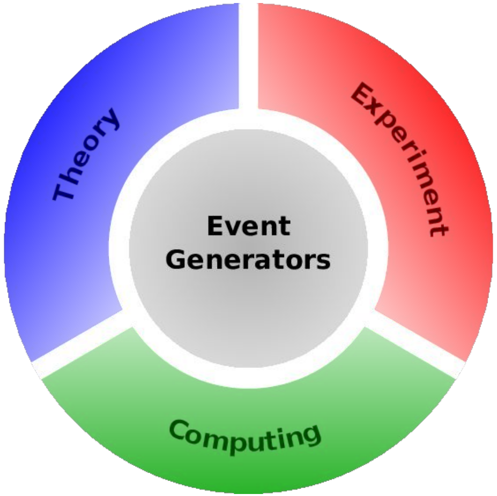
Event Generators Working Group Status Update

J. Taylor Childers (ANL), Stefan Höche (FNAL) (co-convenors),
Enrico Bothmann (Goettingen), Walter Giele (FNAL), Josh Isaacson (FNAL),
Max Knobbe (Goettingen), Olivier Mattelaer (UCLouvain), Nathan Nichols (ANL),
Stefan Roiser (CERN), Andrea Valassi (CERN), Rui Wang (ANL)

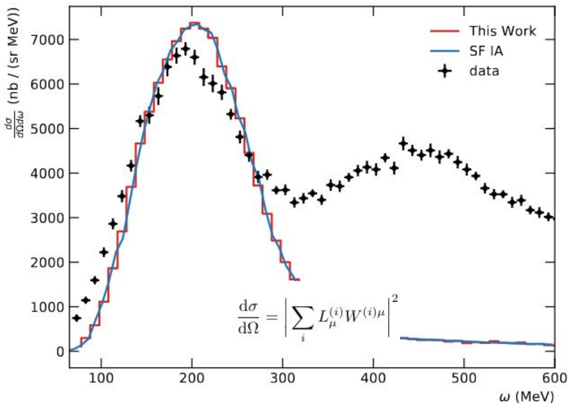
Status of Event Generators [arXiv:2203.11110]

- Overview of status of Monte-Carlo event generators for HEP
  - Contributions from 187 authors at 102 institutions
  - Guided by experimental needs and requirements
- Physics models and computing algorithms employed across facilities and experiments
  - Higher-order QCD & EW perturbative corrections
  - Factorization theorems & parton evolution equations
  - Resummation of QCD & QED effects
  - Hadronization & final-state modeling
  - New physics simulations
- Detailed discussion of MCs for facilities relevant to DOE HEP
  - High Energy Colliders (LHC, μ, …)
  - Neutrino Experiments (Dune, HK, …)
  - Electron-Ion Collider
  - Forward Physics Facility
  - Future Lepton Colliders (FCC-ee, ILC, …)



Status of Event Generators arXiv:2203.11110

- Generators traditionally developed in context of high-energy colliders (LEP, LHC, …) can provide blueprints & components for frameworks at other experiments. For example
  - Novel neutrino generator arXiv:2110.15319
  - Generators for EIC, discussed at MC4EIC workshop, Nov 2021 https://indico.bnl.gov/event/13298/



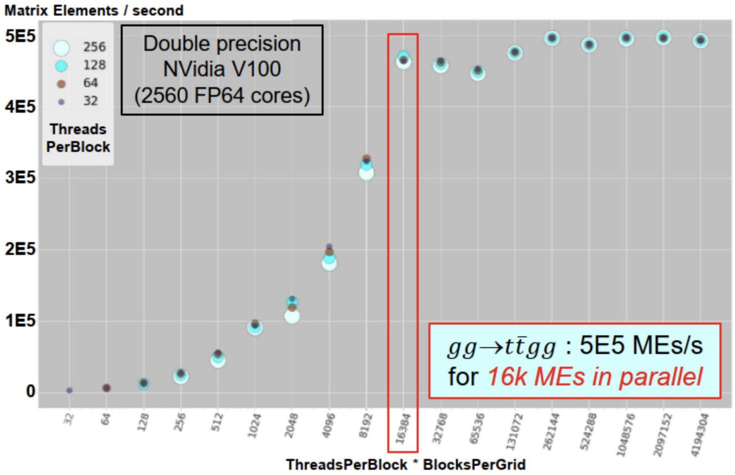
- Event generators are part of a larger ecosystem of tools
  - Interfaces: HepMC, LHEF, LHAPDF, …
  - Analysis & data preservation frameworks: Rivet, Nuisance, …
  - Tuning tools: Professor, Nuisance, …
  - Machine learning techniques: See dedicated document arXiv:2203.07460

Status of Event Generators arXiv:2203.11110

- Computing performance and portability play a key role at LHC arXiv:2109.14938
- Driven by push to high precision simulations in absence of clear signs for new physics
- Main aspects
  - Profiling and benchmarking
  - Efficient phase-space sampling
  - Reduction of negative weights
  - Reweighting for uncertainty estimates
  - Porting software to modern hardware: GPUs and vectorization on CPUs
- Current HEP-CCE activities
  - Portable LO matrix element generators
  - Testing new architectures
  - Negative weight reduction

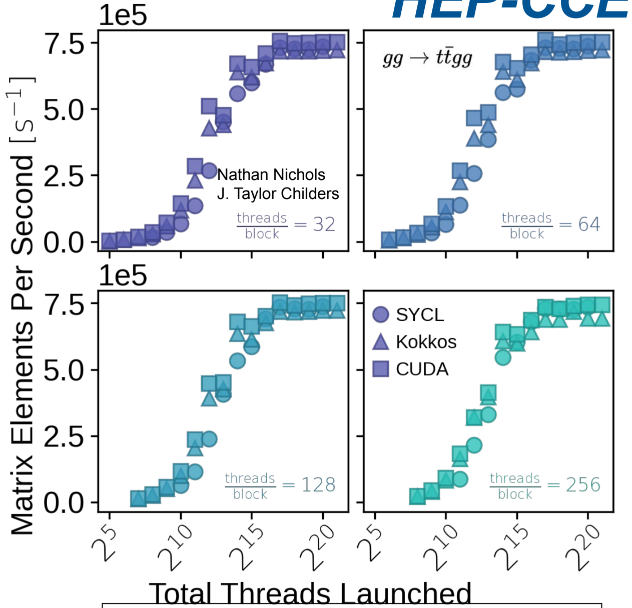
Platform Independent Madgraph5_aMC@NLO

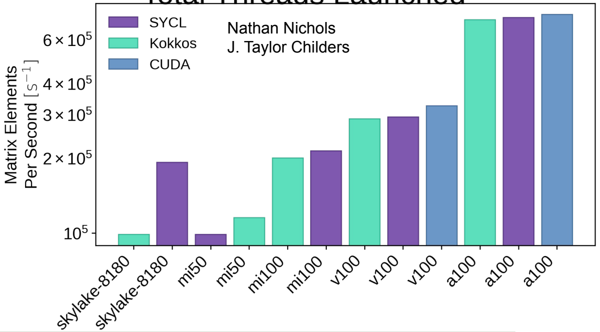
- MadGraph is one of the primary event generators used by ATLAS & CMS and will be responsible for generating a large portion of simulated collisions in the next decade.
- MadGraph uses a python framework that generates process specific compiled codes, originally all FORTRAN based.
- Recently extended to generate serial C++ matrix element calculations called by “MadEvent” which is still FORTRAN
- This code generation framework comes with a PLUGIN capability through which support for other languages and, in our case, portability frameworks can be implemented.
- This enabled the first port of MadGraph to an accelerator [ref], with some performance measures shown here



Platform Independent [Madgraph5_aMC@NLO]

- HEP-CCE is adding backends that use Kokkos and Sycl, broadening reach to architectures in future Exascale machines.
- This effort benefits from an Aurora Early Science Project award received by the ATLAS group at Argonne which provides early access to Intel GPUs and support from Intel/LCF experts.
- These results show matrix element calculations per second for the process *gg* → *tt* + *gg* on NVidia A100 and V100, Intel Skylake CPUs, and AMD MI-50 and MI-100.
- The results show Kokkos & Sycl both perform within 10% of the native CUDA codes, but offer the ability to run on many other systems without rewriting your software.
- We have run on the latest Intel GPUs targeted for Aurora and seen promising results that are currently under NDA.
- The codes are being developed here, still a work in progress: https://github.com/madgraph5/madgraph4gpu

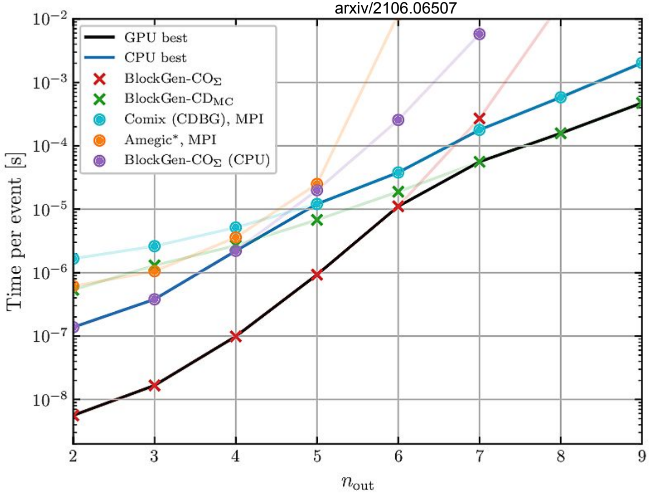




BlockGen - A prototype parton-level event generator for GPUs

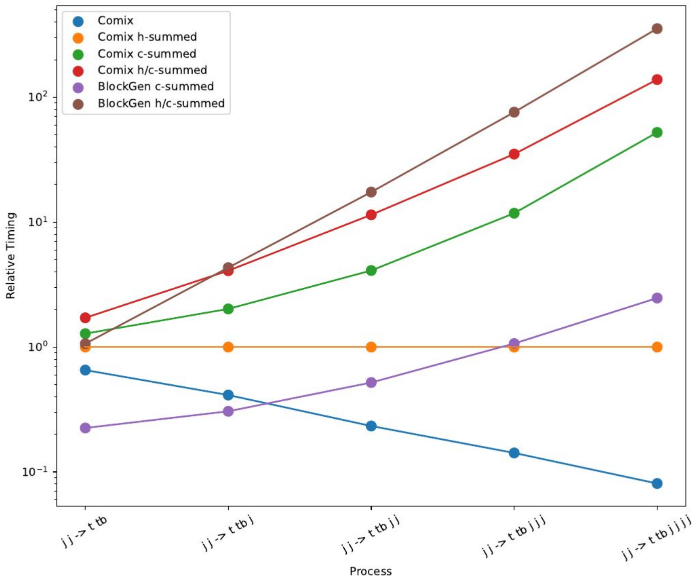
- Matrix element generators like MadGraph use most of the computing time in precision event simulations for the LHC. Providing a scalable solution is important for HL-LHC
- First prototype generator for high multiplicity calculations on GPUs, including multiple algorithms which can be dynamically switched to achieve optimal performance.
- Timing studies in comparison to existing CPU codes (MadGraph, Comix, Amegic) show factor 10+ speedup at low particle multiplicity, factor ~4 at high multiplicity.
- https://arxiv.org/abs/2106.06507

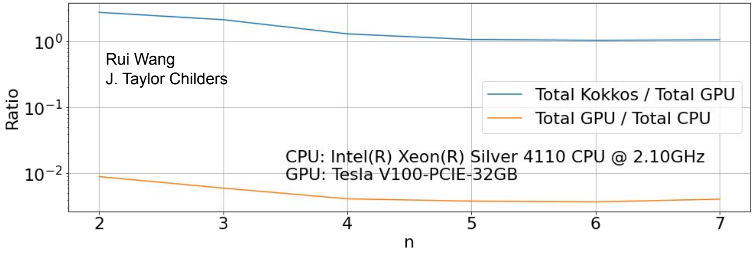
arxiv/2106.06507



BlockGen - A prototype parton-level event generator

- Ported BlockGen code from C++ to Kokkos
  All-gluon version fully functional
- Profiled resulting code and found agreement with dedicated CUDA version in arXiv:2106.06507
- Fermionic currents and phase-space generator being added to code → extension to general-purpose MC
- Going forward, profiling of phase space generator will become a focus point, results from Sherpa point towards a major bottleneck in this component





Computing performance improvements in Sherpa

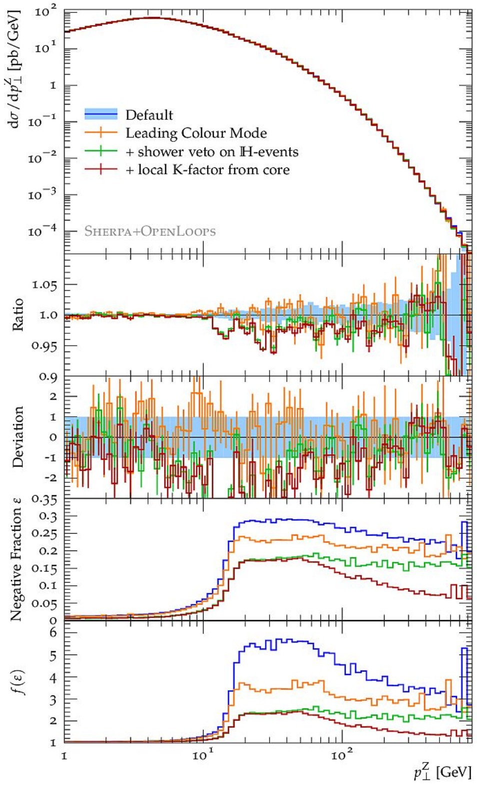
- Sherpa is one of the leading LHC event generators
  Used mostly by ATLAS, somewhat by CMS
- Reduced computational footprint while keeping formal precision targets arXiv:2110.15211, arXiv:2112.09588
  - Negative weight fraction a driver of computing requirements:
    For fraction of neg. weights *f* one needs to generate $1/(1-2f)^2$ as many events to achieve same statistical precision as sample with purely positive weights
  - Can reduce negative weights through controlled approximations

	Negative Weight Fraction
Default	18.1%
Leading Colour Mode	14.0%
+ shower veto on ℍ-events	9.6%
+ local K-factor from core	9.1%

Negative weight fraction in Z+jets

	Negative Weight Fraction
Default	24.8%
Leading Colour Mode	18.7%
+ shower veto on ℍ-events	14.5%
+ local K-factor from core	12.6%

Negative weight fraction in tt+jets



- Not a long-term development, but already used by experiments!

Summary and Outlook

- Current HEP-CCE activities
  - Portable LO matrix element calculators (MadGraph5, BlockGen)
  - Testing new architectures
- Next-to-leading order calculations a natural next target
  - Frameworks are more complex than LO and present some unique challenges
    - Quadruple precision arithmetic
    - Complicated phase space integrals
    - Increased memory consumption
- Important to keep optimizing existing frameworks
  - Low hanging fruit sometimes found by detailed profiling
  - New opportunities, e.g. ML-assisted unweighting arXiv:2109.11964