GPU Teaching Kit

Accelerated Computing

Lecture 1.1 – Course Introduction

Course Introduction and Overview

Course Goals

- Learn how to program heterogeneous parallel computing systems and achieve
  - High performance and energy-efficiency
  - Functionality and maintainability
  - Scalability across future generations
  - Portability across vendor devices
- Technical subjects
  - Parallel programming API, tools and techniques
  - Principles and patterns of parallel algorithms
  - Processor architecture features and constraints

People

- Wen-mei Hwu (University of Illinois, NVIDIA)
- David Kirk (NVIDIA)
- Joe Bungo (NVIDIA)
- Mark Ebersole (formerly NVIDIA)
- Abdul Dakkak (Microsoft, formerly University of Illinois)
- Izzat El Hajj (American University of Beirut, formerly University of Illinois)
- Andy Schuh (University of Illinois)
- John Stratton (Whitman College)
- Isaac Gelado (NVIDIA)
- John Stone (NVIDIA, formerly University of Illinois)
- Javier Cabezas (AMD, formerly NVIDIA)
- Michael Garland (NVIDIA)

Course Content

Module 1: Course Introduction	• 1.1 - Course Introduction and Overview • 1.2 - Introduction to Heterogeneous Parallel Computing • 1.3 - Portability and Scalability in Heterogeneous Parallel Computing
Module 2: Introduction to CUDA C	• 2.1 - CUDA C vs. CUDA Libs vs. OpenACC • 2.2 - Memory Allocation and Data Movement API Functions • 2.3 – Threads and Kernel Functions • 2.4 - Introduction to CUDA Toolkit • 2.5 – Nsight Compute and Nsight Systems • 2.6 – Unified Memory
Module 3: CUDA Parallelism Model	• 3.1 - Kernel-Based SPMD Parallel Programming • 3.2 - Multidimensional Kernel Configuration • 3.3 - Color-to-Greyscale Image Processing Example • 3.4 - Blur Image Processing Example • 3.5 - Thread Scheduling
Module 4: Memory Model and Locality	• 4.1 - CUDA Memories • 4.2 - Tiled Parallel Algorithms • 4.3 - Tiled Matrix Multiplication • 4.4 - Tiled Matrix Multiplication Kernel • 4.5 - Handling Arbitrary Matrix Sizes in Tiled Algorithms
Module 5: Thread Execution Efficiency	• 5.1 - Warps and SIMD Hardware • 5.2 - Performance Impact of Control Divergence

Course Content

Module 6: Memory Access Performance	• 6.1 - DRAM Bandwidth • 6.2 - Memory Coalescing in CUDA
Module 7: Parallel Computation Patterns (Histogram)	• 7.1 - Histogramming • 7.2 - Introduction to Data Races • 7.3 - Atomic Operations in CUDA • 7.4 - Atomic Operation Performance • 7.5 - Privatization Technique for Improved Throughput
Module 8: Parallel Computation Patterns (Stencil)	• 8.1 - Convolution • 8.2 - Tiled Convolution • 8.3 - Tile Boundary Conditions • 8.4 - Analyzing Data Reuse in Tiled Convolution
Module 9: Parallel Computation Patterns (Reduction)	• 9.1 - Parallel Reduction • 9.2 - A Basic Reduction Kernel • 9.3 - A Better Reduction Kernel
Module 10: Parallel Computation Patterns (Scan)	• 10.1 - Prefix Sum • 10.2 - A Work-inefficient Scan Kernel • 10.3 - A Work-Efficient Parallel Scan Kernel • 10.4 - More on Parallel Scan

Course Content

Module 11: Breadth-First (BFS) Queue	• 11.1 – Breadth-First (BFS) Queue
Module 12: Floating Point Considerations	• 12.1 - Floating Point Precision Considerations • 12.2 - Numerical Stability
Module 13: GPU as part of the PC Architecture	• 13.1 - GPU as part of the PC Architecture
Module 14: Efficient Host-Device Data Transfer	• 14.1 - Pinned Host Memory • 14.2 - Task Parallelism in CUDA • 14.3 - Overlapping Data Transfer with Computation • 14.4 - CUDA Unified Memory
Module 15: Application Case Study: Advanced MRI Reconstruction	• 15.1 - Advanced MRI Reconstruction • 15.2 - Kernel Optimizations
Module 16: Application Case Study: Electrostatic Potential Calculation	• 16.1 - Electrostatic Potential Calculation (Part 1) • 16.2 - Electrostatic Potential Calculation (part 2)

Course Content

Module 17: Computational Thinking for Parallel Programming	• 17.1 - Introduction to Computational Thinking
Module 18: Related Programming Models: MPI	• 18.1 - Introduction to Heterogeneous Supercomputing and MPI
Module 19: CUDA Python Using Numba	• 19.1 - CUDA Python using Numba
Module 20: Related Programming Models: OpenCL	• 20.1 - OpenCL Data Parallelism Model • 20.2 - OpenCL Device Architecture • 20.3 - OpenCL Host Code (Part 1)
Module 21: Related Programming Models: OpenACC	• 21.1 - Introduction to OpenACC • 21.2 - OpenACC Subtleties
Module 22: Related Programming Models: OpenGL	• *Module scheduled for a future release of the teaching kit*

Course Content

Module 23: Dynamic Parallelism	• 23.1 - Dynamic Parallelism
Module 24: Multi-GPU	• 24.1 - OpenMP • 24.2 - Multi-GPU Introduction I • 24.3 - Multi-GPU Introduction II • 24.4 - OpenMP and Cooperative Groups • 24.5 - Multi-GPU Heat Equation
Module 25: Using CUDA Libraries	• 25.1 - cuBLAS • 25.2 - cuSOLVER • 25.3 - cuFFT • 25.4 - Thrust
Module 26: Advanced Thrust	• *Module scheduled for a future release of the teaching kit*

GPU Teaching Kit

Accelerated Computing