Proceedings of the DIS'2004, Štrbské Pleso, Slovakia

THEPEG, PYTHIA7 AND ARIADNE

LEIF LÖNNBLAD
Department of Theoretical Physics, Lund University
Sölvegatan 14A, S-223 62 Lund, Sweden
E-mail: leif.lonnblad@thep.lu.se

I describe the status of the THEPEG project, a platform for implementing the next generation of event generators in C++. I also discuss some recent developments of the matching of fixed-order tree-level matrix elements with the dipole cascade in ARIADNE.

1 Introduction

Monte Carlo Event Generators (EGs) have developed into essential tools in High Energy Physics. Without them it is questionable if it at all would be possible to embark on large scale experiments such as the LHC. Although the current EGs work satisfactorily, the next generation of experiments will substantially increase the demands both on the physics models implemented in the EGs and on the underlying software technology. In this talk I will discuss two projects addressing each of these aspects.

The current EGs are typically written in Fortran and their basic structure was designed almost two decades ago. Meanwhile there has been a change in programming paradigm, towards object oriented methodology in general and C++ in particular. This applies to almost all areas of high-energy physics, but in particular for the LHC program, where all detector simulation and analysis is based on C++. When designing the next generation of EGs it is therefore natural to use C++. Below I will briefly describe the THEPEG [1] project for designing a general framework in C++ for implementing EG models, and PYTHIA7 which is a complete reimplementation of the Lund family of EGs using that framework.

LHC is, of course, mainly a machine for discovering new physics. But irrespectively of what new phenomena may exist, we know for sure that the LHC events will contain huge numbers of hadrons, and that a large fraction of these events will have many hard jets produced in standard QCD processes. Such events are interesting in their own right, but they are also important backgrounds for almost any signal of new physics. Unfortunately the standard Parton Shower (PS) based EGs of today are not well suited to describe events with more than a couple of hard jets. The alternative is to use fixed-order matrix element (ME) generators, which typically can generate up to six hard partons. But these generators are not well suited for describing the conversion of these hard partons into jets. Several attempts to combine PS and ME generators have been presented, and below I will describe how the PS generator ARIADNE [2] is interfaced to the ME generator MADGRAPH [3] to simulate W production with several associated hard jets in hadronic collisions, possibly the single most important background to signals of new physics at LHC.

2 ThePEG

ThePEG is a general platform written in C++ for implementing models for event generation. It is made up from the basic model-independent parts of Pythia7 [4,5], the project of rewriting the Lund family of EGs in C++. When the corresponding rewrite of the herwig++ program[6] started it was decided to use the same basic infrastructure as Pythia7 and therefore the ThePEG was factorized out of Pythia7 and is now the base of both Pythia7 and herwig++ [7].

ThePEG uses clhep [8] and adds on a number of general utilities such as smart pointers, extended type information, persistent I/O, dynamic loading and some extra utilities for kinematics, phase space generation etc.

The actual event generation is then performed by calling different *handler* classes for hard partonic sub-processes, parton densities, QCD cascades, hadronization etc. To implement a new model to be used by ThePEG, the procedure is then to write a new C++ class inheriting from a corresponding handler class and implement a number of pre-defined virtual functions. Eg. a class for implementing a new hadronization model would inherit from the abstract `HandronizationHandler` class, and a class for a new parton density would inherit from the `PDFBase` class.

To generate events with ThePEG one first runs a setup program where an `EventGenerator` object is set up to use different models for different steps of the generation procedure. All objects to be chosen from are stored in a *repository*, within which it is also possible to modify switches and parameters of the implemented models in a standardized fashion, using so called *interface* objects. Typically the user would choose from a number of pre-defined `EventGenerator` objects and only make minor changes for the specific simulation to be made. When an `EventGenerator` is properly set up it is saved persistently to a file which can then be read into a special run program to perform the generation, in which case special `AnalysisHandler` objects may be specified to analyze the resulting events. Alternatively it can be read into eg. a detector simulation program where it can be used to generate events.

Currently, ThePEG and Pythia7 are available through their respective web pages [1,5] and include some basic $2 \to 2$ matrix elements, a couple of PDF parameterizations, remnant handling, initial- and final-state parton showers, Lund string fragmentation and particle decays. Also herwig++ is available[7,9].

The future development for ThePEG is mainly to improve the documentation. For Pythia7 the work continues to reimplement models from the old Fortran Pythia version. In addition the Ariadne program will be reimplemented using the ThePEG platform.

3 Matching matrix elements in Ariadne

With ME generators it is possible to generate a handful (typically up to six) hard partons according to exact tree-level matrix elements. But to get properly generated events it is important to interface these to realistic hadronization models, which requires that also soft and collinear partons are generated according PS models to get reliable predictions for the intra- and inter-jet structure. To add on PS to an event from a ME generator, it is important to avoid double counting. Hence the PS must be *vetoed* to avoid generating parton emissions above the cutoff needed to avoid divergencies in the ME generator. In addition the PS assumes that the emissions are ordered in some evolution variable (scale) and uses Sudakov form factors to ensure that there was no additional emissions with a scale between two generated emissions. The ME generator, of course, have no such ordering since all diagrams are added coherently. To naively add a PS to events from a ME generator will therefore give a strong dependence on the cutoff used in the ME.

A solution to this problem was presented by Catani et al.[10]. This so called CKKW procedure is based on using a jet reconstruction algorithm on the ME generated event to define an ordering of the emissions and then reweight the event according to the Sudakov form factors obtained from the reconstructed scales. In this way it was shown that the dependence on the ME cutoff cancels to NNLL accuracy. However, the dependence on the cutoff was still quite visible.

In [11] I presented an improved version of the CKKW procedure implemented in the ARIADNE program. In stead of using a standard jet algorithm to define an ordering, the dipole cascade of ARIADNE is used *backwards*, so that for each ME generated event a likely dipole emission history is found, where complete intermediate states, as well as evolution scales, are reconstructed as if the event had been generated by ARIADNE. These intermediate states are then used to make trial emissions to accept or reject the event, using the fact that the probability of not emitting a parton above the next reconstructed scale is exactly the Sudakov form factor that would have been used, had ARIADNE generated the event. Together with a special treatment of events with the highest parton multiplicity from the ME generator, the dependence on the ME cutoff was in this way much reduced for the e^+e^- annihilation case.

Recently the ARIADNE version of CKKW was implemented also for W production in hadronic collision.[12,13]. Here the situation is complicated by the fact that there are also initial-state PS involved. The procedure is still similar, however. The W+$n$jet events generated by MADGRAPH are reconstructed to find possible intermediate states corresponding to those which would arise if ARIADNE had generated them, including the possibility to have initial state emissions. The *Sudakov veto* algorithm then works in the same way as above, with trial (initial- and final-state) emissions to determine if an event should be rejected or accepted.

Preliminary results for this procedure is shown in figure 1. Here the W-$p_\perp$spectrum is shown for the case when ME generated events with up to only one jet is combined with the ARIADNE dipole shower. For comparison the corresponding result from standard ARIADNE is shown. The latter also contains a ME correction for the first emission[14] (in a way which, however, is not extensible to higher orders), so the curves should in principle agree exactly. Although the curves agree fairly well, there are some differences. The new procedure overshoots the old one near the ME cutoff at 12 GeV, and undershoots a bit at large $p_\perp$. However, we now believe we understand these differences and in a future publication[13] we will present a more detailed description and investigations using also higher order MEs.

It should be noted that although similar to the CKKW based ME+PS correc-

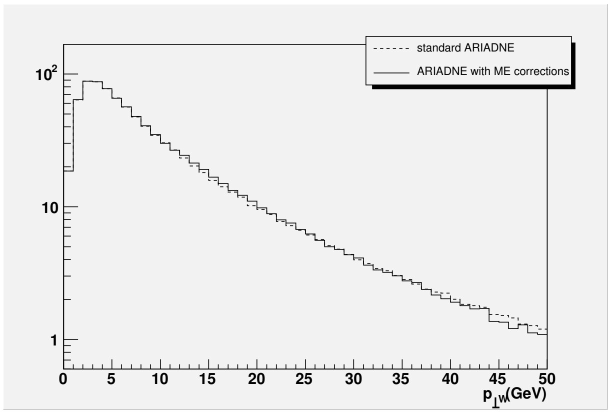

Figure 1. The $p_\perp$ distribution of W's at the Tevatron. Full line is W+0jet and W+1jet MADGRAPH events combined with ARIADNE. Dashed line is standard ARIADNE.

tion of W+$n$jets presented in [15,16,17], the Sudakov form factors are different. While the standard CKKW uses analytical form factors corresponding to standard standard DGLAP resummation, ARIADNE strictly uses the no-emission probability interpretation, which means that some terms proportional to $\log(1/x)$ are also resummed. Especially at the LHC, such terms are expected to be important as $x \propto m_W/\sqrt{S}$ becomes small, and it will be interesting to compare the different procedures to see how large the differences are and if they are diminished by using higher order MEs.

References

1. Leif Lönnblad et al. THEPEG program. `http://www.thep.lu.se/ThePEG`.
2. Leif Lönnblad. *Comput. Phys. Commun.*, 71:15–31, 1992.
3. Fabio Maltoni and Tim Stelzer. *JHEP*, 02:027, 2003.
4. M. Bertini et al. *Comput. Phys. Commun.*, 134:365–391, 2001.
5. Leif Lönnblad et al. PYTHIA7 program. `http://www.thep.lu.se/Pythia7`.
6. G. Corcella et al. *JHEP*, 01:010, 2001.
7. Stefan Gieseke et al. *JHEP*, 02:005, 2004.
8. Leif Lönnblad. *Comput. Phys. Commun.*, 84:307–316, 1994.
9. Stefan Gieseke. These proceedings.
10. S. Catani, F. Krauss, R. Kuhn, and B. R. Webber. *JHEP*, 11:063, 2001.
11. Leif Lönnblad. *JHEP*, 05:046, 2002.
12. Nils Lavesson. Masters thesis, Theoretical Physics, Lund University.
13. Nils Lavesson and Leif Lönnblad. Preprint in preparation.
14. Leif Lönnblad. *Nucl. Phys.*, B458:215–230, 1996.
15. F. Krauss. *JHEP*, 08:015, 2002.
16. Stephen Mrenna and Peter Richardson. *JHEP*, 05:040, 2004.
17. T. Gleisberg et al. These proceedings.