Monte Carlo event generation

MC event generators provide vital inputs to the modern particle physics programme. This is particularly true for current and future collider-based experiments with their need for theoretical predictions at the highest precision. The current emphasis in their deployment at the LHC is driven by the ubiquity of QCD effects, which impacts on practically all analyses: for many LHC measurements the QCD uncertainties are larger than the anticipated experimental ones, and many analysis signatures are dominated by irreducible, QCD-induced backgrounds, in which theoretical predictions are key for statistical extrapolation of background processes into the signal regions. The implementation of precision calculations in generators capable of predicting complete final states is important because of the need to compare state-of-the-art theory to precision differential cross section measurements in a fiducial phase space reflecting the acceptance of the detector.

Demands from future colliders

Already, Runs 1 and 2 of the LHC have seen the computational demands for event generation rise from a negligible fraction of the total LHC experiment CPU budget, to the point where complex process generation may be the slowest part of the entire processing chain, even after pre-generation of the most arduous phase-space and matrix element components. The demand for event complexity is set to rise further, due both to the HL-LHC emphasis on higher-multiplicity electroweak final states (e.g. di-Higgs, 4-top, and tri-boson events), and the continual demand for higher theoretical precision.

The event generator community is in the process of moving toward matching of NNLO QCD matrix elements with parton showers, and as a consequence the CPU demand of such calculations will undoubtedly be even higher compared to current precision simulations. Furthermore, such high-precision simulated events will be needed in unprecedented volumes, to maintain parity with the experimental data statistics of the HL-LHC. Experimental data rates will be made manageable by tightening of trigger and experimental cuts; equivalent cuts on MC generator kinematics and flavour configurations are far from free and will further push up the MC generation demands on available CPU. It goes without saying that for event simulation to be the critical bottleneck in exploitation of the HL-LHC would be a great waste of the experimental facility.

The simulation of realistic final states requires the combination of these higher-order theoretical precision calculations with soft and non-perturbative effects. Event generators are also used as tools to study these complex QCD phenomena in their own right, and increasing interest in heavy ion collisions and high-multiplicity proton collisions brings further computational demands. However, for the purposes of this report we focus primarily on higher-order theoretical precision calculations within event generators, since this is the most CPU intensive aspect.

Theoretical prospects and problems

The step-change in CPU demand for event generation (as opposed to detector simulation) during the first two LHC runs has principally been driven by the advent of merging of matrix element and parton shower calculations at next-to-leading order (NLO) in the QCD coupling. The NLO matrix elements provide a degree of theoretical precision and stability essential for LHC analyses, but for the degree of realism required by detector simulation these calculations must be connected to the parton shower algorithms that generate high final-state particle multiplicities. As many LHC analyses further require consideration of event types with many hadronic jets, amplitudes with many leading-order QCD emissions are also merged and matched into these complex event simulations. The raw computational complexity of NLO amplitudes, combined with many-body phase-space evaluations and the inefficiencies of the matching process has led to the swollen CPU budget for event simulation. Future developments in event generation theory are likely to exacerbate this problem — in exchange for further improvements in precision — with e.g. NNLO amplitudes also used in ME/PS matching.

While event generators are an essential component in the experimental analysis process, they are predominantly developed by the theoretical physics community. This presents a particular challenge to align the need for improvements in computational efficiency with the incentives of theoretical work: the ability to successfully bid for theory grants by publishing journal papers, for example. Any proposal to improve the computational efficiency should acknowledge and address this challenge, for example giving a mechanism by which theoretical advances continue to be rewarded, while supplementing them by complementary computational advances, rewarded in an appropriate way.

The general purpose event generator projects, which supplement matrix element calculations with parton showering, hadronization and soft physics effects, have organised their efforts into a network, MCnet, which has succeeded in expanding the size of this community, through dedicated PhD studentships and by pooling resources on common problems. Nevertheless, the long-term job prospects of those PhD students rest largely on their ability to produce theoretical developments, not on the efficiency of their implementations. It is not that MC generators are intrinsically inefficient, but that the cost of any inefficiencies falls almost entirely on the experiment “users” and that there is little current incentive for generator development teams to devote major effort solely toward experiment support and performance optimisation (which do not produce journal papers).

Looking at the various levels of optimization opportunities, the one that is most interesting theoretically is the development of new or improved algorithms. This however is also the most uncertain avenue to deliver improvements in a given timeframe. Dedicating many more people to the same problem will typically not help much here.

At a more technical level, the situation is different. Concurrency and parallelization are avenues that have yet to be explored in depth for event generation (beyond pure MC integration of matrix elements, where GPGPU approaches can offer large benefits). Given

that event generation can be trivially parallelized over individual events, all available CPU resource can be fully utilized by splitting the required load, provided end effects are minimal. However, some higher-order generators have significant initialization phases, which are less easily parallelized, but there has so far been no strong incentive to investigate these directions further.

Another underexplored avenue is the efficiency of event generation as used in the experiments. An increasingly common usage is to generate very large inclusive event samples, which are filtered on event final-state criteria to decide which events are to be retained and passed on to detector simulation and reconstruction. This naturally introduces a large wastage fraction of very CPU-expensive event generation, which could be reduced by an emphasis on filtering tools within the generators themselves, designed for compatibility with the experiment requirements. A particularly wasteful example is where events are separated into orthogonal sub-samples by filtering, in which case the same large inclusive sample is generated many times, with each stream filtering the events into a different group: significant CPU resources may be recoverable on the experiment side by allowing a single inclusive event generation to be filtered into several orthogonal output streams, avoiding the expensive duplication.

Additional effort dedicated mainly to these software engineering considerations would have a major positive impact on the experimental use of the generators. However, the incentives do not align well to encourage direct hire of such people on the theory grants that support MC development, and experience has shown that close embedding/ integration in the MC developer teams is essential to success. Some longevity is also required — one-off, “drop-in” technical assistance with MC development has not generally made a lasting impact.

Proposals and discussion points

As the MC projects are funded mainly to develop theoretical improvements, and not mainly as “suppliers” to the experimental HEP programme, any strong requests towards efficiency improvements from the experimental community would need to be backed up by plausible avenues of support.

A model that has been successful in the past to foster interaction between the communities is the MCnet short-term studentship programme, where interested experimental PhD students join a generator group for several months to work on improving a physics aspect of the simulation that is relevant for their work, or to improve the integration of the generator into the experimental framework. A similar matchmaking scheme could focus on the software engineering side, and transfer some of the expertise available in the experiments to the generator projects. Whether improvement can be delivered by graduate students “learning on the job” and then leaving after a few months is an open question: such studentships have been fundamentally viewed as being for student training and relationship-building as much as for the resulting technical development. To meet the requirement of transferring technical expertise and effort from the experiments would likely

require placements for experienced optimisation specialists rather than students, and a medium/long-term connection to the generator project.

Hence we also consider another useful model, found in the development of free/open tools, such as the Linux kernel or the GNU tools. Core contributors to these volunteer projects are funded full-time by companies that rely heavily on them, in the spirit that improvements in common tools provide value that justifies direct investment by all stakeholders. For HEP, the equivalent of a large corporation is the LHC experiments who have large reserves of researchers, including a significant number of software specialists. This model would also be beneficial for core HEP tools like LHAPDF, HepMC and Rivet, where future improvements have no theoretical physics interest anymore, putting them in a similar situation to generator performance improvements. The structural issue blocking such a mode of operation is that experiments do not currently count contributions to external projects as experiment service work — a situation perhaps deserving of review in areas where external software tools are critical to experiment success.

Context and mandate

Chat with Graeme clarified aim of document contribution:

identify problems facing Generator community, emphasis on problems for long-term LHC program, future experimental use. Computational scaling issues.

Other HSF docs, for reference:

- https://www.overleaf.com/read/wyyybnvxyfyn#/36111595/
- http://hepsoftwarefoundation.org/activities/cwp.html
- https://docs.google.com/document/d/1rcPIJQc3LNAh5tjHKjfuq80StrMO5ksiLwhDlJzeg9U/edit#heading=h.wq4wngymp8nh