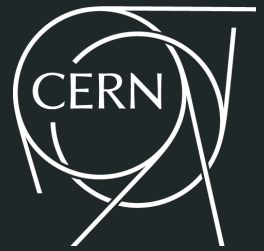


HSF & LPCC Event Generator Workshop at CERN

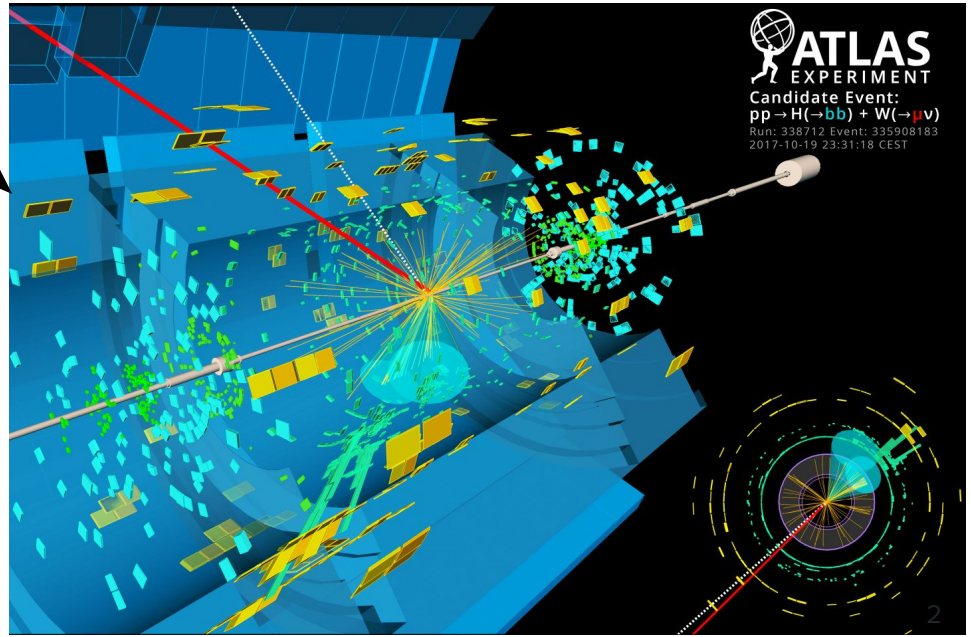
Graeme Stewart

2018-12-17

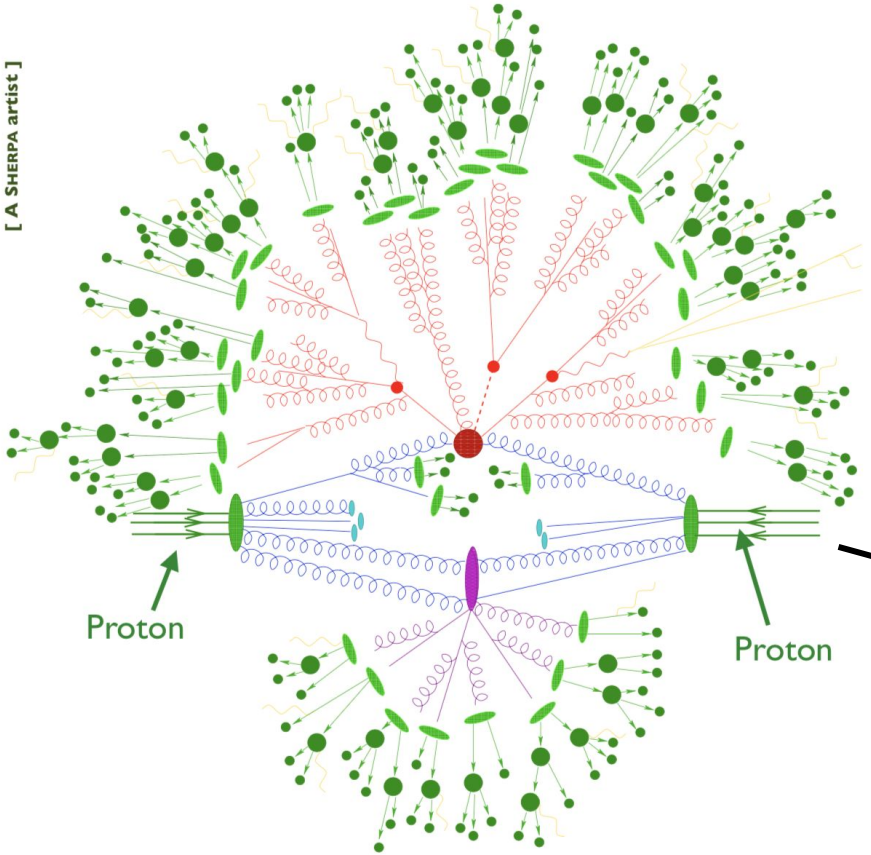
What are event generators anyway?

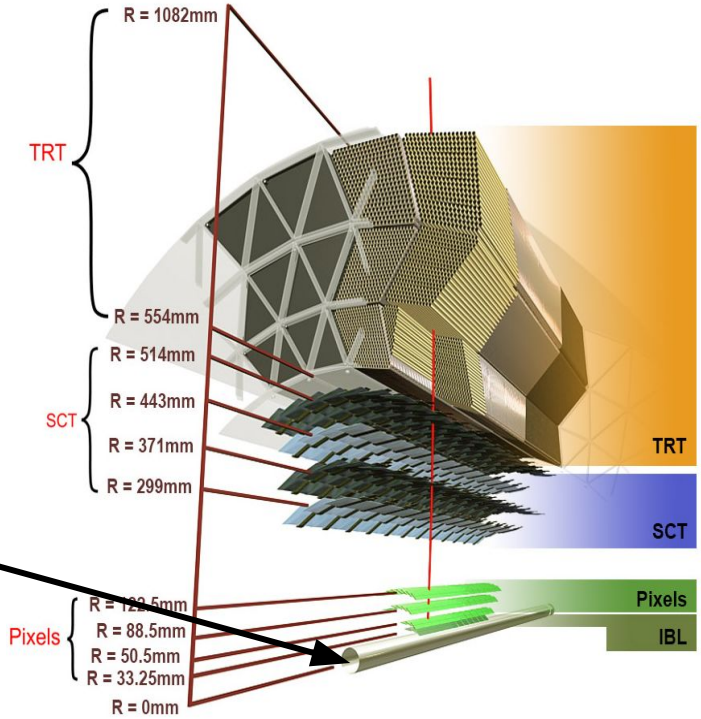
$$\mathcal{L}_{QCD} = \sum_q \left(\bar{\psi}_{qi} i\gamma^\mu \left[\delta_{ij}\partial_\mu + ig\left(G_\mu^\alpha t_\alpha\right)_{ij} \right] \psi_{qj} - m_q \bar{\psi}_{qi}\psi_{qi} \right) - \frac{1}{4} G_{\mu\nu}^\alpha G_\alpha^{\mu\nu}$$

No non-trivial analytic solutions to the QCD Lagrangian, never mind for what happens inside an LHC experiment

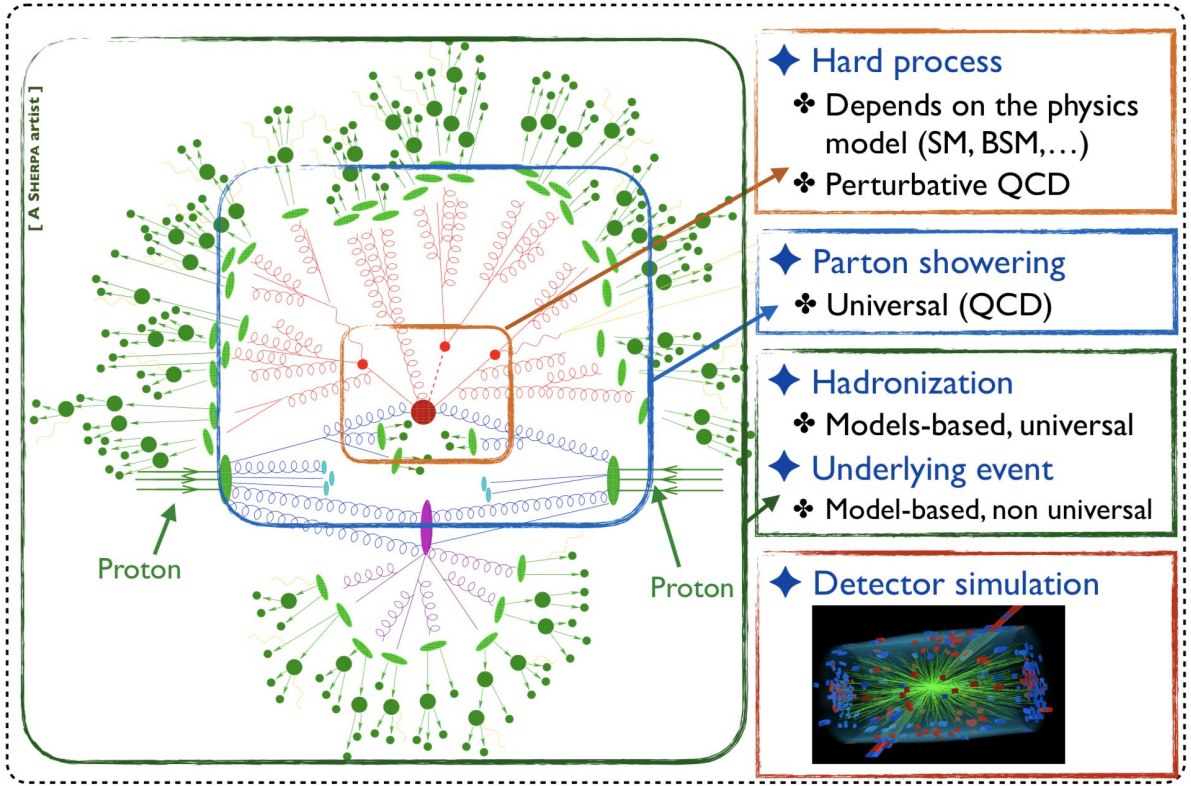


Physics Event Generation





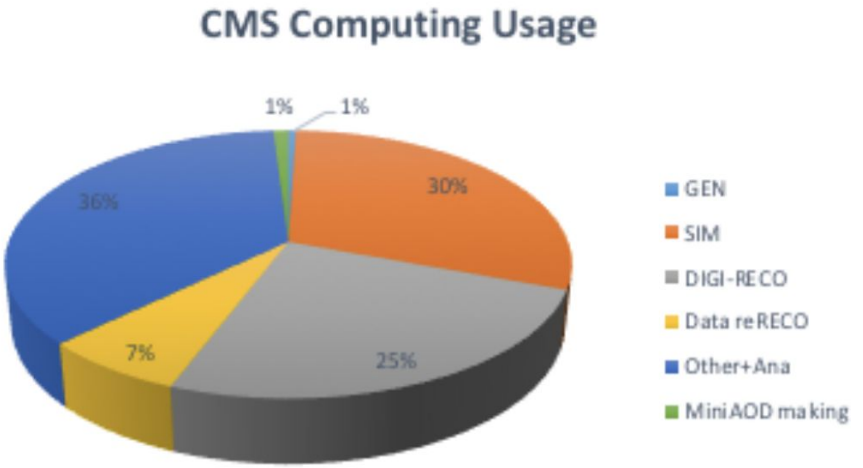
Process Cascades

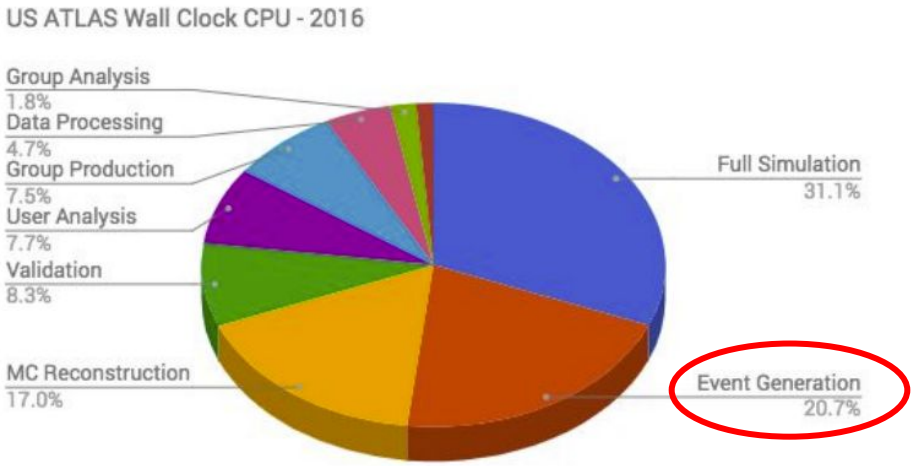


- Problem is rather factorisable
- Solutions are approximate in each regime
- **Many** programs exist that each tackle a specific part and have particular strengths and weaknesses

Slide from Benjamin Fuks

Event Generation from Run 1 and Run 2



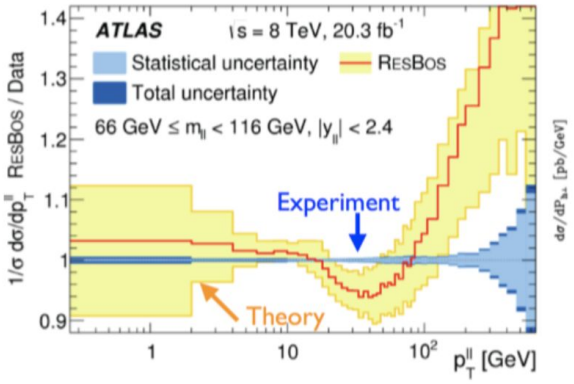


- Event generation is already a noticeable part of the compute budget for LHC experiments, particularly for ATLAS, who use more higher order event generation

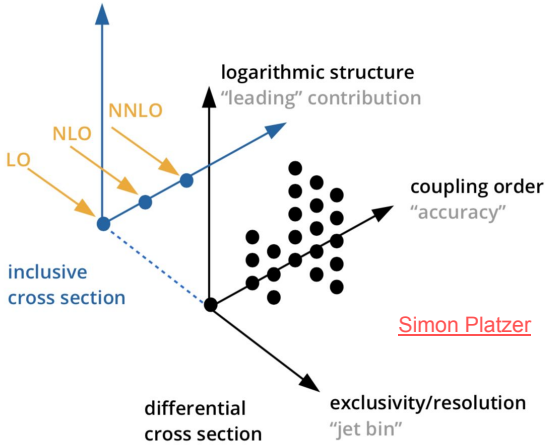
To High-Luminosity LHC

- New physics is not anywhere obvious
- High-Luminosity programme will be characterised by *precision measurements*
  - Look for deviations from Standard Model predictions
  - Searches for weak signals with very large backgrounds
- In particular this means higher order calculations
  - Leading Order (LO)
  - Next-to-leading Order (NLO)
  - Next-to-next-to-leading Order (NNLO)
- Which become computationally harder along multiple dimensions
- Event generation is getting *more important* and getting computationally *more demanding*

Radja Boughezal



Theory errors more than an order of magnitude larger than experimental ones



Simon Platzer

Community White Paper

- Recognised as a key feature in our evolving computational landscape
- Acknowledge the unique challenges:
  - Monte Carlo event generators used by experiments are a side effect of theory interests
  - Work on improving the technical aspects of generators does not advance theorists' careers
    - Fixing threading issues
    - Scaling-up workloads
    - Porting to new architectures
- CWP built further links between experimental software community and generator authors
- *A workshop was the natural next step*

Contents

HSF/LPCC Event Generator Workshop

- Co-organised between HEP Software Foundation and LHC Physics Center at CERN [Indico], 56 registered
- Bring together theorists, experimentalists and software engineers
  - Experiment reports and needs
  - Generator teams status and plans
  - Introduction to software performance optimisation
  - HPC projects
  - Discussion on optimising generator use
  - NLO, NNLO and beyond session
  - Final discussion and next steps

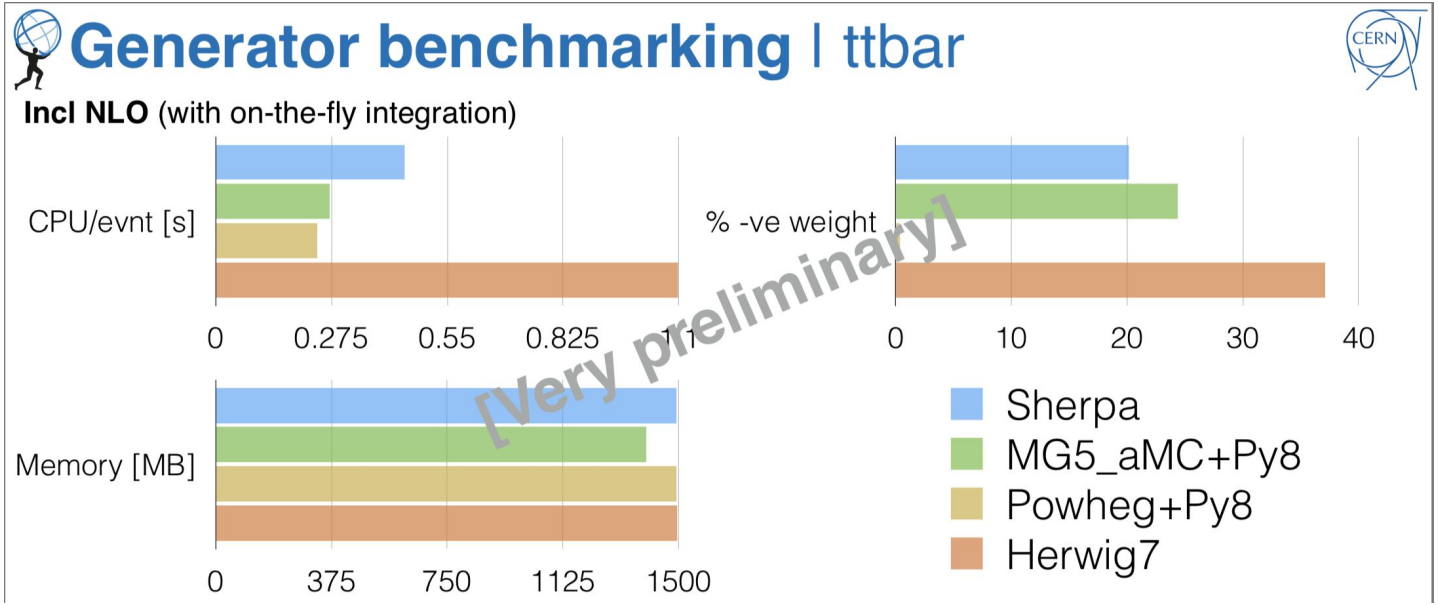
Experiment Reports

- ATLAS and CMS using different strategies right now to support Run 2 physics
  - CMS: Leading Order multi-leg calculations - lots of MadGraph and Pythia; NLO is MadGraph_aMC@NLO and POWHEG
  - ATLAS: Significant CPU spend on Sherpa, V + 0, 1, 2 jets@NLO and +3, 4 jets @LO
- Experiments very happy with increases in calculation accuracy in recent years
  - Thanks to the theory community for this progress
- Moving to NLO hampered by negative weight fraction
  - Substantial increase in the number of events needed to achieve same statistics

ttbar production:

Sample	Fraction of events with neg. weights [%]
Sherpa (lepton+jets)	20.5
Sherpa (lepton+jets)	20.4
Sherpa (dilepton)	20.4
Sherpa ttbb (lepton+jets, CSSKIN, 4FS)	24.4
Sherpa ttbb (lepton+jets, CMMPS, 4FS)	25.7
aMC@NLO+Py8 (lepton+jets)	23.7
aMC@NLO+Py8 (dilepton)	23.7
aMC@NLO+Py8 (FxFx, 70 GeV)	28.4
aMC@NLO+H++ (4FS, ttbb)	37.2
Powheg+Herwig7 (lepton+jets)	0.4
Powheg+Herwig7 (dilepton)	0.4

Events with negative weights need to be simulated, reconstructed and the *subtracted* from final histograms



- Very difficult to ensure that generators are doing comparable calculations
  - Need expert input from the generator authors
- These plots are not to be treated as definitive (very preliminary!)
- But they are a good start to understanding better what generators are doing in practice for the experiments

Experiment Reports

- LHCb and CMS do a lot of event generation in the same grid job as simulation
  - Need event generator to be multi-thread friendly
    - Avoid serial bottleneck in an otherwise parallel job
  - EvtGen used a lot and this has not been much developed recently
- Filter efficiencies have a huge effect on how much CPU is spent on each accepted event
  - W+8 jets is about 0.2% of sample
  - Want to bias phase space generation to increase acceptance further
- GPUs seem to be quite promising for event generation
  - Cross section calculations x100
  - Phase space integration x60

  (See last slide of Efe's talk for references)

Generator Team Reports

- Really hard to do this justice - each generator has a different team and developing and optimising in many individual ways...
- ...but I picked a few of what were the highlights (my biases)
- Illustrate the problems and challenges from many different perspectives
  - Some of these are more physics oriented
  - Some are very technical
  - Awareness of optimisation issues is rising

Sherpa

- Complex program with great flexibility in tacking different stages and processes of event generation
- Profiling now to find out which pieces are most expensive for the LHC experiments

Code performance

LO merging using COMIX

Process W^-+	0j	≤1j	≤2j	≤3j	≤4j
RAM Usage	39 MB	44 MB	49 MB	64 MB	173 MB
Initialization time	<1s	<1s	3s	22s	7m 7s
Startup time	<1s	<1s	<1s	<1s	2s
Integration time	25s	3m 19s	34m 8s	3h 12m	2d 17h
10k weighted evts	3m 24s	3m 51s	4m 2s	4m 4s	4m 21s
10k unweigthed evts	3m 20s	4m 39s	11m 47s	35m 54s	4h 3m

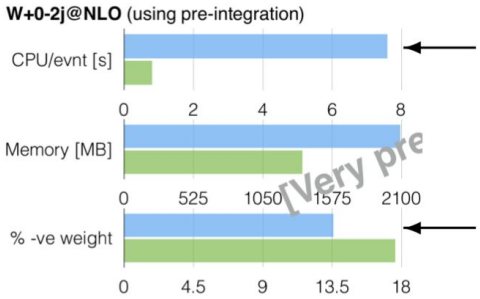
NLO merging using AMEGIC+BLACKHAT/COMIX (S/H-events)

Process W^-+	0j	≤1j	≤2j
RAM Usage	51 MB	112 MB	572 MB
Initialization time	1s	20s	4m 6s
Startup time	<1s	2s	18s
Integration time	20m 48s	4h 45m	5d 23h
10k weighted evts	3m 58s	4m 38s	6m 48s
10k unweighted evts	4m 14s	4h 8m	24h 54m

On dual 8-core Intel® Xeon® E5-2660 @ 2.20GHz using MPI parallelisation, timings quoted cumulative.

bottleneck, steps taken to reduce this look promising

factor 10 improvements seem feasible



Dominated by $W + 2j$ H events, which are dominated by CKKW clustering, ie. scale determination. Using H'_T approximation instead (like in MG) gives **factor 4** improvement: 2s/evt
Available in SHERPA-2.2.x.

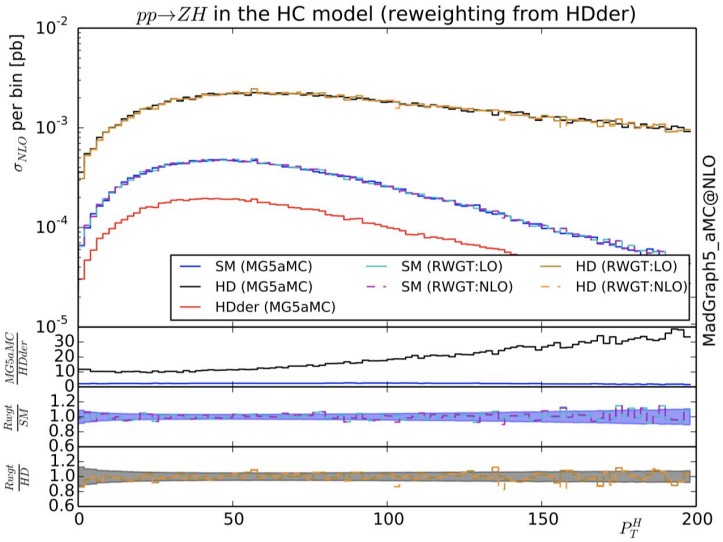
Dominated by $W + 2j$ H events, can also be improved upon.

[taken from Josh's slides]

⇒ **more improvements possible with non-default settings**
→ approximate physics, but substantially improve performance

MadGraph5_aMC@NLO

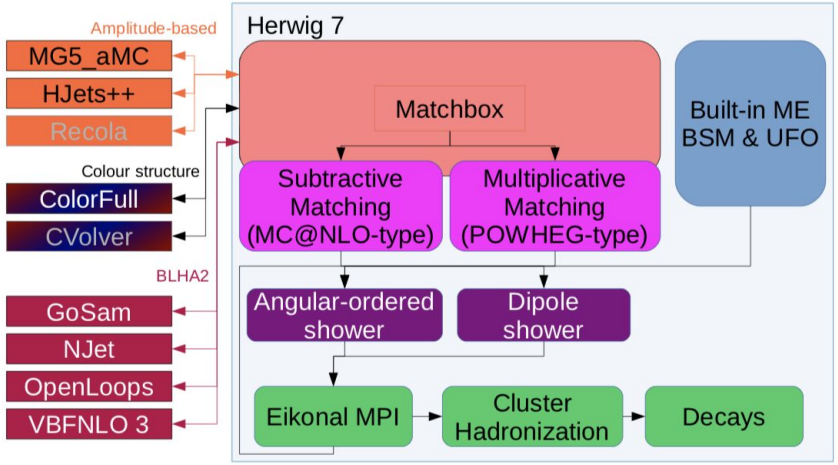
- Compilation settings are rather conservative
  - Seem to have very cheap gains from increasing to `-Ofast` (under validation)
- Developing options to allow for event reweighting
  - Reuse calculations with different pdf
  - Can also be used at NLO
  - Trades disk space for CPU cycles
- Developing MPI strategy for scaling to larger resources
- GPU port available
  - Validated in 2013
  - No interest from the experiments at that time



Pythia8

- Leading Order generator, particularly used for showering and hadronisation
  - Does not use a lot of CPU, but is run widely (so integrated gains could be large)
  - Often combined with other hard scatter (NLO) generators
- Recent work to make the generator multi-threaded
  - Only one option that violates this now
- New Heavy-Ion event generator (Angantyr)
- More accurate shower and dark matter models

Herwig7



- History
  - C++ rewrite of FORTRAN Herwig6 code
- Tried to learn the lessons of the past
  - Adopt better coding practices
  - Emphasise code sustainability over speed
  - And improve the physics content (which was hampered by design issues in the previous code)
- Recent improvements are all in physics code
  - This is what gets theory recognition and offers career advancement
  - Code re-entangled… becoming fragile again
- Code could now do with substantial parts being rewritten
  - “years of work for little recognition”

Photos/Tauola and EvtGen

- Specialist ‘afterburner’ generators for decays of heavy flavour hadrons and taus, developed for BaBar
- Photos and Tauola have not been developed much recently
  - Tauola still in F77
  - Photos largely in C, but with C++ interfaces
  - Still no HepMC3 outputs
  - Still in SVN, no appetite from developer to move
- EvtGen
  - Code was in maintenance for many years
  - Now technical development reinvigorated
    - Gerhard Raven contributing - this decayer is very important to LHCb
  - Has Photos and Tauola as dependencies, but can now replace Tauola with Pythia8

Using Event Generators More Efficiently

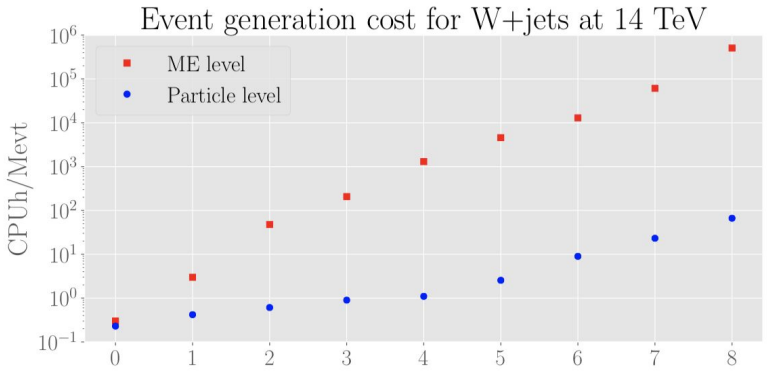
- Discussion developed by Marek Shönherr, helped by Stefan Höche and Andy Buckley
- Various ideas proposed
  - Use LO calculations for sample bulks, then just rescale according to some NLO/NNLO scaling
    - Not hugely liked by the theorists and some disagreement between generator teams as to how effective it would be
  - Generate Matrix Elements for hard process and share them between ATLAS and CMS (x2 saving)
    - Each experiment would do its own hadronisation and showering
    - Worries about independence, but not considered that serious
    - Looks like it could fly
  - How to get experiment support for generators
    - Willingness to do this, but not really so clear what the scope is

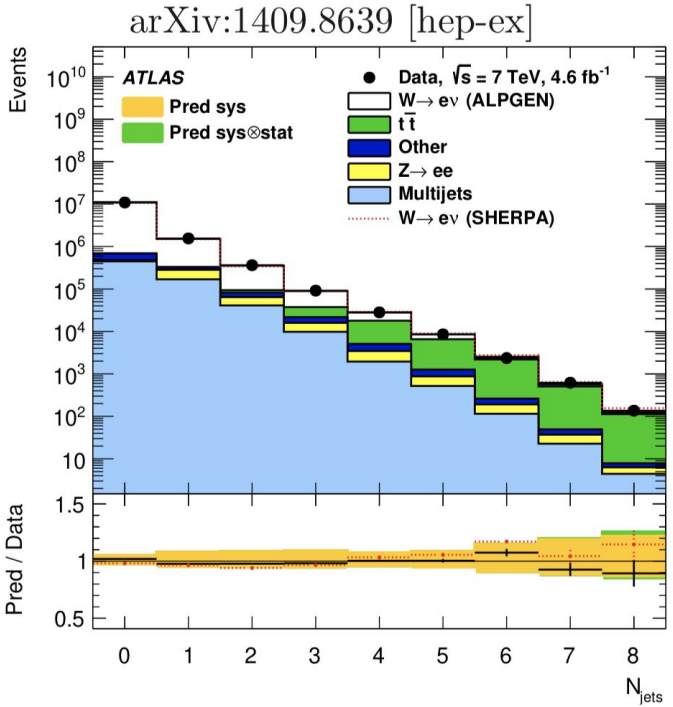
HPC Workflows

- HPCs are characterised by
  - Large scale resource allocations (hundreds to thousands of nodes)
  - Fast interconnects
  - Accelerators (usually GPUs)
- Getting event generation running on these machines can be interesting
  - N(N)LO can benefit from the resource scale (although far from uniquely)
  - Fast interconnects aren't really needed, but phase space integration does require internode communication
  - Code bases are much smaller and simpler than other experimental code
    - Porting to non-x86 CPU architectures should be quite easy (Sherpa on Power8@MIRA)
    - Porting to accelerated architectures should be possible (i.e. MadGraph)
- The last point is particularly interesting for HEP given the paucity of other codes that could reasonably run on GPUs

HPC Scaling Workloads

- W + jets simulated using Sherpa to produce matrix elements
  - Use an HPC friendly format, HDF5 instead of XML
  - Allows parallel read and write
- Hadronisation handled by Pythia8 on CORI
- Use a lightweight HPC workload paralleliser, DIY
- Splitting ME and particle simulation allows for various parameter scans and tuning

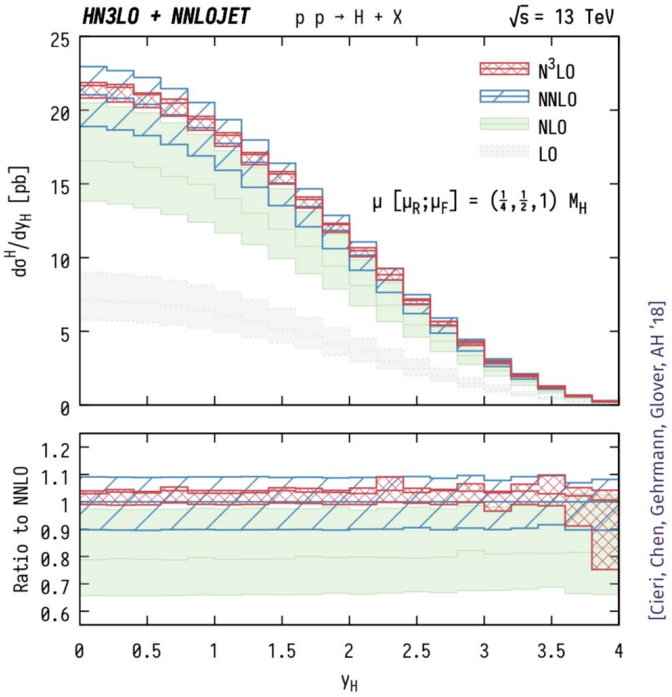




ATLAS observation of up to 8 jets

NLO / NNLO / N3LO ...

- Workshop was focused on computational software matters
- However, most dramatic advances come from *algorithmic* improvements from the theory community
  - These make inclusive cross section calculations reachable now, even at high orders
- We hope for more progress here - in the hands of our theory colleagues



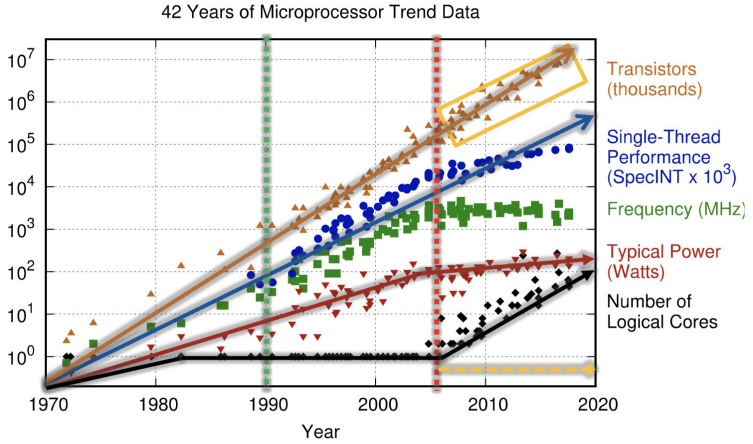
- **LO**
  ↪ $\mathcal{O}(1\ \text{CPU min})$
- **NLO** [Antenna]
  ↪ $\mathcal{O}(30\ \text{CPU min})$
- **NNLO** [Antenna]
  ↪ $\mathcal{O}(100\ \text{CPU h})$
- **N^{3}LO** [q_T sub.]
  ↪ $\mathcal{O}(7\text{M CPU h})$

Performance Optimisation

- One of the successes of the workshop was to bring two different worlds together...
  - MPI - Message Passing Interface for software engineers
  - MPI - Multi-Patron Interaction for theorists
- We wanted to introduce some of the core concepts of performance optimisation to those more concerned with the physics theory
- Organised a dedicated session during the main workshop
  - Third day as an optional hackathon

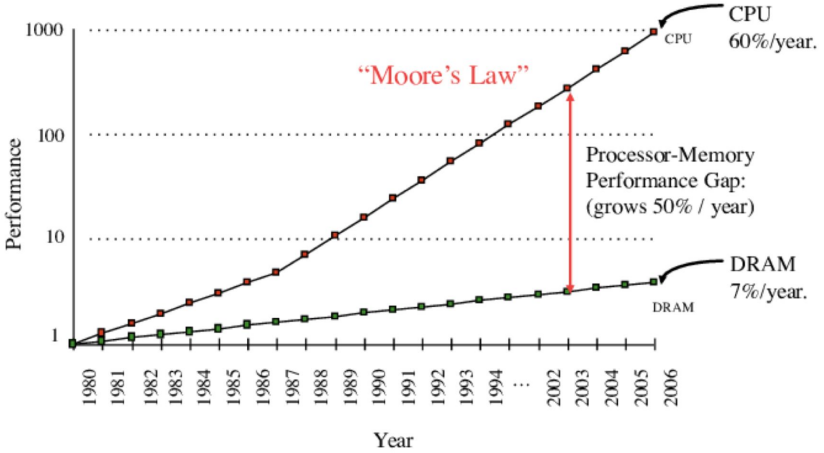
Introduction to Performance Optimisation

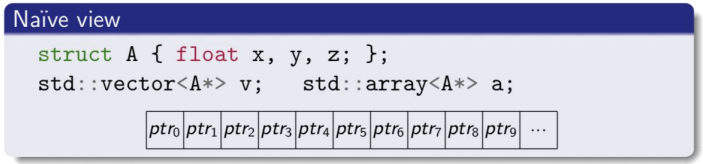
- Introduction to Performance Optimisation
  - Explain the performance of modern processors
    - What it relies on and how it has changed over the years
- Identify which parts of the code should be optimised
- Importance of parallelism today
  - Key concepts: Amdahl, Gustavson
- Profiling is a tricky business

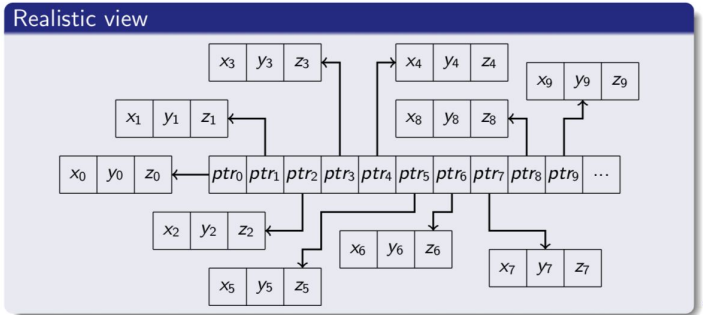


Memory Optimisation

- Stress the importance of data layout to code performance
- Cover the basics of stack, heap, pointers
  - And what happens in C++ STL containers
  - Simple tips to improve
- Arrays of Structures vs. Structures of Arrays
- How to spot poor memory layout through profiling







Hackathon

- A few people stayed on for a Hackathon on Wednesday
- A longer introduction to performance
  - Pipelining to Multi-processing
  - Why the compiler is your best friend
    - And your best friend likes clear optimisable code!
- Then a tutorial on Intel VTune and Amplifier
- Success was to sit with a few generator authors and get them to look at key parts of the code with ‘performance goggles’
- A lot of generator codes are quite amenable to optimisation
  - Self contained smaller code bases
  - More computational elements that are ‘kernel like’

Outcomes

- Workshop was a success
  - Brought different communities together for a different, but increasingly important, purpose
  - Feedback was positive
- Initiative has been well received by management
  - It’s part of the HSF’s job to do these things
- The problem of supporting technical work on generators is recognised
  - Experiment support will be looked at seriously
  - Engineering support from new funding initiatives

Next Steps

- Several people have agreed to be convenors of a new Event Generator Working Group
  - Andrea Valassi, Josh McFayden, Steve Mrenna, Stefan Höche, Taylor Childers
- Areas to focus on
  - Workshop proceedings - setting a baseline
  - Further work on benchmarking and understanding experiment use
  - Experiment support for technical work
  - Understand better what the HL-LHC generator needs will be
  - Sharing of Matrix Element calculations
  - Use the MadGraph GPU port in earnest (would like some validation!)
  - Longer (five day) hackathon for next year, really drill into the code
- Workshop was an excellent start - lots to do!