Subject: problem with a processor (software-wise)
Posted by thuyuk on Fri, 08 Aug 2014 14:23:39 GMT
View Forum Message <> Reply to Message

Hi all,

I want to test an array of data with some 1D conditions and then set output of another array of data when the first data set fulfill the condition.

So, I modified Gate1D.cpp and .hpp as the following:

```
#include "Gate1D.hpp"
#include <iostream>
#include <cmath>

Gate1D::Gate1D(const std::string &config_dir,
       const std::string &name,
       const bool no_files)
  : Processor(name, config_dir, no_files, std::vector<int>())
{
  // make all the in/outputs, parameters and coefficients
  // known by name.
  NAME_PARAMETER(num_channels);
  init();
  read_parameters();

  NAME_CONDITION_ARRAY(reference_gate, Window1D, parameter(num_channels));

  NAME_INPUT_ARRAY(value, parameter(num_channels));
  NAME_INPUT_ARRAY(value2test, parameter(num_channels));

  NAME_OUTPUT_ARRAY(GatedValue, parameter(num_channels));



  init();
  read_conditions();
}

Gate1D::~Gate1D()
{
}


void Gate1D::process(prespec::viscon::Interface &viscon_interface, int trigger)
{

int chnr = parameter(num_channels);

  for (int i = 0; i<chnr; i++)
 {
```

```
double left_gate = condition_array(reference_gate, i, 0);
double right_gate = condition_array(reference_gate, i, 1);

  if(input_array_value(value, i) == 0 && input_array_value(value2test, i) == 0 )
   return;

 if (input_array_value(value2test, i) > left_gate && input_array_value(value2test, i) <
right_gate)
 {
 double girdi = input_array_value(value, i);
 fill_output_array(GatedValue, i, girdi);
 }

  }

}
```

I created required Processor.par and Processor.con files which have parameter and the conditions, respectively.

When I want to run this processor, I get the following error:

go4analysis: prespec/process/Processor.cpp:665: const double&
prespec::process::Processor::input_array_value(int, int): Assertion
`input_arrays_[channel].size() > i' failed.

I checked if any of input_array_value requests data from any of the channel which has a number that is out of range of the array, but I couldn't see it.

If anyone could tell me what seems to be the problem, I would be grateful.

Thanks!
Tayfun

Subject: Re: problem with a processor (software-wise)
Posted by miree on Mon, 11 Aug 2014 09:48:03 GMT
View Forum Message <> Reply to Message

Hi Tayfun,

I like a lot that you want to add new functionality!

But the existing "Gate1D" processors might already be used at some point in your (or others) analysis. And if you modify the behavior of an existing processor it might break your analysis.

It is always good to add new functionality by adding another processor. Doing this, you also have the opportunity to give your processor a name that describes its purpose. To add a new

processor, do the following:

1) copy an exsisting processor's header (.hpp) and implementation (.cpp) files
Example:

cp Gate1D.hpp MyNewProcessor.hpp; cp Gate1D.cpp MyNewProcessor.cpp

2) add the new processor to the file module.cpp in the direcory of the plugin. You have to add a line on the top of the file to include your header file:

#include "process/MyNewProcessor.hpp"

and you have to add a line to make the framework aware of the new Processor:

```
ELDER_MODULE_PROCESSOR_BEGIN;
...
ELDER_MODULE_PROCESSOR_ADD(MyNewProcessor);
...
ELDER_MODULE_PROCESSOR_END;
```

3) add your processor header and implementation files to the toplevel Makfile.am in the prespec directory.
You have to add it to the correct plugin. If your new processor is in the UTILS plugin, it has to go here:

```
libprespecUTILS_la_SOURCES = ...    \
                             ...    \
                             plugins/UTILS/process/MyNewProcessor.cpp \
                             plugins/UTILS/process/MyNewProcessor.hpp \
                             ...
```

Be careful not to put whitespaces after the '\' character!!!!

4) Now you can implement your Processor, but first rename all "Gate1D" to "MyNewProcessor" in the two files MyNewProcessor.cpp and MyNewProcessor.hpp.

5) type 'make' in the prespec toplevel directory.

Once you have written a useful processor, please push it to the repository so that everybody can benefit from your efforts.

I'll post another reply about the assertion failure that you get.
You can also look at the processor "ArrayFilter" in UTILS plugin. It might already do what you need.

Best regards,
Michael

Subject: Re: problem with a processor (software-wise)
Posted by thuyuk on Mon, 11 Aug 2014 09:56:10 GMT
View Forum Message <> Reply to Message

Hi Michael,

Thanks for the suggestions.

You are right, I will try to create a new processor next time. But, especially for my unique case, I checked first if this processor is used in any other config file, and it is not. So, I didn't expect to see any complications in the other parts of the analysis.

Regarding to your suggestion, using UTILS/ArrayFilter, it doesn't do what I need. I want a put a condition on an array, and if the condition is fulfilled, I want to get the output of another array.

Thank you!
Tayfun

Subject: Re: problem with a processor (software-wise)
Posted by miree on Mon, 11 Aug 2014 11:48:07 GMT
View Forum Message <> Reply to Message

Hi Tayfun,

this post is related to the error message Assertion `input_arrays_[channel].size() > i' failed.

If you want to write processors with arrays, you need to know the structure of an Array. This is explained in the tutorial.pdf in Section 4.2 "Elder Arrays: Writing a DSSSD Processor". I'll try to point out the relevant thing:

Arrays are composed of N entries, where N can be different for each event. If an array is defined by

NAME_INPUT_ARRAY(my_array, 4) // 4 different indices allowed: 0,1,2,3

It can look like this: (input_array_size(my_array) == 4)

```
      |entry0 | entry1 | entry2 | entry3 |
-----------------------------------------
index  |  1   |   3   |  2   |  3   |
value  |  3.2  |  100.5 |   3.3  |  234.5 |
-----------------------------------------
```

Or it can look like this: (input_array_size(my_array) == 6)

```
      |entry0 | entry1 | entry2 | entry3 | entry4 | entry5 |
-------------------------------------------------------------
index  |  3   |   0   |  1   |  2   |  2   |  1   |
value  |  3.2  |  100.5 |   3.3  |  234.5 |  150.1 |   3.14 |
-------------------------------------------------------------
```

Or it can be empty: (input_array_size(my_array) == 0)

```
      |
--------
index  |
value  |
--------
```

The index field will only contain numbers 0,1,2 or 3. It can have an arbitrary number of entries. Each index may come zero or one or many times. Each entry has one value.
The meaning of the index depends on the context. It can be the channel of MultihitTDC, the crystalID of a HPGe-ARRAY, or the LYCCA-module number.

The correct way to loop over an array is like this:

```
int N = input_array_size(my_array); // N can be any positive integer or zero
for (int i = 0; i < N; ++i)
{
  int    index = input_array_index(my_array, i); // index refers to the index of the i-th array entry
  double value = input_array_value(my_array, i); // value refers to the value of the i-th array
entry


  // do something with the value and the index
  // .. for example testing
}
```

The assertion failure is caused by accessing an entry that is beyond the size of the array. Example: If you get an array with 3 entries and you call entry 5, it will not work.

```
input_array_index(my_array, 5); // this will cause an assertion failure if the number of entries is
smaller than 6
```

That's it
There are a few traps that can happen if you are new to that. Looping over an array is one of them.
So don't hesitate to ask.

Also, there is a possibility to make the condition test by calling the function 'condition_valid'

```
for( //...
{
  int    index = input_array_index(my_array, i);
  double value = input_array_value(my_array, i);
  //...

  // the following works if 'reference_gate' is defined
  //   with 'NAME_CONDITION_ARRAY(reference_gate, Window1D,
```

```
parameter(num_channels))'
  if (condition_valid(value, reference_gate, index))
  {
    //...
  }
}
```

Best regards,
Michael

Subject: Re: problem with a processor (software-wise)
Posted by miree on Mon, 11 Aug 2014 12:07:59 GMT
View Forum Message <> Reply to Message

Hi Tayfun,

A comment to your statement: "I want a put a condition on an array". This can mean many different things:

A few examples: It can mean that the condition is fulfilled if
1) all of the array entries fulfill a condition.
2) at least one of the array entries to fulfill a condition
3) at least half of the entries are inside a gate.
4) the size of the array is larger than 5
....

Some of these cases can be achieved by combinations of existing processors.
In order to achieve 2), you could do the following:
Use an UTILS.ArrayFilter that picks out all array entries that fulfill a condition.
Determine the size of the filtered array with the processor UTILS.ArraySize.
Make a condition with UTILS.ConditionWindow1D and make the gate go from 1 to 1000 (a large value).

Michael

Subject: Re: problem with a processor (software-wise)
Posted by thuyuk on Wed, 13 Aug 2014 10:32:37 GMT
View Forum Message <> Reply to Message

Hi Michael,

Thank you for the explanation on how arrays work in the code, although I read this part of the tutorial before, it's useful to have it here if somebody wants to get it easier.

I searched through the available processors in plugins/UTILS, but, I couldn't find one or a combination of them to do what I want to do.

So, I followed your suggestion, and created another processor. Please find below how it looks

like:

This the cpp:

```
#include "ArraysWithCondition.hpp"
#include <iostream>
#include <cmath>

ArraysWithCondition::ArraysWithCondition(const std::string &config_dir,
        const std::string &name,
        const bool no_files)
  : Processor(name, config_dir, no_files, std::vector<int>())
{
  // make all the in/outputs, parameters and coefficients
  // known by name.


  NAME_INPUT_ARRAY(value1, 7);
  NAME_INPUT_ARRAY(value2, 7);

  NAME_OUTPUT_ARRAY(GatedValue, 7);

  NAME_CONDITION_ARRAY(gate, Window1D, 7);

  init();
  read_conditions();
}

ArraysWithCondition::~ArraysWithCondition()
{
}

void ArraysWithCondition::process(prespec::viscon::Interface &viscon_interface, int trigger)
{


 for(int i = 0; i < input_array_size(value1); i++)
 {
  int index = input_array_index(value1, i);
  double val2 = input_array_value(value2, index);
  double val1 = input_array_value(value1, index);
  double left_gate = condition_array(gate, index, 0);
  double right_gate = condition_array(gate, index, 1);
  if(val2 < right_gate && val2 > left_gate)
  {
   fill_output_array(GatedValue, index, val1);

  }

 }
```

```
}
```

and the hpp:

```
#ifndef UTILS_ArraysWithCondition_HPP_
#define UTILS_ArraysWithCondition_HPP_

#include <prespec/process/Processor.hpp>


//! @brief A processor that evaluates the information coming from a
//!        thin circular plastic membrane detectors, surrounded by
//!        many photomultipliers. These detectors are used in the
//!        Lycca time of flight subsystem.  WTF!!!!! wrong doc!
//!
//! It evaluates the time of the particle impact on the membrane, given
//! the particle position (deduced by other tracking detectors).
//! If this information is missing, the processor tries to deduce the
//! impact position from the time information. That is possible with an
//! accuracy of between (6-10 milimeters).
class ArraysWithCondition : public prespec::process::Processor
{
public:
  ArraysWithCondition(const std::string &config_dir,
  const std::string &name,
  const bool no_files = false);
  ~ArraysWithCondition();

  virtual void process(prespec::viscon::Interface &viscon_interface, int trigger);

  double analyze(double x_particle, double y_particle, bool find_position = false);

  // the inputs to this processor
  enum InputArray
    {
      value1,
      value2,
    };

  // the outputs of this processor
  enum OutputArray
    {
      GatedValue,
    };


   enum Conditions
    {
      gate,
```

```
    };

};


#endif
```

This the part in Store.config:

```
processor HighLevel/particle/incoming/MusicPileUp UTILS.ArraysWithCondition
 value1[0:6] <- MuiscPileupCor.music[0:6]
 value2[0:6] <- MuiscPileupCor.music[8:14]

 display GatedValue
end
```

And this is how the condition file looks like:

```
gate[0]  25 160
gate[1]  25 160
gate[2]  25 160
gate[3]  25 160
gate[4]  25 160
gate[5]  25 160
gate[6]  25 160
```

Well, with these files, the compilation is done without errors. But when I want to run this processor, I get this:

```
prespec/process/Processor.cpp:665: const double&
prespec::process::Processor::input_array_value(int, int): Assertion
`input_arrays_[channel].size() > i' failed.
```

First of all, I looked through the written processors, and couldn't find one to process the MUSIC Pile up data. Second, I have no idea how to process these data, due to the high counting rate, we possibly suffer because of the pile-up in MUSIC, which makes noise in the higher Z region in the ion selection in FRS. So this code intents to put a 1-D gate on the peaks that appear on the spectra of the last 8 channels of the MuiscPileUp Crate, and accept the data from the first 8 channels with this condition.

Maybe I'm wrong, maybe this is not the way to process these data, this is another subject to discuss, but I don't see the reason why the code fails with this kind of error message. The array size is 7 in this case, because two of the channels are missing in the crate. I suppose to fill 7 indices of the input arrays, but why I cannot ask the input value inside any of them?

Subject: Re: problem with a processor (software-wise)
Posted by miree on Wed, 13 Aug 2014 16:10:33 GMT
View Forum Message <> Reply to Message

Hi Tayfun,

I never was looking in detail to the pile-up correcting readout of the MUSIC. First of all: Only one of the two MUSICS was read out with the pile-up correction module. The other one was read out in the conventional way. The pile-up correction readout is done with a SIS3302 digitizer from Struck with a special firmware from KVI Groningen. This digitizer has 8 channels for incoming signals.
The unpacker delivers for each of the channels an amplitude (having indices from 0 to 7), and time information (having indices ranging from 8 to 15, where (index-8) will tell you the module channel). Or, in other words: you get 16 output channels (or indices), where the first half contains amplitude information, the second half contains time information.
This module is multi-hit capable, that means that you can have several hits in each channel. You need to select somehow the "good" hit if there are is more than one hit.

One thing that you could do in addition: check the correlation of the SIS-MUSIC amplitudes with the amplitudes from the classical music readout of the other MUSIC. I'have seen an experiment, where there was no correlation at all between the SIS and classical readout.

Regarding the error message in the implementation of your processor. I've written the code as I think it should work in a verbose manner, to point out what is causing the problem in your implementation:

```
// check first if both arrays have the same number of entries
if (input_array_size(value1) == input_array_size(value2))
{
  for (int i = 0; i < input_array_size(value1); ++i)
  {
    // At this point, your code has a problem, because you use "index" instead of "i".
    // The second parameter in the functions input_array_index/input_array_value
    // must not exceed the size of the array (i.e. the number of entries of the array)
    // Otherwise you'll get an assertion failure.
    int    index_of_ith_entry_in_value1 = input_array_index(value1, i);
    double value_of_ith_entry_in_value1 = input_array_value(value1, i);
    int    index_of_ith_entry_in_value2 = input_array_index(value2, i);
    double value_of_ith_entry_in_value2 = input_array_value(value2, i);

    // eventually, you want to check if the indices are the same
    if (index_of_ith_entry_in_value1 == index_of_ith_entry_in_value2)
    {
      if (condition_valid(value_of_ith_entry_in_value2, gate, index_of_ith_entry_in_value1))
      {
        fill_output_array(GatedValue, index_of_ith_entry_in_value2,
value_of_ith_entry_in_value1)
      }
    }
  }
}
```

Just keep in mind the following: Index of an array entry index_of_ith_entry_in_value1 is something different than the entry number i. Mixing up these two things causes your code to stop with an assertion failure.

Subject: Re: problem with a processor (software-wise)
Posted by thuyuk on Wed, 13 Aug 2014 16:55:53 GMT
View Forum Message <> Reply to Message

Hi Michael,

I really thank you very much for the valuable information regarding to the MUSIC pile-up detectors. This will make my life easier!

With respect to the correction on the processor code, thanks a lot for clarifying what was wrong. Later I thought also to assign different indices to different inputs but I couldn't figure out to check if the two indices are equal.

Many thanks!

Tayfun
