Tree Looper

(DocDB 7493)

Zhe Wang

Tsinghua University

Jan. 27, 2012

While doing data analysis, all the work in the end boils down to looping over a tree. Since it becomes a routine work, this note briefly document a simple root tree looper which has the basic looping functionality and accommodate the feature needed for the Daya Bay Experiment.

Several features are usually expected for this: 1. Fast; 2. Random access to any tree entry; 3. Simple to program (include the tedious I/O part, BeginJob, BeginRun, etc.) and fully-automation is the best; *4. No mystery and smooth learning curve. A college student should be able to grasp the technique in a day and to start on physics quickly. With some basic C++ and root knowledge, one will have the ability to fix problems.* The last point hasn't been emphasized enough in the past days.

Following is a step-by-step instruction and a template is written together with this example.

Step 0. Preparation

1. Setup a root environment.
2. Get the template including an example root tree file and some code. (In the following, all the commands needed to type in are enclosed in a grey box. Root prompt is like root [1], and shell prompt is like $. In the following we will always work in the directory "LoopExample".)

```
$ svn co http://dayabay.ihep.ac.cn/svn/dybsvn/people/wangzhe/TWin/LoopExample
$ cd LoopExample
```

Step 1. Handle root tree input

Since every tree is different, then there is not a unique template. Need to generate some root tree reading code for each case. Fortunately this all can be done automatically. The example tree file "0012653._0189.stream.root" contains a tree called "Stream". Now the a C++ program "TreeReader" will be created.

```
$ root 0012653._0189.stream.root
root [1] Stream->MakeSelector("TreeReader")
root [2] .q
```

Now in current directory, "TreeReader.h" and "TreeReader.C" are created. Since later all the program needs to be compiled running in binary mode, a dictionary file is also needed. This can be created with the following command.

```
$ rootcint TreeReaderDict.C -c TreeReader.h
```

Step 2. Get a main file and Makefile

You already downloaded them while doing the checkout of the template. They are already enough for the basic looping functionality. List the current directory to verify their existence (Looper.C and Makefile). You may need to edit them to accommodate a different analysis. This will be covered later.

Step 3. Run it!

```
$ make
$ Looper 0012653._0189.stream.root
```

In this example, the trigger number “TrigNum” and event energy “E” are printed out. They will flush your screen.

Step 4. Need some modification for your purpose

For different tree, you need to repeat step 1, and the tree name “Stream” should be modified.

In the main file, Looper.C, the tree name “Stream” also shows up once, it also need to be modified correspondingly.

In this example, “TrigNum” and “E” are printed out in line 55 of Looper.C. They should be removed if they don’t exist.

In the end

The total length of the main file is only 77 lines including 197 words. The central loop is:

To get random access of any tree entries, assign different entry number in TR->GetEntries(entry).

```
/* The main loop over every stream entries */
/* --------------------------------------- */
unsigned int entries = ReadinChain->GetEntries();
for( unsigned int entry=0; entry<entries; entry++ )  {
  unsigned int localentry = ReadinChain->LoadTree(entry);
  int ret = TR->GetEntry( localentry );
  if( ret==0 ) {
    cout<<"Warning: Read error"<<endl;
  }
  /*** Test each entry here ***/
  cout<< "TrigNum: "<<TR->TrigNum<<" Energy= "<<TR->E <<endl;
  /***  End of each entry   ***/
}
```

An analysis orientated to physics, hopefully, can start easier.