THEPEG (and ARIADNE)

Current status and future plans

Leif Lönnblad

Department of Theoretical Physics
Lund University

HERA-LHC/MCnet Network Meeting
CERN 08.05.29

The THEPEG project

- A basic framework for implementing models of event generation.
- Includes all model-independent parts you would find in PYTHIA, HERWIG, . . .
- Includes plug-in (abstract base) classes for any(?) event generator model.

THEPEG components

- Basic infrastructure (things Bjarne Stroustrup forgot)
  - Smart pointers, extended type info, serialization, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, unitful numbers, . . .
- Handler classes
  - to inherit from to implement specific physics models, PDFs, luminosity functions, phase space generators, . . .
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.

The basic idea

THEPEG defines a set of abstract Handler classes for hard partonic sub-processes, parton densities, QCD cascades, hadronization, etc. . .

These handler classes interacts with the underlying structure using a special Event Record and a pre-defined set of virtual function definitions.

The procedure to implement e.g. a new hadronization model, is to write a new (C++) class inheriting from the abstract HadronizationHandler base class, implementing the relevant virtual functions.

When implementing models for event generation there is typically a number of parameters and options available (in addition to the parameters of the Standard Model).

THEPEG defines a uniform way of interacting with the handler classes. The sub-classes may define a set of `InterfaceBase` objects corresponding to parameters, switches or references to objects of other `Interfaced` classes.

These are then used by the `Repository` to manipulate the corresponding member variables in the handler classes.

How to use THEPEG

Running THEPEG is separated into two phases.

- Setup:
  - A setup program is provided to combine different objects implementing physics models together to build up an `EventGenerator` object. Here the user can also change parameters and switches etc.
  - No `C++` knowledge is needed for this. Either use simple setup files with commands or Java-based GUI.
  - The Repository already contains a number of ready-built `EventGenerators`.
  - In the end the built `EventGenerator` is saved to a file.

- Running:
  - The saved `EventGenerator` can be simply read in and run using a special slave program. If `AnalysisHandlers` have been specified, this is all you have to do.
  - Alternatively the the file with the `EventGenerator` can be read into any program where it can be used to generate events which can be sent to analysis or to detector simulation.
  - The `ThePEG::Events` can, of course, be translated into `HepMC::GenEvents` or whatever.

The `EventGenerator` class

- The main class administrating an event generation run.
- It maintains global information needed by the different models: The `ParticleData` objects to be used, a `StandardModel` object with couplings etc, a `RandomGenerator`, a list of `AnalysisHandlers` etc.
- It also has an `EventHandler` object to administer the actual process generation.

Recent developments

(since last MCnet meeting at CERN in January 2007)

- No longer depends on CLHEP
- Only dependence is the GNU scientific library
- Compile-time checking of units
- Les Houches Event Files
- Helicity classes for construction of matrix elements.

Units checking

THEPEG discourages the use of `doubles` for unitful numbers. Instead there are templated classes for `Energy`, `Length`, `CrossSection`, . . .

```
Energy doSomething(LorentzMomentum p) {
  //Do something with p.
  return p.perp2() + p.m2();
}
```

will produce a compilation error because of unit mismatch.

No run-time overhead. But slows down compilation (hence switched off by default in configure script).

Les Houches Event Files

The `LesHouchesEventHandler` can handle one or more `LesHouchesFileReader`

Also specialized sub-class `MadGraphReader` which benefits from extra information in MADGRAPH files. Also reads pre-LHEF MADGRAPH files, but support for this will be discontinued.

Wish list for LHEFv2

- Standardized way of specifying cuts
- Specify the number of events in a file.
- Specify of the values of the PDFs used in each event
- Specify matching/merging scheme used and possible per-event weigths introduced.

Status of THEPEG

- Current version 1.2.0 (`www.thep.lu.se/ThePEG`)
- Reasonably stable piece of software

The Colour Dipole Cascade Model

- Describe gluon emissions in terms of radiation from colour dipoles
- Instead of one parton splitting into two, we have one dipole splitting into two, or two (colour-connected) partons into three.
- $g \longrightarrow q\bar{q}$ is still treated as normal parton splitting
- Time-like dipole shower is equaivalent to normal (angular ordered) parton shower
- Excellent description of LEP event shapes etc.

Initial-state radiation

- All gluon radiation is treated as final-state emissions
- Also radiation from dipoles connecting the proton remnants
- High $p_\perp$ gluons may be emitted in forward directions before softer emissions close to the hard sub-process.
- Corresponds to a resummation of large $\log 1/x$ terms, although not exactly BFKL or CCFM.
- Reasonable agreement with HERA data
- The only event generator able to reproduce HERA forward jets.

The ARIADNE program

- Current version 4 has been around since 1992
- Not much has been improved the last 5 years
- CKKW(L) possible but cumbersome
- Heavily used by LEP and HERA
- Not used at all at the Tevatron
- Not suitable for Higgs production
  (no initial-state $g \to q$ splitting)
- Need to get into shape for LHC

Current status of ARIADNE

- Completely rewritten in C++ using THEPEG Main work by Nils Lavesson
- Almost all components are in place
- Simple CKKW(L) matching
- Modified model for initial-state radiation needed
- $q \to g$ splitting included
- String fragmentation with PYTHIA7
- Validated for e^+e^-

Recent progress for ARIADNE

Recent progress for ARIADNE

none

- THEPEG is now a working framework for implementing event generators
- ARIADNE will not be ready for LHC startup

- THEPEG is now a working framework for implementing event generators
- ARIADNE will not be ready for LHC startup