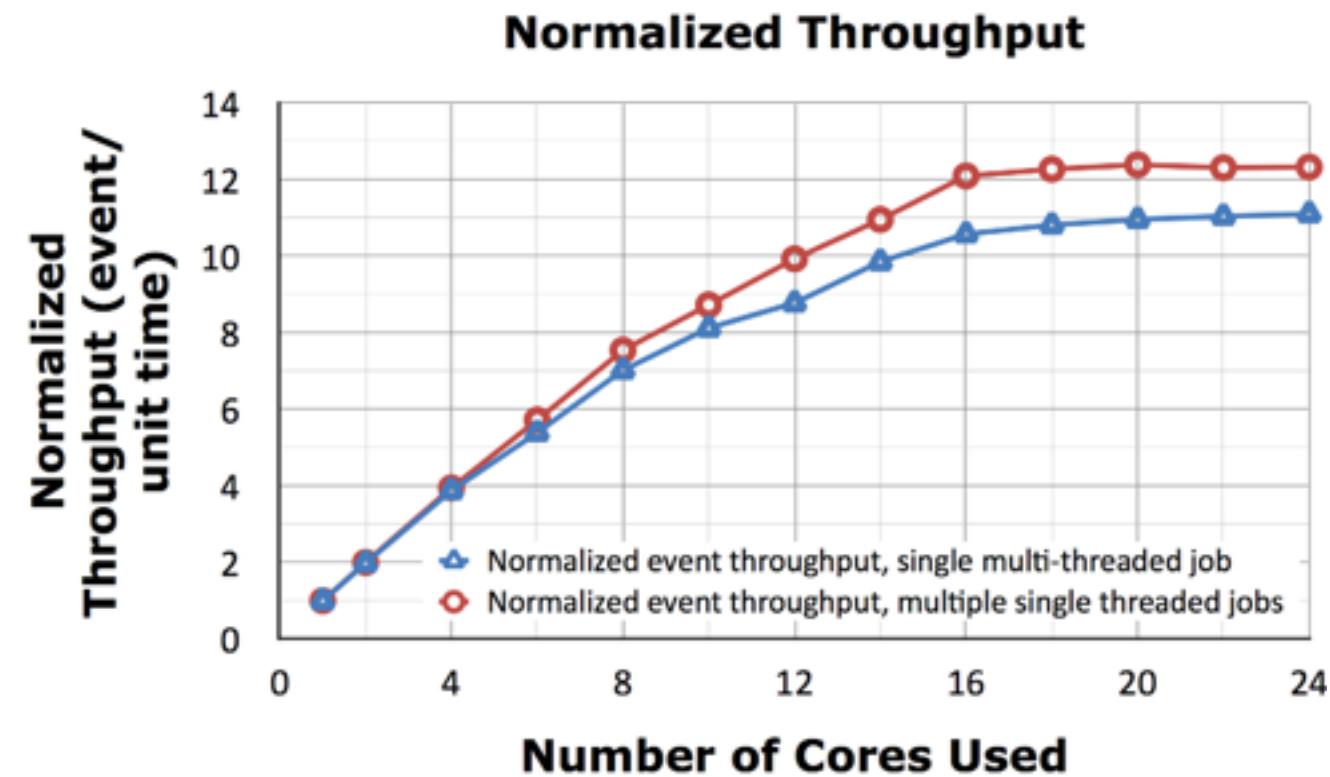
# CMSSW Multi-threaded

- 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
- 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





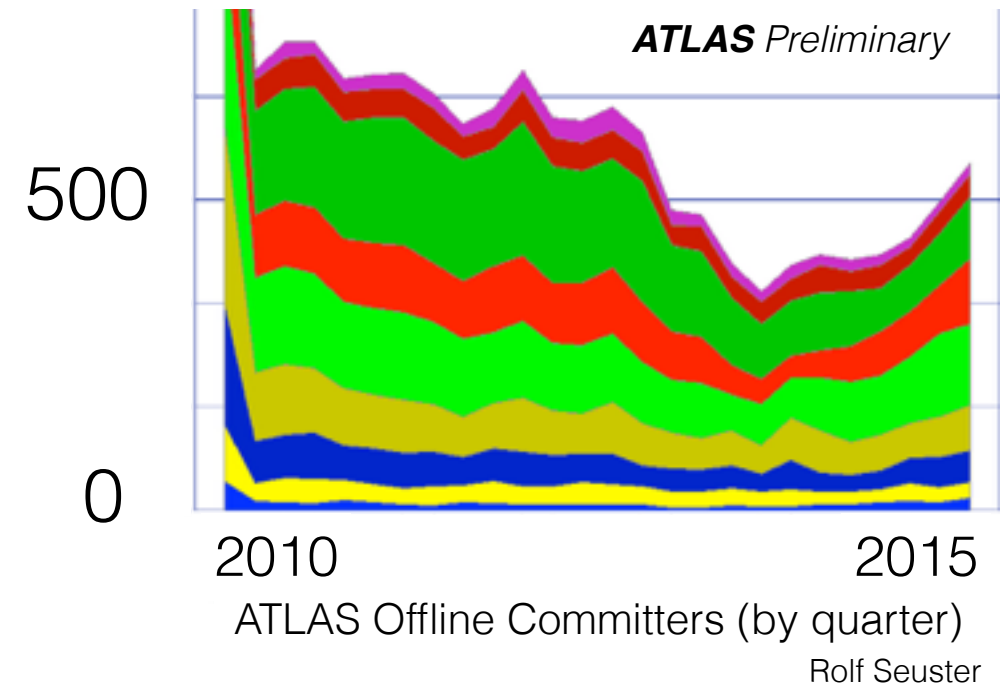
# CMSSW Results



Christopher Jones, FNAL

- High throughput maintained by multi-thread job
- Memory consumption hugely reduced
- Work on technical improvements to improve throughput continues
  - e.g., bottlenecks at condition change boundaries

# Algorithmic Code Evolution



- Highest investment in algorithmic code —  $O(100M\$)$  for LHC experiments
  - Vast majority of offline packages
- Migration of this code to new software patterns and paradigms is not trivial
- Frameworks have a challenge to provide services to algorithmic code
  - Insulate most code from needing to care deeply about multi-threading
- Real skills gap opening up between what is needed to develop for new architectures and where our students and post-docs are
  - Collaboration with experts from other areas will really be needed
  - Training has to become a real focus
  - Cannot afford overly complex solutions just to squeeze out a last few percent or deal with corner cases
    - Software has to be *maintainable* to be evolved

# Analysis Code



- Smart slimming and skimming frameworks used to bring data volumes under control
  - At the expense of some data duplication (though also augmentation used)
  - Must keep data volume and cpu costs under control
- Limited i/o capacity pushes us towards *train models*
  - One job reads an reconstruction output file, writes multiple analysis output formats
  - Maximise use of staged data
    - Staged can mean staged from **tape**, moved from **mass storage disk** to local **SSD**, data that has undergone persistent to **transient conversion**, data that has been moved from **main memory** into the **CPU cache** hierarchy...
- Internally analysis may use multi-threading to have multiple events in flight
  - Remains to be seen how useful/possible this model is re. programming difficulties (sandbox each event?)
- Object stores as a disruptive technology here...?
  - But how to catalog everything?

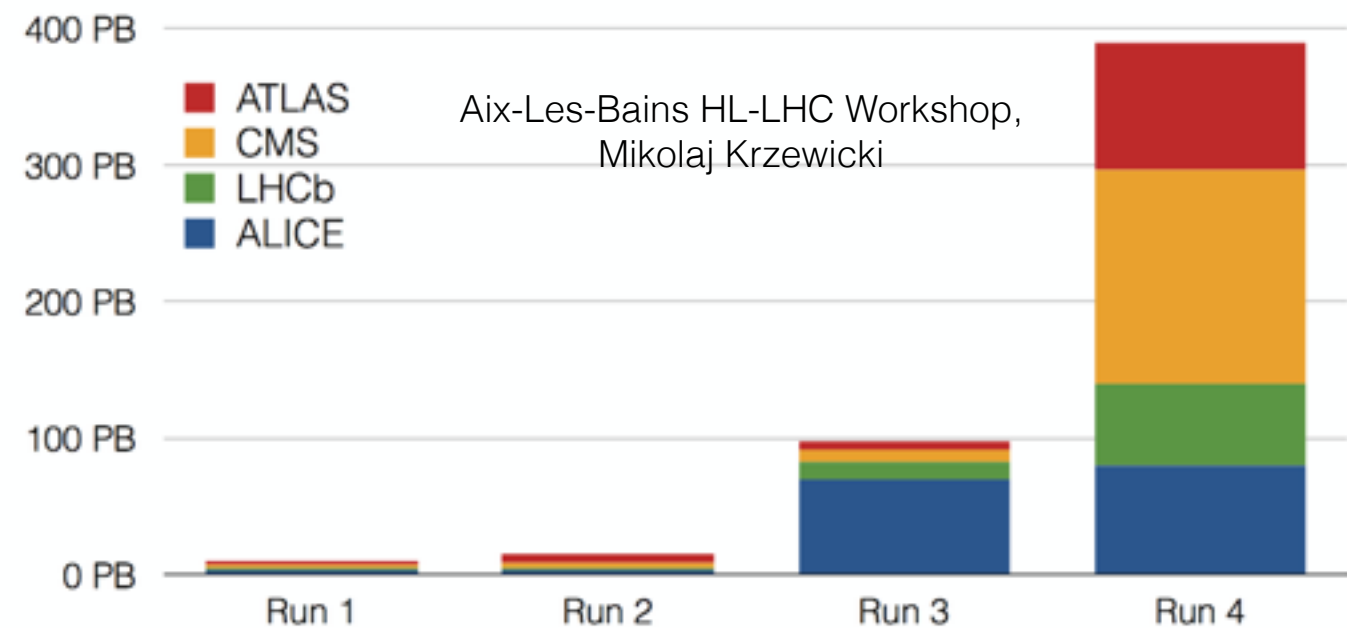
# Software at HL-LHC



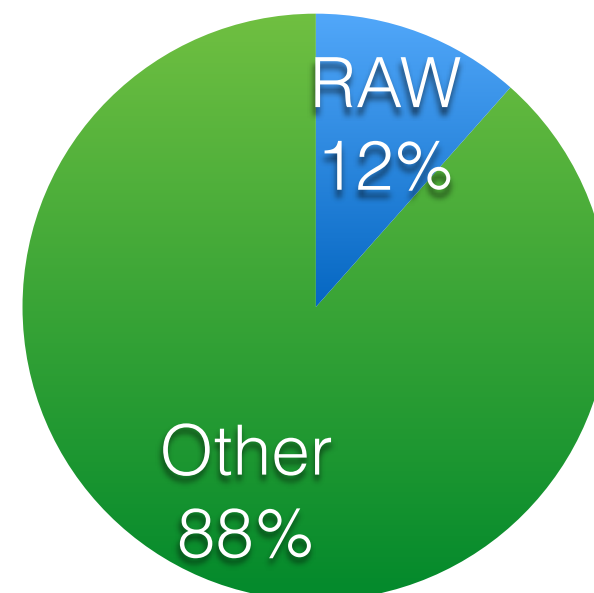
- Hardware will not save us for HL-LHC
  - Hardware only provides the boundary conditions in which software will operate
- Knowledge of the precise details of the problem are vital to a effective solution
  - Differences between experiments are *real* and make a *real difference* when aiming for ultimate physics performance (e.g., b-tagging performance)
  - One-size-fits-all is always possible, but actually rarely desirable
    - There is a point at which we need to become quite specific in our code
- However, we can (and must) effectively share our knowledge and experience
  - Funding agencies will not look kindly on poorly motivated parallel developments
  - Share in cost effective ways, aim to minimise specialisation
- HEP Software Foundation meeting on Friday afternoon

# Data Taking Challenge

- Large increase in Run3 integrated in ALICE's new O<sup>2</sup> DAQ HLT system
- Very large increases in Run4 for ATLAS and CMS at high luminosity
- RAW data is currently a *modest fraction* of offline data for ATLAS
  - x8 more derived data than RAW during Run1
    - Including all monte-carlo
  - Improved already in Run2 (far less duplication of analysis formats)
    - *But how low can we go?*



RAW data to offline for LHC experiments



ATLAS RAW Data Fraction, Run1 (including multiple replicas)



# Data Taking Challenge

- Challenge for ATLAS and CMS will be to keep the multiplication factor for derived data formats under control
  - x8 is not feasible:
    - 150PB (75PB x 2) could become 1350PB (per year!)
- Make derived data fast and flexible
- More flexible data access, such as processing on demand might be used
- Still the problem that some data has to be read from *somewhere*, even if the network was infinite
  - Fast caches might help, if they have the correct data

	RAW (2 replicas)	Derived	Annual Total	Increase over now
Now	8PB/yr	x8	72PB	x1
HL-LHC do nothing	150PB/yr	x8	1350PB	x18
HL-LHC smart	150PB/yr	x4	750PB	x10

x10 right on the edge upper of affordable disk\*: more tape needed...

# Estimating the Computing Challenge (for General Purpose Detectors)

- To make an estimate of how much computing we need at LHC we have to understand the scaling between  $\mu=40$  and  $\mu=140$  and HLT output rates of 1kHz vs 5kHz

Step	HL-LHC do nothing factor	HL-LHC smart factor	Comment
Generation	x20	x5	Need more precise and heavily filtered generators
Simulation	x5	x3	Does not scale with $\mu$ , but we will need more simulation; 'smart' includes GEANT improvements and fast sim
Digitisation	x20	x10	Linear with $\mu$ , technical improvements possible
Reco (MC)	x100	x15	Non-linear with $\mu$ , plus need more events; smart includes truth tracking and algorithmic improvements
Reco (Data)	x100	x25	Non-linear with $\mu$ , plus need more events; smart includes algorithmic improvements and maybe track trigger info
Analysis	x10	x5	Scales mostly with data volume, main problem probably i/o

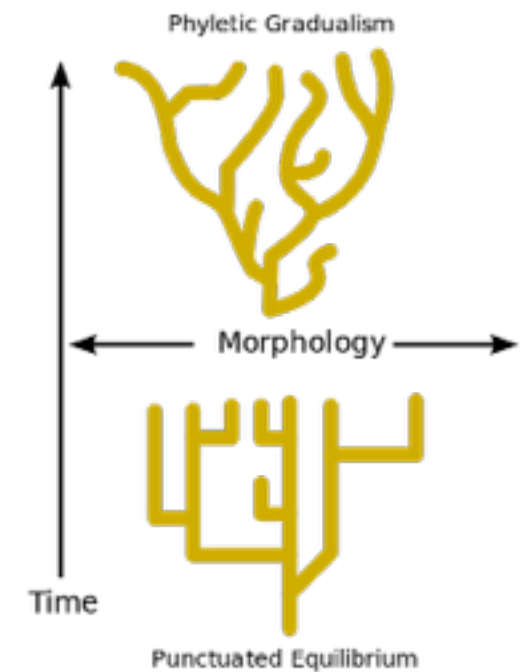
Personal  
guesstimates

# GPD Processing Challenge at High Luminosity

Step	Approx. Fraction Today	HL-LHC do nothing multiplication factor	HL-LHC do nothing CPU increase	HL-LHC smart multiplication factor	HL-LHC smart CPU increase
Generation	0.05	20	1	5	0.25
Simulation	0.45	5	2.25	3	1.35
Digitisation	0.05	20	1	10	0.5
Reco (MC)	0.15	100	15	15	2.25
Reco (Data)	0.1	100	10	25	2.5
Analysis	0.2	10	2	5	1
<b>Total (in units of today's compute)</b>	<b>1</b>		<b>31.25</b>		<b>7.85</b>

- This is a straw man model, but I really believe that
  - Do nothing is not an option (technically as well as politically)
  - Smart gets us into the domain of the possible

# Infrastructure Future - Some Observations



- WLCG and general scientific computing infrastructure is an *ecosystem*
  - Rather healthy to have many 'species' and evolution will happen — clouds, IaaS, etc.
    - Tier-0, 1, 2 and 3 probably all have their (evolved) place
  - Expect regional differences
- HPC trending...?
  - Part of the integration of LHC computing into general scientific computing infrastructures
  - Very good for simulation and event generation: small i/o, CPU bound
- Opportunistic computing pretty well suited to low priority simulation: BOINC, spot priced clouds
  - Caveat: things which we don't pay for will not address core computing needs effectively

# Conclusions

- Evolution of and upgrades to the LHC are well motivated
  - This *is* the energy frontier machine for the next 20 years (plus intensity at LHCb)
- Physics demands require corresponding advances in software and computing to support them
  - This is not just a one way street: it's not what computing we *need*, but what computing we can *develop affordably* to support the overall physics goals
- Hardware advances are producing interesting solutions to overall computing throughput
  - But not in a way that old models of software can easily take advantage of
  - There is no hardware 'magic bullet'
- Software needs to adapt to the HL-LHC running conditions
  - Some of this is scaling up, but some is really a new problem for which we require new solutions, working as a community where we can ([HSE](#), Concurrency Forum)
  - Training, re-training and a new generation of experts in algorithms and computing needed
- Computing will evolve and sites will evolve and diversify
  - Provision of effective storage solutions remains one of our hardest problem in WLCG
- *A lot of challenges to face and a lot of really interesting work to do*



# Acknowledgements

Many thanks to the CHEP organisers for the invitation to give this talk.

I am greatly indebted to all of my colleagues in the field for all I learned from them, directly and indirectly, to prepare this talk.  
(Mistakes probably mean I didn't listen to them closely enough.)

However, the following people deserve special thanks for their input and feedback:

Marco Cattaneo, Richard Mount, Dave Britton, Roger Jones, Ian Bird,  
Zach Marshall, Vladimir Gligorov, Markus Elsing, Niko Neufeld,  
Vincenzo Innocente, David Lange, Andi Salzberger, Latchezar Betev

... and to Hayao Miyazaki and Totoro

