CS6235: Parallel Programming for Many-Core Processors (3 units)

Schedule: MW 8:05-9:25 AM
Location: WEB L110
Instructors: Mary Hall (MEB 3252, mhall@cs.utah.edu**) & Saday Sadayappan (MEB 3458,** saday@cs.utah.edu**)**
Office Hours: TBA
Teaching Assistants: Lizhi Xiang (u0814474@utah.edu**) & Bo Zhang (**bozhang@sci.utah.edu**)**
Office hours: TBA

Course Summary
This course examines an important trend in high-performance computing, the use of special-purpose hardware originally designed for graphics and games to solve general-purpose computing problems. Such graphics processing units (GPUs) have enormous peak performance for arithmetically-intensive computations, and at relatively low cost and low energy consumption as compared to their general-purpose counterparts with similar performance levels. Technology trends are driving all microprocessors towards many-core designs, and a diversity of related accelerator hardware. Therefore, techniques for parallel programming represent a rich area of recent study. Students in the course will learn how to develop scalable parallel programs targeting the unique requirements for obtaining high performance on GPUs and related accelerators. We will compare and contrast parallel programming for GPUs and conventional multi-core microprocessors.

The course will largely consist of small individual programming assignments, in-class homework and programming assignments, and a larger term project to be presented to the class. Several of these projects from previous years were presented in conferences, workshops and poster sessions, and contributed to Masters and PhD research. As this course combines hands-on programming and a discussion of research in the area, it is suitable for students who wish to learn how to write parallel applications or are engaged in research in related areas.

Prerequisites
Basic knowledge of: programming in C (CS4400 or equivalent); algorithms and data structures (CS4150 or equivalent) and computer architecture or game hardware architecture (CS3810 or equivalent).

Textbooks and Resources
[Recommended] P*rogramming Massively Parallel Processors: A Hands-on Approach*. *David B. Kirk and Wen-mei W. Hwu Morgan Kaufmann, Third Edition (Dec. 2016)*, *ISBN* 0128119861.

[Online Resource] NVidia, *CUDA Programming Guide,* https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html for Windows, Linux or MAC OS.

[Reference] Grama, A. Gupta, G. Karypis, and V. Kumar, *Introduction to Parallel Computing, 2nd Ed.* (Addison-Wesley, 2003).

Grading

Projects	50%	
Assignments	30%	
Midterm test	20%	(March 15)

Assignments

Late programming assignments will incur a 20% penalty per day.

Working Together

Students are encouraged to discuss individual project assignments with fellow classmates, but each student is responsible for writing her own answer. *Cheating is:* sharing code or other electronic files either by copying, retyping, looking at, or supplying a copy of a file. *Cheating is not:* discussing concepts, answering questions about concepts or clarifying ambiguities, helping someone understand how to use the computer systems or basic tools (compiler, debugger, etc.), or helping with high-level design issues or general debugging.

The major project will be a group effort. In this case, working together is encouraged. However, students must attribute any sources of code that are used in the project.

Of course, there must be no collaboration during the midterm examinations. Please see the University of Utah Student Code for a detailed description of the university policy on cheating.

Classroom Behavior

All students are expected to maintain professional behavior in the classroom setting, according to the University of Utah Student Code, which is posted at http://www.regulations.utah.edu/academics/6-400.html

Students should read the Code carefully and know that they are responsible for the content. According to Faculty Rules and Regulations, it is the faculty responsibility to enforce responsible classroom behaviors, beginning with verbal warnings and progressing to dismissal from class and a failing grade. Students have the right to appeal such action to the Student Behavior Committee.

College of Engineering Guidelines

For more information, see
https://www.coe.utah.edu/wp-content/uploads/2022/08/COE-Guidelines-Fall-2022-1.pdf

Inclusivity Statement

It is our intent that students from all diverse backgrounds and perspectives be well-served by this course, that students' learning needs be addressed both in and out of class, and that the diversity that the students bring to this class be viewed as a resource, strength and benefit. It is our intent to present materials and activities that are respectful of diversity: gender identity, sexuality, disability, age, socioeconomic status, ethnicity, race, nationality, religion, and culture. Your suggestions on how we can make the course more inclusive and welcoming are encouraged and appreciated.

We also expect students to treat others in the class, including the teaching staff, with the same level of respect.

We take incidents of discrimination, bias, and harassment seriously. We will file reports with the Office or Equal Opportunity, Affirmative Action, and Title IX (OEO) about such incidents. If you are unsure what differentiates free speech and professional behavior from discrimination, bias, and harassment we are happy to have an open, judgment-free, and confidential conversation with you, or refer you to the OEO.

Students with Disabilities

The University of Utah seeks to provide equal access to its programs, services and activities for people with disabilities. If you will need accommodations in the class, reasonable prior notice needs to be given to the Center for Disability Services, 162 Union Building, 581-5020 (V/TDD). CDS will work with you and the instructor to make arrangements for accommodations.

Syllabus and Topic Outline (subject to change)

PART I: FOUNDATION

Introduction
CUDA, GPUs, Nvidia Architecture
(Kirk and Hwu, Ch. 1, 3)

Writing Correct Parallel Programs
Dependences, synchronization, barriers, atomic operations

Hardware and Execution Model
SIMT, Multithreading, Scheduling
(Kirk and Hwu, Ch. 2-3)

Memory Hierarchy Optimization
Data placement, locality optimizations, memory bandwidth optimizations, data layout
(Kirk and Hwu, Ch. 4-5)

PART II: OPTIMIZATION FRAMEWORKS

Performance Analysis
Roofline model, Nvidia NSight tool for performance monitoring, performance bottleneck analysis

Data Representation
Floating point accuracy, reduced precision
(Kirk and Hwu, Ch. 6)

Programmer Productivity
Libraries, Performance-portability frameworks (Kokkos), Autotuning, Domain-specific frameworks (CUTLASS for dense matrix computations)
(Kirk and Hwu, Ch. 19)

Mapping to Special-Purpose Hardware
Tensor cores and sparse tensor cores

PART III (throughout): PARALLEL PATTERNS

Parallel Programming Patterns
Dense and sparse linear algebra, stencils and image processing, parallel primitives, application case studies
(Kirk and Hwu, Chs. 8-12)

ADDITIONAL TOPICS, AS NEEDED FOR PROJECTS

Multi-GPU implementations, Dynamic Parallelism, Memory Management, Other Patterns