Memory Usage Studies

Jake Calcutt
March 13, 2023

Introduction

Recently, a memory usage task force was created to study any shortcomings/places for improvement for memory usage within DUNE's standard production workflows

Started looking at the PDSP production chain

- 4 stages: event gen, g4/photon, detsim, reco
- Ran valgrind's massif/memcheck tools on one event passed through each stage
- Tried to identify areas in DUNE code we can start to address

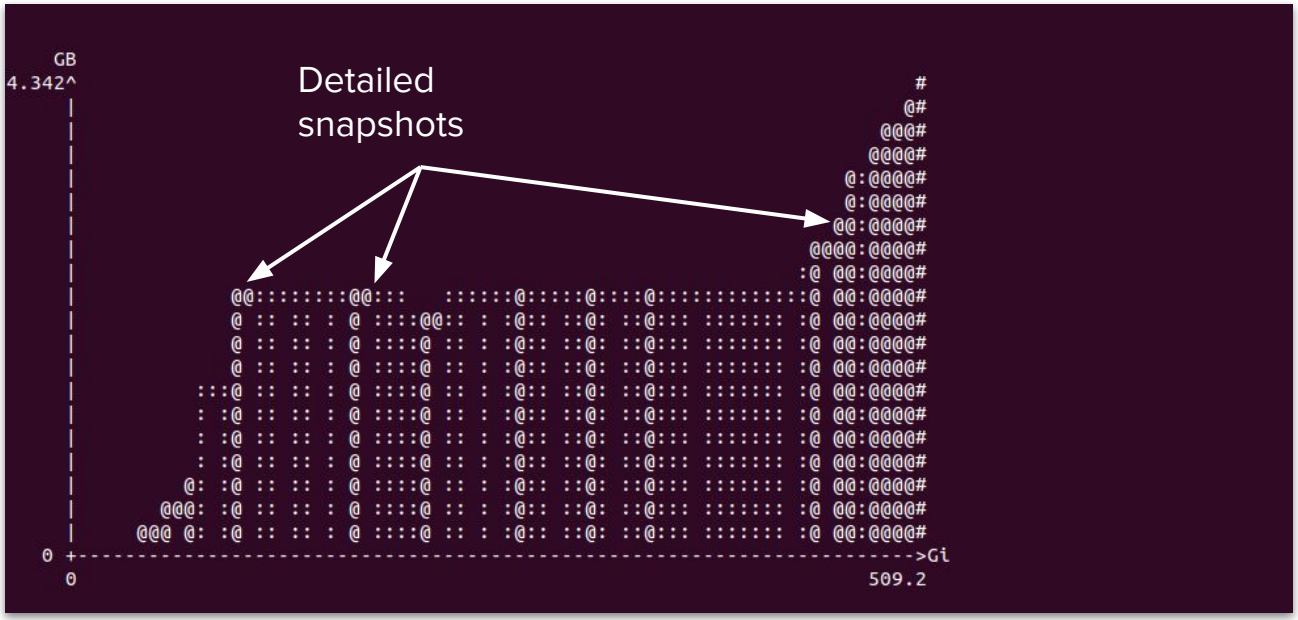
Valgrind

Valgrind is a profiler that comes with several tools for studying memory usage

- Massif categorizes memory allocated on the heap
  - I.e. global objects that persist regardless of scope – not local variables deallocated after functions are called
- Memcheck looks for memory issues like leaks (i.e. not deallocating/deleting pointers)

Massif Output

Total memory usage



Instructions performed

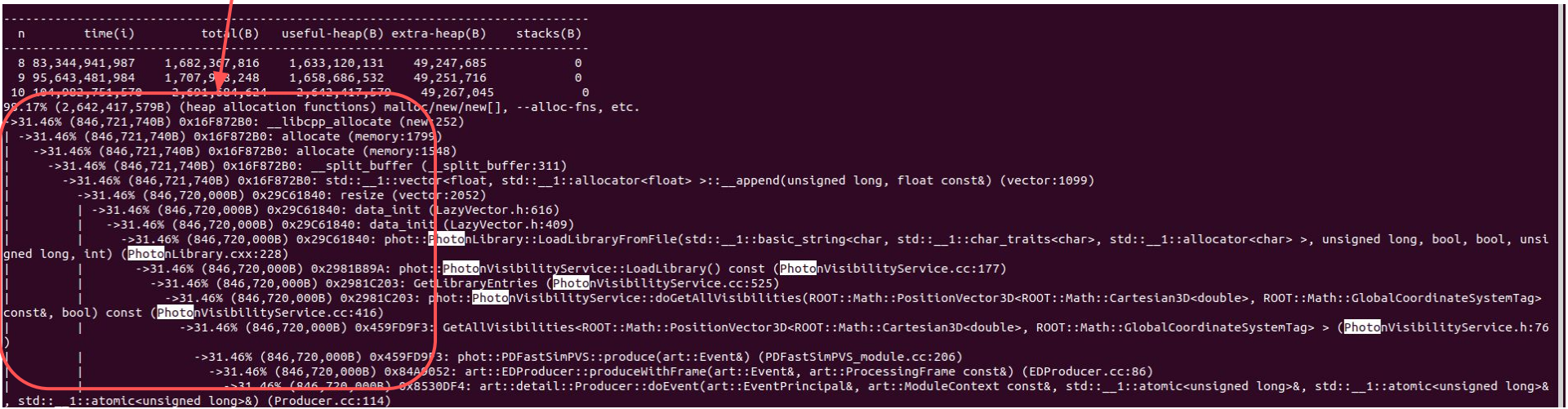
Massif Output

```
--------------------------------------------------------------------------------
  n        time(i)         total(B)   useful-heap(B) extra-heap(B)    stacks(B)
--------------------------------------------------------------------------------
  8 83,344,941,987    1,682,367,816    1,633,120,131    49,247,685            0
  9 95,643,481,984    1,707,938,248    1,658,686,532    49,251,716            0
 10 104,982,751,570    2,691,684,624    2,642,417,579    49,267,045            0
98.17% (2,642,417,579B) (heap allocation functions) malloc/new/new[], --alloc-fns, etc.
->31.46% (846,721,740B) 0x16F872B0: __libcpp_allocate (new:252)
| ->31.46% (846,721,740B) 0x16F872B0: allocate (memory:1799)
|   ->31.46% (846,721,740B) 0x16F872B0: allocate (memory:1548)
|     ->31.46% (846,721,740B) 0x16F872B0: __split_buffer (__split_buffer:311)
|       ->31.46% (846,721,740B) 0x16F872B0: std::__1::vector<float, std::__1::allocator<float> >::__append(unsigned long, float const&) (vector:1099)
|         ->31.46% (846,720,000B) 0x29C61840: resize (vector:2052)
|           | ->31.46% (846,720,000B) 0x29C61840: data_init (LazyVector.h:616)
|           |   ->31.46% (846,720,000B) 0x29C61840: data_init (LazyVector.h:409)
|           |     ->31.46% (846,720,000B) 0x29C61840: phot::PhotonLibrary::LoadLibraryFromFile(std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allocator<char> >, unsigned long, bool, bool, unsi
gned long, int) (PhotonLibrary.cxx:228)
|           |       ->31.46% (846,720,000B) 0x2981B89A: phot::PhotonVisibilityService::LoadLibrary() const (PhotonVisibilityService.cc:177)
|           |         ->31.46% (846,720,000B) 0x2981C203: GetLibraryEntries (PhotonVisibilityService.cc:525)
|           |           ->31.46% (846,720,000B) 0x2981C203: phot::PhotonVisibilityService::doGetAllVisibilities(ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<double>, ROOT::Math::GlobalCoordinateSystemTag>
const&, bool) const (PhotonVisibilityService.cc:416)
|           |             ->31.46% (846,720,000B) 0x459FD9F3: GetAllVisibilities<ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<double>, ROOT::Math::GlobalCoordinateSystemTag> > (PhotonVisibilityService.h:76
)
|           |               ->31.46% (846,720,000B) 0x459FD9F3: phot::PDFastSimPVS::produce(art::Event&) (PDFastSimPVS_module.cc:206)
|           |                 ->31.46% (846,720,000B) 0x84A9052: art::EDProducer::produceWithFrame(art::Event&, art::ProcessingFrame const&) (EDProducer.cc:86)
|           |                   ->31.46% (846,720,000B) 0x8530DF4: art::detail::Producer::doEvent(art::EventPrincipal&, art::ModuleContext const&, std::__1::atomic<unsigned long>&, std::__1::atomic<unsigned long>&
, std::__1::atomic<unsigned long>&) (Producer.cc:114)
```

Total useful memory at snapshot

Massif Output

Allocation tree



```
--------------------------------------------------------------------------------
  n        time(i)         total(B)   useful-heap(B) extra-heap(B)    stacks(B)
--------------------------------------------------------------------------------
  8 83,344,941,987    1,682,367,816    1,633,120,131    49,247,685            0
  9 95,643,481,984    1,707,9[illegible]8,248    1,658,686,532    49,251,716            0
 10 [illegible]    [illegible]    [illegible]    49,267,045            0
98.17% (2,642,417,579B) (heap allocation functions) malloc/new/new[], --alloc-fns, etc.
->31.46% (846,721,740B) 0x16F872B0: __libcpp_allocate (new:252)
| ->31.46% (846,721,740B) 0x16F872B0: allocate (memory:1799)
|   ->31.46% (846,721,740B) 0x16F872B0: allocate (memory:1548)
|     ->31.46% (846,721,740B) 0x16F872B0: __split_buffer (__split_buffer:311)
|       ->31.46% (846,721,740B) 0x16F872B0: std::__1::vector<float, std::__1::allocator<float> >::__append(unsigned long, float const&) (vector:1099)
|         ->31.46% (846,720,000B) 0x29C61840: resize (vector:2052)
|           | ->31.46% (846,720,000B) 0x29C61840: data_init (LazyVector.h:616)
|           |   ->31.46% (846,720,000B) 0x29C61840: data_init (LazyVector.h:409)
|           |     ->31.46% (846,720,000B) 0x29C61840: phot::PhotonLibrary::LoadLibraryFromFile(std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allocator<char> >, unsigned long, bool, bool, unsigned long, int) (PhotonLibrary.cxx:228)
|           |       ->31.46% (846,720,000B) 0x2981B89A: phot::PhotonVisibilityService::LoadLibrary() const (PhotonVisibilityService.cc:177)
|           |         ->31.46% (846,720,000B) 0x2981C203: GetLibraryEntries (PhotonVisibilityService.cc:525)
|           |           ->31.46% (846,720,000B) 0x2981C203: phot::PhotonVisibilityService::doGetAllVisibilities(ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<double>, ROOT::Math::GlobalCoordinateSystemTag> const&, bool) const (PhotonVisibilityService.cc:416)
|           |             ->31.46% (846,720,000B) 0x459FD9F3: GetAllVisibilities<ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<double>, ROOT::Math::GlobalCoordinateSystemTag> > (PhotonVisibilityService.h:76)
|           |               ->31.46% (846,720,000B) 0x459FD9F3: phot::PDFastSimPVS::produce(art::Event&) (PDFastSimPVS_module.cc:206)
|           |                 ->31.46% (846,720,000B) 0x84A0052: art::EDProducer::produceWithFrame(art::Event&, art::ProcessingFrame const&) (EDProducer.cc:86)
|           |                   ->31.46% (846,720,000B) 0x8530DF4: art::detail::Producer::doEvent(art::EventPrincipal&, art::ModuleContext const&, std::__1::atomic<unsigned long>&, std::__1::atomic<unsigned long>&, std::__1::atomic<unsigned long>&) (Producer.cc:114)
```

Massif Output

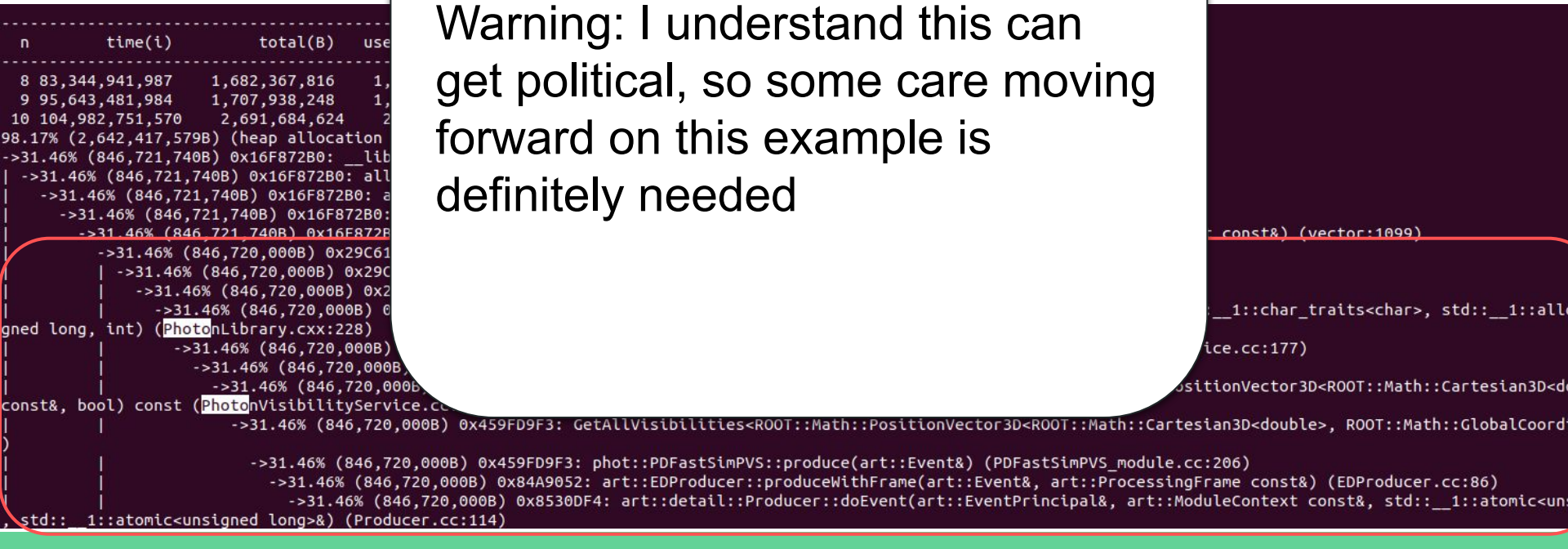
at this snapshot (and most others in this program) larsim's PhotonVisibilityService is one of the top two memory users ➜ Can this be improved?

```
--------------------------------------------------------------------------------
  n        time(i)         total(B)   useful-heap(B) extra-heap(B)    stacks(B)
--------------------------------------------------------------------------------
  8 83,344,941,987    1,682,367,816    1,633,120,131    49,247,685            0
  9 95,643,481,984    1,707,938,248    1,658,686,532    49,251,716            0
 10 104,982,751,570    2,691,684,624    2,642,417,579    49,267,045            0
98.17% (2,642,417,579B) (heap allocation functions) malloc/new/new[], --alloc-fns, etc.
->31.46% (846,721,740B) 0x16F872B0: __libcpp_allocate (new:252)
| ->31.46% (846,721,740B) 0x16F872B0: allocate (memory:1799)
|   ->31.46% (846,721,740B) 0x16F872B0: allocate (memory:1548)
|     ->31.46% (846,721,740B) 0x16F872B0: __split_buffer (__split_buffer:311)
|       ->31.46% (846,721,740B) 0x16F872B0: std::__1::vector<float, std::__1::allocator<float> >::__append(unsigned long, float const&) (vector:1099)
|         ->31.46% (846,720,000B) 0x29C61840: resize (vector:2052)
|           ->31.46% (846,720,000B) 0x29C61840: data_init (LazyVector.h:616)
|             ->31.46% (846,720,000B) 0x29C61840: data_init (LazyVector.h:409)
|               ->31.46% (846,720,000B) 0x29C61840: phot::PhotonLibrary::LoadLibraryFromFile(std::__1::basic_string<char, std::__1::char_traits<char>, std::__1::allo
gned long, int) (PhotonLibrary.cxx:228)
|                 ->31.46% (846,720,000B) 0x2981B89A: phot::PhotonVisibilityService::LoadLibrary() const (PhotonVisibilityService.cc:177)
|                   ->31.46% (846,720,000B) 0x2981C203: GetLibraryEntries (PhotonVisibilityService.cc:525)
|                     ->31.46% (846,720,000B) 0x2981C203: phot::PhotonVisibilityService::doGetAllVisibilities(ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<d
const&, bool) const (PhotonVisibilityService.cc:416)
|                       ->31.46% (846,720,000B) 0x459FD9F3: GetAllVisibilities<ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<double>, ROOT::Math::GlobalCoord
)
|                         ->31.46% (846,720,000B) 0x459FD9F3: phot::PDFastSimPVS::produce(art::Event&) (PDFastSimPVS_module.cc:206)
|                           ->31.46% (846,720,000B) 0x84A9052: art::EDProducer::produceWithFrame(art::Event&, art::ProcessingFrame const&) (EDProducer.cc:86)
|                             ->31.46% (846,720,000B) 0x8530DF4: art::detail::Producer::doEvent(art::EventPrincipal&, art::ModuleContext const&, std::__1::atomic<un
, std::__1::atomic<unsigned long>&) (Producer.cc:114)
```

Massif Output

at this snapshot (ar … isibilityService is one of the top two …

Warning: I understand this can get political, so some care moving forward on this example is definitely needed



Massif Peak Summaries

Gen: small amount of memory usage ~ 470 MB

G4/Photon: ~4.5 GB

- PhotonVisibility Service is a main contributor

Detsim: ~4 GB

- Biggest contributor appears to be art's delayed reader

Reco: ~3 GB

- ~800 MB, from various Wirecell methods
- 370 MB from DataPrep tool
- 115 MB Tensorflow

Memcheck output

Ran into issues with memcheck taking too long to run interactively and various grid issues so I didn’t get the chance to add the output to the presentation. Will update when ready