Copy of HPS Data Summary Tapes

Overview

The HPS Data Summary Tape (DST) maker uses the LCIO C++ API to read collections of physics objects (e.g. ReconstructedParticle, Track) from reconstructed LCIO files and convert them to ROOT data structures. The resulting ROOT files can then be analyzed using the HpsEvent API in combination with the ROOT data analysis framework.

This guide will outline the steps required to generate DSTs from a reconstructed LCIO file and will provide some examples of a simple analysis.

Installation

Preliminaries

The DST maker and HpsEvent API require the following packages before they can be built:

- The LCIO C++ API
- ROOT data analysis framework
- Generalized Broken Lines
- CMake > 2.8
- Doxygen (Optional. Used to build the documentation.)

Getting the source

The DST maker and HPS Event API source is stored in a public github repository. The code can be "cloned" i.e. copied to a users local machine by issuing the following commands from a terminal

```
cd /path/to/workdir
git clone https://github.com/omar-moreno/hps-dst.git
```

A github account is not required to clone the source code.

NOTE: If you want to contribute to the development of the HPS DST, please contact Omar Moreno.

Building the DST Maker and HPS Event API

Before the DST source can be built, the following environmental variables need to be set:

```
ROOTSYS=/path/to/root
GBL_DIR=/path/to/gbl/cpp
LCIO=/path/to/lcio
```

In order to avoid polluting the source tree, an out-of-source build is encouraged. This can be done as follows:

```
cd hps-dst/
mkdir build; cd build
cmake ../
make
```

This will create the binaries (in the build/bin directory) along with the shared library HpsEvent.so (in the build/lib directory) which contains the ROOT dictionary and HPS Event API. Building of the API documentation is done by Doxygen as follows:

```
make doc
```

This will generate both LaTeX and html documentation in the directory hps-dst/doc.

The DST Maker

Generating DST's from reconstructed LCIO files can be done by issuing the following command from a terminal:

```
dst_maker LCIO_INPUT_FILE [additional input files] -o OUTPUT_FILE_NAME.root
```

This command will generate a DST with all GBL collections empty. In order to run the GBL fitter and generate the GBL collections, the following command can be issued from a terminal:

```
dst_maker LCIO_INPUT_FILE [additional input files] -o OUTPUT_FILE_NAME.root -g -b B_FIELD
```

Note that when enabling the GBL collections, the magnetic field strength needs to be passed as a parameter to the DST maker.

DST Structure

The ROOT based DST's are composed of HpsEvent objects which are used to encapsulate collections (TClonesArray) of physics objects e.g. tracks and particles. The collections encapsulated by an HpsEvent are shown on Table 1.

Collection Name (TClonesArray)	Type	Variables (ROOT leaves)	References (TRefArray)
tracks	SvtTrack	# hits, d0, phi, omega, tan_lambda, z0, chi^2	SvtHit
svt_hits	SvtHit	layer #, corrected position and covariance matrix, hit time	
ecal_clusters	EcalCluster	# ecal hits, position, energy, hit time, m2, m3	EcalHit, SeedHit
ecal_hits	EcalHit	position, crystal indices, energy	
fs_particles (Final State Particles)	HpsParticle	charge, momentum, energy	SvtTrack, EcalCluster
uc_vtx_particles (Unconstrained Vertexed Particles)	HpsParticle	# daughters, charge, fitted momentum, energy, vertex position	HpsParticle
bsc_vtx_particles (Beamspot Vertexed Particles)	HpsParticle	# daughters, charge, fitted momentum, energy, vertex position	HpsParticle
tc_vtx_particles (Target Constrained Vertexed Particles)	HpsParticle	# daughters, charge, fitted momentum, energy, vertex position	HpsParticle
mc_particles	HpsMCParticle	pdg id, charge, generator status, energy, mass, momentum, endpoint	
gbl_tracks	GblTrack	kappa, theta, phi, d0, z0, seed_kappa, seed_theta, seed_phi, seed_d0, seed_z0, chi^2, ndf, momentum, covariance	
gbl_tacks_data	GblTrackData	# of strip hits, kappa, theta, phi, d0, z0	
gbl_strips_data	GblStripData	id, path3D, path ...	

Table 1: Collections and variables encapsulated by HpsEvent.

The HpsEvent class also contains additional event information, e.g. run number, as shown on Table 2.

ROOT Branch	Variables (ROOT leaves)
HpsEvent	event #, run #, # tracks, # SVT hits, # Ecal clusters, # Ecal hits, # final state particles, # unconstrained/beam constrained/target constrained particles, # of MC particles, # GBL tracks, # GBL track data, # GBL strips data

Table 2: Variables encapsulated by HpsEvent.

Getting Started with Analysis