Master Course Description for EE-479

Title: High-Performance GPU Computing

Credits: 4

EE 479: High-Performance GPU Computing Introduction to high performance computing (HPC) and GPUs. GPU based systems, microarchitectural structures, memory hierarchies, programming models, and general strategies to harness their computational power. Emphasis is placed on GPU architectures due to their computation power, ubiquity, broad applicability to a variety of application domains (including machine learning), and extraordinary low cost relative to historical high performance supercomputers.

Prerequisite: CSE 373 and CSE 374.

Coordinator: Jeffrey Bilmes, Professor, Electrical and Computer Engineering

Goals: The broad goals are to introduce the student to various concepts within the field of high-performance computing and its applications, as embodied in GPU-based systems. As computing system hardware improvements continue to decelerate, and as Moore's law runs out of steam, it is becoming ever more important for the student to learn to be aware of: (1) various microarchitectural features that are common in commodity microprocessors; (2) the architectural features common to what we consider to be commodity supercomputers (i.e., GPUs); and (3) advanced HPC programming concepts and techniques that can exploit the above to achieve high machine efficiency. This will prepare students in the ways of HPC, thereby offering them the ability to create systems that run faster, use less energy, and also reduce overall cost.

Learning Objectives: At the end of this course, students will be able to:

1. Understand basic memory hierarchies, and caches.
2. Produce software that can exploit such hierarchies to achieve high speed.
3. Comprehend parallel models of computing.
4. Recognize data flow and various programming models.
5. Identify sources of parallelism in standard control flows.
6. Test and optimize vector high performance kernels.
7. Contrast the computation/communication dichotomy.
8. Write fast code for matrix-matrix multiply and other core kernels.
9. Understand SIMD and other data parallel models.
10. Feel comfortable with standard vector instruction sets, such as CUDA and openCL as one finds with GPUs.
11. Know about grid-based parallelism.

12. Realize why machine learning (ML) is so important, ubiquitous, and the role commodity supercomputing such as GPUs has played in making ML such a huge and widespread success.
13. Recognize the role other important algorithms, such as the Fourier transform, have in HPC.
14. Know cloud computing and its role in fostering HPC going into the future.
15. Have experience writing GPU code and running it on a modern GPU.

Textbook: No book, only handouts (URLs to online PDF files) and slides.

Prerequisites by Topic:

1. Mature knowledge of computer programming, and the C language (CSE 374).
2. Data structures and algorithms (CSE 373).

Topics:

1. Memory Hierarchies and Matrix Multiplication
2. Shared Memory Parallelism
3. Roofline and Performance Modeling
4. Sources of Parallelism and Locality
5. Communication-avoiding matrix multiplication
6. Data Parallel Algorithms
7. GPUS, CUDA and OpenCL
8. Distributed Memory Machines and Programming, SMPs
9. MPI and Collective Communication Algorithms
10. Parallel Matrix Multiply
11. Dense Linear Algebra and Convolution
12. Sparse-Matrix-Vector-Multiplication
13. Grids
14. Deep Machine Learning
15. Fast Fourier Transform
16. Graphs
17. Cloud Computing and HPC

Course Structure: The class meets for two 110-minute lectures a week. Homework and programming assignments are given approximately weekly. The course includes one midterm and a comprehensive final exam.

Computer Resources: Programming assignments require a high-performance GPU, such as those provided in the ECE department's Linux Lab (Sieg 118).

Grading: Approximate distribution: Homework 10%, Programming Assignments 40%, Midterm 20%, Final Exam 30%. The grading scheme in any particular offering is the prerogative of the instructor.

ABET Student Outcome Coverage: This course addresses the following outcomes:

H = high relevance, M = medium relevance, L = low relevance to course.

(1)*An ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics* **(H)** Students create high-performance software on GPU-based systems via best practices of engineering and mathematics..

(2) *An ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors* **(M)** Students design computational solutions to important problems via high-performance computing resources. These consider economic factors by optimizing the cost/performance and power/performance tradeoffs, and can provide efficient solutions to problems in health, environment, and other domains.

(5) *An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives* **(M)** Programming assignments include group work and collaboration.

(6) *An ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions* **(M)** Programming assignments require experimentation, analysis, and reoptimization to create efficient implementations.

(7) *An ability to acquire and apply new knowledge as needed, using appropriate learning strategies* **(M)** Students must access literature on problem domains, web resources on programming strategies, and other techniques to seek out and apply new knowledge to solve the assignments for this class..

Prepared By: Jeffrey Bilmes, Scott Hauck

Last Revised: May 18, 2020