Using HDF5 for CMS

Saba Sehrish

Writing serialized data products to HDF5

- The focus of this activity is on taking the serialized data products (using ROOT) and writing to HDF5 files.
- Working with the serialization code provided by Chris Jones
- Using a ROOT file with only 20 events and 7 data products
- Also have deserialization code available but not used it yet

Current status: a “serial” C++ program that currently …

- reads event by event data from a given ROOT file - the ROOT file has realistic structure but dummy data
- serializes data products in each event
- accumulates a given number of events in memory
- writes different data sets, where an HDF5 dataset corresponds to a data product
- the current datatype is a string,
- one dimension is extensible to allow for adding events as we don’t know how many events in advance
- used chunking

h5ls of the output file

```
[ssehrish@grunt3 build]$ h5ls cmsevents_ds_acc.h5
BranchListIndexes         Dataset {20/Inf, 1}
EventAuxiliary            Dataset {20/Inf, 1}
EventProductProvenance    Dataset {20/Inf, 1}
EventSelections           Dataset {20/Inf, 1}
edmTriggerResults_TriggerResults__TESTOUTPUT. Dataset {20/Inf, 1}
edmtestOtherThings_OtherThing_testUserTag_TESTOUTPUT. Dataset {20/Inf, 1}
edmtestThings_Thing__TESTOUTPUT. Dataset {20/Inf, 1}
[ssehrish@grunt3 build]$
```

h5dump of the output file

```
[ssehrish@grunt3 build]$ h5dump --header cmsevents_ds_acc.h5
HDF5 "cmsevents_ds_acc.h5" {
GROUP "/" {
   DATASET "BranchListIndexes" {
      DATATYPE  H5T_STRING {
         STRSIZE H5T_VARIABLE;
         STRPAD H5T_STR_NULLTERM;
         CSET H5T_CSET_UTF8;
         CTYPE H5T_C_S1;
      }
      DATASPACE  SIMPLE { ( 20, 1 ) / ( H5S_UNLIMITED, 1 ) }
   }
   DATASET "EventAuxiliary" {
      DATATYPE  H5T_STRING {
         STRSIZE H5T_VARIABLE;
         STRPAD H5T_STR_NULLTERM;
         CSET H5T_CSET_UTF8;
         CTYPE H5T_C_S1;
      }
      DATASPACE  SIMPLE { ( 20, 1 ) / ( H5S_UNLIMITED, 1 ) }
   }
   DATASET "EventProductProvenance" {
      DATATYPE  H5T_STRING {
         STRSIZE H5T_VARIABLE;
         STRPAD H5T_STR_NULLTERM;
         CSET H5T_CSET_UTF8;
         CTYPE H5T_C_S1;
      }
      DATASPACE  SIMPLE { ( 20, 1 ) / ( H5S_UNLIMITED, 1 ) }
```

Next Steps

- The code writes the schema as told but has not been tested and verified. I have “deserialization” code form Chris, which I would like to use on the data read from the HDF5 file and compare with the ROOT file.
- Make the code available with instructions to run, and example schema too, Github?
- Also, the batch writes are not taking care of the cases where the last write may have fewer events to write. There is no check in place yet.
- The writes are sequential, once the code is tested and verified that we can write and read back correct data, the next step is parallel writes using MPI, and also use compression and look into what IO optimizations will make sense here.
- Discuss with LBNL HDF5 group about the current approach and alternate design choices in the off-week Wednesday
- We will need intermediate and large dataset to evaluate what chunking and compression options to use, and also overall design.

More details

- Developing in CMS build environment and using all the libraries available that are available
  - HDF5 v1.10.6, recently requested parallel support, will wait for h5py support before we
  - HighFive (using current head of develop)
  - OpenMPI available v2.1.5
- Plan to use H5CPP
- Do we have any repository we should use or should I go ahead and create something?

Data types and organization

In the HDF5 writer application, we will be getting data event by event, so we need to accumulate events before we start writing

- Write each data product to a different data set (current approach)
- Write events as a compound type to one data set (?)
- Write some data products grouped into compounds, and others as individual (?)