Large-scale distributed usage of NLO and multi-leg Monte Carlo event generators on the grid and at Argonne Leadership Computing Facility

Josh Bendavid (Caltech)
On behalf of the CMS collaboration

Caltech

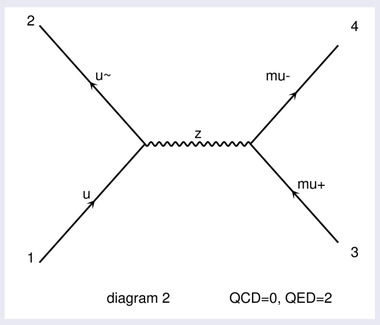
Oct. 10-14, 2016, CHEP, San Francisco USA

Monte Carlo in CMS

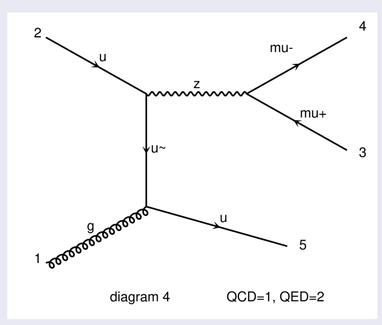
- CMS relies on detailed and large scale Monte Carlo production for modeling of the detector and underlying physics
- At LHC additional QCD radiation is prevalent
- Many measurements or searches explicitly select or veto on the presence of extra jets
- Strong motivation for NLO and/or multi-leg/merged-multiplicity Monte Carlo generators in order to achieve highest possible accuracy for final states with additional jets
- Typical Monte Carlo workflow has a few distinct steps:
  - **Hard process/Matrix Element generation:** Generating kinematic configuration for a desired process up to parton level using perturbative QCD.
  - **Parton Shower/Hadronization**: Adds additional QCD and QED emissions down to a low energy scale, and produces hadrons from QCD partons.
  - **Detector Simulation:** Detailed Geant4 simulation of the interactions of the outgoing particles with the CMS detector.
  - **Digitization:** Simulation of detector electronics and creation of simulated raw data.
  - **Reconstruction:** Reconstruction of simulated raw data into higher level physics objects. (To a good approximation, identical code as runs on real data.)

Multi-leg generators

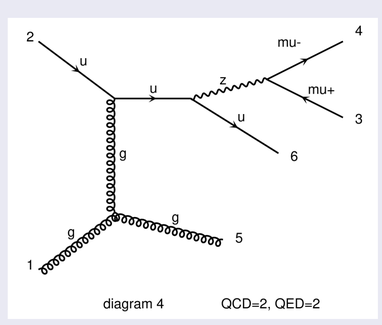
- Several Monte Carlo generators have the capability to automatically generate matrix elements at LO for several jet multiplicities
- Results can be **consistently combined** across multiplicities when treated consistently in parton shower (jet matching/merging).
- Most widely used configuration in CMS Run 2: Madgraph_aMC@NLO (LO) + Pythia 8 with MLM matching
- Most complex processes with up to 4 additional jets:



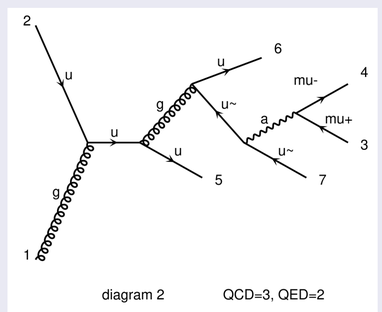
(a) DY+0-jet



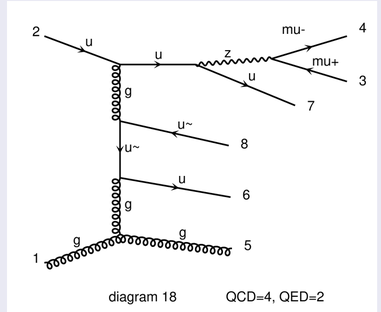
(b) DY+1-jet



(c) DY+2-jet



(d) DY+3-jet



(e) DY+4-jet

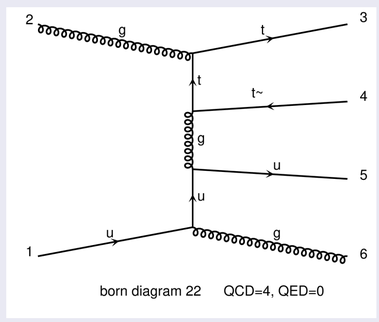
- CPU time up to about 10s per matrix element event (averaged over jet multiplicities), ∼10% matching efficiency at the parton shower step → 100s cpu for matrix element per fully-simulated event

Gridpacks

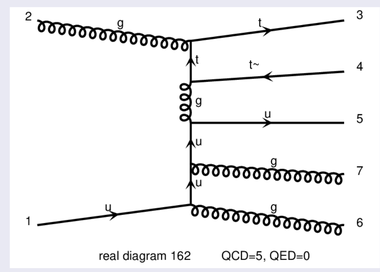
- Matrix element generators typically have two discrete steps:
  1. Matrix element/code generation and phase space integration
  2. Generation of events
- For efficient generation of events on the grid, compiled code and the results of the phase space generation are stored in **gridpacks**, largely self-contained tarballs which are prepared in advance and stored on CVMFS
- Gridpack preparation so far done with single machines or with batch-job parallelism (LSF or Condor)
- Subsequent grid jobs read the gridpacks from CVMFS and generate events with trivial process-level parallelism

NLO Multi-leg generators

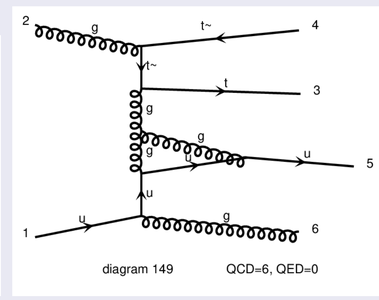
- A few Monte Carlo generators now have the capability to (semi)-automatically generate matrix elements at **NLO in QCD for several jet multiplicities** and consistently merge them
- Most widely used configuration in CMS Run2: Madgraph_aMC@NLO (NLO) + Pythia 8 with FxFx merging
- At NLO in QCD, **each multiplicity** consists of **born**, **real**, and **virtual** contributions to the matrix element
- Most complex processes with up to two additional jets at NLO



(a) tt+2-jet born



(b) tt+2-jet real

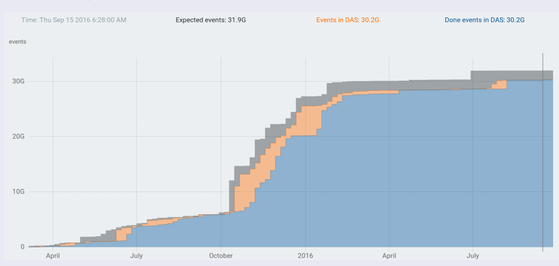


(c) tt+2-jet virtual

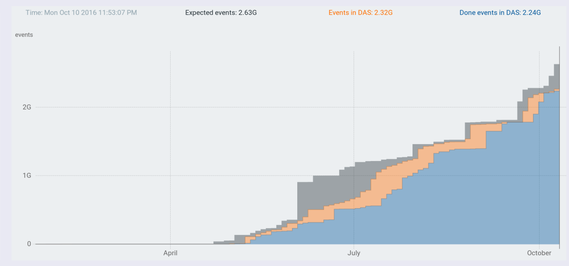
- CPU time scaling up to to ∼ 30 s per ME event with matching efficiencies of ∼ 30% → 90 s of cpu time for matrix element per fully-simulated event
- Events are also accompanied by a possibly large fraction of negative weights (up to 40%) which reduces statistical precision and necessitates larger samples
- Diagram/code generation also very CPU and memory intensive → recent contribution to 2.4.x series to significantly improve (thread-level) parallelization and memory footprint of this step, eventually enabling more complex processes

CMS Software Architecture and Production Infrastructure

- Event generators which perform the parton shower step (Pythia, Herwig, Sherpa) are fully integrated as "externals" and packaged with CMSSW/directly linked from CMSSW interfaces
- Matrix element generators which generate LHE files (Madgraph, POWHEG, etc) are only loosely coupled, tarball extraction from CVMFS and calling of external generation script handled by an integrated CMSSW module "externalLHEProducer"
- ascii LHE files are transient and are immediately packed into binary format and compressed.
- Generation of Matrix Element events is configured and executed as a standard CMSSW job, and is therefore fully integrated with CMS Production Infrastructure
- Over 30B LHE events (before matching) produced in initial Run 2 production campaign



(a) LHE produced as a separate step (number of events is before matching)



(b) LHE produced together with showering/simulation (number of events is after matching)

Exploiting Large Parallel Computing Systems

- Several important limitations in particular in the gridpack generation step:
  - Madgraph_aMC@NLO does not presently support MPI parallelization → poor balancing of work between jobs, not straightforward or efficient to run on large MPI-oriented clusters
  - Sherpa currently supports mainly MPI-level parallelization → not possible to run complex phase space integrations on conventional batch resources widely available to the collaboration
- Work-in-progress to exploit MC generators on large parallel computing facilities
- First test case is Sherpa on ALCF Mira system → already significant improvements to efficiency/parallelization in 2.2 series coming out of previous/ongoing work from ATLAS
- Interest together with authors to extend range of parallelization options in Madgraph_aMC@NLO as well
- Ongoing work to improve the efficiency of underlying phase-space integration/generation with machine learning techniques