Running Geant4 Functions on a GPU
Discussion of Results

Stuart Douglas – dougls2
Rob Gorrie – gorrierw
Matthew Pagnan – pagnanmm
Victor Reginato – reginavp

McMaster University

April 14, 2016

Overview

Introduction
Features
Conclusion

Brief Project Overview
Explanation of Terms
Scope
Purpose

Brief Project Overview

Take an existing particle simulation toolkit - Geant4 - and have some functions run on a GPU device to improve performance.

Introduction
Features
Conclusion

Brief Project Overview
Explanation of Terms
Scope
Purpose

What is Geant4

- Geant4 is a toolkit that is meant to simulate the passage of particles through matter.
- It has been developed over the years through collaborative effort of many different institutions and individuals.
- Geant4's diverse particle simulation library has a wide variety of applications including
  - High energy physics simulations
  - Space and radiation simulations
  - Medical physics simulations

Introduction
Features
Conclusion

Brief Project Overview
Explanation of Terms
Scope
Purpose

What is GP-GPU

- General-purpose graphic processing unit computing is a re-purposing of graphics hardware
- Allows GPUs to perform computations that would typically be computed on the CPU
- If a particular problem is well suited to parallelization, GP-GPU computing can greatly increase performance

Introduction
Features
Conclusion

Brief Project Overview
Explanation of Terms
Scope
Purpose

Scope

- Make current CPU functions available for use on GPU
  - Add appropriate prefixes to function definitions
  - Make use of multiple parallel threads to execute each function
- Ensure correctness of each GPU available function by matching results to the corresponding CPU function
- Compare performance of GPU available functions to CPU functions

Introduction
Features
Conclusion

Brief Project Overview
Explanation of Terms
Scope
Purpose

Purpose

- Determine if target functions are suitable to parallelization
- Increase performance of functions when run on GPU
- Decrease time required to run simulations involving ported functions

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Features

- GPU acceleration available on an "opt-in" basis
- Easy to enable/disable GPU acceleration
- Same results whether acceleration enabled or disabled

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Easily Enable/Disable GPU Acceleration

- Existing projects can use GPU acceleration without having to change any code
- Flag during build phase enables/disables GPU acceleration
- No new functions to learn [1]

---

[1]implementation 1 only

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Implementation

- Header forwards function calls to GPU or CPU Implementation
- This decision is made at compile time.

```
inline G4double GetY(G4double x)
{
  #if GEANT4_ENABLE_CUDA
    return cudaVector->GetXsec(x);
  #else
    return GetXsec(x);
  #endif
}
```

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Accelerating Module on GPU

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Why `G4ParticleHPVector`

- Represents empirically-found probabilities of collisions for different particles based on their energy
- Identified as starting point by relevant stakeholders
    - Used heavily in simulations run by stakeholders
- Seems well-suited to parallelization
    - Based on large vector of 2D points
    - Performs calculations over this vector
    - Sorted by x-value (particle energy)

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Two Implementations

1 Forward all calls to existing `G4ParticleHPVector` interface to a GPU-based implementation of the module
   - Store data vector in GPU memory
   - Copy results back to the CPU to return to the caller

2 Add new methods to `G4ParticleHPVector` interface that are well-suited to GPU computing
   - Copy data vector to GPU memory on method call
   - Existing `G4ParticleHPVector` methods unchanged, continue to run on CPU

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Impl. 1: Existing Module in GPU Memory

Calls to `G4ParticleHPVector` forwarded to new GPU-based class

Pros:

+ Do not have to maintain a copy of the vector on the CPU
+ Do not have to maintain a hashed vector
+ Reduces how much is being copied to the GPU

Cons:

− All methods are run on the GPU

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Implementation – Times

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Implementation – `GetXSec`

Implementation – `SampleLin`

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results Summary

- Most methods slower on GPU until ~10,000 entries in data vector
- Most *commonly-used* methods significantly slower on GPU, even with large data vector
  - Lots of data accesses
- Many problems in vector class not well-suited to parallelism

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results – Times

- Multiplies each point in vector by factor
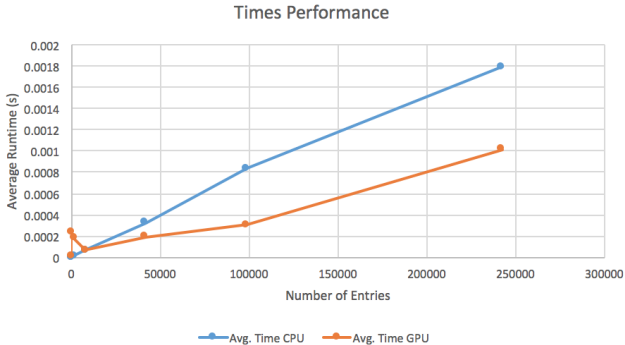
Figure: Runtime vs. Number of Data Points – Times

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results – `GetXSec`

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results – `SampleLin`

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results – System Tests

Performance Discussion

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Impl. 2: Add New GPU-Accelerated Methods to Interface

Add new methods to `G4ParticleHPVector` interface that are well-suited to parallelism

Pros:

+ Only methods that run faster on the GPU are implemented
+ Not forced to run methods that run slowly on GPU

Cons:

− Will have to maintain two copies of the vector
− More copying the vector to and from the GPU

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Implementation – `GetXSecList`

- Fill an array of energies we want to get xSec values for
- Send the array to the GPU to work on

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results Summary

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results – `GetXSecList`

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Results – System Tests

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Performance Discussion

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Accuracy

Introduction
Features
Conclusion

Easily Enable/Disable GPU Acceleration
Impl. 1: Existing Module in GPU Memory
Impl. 2: Add New GPU-Accelerated Methods to Interface
Accuracy / Testing

Testing

Summary of Results

Recommendations