HepMC Status & Plans

Lynn Garren

January 28, 2009

HepMC Status

- 2.03.09
  - seems to be used by nearly everyone
- 2.04.01
  - according to the March agreement, everyone should be using 2.04.xx
- Homepage: http://lcgapp.cern.ch/project/simu/HepMC/
- Bug Reports: https://savannah.cern.ch/projects/hepmc/
- downloads: http://lcgapp.cern.ch/project/simu/HepMC/download/

Supported Platforms

- osx105_ia32_gcc401
- i686-slc5-gcc34 / i686-slc5-gcc43
- slc4_amd64_gcc34 / slc4_amd64_gcc43
- slc4_ia32_gcc34 / slc4_ia32_gcc43
- x86_64-slc5-gcc34 / x86_64-slc5-gcc43
- i686-winxp-vc9
- win32_vc71 (soon to be dropped)
- platform list driven by general LCG support: http://lcgsoft.cern.ch/

A Brief Digression

- HepPDT is now used by both Atlas and CMS
- HepPDT is part of GENSER, but not listed
  - list it explicitly on the GENSER web page
  - http://lcgapp.cern.ch/project/simu/generator/
- both HepMC and HepPDT are external packages
  - /afs/cern.ch/sw/lcg/external/HepMC
  - /afs/cern.ch/sw/lcg/external/HepPDT
  - /afs/cern.ch/sw/lcg/external/MCGenerators

Issues

- status codes
- barcodes
- FourVector mag()
- HepMC_CLHEP20
- cross section
- iterators
- IO issues
- defs.h
- These issues are in random order.

Status Code Convention

0	null entry
1	final state (not decayed or fragmented)
2	decayed or fragmented
3	documentation line
4-10	reserved for future standards
11-200	at the disposal of event generators and equivalent to a null entry
201-	at the disposal of the user (e.g., for tracking in the detector)

New Status Code Convention

0	null entry
1	final state (not decayed or fragmented)
2	decayed or fragmented
3	documentation line
4	beam particle
11-200	at the disposal of event generators and equivalent to status 2
201-	at the disposal of the user (e.g., for tracking in the detector)

HepMC Status Methods

- int GenParticle::status()
- void GenParticle::set_status(int)
- experiments use if(p.status()==2)
- PROPOSAL
  - bool GenParticle::is_stable()
    - true ONLY if status == 1
    - possible alternate names: is_final(), is_undecayed()
  - bool GenParticle::has_decayed()
    - true if status == 2 (or equivalent)
  - bool GenParticle::is_beam()
  - should HepMC IO methods convert status codes 11-200 to status 2??
    - lose information

Barcodes

- barcodes are intended for internal use in HepMC
  - unique identifier for the particles and vertices
- the barcode is sometimes used to encode extra information about the event
  - e.g., MCTruthManager: orig barcode + N*10000000
  - this is an abuse of the barcode data member
- PROPOSAL
  - make suggest_barcode() a protected function
  - not backwards compatible

FourVector

- FourVector::mag() returns 3 vector magnitude
- FourVector::mag() inconsistent with HepLorentzVector::mag()
- bug #38319
- PROPOSAL
  - remove FourVector::mag() and ThreeVector::mag()
  - FourVector::rho() provides magnitude of 3 vector
  - ThreeVector::r() provides magnitude of 3 vector
  - FourVector::m() provides magnitude of 4 vector
  - not backwards compatible, but clears up confusion

HepMC_CLHEP20

- courtesy header
- allowed HepMC to work with CLHEP 2.0.x.y without changes
- HepMC no longer uses CLHEP
- FIRM PROPOSAL
  - remove HepMC_CLHEP20.h
  - this header no longer serves any useful purpose

Cross Section

- request to store cross section in HepMC
- bug #38051
- could be added to a new GenRun or GenJob class
- BUT some users stream events as they are created
- need to include cross section in each GenEvent
- PROPOSAL
  - add 2 doubles: cross_section() and cross_section_error()
  - define units: pb???
    - no enforcement, but be very clear about expectations

Iterators

- HepMC defines particle and vertex iterators
  - inner classes within GenEvent and GenVertex
- bug #35658
- PROPOSAL
  - move iterators into their own headers
    - easier to read the code (flashing lights, fireworks...)
    - easier to use the iterators in some contexts (e.g., SWIG)
    - can do this and be backwards compatible

IO Issue #1

- IO_Ascii
  - deprecated since 2.02.00
  - does not persist all information in GenEvent
  - replaced by IO_GenEvent
  - any existing files written with IO_Ascii can be read with IO_Genevent
- IO_Ascii will be removed as of 2.05.00
  - will allow other code cleanup
  - NON-NEGOTIABLE

IO Issue #2

- allow variable ascii output precision
- bug #41602
- only affects ascii output methods
- easily backwards compatible
- PROPOSAL
  - add precision() method
    - probably in IO_Base
  - pass precision to GenEvent::print()

IO Issue #3

- enable standard stream IO
- in addition to IO_GenEvent, but with the exact same format
- helpful in many contexts
- PROPOSAL
  - operator<<(std::stream&, GenEvent&)
  - operator>>(std::stream&, GenEvent&)

IO Issue #4

- HepMC stops reading ascii input when it encounters NaN's
  - part of the original design
  - does not allow graceful exit
- FIRM PROPOSAL
  - keep reading when invalid data is encountered
  - return empty event and flag that we are not at end
  - use throw/catch internally to do this
  - user controls event loop and decides whether to continue

Existing User Code

- will continue to work exactly as before
- will stop when corrupt event is encountered

```
HepMC::IO_GenEvent xin("test.input",std::ios::in);
HepMC::GenEvent* evt = xin.read_next_event();
while ( evt ) {
        // process event     - then get next event
        delete evt;
        xin >> evt;
}
```

Possible New User Code

```
HepMC::IO_GenEvent xin("test.input",std::ios::in);
bool ok = true;
HepMC::GenEvent* evt = xin.read_next_event();
while ( ok ) {
      if( evt ) {
             // process event            - then get next event
             delete evt;
             xin >> evt;
      } else if (xin.error_type() == HepMC::IO_Exception::InvalidData ) {
             std::cerr << "INPUT ERROR: " << xin.error_message() << std::endl;
             // clean up and get next event
             delete evt;
             xin >> evt;
      } else {
             ok = false;
      }
}
```

defs.h

- want to ask HepMC which features it has
  - ThePEG
- PROPOSAL
  - add appropriate #defines to defs.h
  - #define HAS_UNITS
  - #define HAS_PDF_INFO
  - #define HAS_HEAVY_ION
  - etc.

Going Forward

- Coding will be based on input at this meeting
- Technical discussion under appropriate support threads:
  - https://savannah.cern.ch/support/?group=hepmc
- beta release within a month
  - allow about a month for comments and discussion
- 2.05.00 released
- no more bug fix releases on 2.03 branch?
- work on the documentation