HDF5 and Row-wise Ntuple in Analysis Tools in Geant4 10.3

G. Barrand[1)], I. Hřivnáčová[2)]

[1)] Laboratoire de l'Accélérateur Linéaire (LAL), Université Paris-Sud, CNRS-IN2P3
[2)] Institut de Physique Nucléaire (IPNO), Université Paris-Sud, CNRS-IN2P3

Geant4 Analysis / g4tools (see also CHEP 2016 poster)

- The **Geant4 analysis** category provides the users a "light" analysis tool available directly with Geant4 installation. It offers a uniform, user-friendly interface to **g4tools**, hiding the differences of selected output formats. Its integration in the Geant4 framework offers interactive commands, units, (in)activation of selected objects.
- **g4tools**, coming with Geant4, provides histograms, profiles, ntuples, plots and code to write/read in various formats: CSV, ROOT, AIDA-XML, Postscript (for plots) and now **HDF5**. It is strongly OO with no implicit global management. It is thread safe since having no writable statics. It is now also available on **GitHub** at : https://github.com/gbarrand/g4tools.git with pages at **https://gbarrand.github.io**

HDF5

From https://portal.hdfgroup.org/display/support :

HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety of datatypes, and is designed for flexible and efficient I/O and for high volume and complex data. HDF5 is portable and is extensible, allowing applications to evolve in their use of HDF5. The HDF5 Technology suite includes tools and applications for managing, manipulating, viewing, and analysing data in the HDF5 format.

Already used by various applications (see Third Party pages).

Already used in other domains of science where Geant4 is also used (for example in medical, G4Linac_MT is in their list).

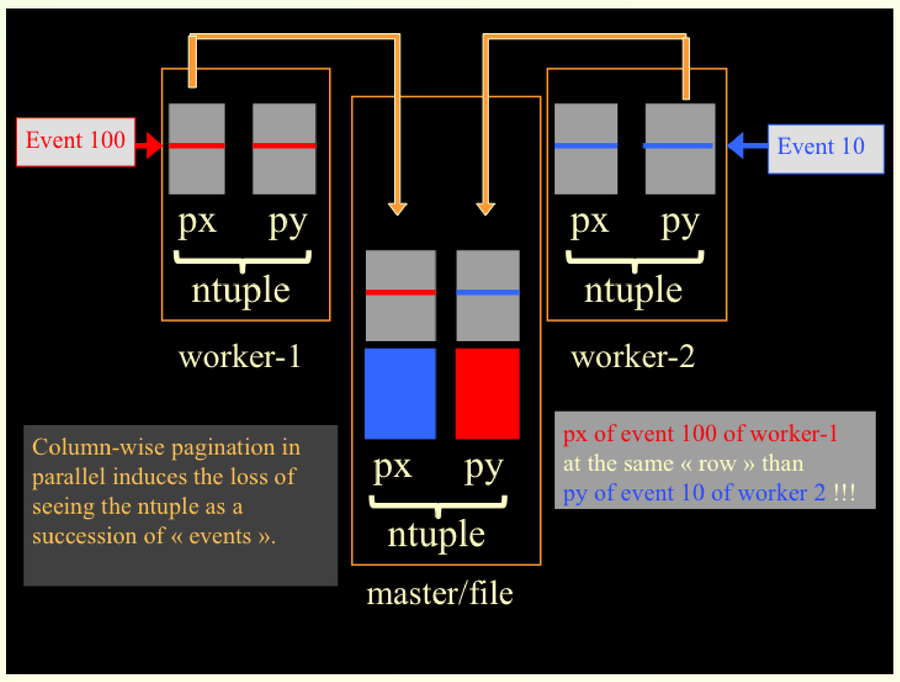
It sounded natural to introduce this format to write/read histograms and ntuples.

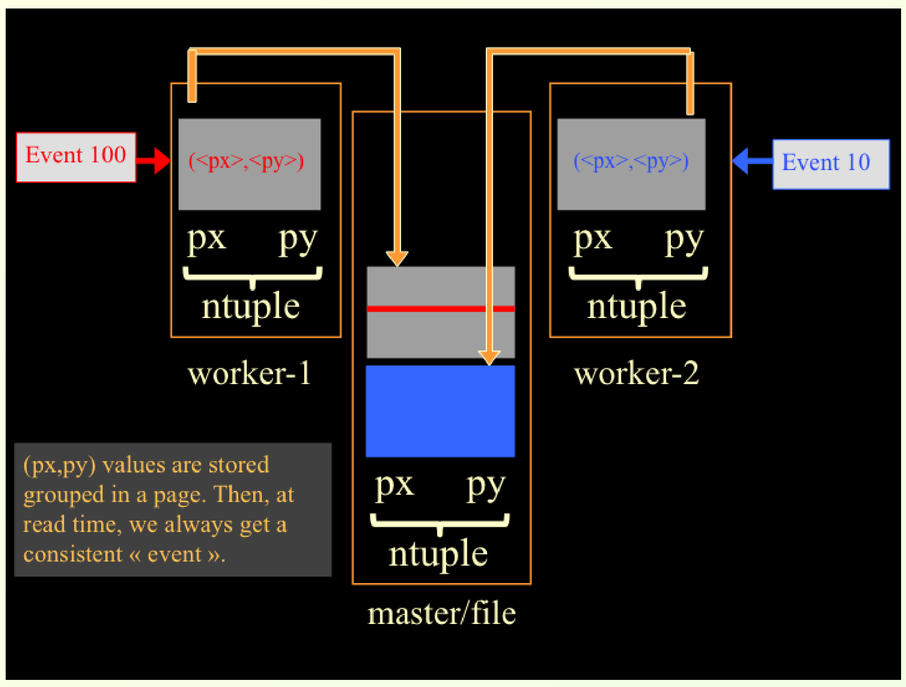
Merging ntuples in a parallel context (**MT** or **MPI**) :

- On users requests : handle one file when filling an ntuple (with same booking) from different workers (threads or MPI/ranks).
- Done with **column-wise** ntuples at the root format in 10.3 by exploiting the **page/TBasket** logic of a **column/TBranch**.
- But, as the pages arrive from workers in random order for a given column, we loose an "event point of view" when reading back the data.
- We then introduced a **"row-wise"** mode to sort out this case. In this mode, columns are leaves of a single TBranch attached to each ntuple per worker. A page contains a set of "row" which could be interpreted consistently as an "event" when reading data from the lonely file. This mode had been introduced for multithreads for the root format in Geant4 10.4
- Row-wise merging with MPI is foreseen for 10.5.

Way to write HDF5 data : priority to simple schema/semantic :

- For a given object (for example an histogram), there is a lot of ways to store it in HDF5. We could have defined a "compound datatype" that does a one to one mapping of the class/fields to a H5 structure and then write one histogram in a H5 dataset by writing the compound. Or by using a hierarchy of **H5 groups** storing at the end **array of simple data** (for example an array of bin entries, an array of bin weights).
- In order that data file be easily read back by various tools, we choose the second way that has the advantage of having a more simple semantic/data schema. In particular the h5ls tool permits one to have a glance of what is in the file.
- For **ntuples**, it is the same. Each column of simple data types is a **H5group gathering "pages" of data, each page being mapped to a H5 dataset**. When a page is filled in memory, it goes straight in a H5 dataset in the file. Here too, the semantic of data organisation is rather simple, and may be read back quite easily (we think).





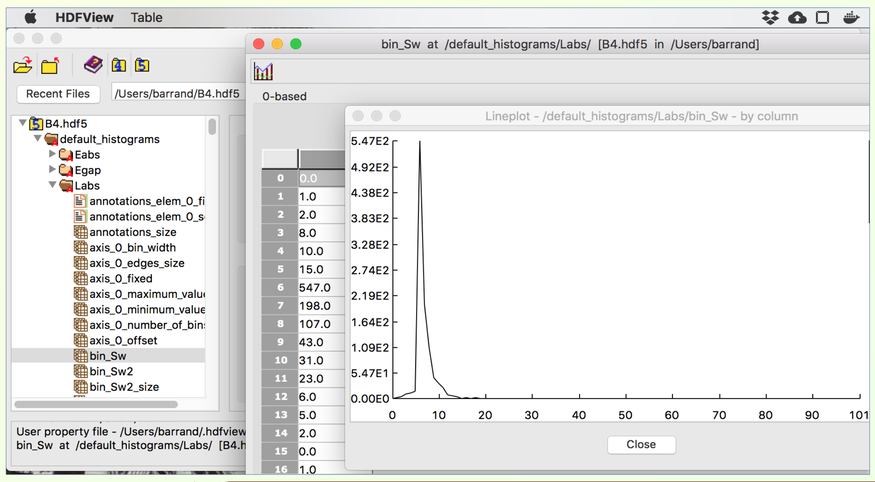
Examples and apps to read files :

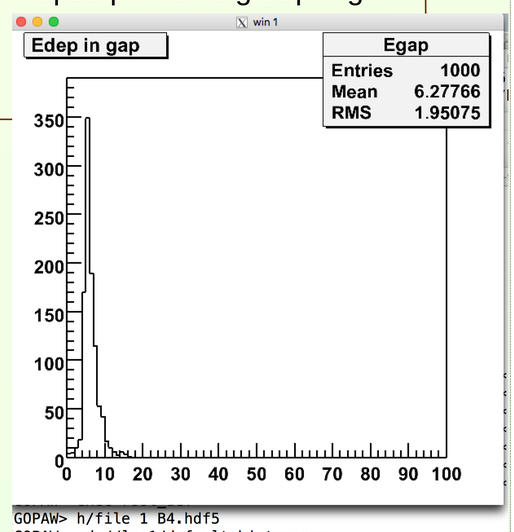
Geant4 **B4, B5** examples : using g4hdf5.hh will switch from the default output to HDF5.

In g4tools/test/cpp (available on github https://github.com/gbarrand/g4tools.git) :
- hdf5_histos.cpp, hdf5_ntuple.cpp : write/read histos, ntuple.
- hdf5_threads.cpp : write ntuples in threads.
- cern_root_hdf5_ntuple.cpp : read an ntuple and do a projection and plot by using a TH1D and a TCanvas.

Apps :
- **HDFView** reads the file written by g4tools. It is available at https://portal.hdfgroup.org
- **ioda** (1.14.x) knows the semantic of g4tools hdf5 files.
- **gopaw** too. (See G.Barrand dedicated poster).

ioda and gopaw are also at github.com/gbarrand





Performances

- I/O performances to store histograms and ntuples is not the issue in this context (but would be for someone wanting to deal with a full event model with HDF5).
- If needed, for histograms, we could introduce a "compound" mode which would be more effective.
- For ntuple, the "page" logic is similar to the TBasket of ROOT, and if choosing the same size and compression level for them, we have similar file sizes and write time since, in both cases, we are anyway dominated by the speed of writing on disk.
- Concerning parallelisation, HDF5 has some logic of its own that we plan to look at.

23nd International Conference on Computing in High Energy and Nuclear Physics, July 9-13 2018, Sofia