Event Generators - Status and Plans

Taylor Childers, Stefan Höche

HEP-CCE Kickoff Meeting
9-10 March, 2020

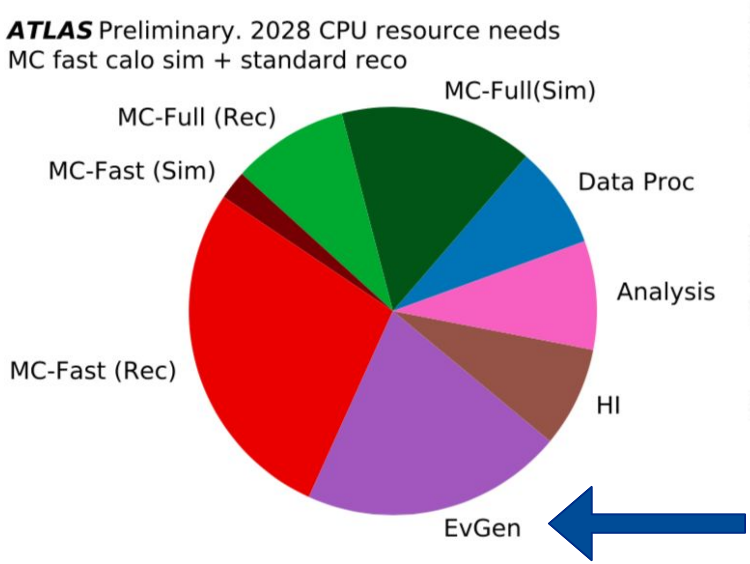
Why improve event generators?

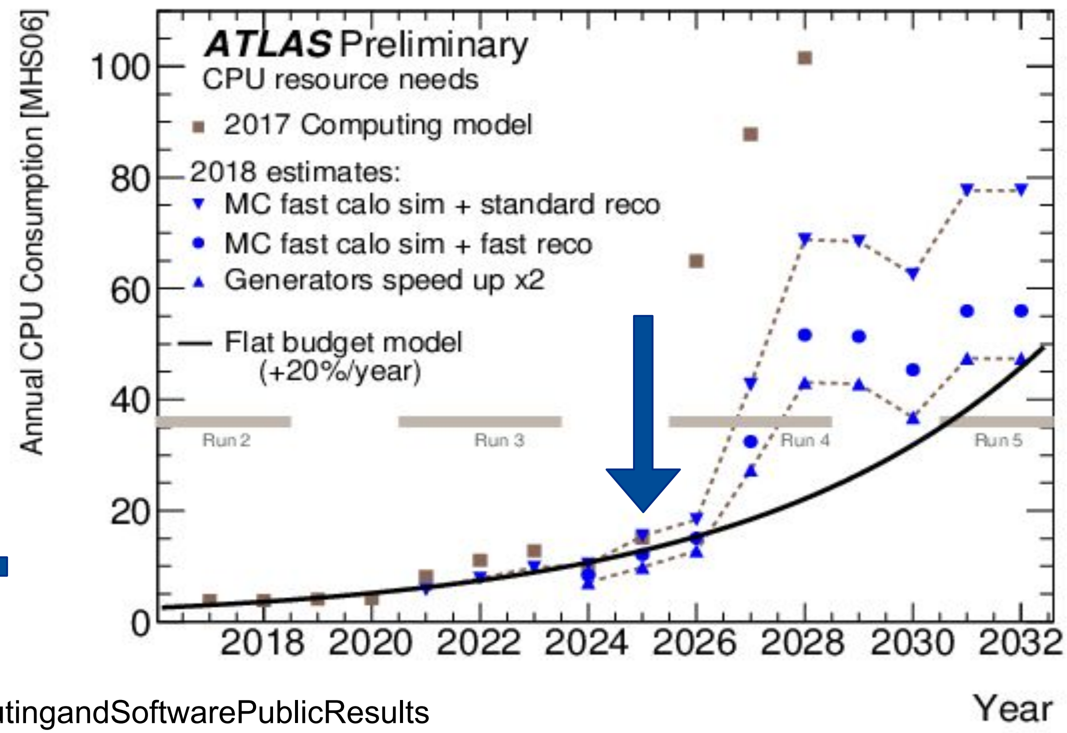
Event generation will consume significant fraction of resources at LHC soon

Need to scrutinize both generator usage and underlying algorithms

Dedicated effort in HEP Software Foundation (HSF)

https://hepsoftwarefoundation.org/workinggroups/generators.html

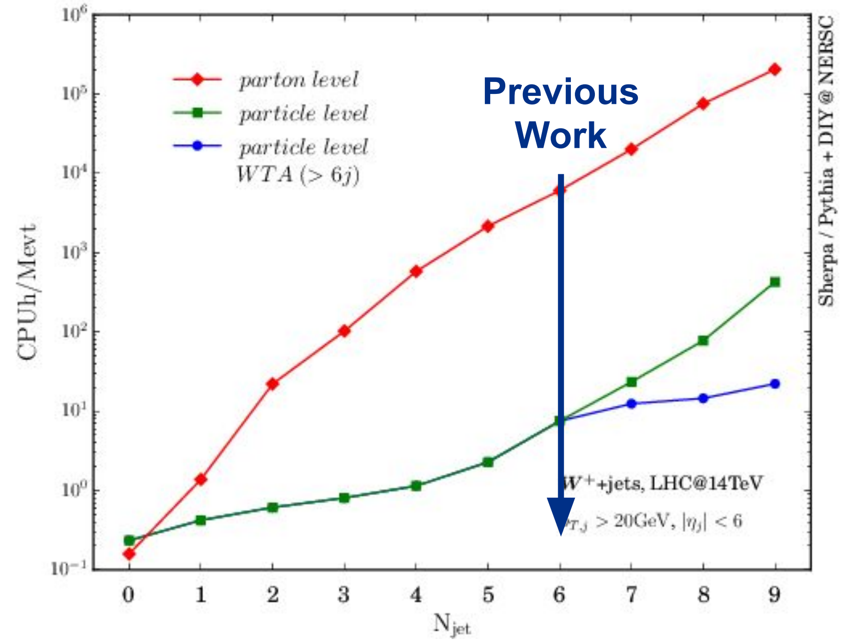


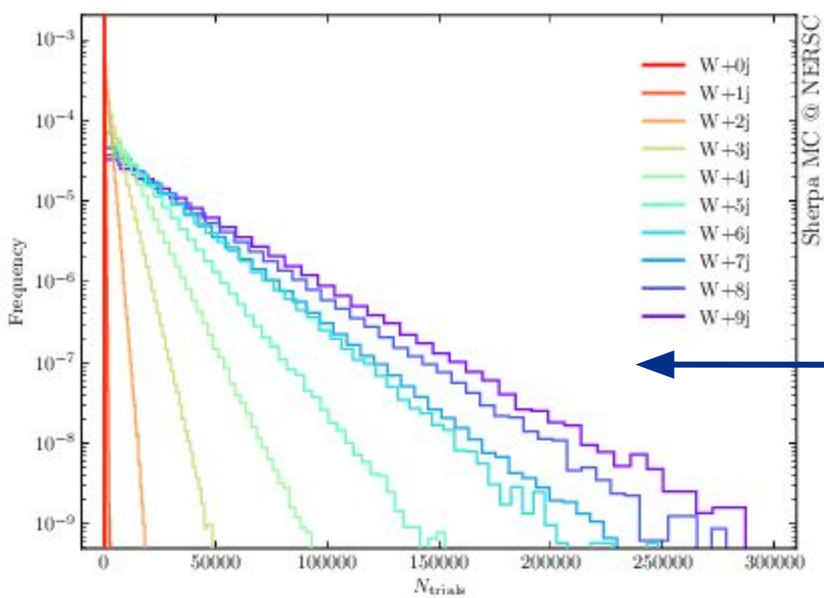


https://twiki.cern.ch/twiki/bin/view/AtlasPublic/ComputingandSoftwarePublicResults

Timing and Scaling

- Hard scattering simulation (ME) much more demanding than parton shower (PS) & hadronization
- Complexity of merging ME&PS grows quickly due to inherent N! scaling of underlying algorithms

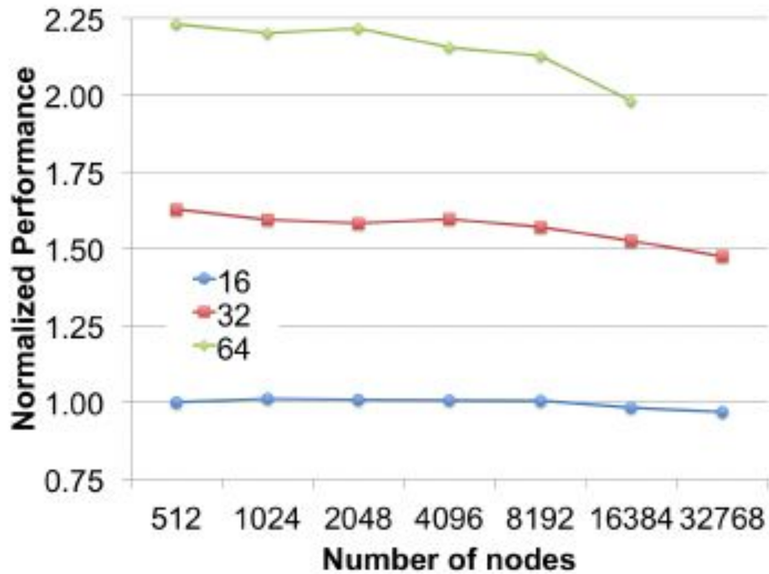


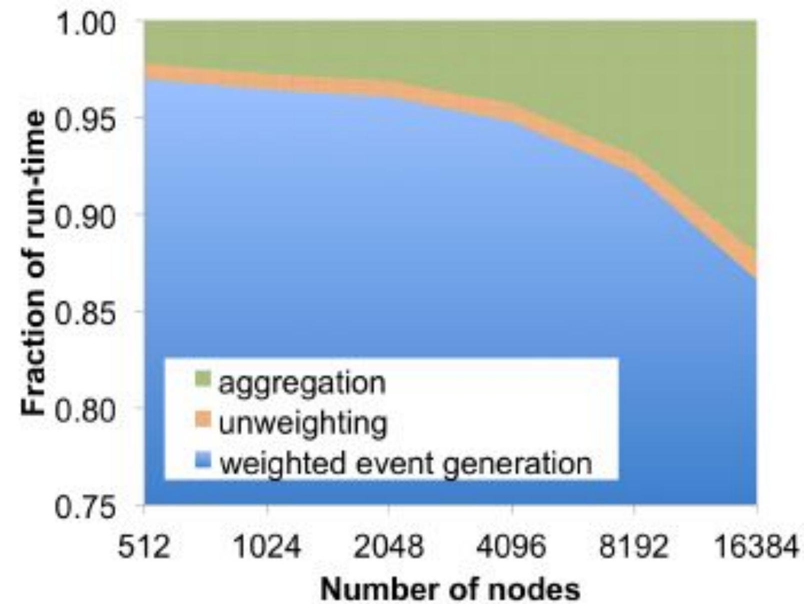


- ME evaluation time naively scales as ~ O(3^N)
- Monte-Carlo unweighting efficiency degrades quickly, as dimensionality of integral is 3N-4
- Overall scaling ~ O(4^N)

Adapting current generators for HPC

T. Childers, T. Uram, T. Lecompte, M. Papka, D. Benjamin arXiv:1511.07312





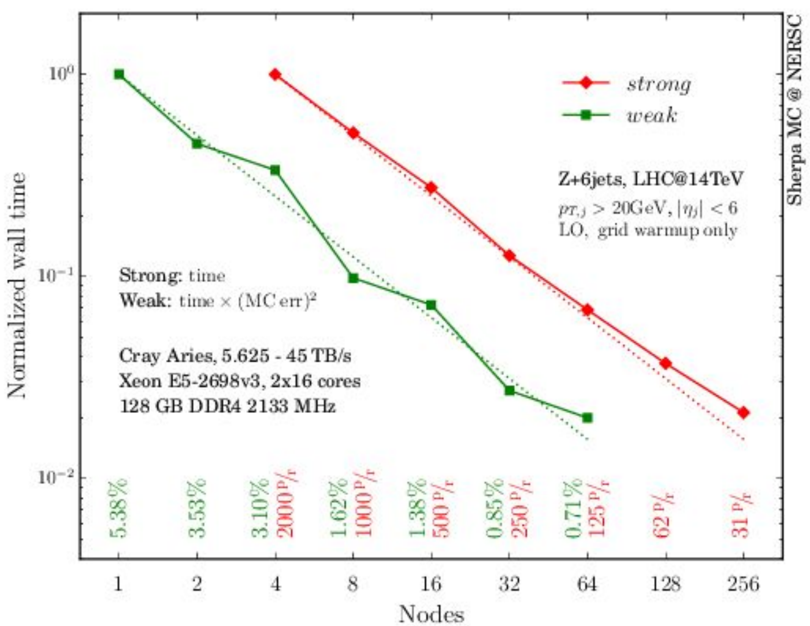
- Parallelized version of Alpgen scaled to millions of threads on Mira
- Delivered large scale data sets to ATLAS for production runs
- Parallelization model adopted by Sherpa+Pythia framework
  ↗️ next slide, but bottlenecks remain there, as well as in MadGraph

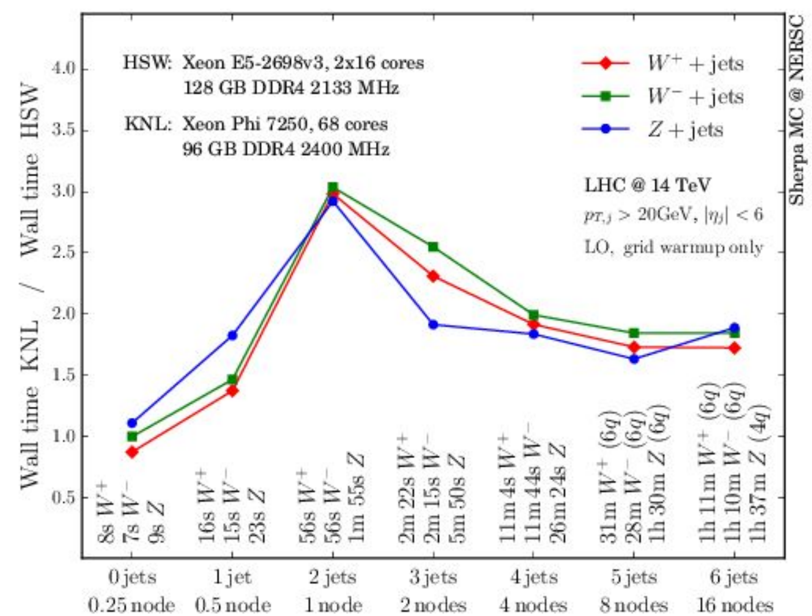
Adapting current generators for HPC

S. Prestel, H. Schulz, SH arXiv:1905.05120

- Separate ME generation from PS evolution and ME+PS merging
- Store intermediate status of event in HDF5 files
- Parallelize event processing at particle level using ASCR's DIY

- Performance limited by number of events being processed per rank
- Scaling of optimization step (strong & weak) up to ~ 2048 cores
- Acceptable (though limited) performance on KNL w/o modifications





Phase-space integration with Normalizing Flows

C. Gao, J. Isaacson, C. Krause, H. Schulz, SH arXiv:2001.10028

- Development of a novel framework for collider event simulation based on Neural Networks and Normalizing Flow algorithms
- Allows to exploit developments in Machine Learning for accelerated numerical simulations at the Large Hadron Collider and future facilities

- Created new open-source toolkit based on Tensorflow that can be used to perform adaptive Monte-Carlo integration in connection with any parton-level event generator
- Exemplified performance in connection with Sherpa framework for the most costly simulations at the LHC → W/Z + jets production at leading and next-to-leading order in QCD perturbation theory

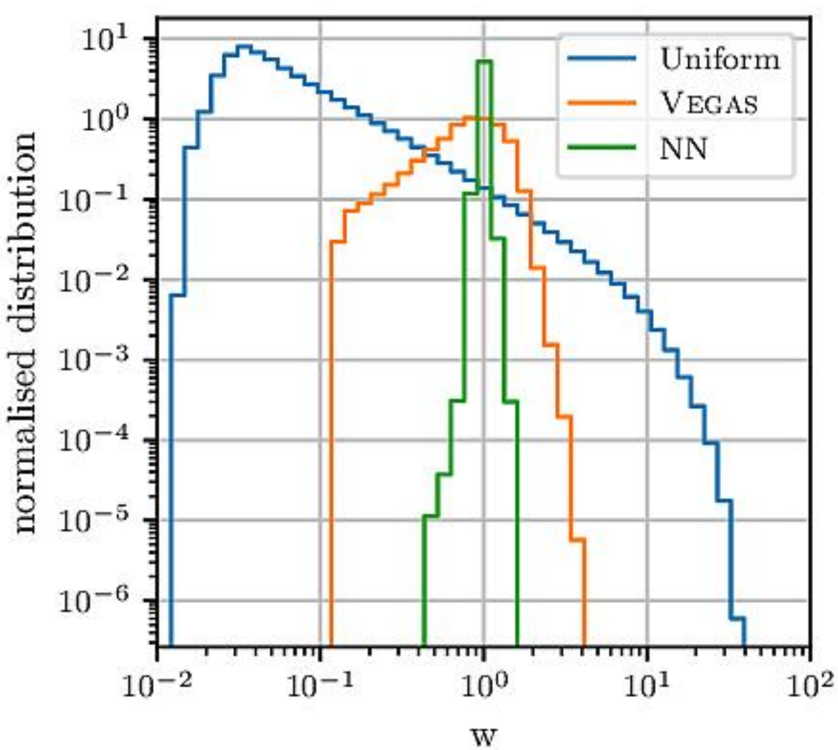
unweighting efficiency		LO QCD					NLO QCD (RS)	
$\langle w\rangle/w_{\max}$		$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=0$	$n=1$
$W^+ + n$ jets	Sherpa	$2.8\cdot10^{-1}$	$3.8\cdot10^{-2}$	$7.5\cdot10^{-3}$	$1.5\cdot10^{-3}$	$8.3\cdot10^{-4}$	$9.5\cdot10^{-2}$	$4.5\cdot10^{-3}$
	NN+NF	$6.1\cdot10^{-1}$	$1.2\cdot10^{-1}$	$1.0\cdot10^{-3}$	$1.8\cdot10^{-3}$	$8.9\cdot10^{-4}$	$1.6\cdot10^{-1}$	$4.1\cdot10^{-3}$
	Gain	2.2	3.3	1.4	1.2	1.1	1.6	0.91

Phase-space integration with Normalizing Flows

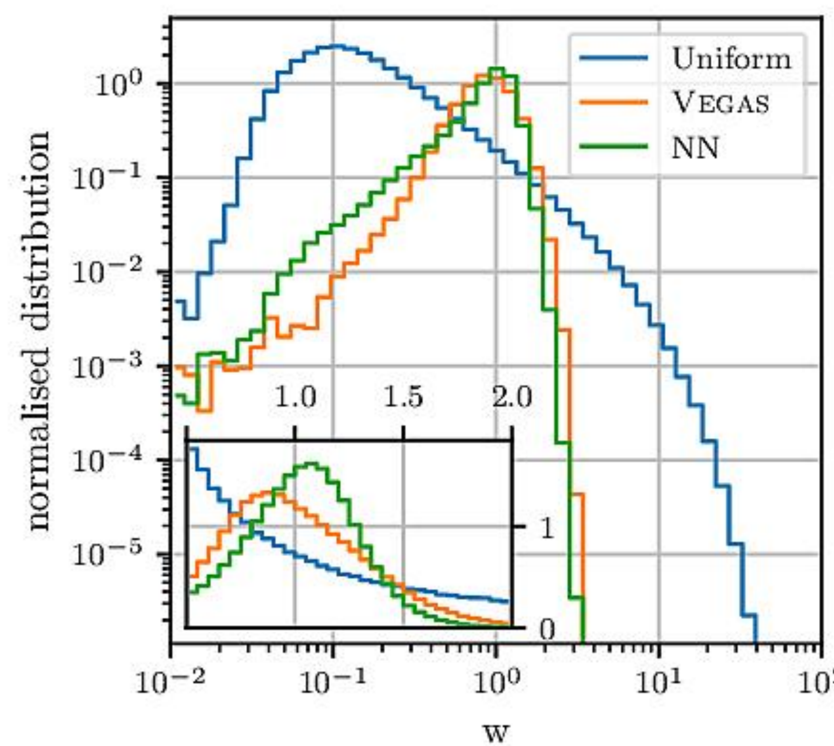
E. Bothmann, T. Janßen, M. Knobbe, T. Schmale, S. Schumann arXiv:2001.05478

- Same developments in parallel by group in Goettingen with same result
- Normalizing flow technique promising at low multiplicity, but not enough computing power to sufficiently train NN in interesting scenarios

Partonic 3-gluon production

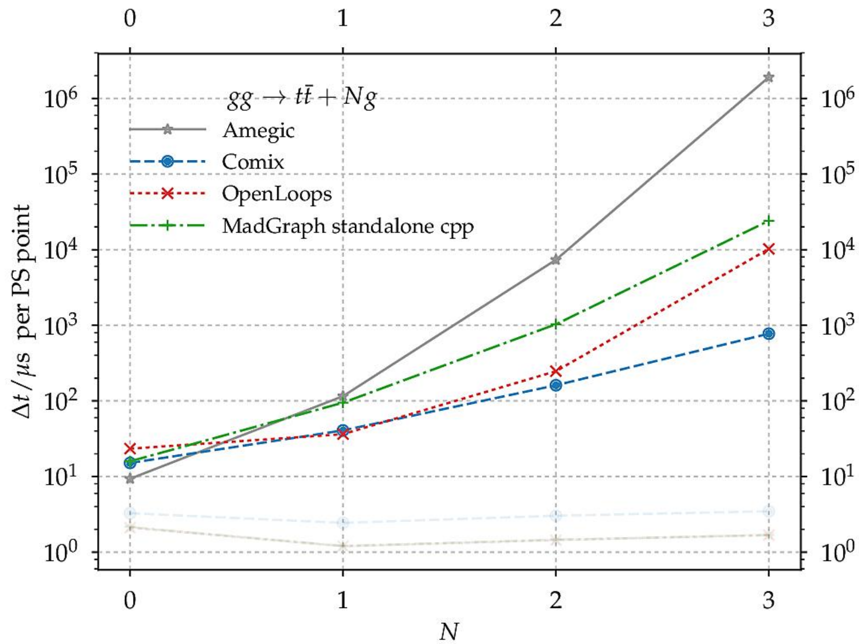


Partonic 4-gluon production



The road ahead

- Restructured modern evgen frameworks working at LO, still being developed at NLO
- Continued SciDAC work on MC integration efficiency, but
- Further improvements likely require rewrite of ME generator for different architectures



- Past efforts in MadGraph and new HEP-CCE project
- Plans to explore both low-level and high-level languages
- Likely to restructure computation to use sub-optimal algorithm vz scaling to gain higher performance for practical cases (W+<=6j)
  ↗️ next slide for details

Fermilab

The road ahead

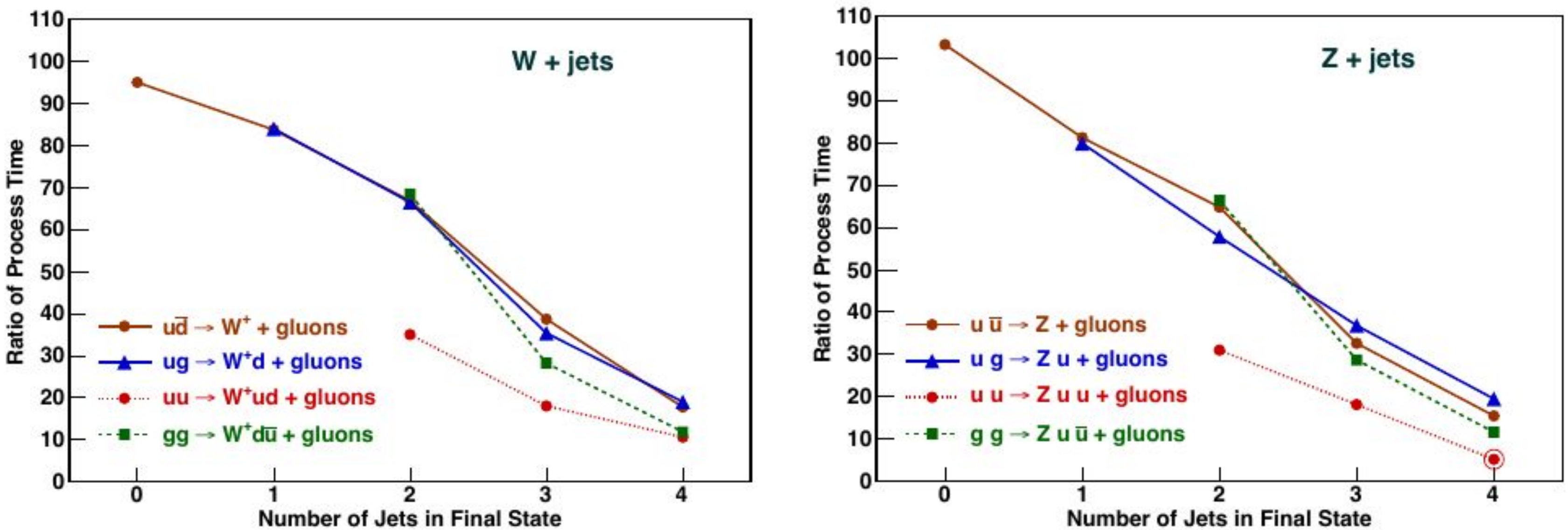
C. Duhr, F. Maltoni, SH hep-ph/0607057

Final State	BG		BCF		CSW	
	CO	CD	CO	CD	CO	CD
$2g$	0.24	0.28	0.28	0.33	0.31	0.26
$3g$	0.45	0.48	0.42	0.51	0.57	0.55
$4g$	1.20	1.04	0.84	1.32	1.63	1.75
$5g$	3.78	2.69	2.59	7.26	5.95	5.96
$6g$	14.2	7.19	11.9	59.1	27.8	30.6
$7g$	58.5	23.7	73.6	646	146	195
$8g$	276	82.1	597	8690	919	1890
$9g$	1450	270	5900	127000	6310	29700
$10g$	7960	864	64000	-	48900	-

- Current best algorithm based on Color Dressed Berends-Giele recursive relations (a type of dynamic programming technique)
- Provably the algortihm with best scaling, but suffers from dynamic nature: many branch points and allocation / deallocation overhead
- Current idea: return to less dynamic algorithm with worse scaling: Color Ordered Berends-Giele or color summed Berends-Giele

The road ahead

K. Hagiwara, J. Kanzakia, Q. Lib, N. Okamura, T. Stelzer arXiv:1305.0708



- Existing GPU implementation of HELAS routines and MadGraph that can be used as a guideline for future development
- Performance improvement likely enough to match experiments' computing needs but currently not working in production
- Maybe quickest solution is to port this to newer GPU architectures?

Summary and Outlook

Improvements under SciDAC

- Parallelized Pythia particle-level event generation using DIY
- Improved performance of Sherpa, particularly I/O (HDF5)
- Optimization & event generation now scale up to ~2k cores
- First novel results not previously attainable on WLCG
- Novel integrator using Neural Networks and Normalizing Flows

Plans for CCE

- Complete rewrite of matrix-element generator for CPUs & GPUs
- Paradigm shift from “best for scaling” to “best for computation”
- Exploration of different frameworks (low-level, Kokkos, …)
- Alternatively try to revive existing implementations (HELAS/MG)
- In either case, learn from work on ALPGEN here at ANL
- Coordinate with HSF and worldwide efforts