LUND UNIVERSITY

ARIADNE and THEPEG

Current status and future plans

Leif Lönnblad

Department of Theoretical Physice
Lund University

1st MCnet Network Meeting
CERN 07.01.10

Outline

Background

- The THEPEG project
- The Colour Dipole Cascade Model
- The ARIADNE program

Current Status

- THEPEG is working and available
- ARIADNE version 5 is getting ready

Future Plans

- THEPEG
- ARIADNE

The THEPEG project

- Started out as PYTHIA7 to rewrite the Lund programs in C++
- Factorized out THEPEG as the model-independent parts
- PYTHIA7, HERWIG++ and ARIADNE are built on THEPEG
- PYTHIA8 is not

The THEPEG project

- Started out as PYTHIA7 to rewrite the Lund programs in C++
- Factorized out THEPEG as the model-independent parts
- PYTHIA7, HERWIG++ and ARIADNE are built on THEPEG
- PYTHIA8 is not

The THEPEG project

- Started out as PYTHIA7 to rewrite the Lund programs in C++
- Factorized out THEPEG as the model-independent parts
- PYTHIA7, HERWIG++ and ARIADNE are built on THEPEG
- PYTHIA8 is not

The THEPEG project

- Started out as PYTHIA7 to rewrite the Lund programs in C++
- Factorized out THEPEG as the model-independent parts
- PYTHIA7, HERWIG++ and ARIADNE are built on THEPEG
- PYTHIA8 is not

THEPEG components

- Basic infrastructure
  - Smart pointers, extended type info, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, . . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes
  - to inherit from to implement specific physics models.
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .

THEPEG components

- Basic infrastructure
  - Smart pointers, extended type info, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, . . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes
  - to inherit from to implement specific physics models.
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .

THEPEG components

- Basic infrastructure
  - Smart pointers, extended type info, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, . . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes
  - to inherit from to implement specific physics models.
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .

THEPEG components

- Basic infrastructure
  - Smart pointers, extended type info, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, . . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes
  - to inherit from to implement specific physics models.
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .

THEPEG components

- Basic infrastructure
  - Smart pointers, extended type info, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, . . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes
  - to inherit from to implement specific physics models.
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .

THEPEG components

- Basic infrastructure
  - Smart pointers, extended type info, dynamic loading, . . .
- Kinematics
  - 5-vectors, Flat n-body decays, . . .
- Repository
  - Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes
  - to inherit from to implement specific physics models.
- Event Record
  - Used to communicate between handler classes.
- Particle data
  - particle properties, decay tables, decayers etc. . .

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.

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.

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.

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.

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.

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.
- $g \to q\bar{q}$ 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.

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.
- $g \to q\bar{q}$ 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.

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.
- $g \to q\bar{q}$ 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.

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.
- $g \to q\bar{q}$ 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.

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.
- $g \to q\bar{q}$ 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.

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.

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.

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.

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.

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)pause
- Need to get into shape for LHC

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)pause
- Need to get into shape for LHC

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)pause
- Need to get into shape for LHC

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)pause
- Need to get into shape for LHC

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)pause
- Need to get into shape for LHC

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)pause
- Need to get into shape for LHC

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

THEPEG is working and available

- Current version 1.0.1 `www.thep.lu.se/ThePEG`
- Some simple $2 \to 2$ MEs
- Les Houches interface (+LHEF)
- LHAPDF interface
- Simple decays
- Templated HepMC interface
- Light-weight AIDA-based histograming
- Java-based GUI
- Inadequate documentation

Current development version

- New remnant handling strategy
- Infrastructure for CKKW matching
- Minor changes to allow for *ep* DIS
- Interface to HZTOOL

Current development version

- New remnant handling strategy
- Infrastructure for CKKW matching
- Minor changes to allow for *ep* DIS
- Interface to HZTOOL

Current development version

- New remnant handling strategy
- Infrastructure for CKKW matching
- Minor changes to allow for *ep* DIS
- Interface to HZTOOL

Current development version

- New remnant handling strategy
- Infrastructure for CKKW matching
- Minor changes to allow for *ep* DIS
- Interface to HZTOOL

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

ARIADNE version 5 is almost ready

- Completely rewritten in C++ using THEPEG
  Main work by Nils Lavesson
- Modified model for initial-state radiation
  (no more recoil gluons)
- $q \to g$ splitting included
- Simple CKKW(L) matching
- Almost all components are in place
- String fragmentation with PYTHIA7
- e^+e^- validation has started

Plans for ThePEG

- Drop dependence on CLHEP
- Introduction of templated units checking
- Moving helicity classes from Herwig++ to ThePEG
- Producing proper documentation and manual

Plans for ThePEG

- Drop dependence on CLHEP
- Introduction of templated units checking
- Moving helicity classes from Herwig++ to ThePEG
- Producing proper documentation and manual

Plans for ThePEG

- Drop dependence on CLHEP
- Introduction of templated units checking
- Moving helicity classes from Herwig++ to ThePEG
- Producing proper documentation and manual

Plans for ThePEG

- Drop dependence on CLHEP
- Introduction of templated units checking
- Moving helicity classes from Herwig++ to ThePEG
- Producing proper documentation and manual

Plans for ARIADNE

- Implement the last bits and pieces
- Validate with HERA DIS data
- Validate with Tevatron data
- Produce manual
- Release version 5.0 before . . .
- (without underlying events)

Plans for ARIADNE

- Implement the last bits and pieces
- Validate with HERA DIS data
- Validate with Tevatron data
- Produce manual
- Release version 5.0 before . . .
- (without underlying events)

Plans for Ariadne

- Implement the last bits and pieces
- Validate with HERA DIS data
- Validate with Tevatron data
- Produce manual
- Release version 5.0 before . . .
- (without underlying events)

Plans for Ariadne

- Implement the last bits and pieces
- Validate with HERA DIS data
- Validate with Tevatron data
- Produce manual
- Release version 5.0 before . . .
- (without underlying events)

Plans for Ariadne

- Implement the last bits and pieces
- Validate with HERA DIS data
- Validate with Tevatron data
- Produce manual
- Release version 5.0 before . . .
- (without underlying events)

Plans for Ariadne

- Implement the last bits and pieces
- Validate with HERA DIS data
- Validate with Tevatron data
- Produce manual
- Release version 5.0 before . . .
- (without underlying events)

- THEPEG is now a working framework for implementing event generators
- ARIADNE will be ready for LHC

- THEPEG is now a working framework for implementing event generators
- ARIADNE will be ready for LHC

- THEPEG is now a working framework for implementing event generators
- ARIADNE will be ready for LHC