# HEP Physics Generators:

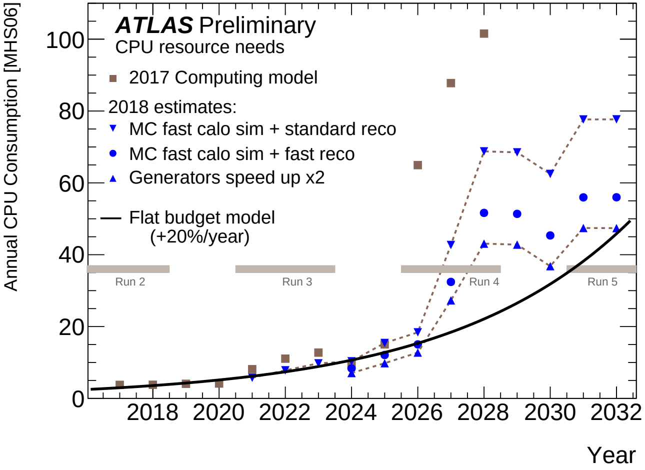
## The Community Roadmap for the Next Decade

Stephen Mrenna

Fermilab, CMS, Pythia

21 November 2019

# Why do we need a roadmap?




■ 2017 ▼ fast calo sim used for 75% of the Monte Carlo simulation

■ faster reconstruction seeded by the event generator information

▲ event generation $\times\frac{1}{2}$ **solid:** flat funding.

## Clarifications/Details

ATLAS extrapolation: Sherpa W/Z + jets

MG is 2x faster

Difference is the choice of scale: MG uses parton kinematics; Sherpa uses an iterative (slow) clustering algorithm

(Preliminary) Physics the same

Thus, factor of $\frac{1}{2}$ is (maybe) already there!

Not so fast: we want all the improvement we can get, plus this is not necessarily a universal statement

## The Destination

### Predictions that “looks” like the data:

Hard process (parton level) at highest possible order (NLO?) and multiplicity (usually # QCD partons)

1. construct the amplitudes (functions, recursion)
2. evaluate cross section (VEGAS)
3. unweight (rejection)

Merged with parton shower-based

1. prepare state for showering (factorial # histories)
2. accept/reject (Sudakov)

Improvements are needed on all facets of event generation

# Case Study:
## Scientific Discovery Through Advanced Computing

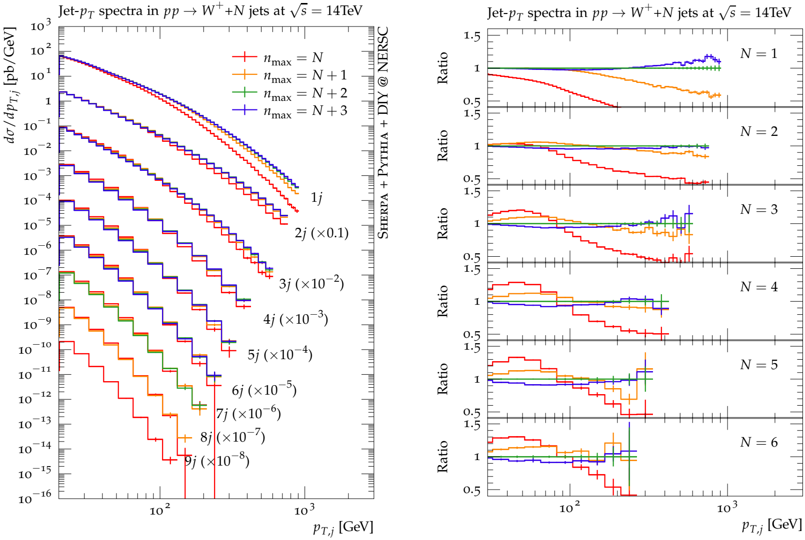
> [1] *SciDAC projects are collaborative basic research efforts involving teams of physical scientists, mathematicians, computer scientists, and computational scientists working on major software and algorithm development to conduct complex scientific and engineering computations on leadership-class and high-end computing systems at a level of fidelity needed to simulate real-world conditions.*

Crossover between two SciDAC projects:

1. Matrix Element and Event Generation (SH, SM, SP, TC)
2. Framework + Optimization (FNAL+ATLAS+ANL+SM)

Work of: Höche, Prestel, Schulz: PhysRevD.100.014024

[1] `https://www.scidac.gov/about.html`

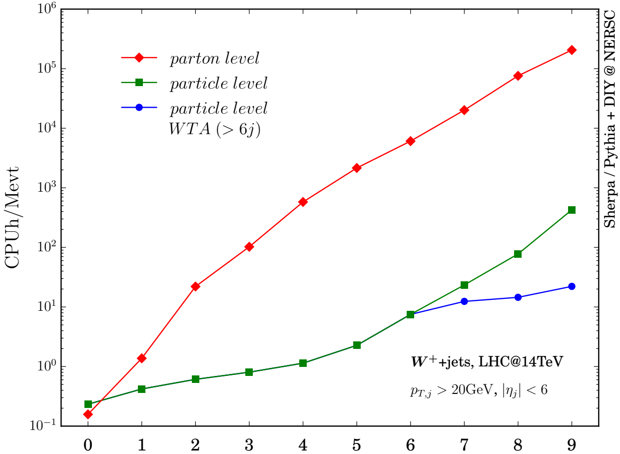



Jet transverse momentum distributions in $W^+$+jets events. We show a comparison of multi-jet merged simulations where the maximum jet multiplicity, $n_{\max}$, is set to the number of measured jets, $N$ (red), to $N+1$ (green), $N+2$ (blue) and $N+3$ (purple).

## Novel event generation framework
### efficient simulation of vector boson plus multi-jet events

- MPI parallelization of parton-level and particle-level event generation
  - Calculated at NERSC (High Performance Computing)
- storage of parton-level event information using the HDF5 data format
  - HPC –friendly, table based
- leading-order merged Monte-Carlo predictions with up to nine jets in the final state.
- parton-level event samples for $3\mathrm{ab}^{-1}$ and code to produce particle level

# Scaling (CPU hours / 1M-events)
## parton-level vs particle-level for $W^+$+jets at the LHC

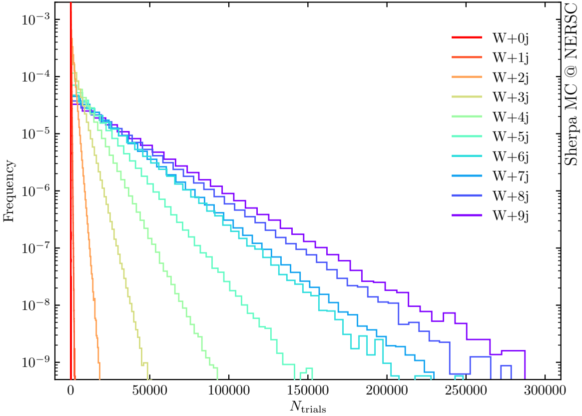



■ standard clustering ● winner-takes-all (WTA)

Number of quarks $\leq 6$ in $W^+ + 6, 7$jet and $\leq 4$ in $W^+ + 8, 9$jet

# Complications: Number of trials in the unweighting

## # Trials to get 1 Unweighted Event




Timing of overall MPI run set by most inefficient rank – If allow variable number of unweighted events, scaling from I/O – if saving weighted events, then inefficient – if fixed # trials, then must handle 0-weight events

# Complications: Parton Showers and Merging

- CKKW-L method constructs *all* histories corresponding to a given parton-level final state in the fixed-order calculation.
- The number of possible histories grows at least factorially: timing and memory usage of the executable therefore increase rapidly with the final-state multiplicity.
- The computational complexity increases by approximately a factor 1.5 for each additional final-state jet.
- Starting with $pp \to W/Z + 7$jet final states, the jet clustering procedure also begins to exhaust the memory of modern computers (at $\approx$4 GB/core).
- Adopt WinnerTakesAll strategy: accept only largest prob. history.
- Note: use tool DIY for handling parellization of Pythia 8 runs

# Handling of parton configurations
## I/O is not the forte of HPC machines

- LHEF standard is based on XML: a challenge for I/O, in particular the simultaneous read access when processing event information in heavily parallelized workflows.

- Adopt a new format based on HDF5: designed specifically for processing large amounts of data on HPC machines.
- HDF5 uses a computing model not too dissimilar from databases.
- Datasets can be organized in groups in order to create hierarchical structures.

- Event processing of MG+Pythia is 5 $\times$ faster – just from reading HDF5 over XML

# Example

init:

| Dataset | data type |
|---|---|
| PDFgroupA | int |
| PDFgroupB | int |
| PDFsetA | int |
| PDFsetB | int |
| beamA | int |
| beamB | int |
| energyA | double |
| energyB | double |
| numProcesses | int |
| weightingStrategy | int |

event:

| Dataset | data type |
|---|---|
| nparticles | int |
| start | int |
| pid | int |
| weight | double |
| scale | double |
| fscale | double |
| rscale | double |
| aqed | double |
| aqcd | double |
| npLO | int |
| npNLO | int |
| trials | double |

particle:

| Dataset | data type |
|---|---|
| id | int |
| status | int |
| mother1 | int |
| mother2 | int |
| color1 | int |
| color2 | int |
| px | double |
| py | double |
| pz | double |
| e | double |
| m | double |
| lifetime | double |
| spin | double |

# Lessons and Generalizations

- I/O operations related to the read-in of information related to the construction of the hard matrix elements and to the parameters of the adaptive integrator can be costly
- Unweighting also costly, due to tails in # trials to unweighted event
- Cut efficiencies tend to be non-uniform across various ranks when the number of points per rank is too low
- File size for event storage is determined dynamically. Optimizing the HDF5 output parameters may lead to further improvements

# Some other comments

- Negative weights at NLO reduce significantly statistical precision

  $\sigma^2 = \overline{w^2} - \bar{w}^2 = \frac{1}{4} w^2 f(1-f)$, $f$ = fraction $w < 0$

- GPUs have not really been exploited

  Madgraph LO studies (arXiv:0909.5257, 0908.4403)[2]

  $u\bar{u} \rightarrow (2-8)\gamma$

  5-jet production processes

  Roughly 40 - 150 $\times$ improvement

  NLO codes too big for GPU memory

  Allocations on HPCs will require efficient use of GPUs

[2] K. Hagiwara, J. Kanzaki, N. Okamura, D. Rainwater, T. Stelzer

## Directions for reaching our destination

See arXiv:1712.06982

### Possibilities for Near-term Collaborations

1. new theoretical algorithms
2. reweighting event samples
3. concurrency in phase space evaluation
4. concurrency/thread friendliness in general
5. framework with parallelism built-in from start
6. evaluation of inefficiency in filtering/selection
7. general profiling of codes and sub-codes

# Tuning

## Utilize High Performance Computing resources for HEP problems

- Still need good tunes or perturbative improvements are lost

- Developed parallel workflow for parameter scans at HPC facilities
- Exploit fastMath/MathScience resources at LBL, ANL
- Rational polynomial approximations for building a surrogate function for predictions from fixed number of points in multiple dimensions
- “Smarter” parameter sampling (Latin Hypercube, etc.)
- “Better” minimization techniques (reduce variance, etc.)

Example: 20-D scan of Pythia parameters for LEP

## Summary

- Theory predictions (event generators) are an important part of the *expt'l* HEP program
- More accurate, but also more complicated and expensive
- Improvements require brawn and brains
- Current grid-computing model will be supplemented (replaced?) by HPC facilities
- Near-term projects could have a big impact on long term planning