University of Delaware – Computer and Information Science

CISC 879-Spring 2015

Advanced Parallel Programming

Instructor: Dr. John Cavazos **Office Hours:** By appointment
Email: cavazos@cis.udel.edu

Textbook: No Required Text

Exams: No Midterm/Final

Grading:
1. Paper presentation (30%)
2. Projects (80%)
   - Project 1 (20%)
     - Project report/1 checkpoint
   - Project 2 (50%)
     - Project report/presentation/checkpoint

Webpage: www.cis.udel.edu/~cavazos/cisc879

Background:

Parallel programming and computer architecture background helpful, but not required. A reasonable familiarity of the C programming language is necessary. You might have to know some C++ depending on the programs you choose for your project.

Objectives of the course:

This course is intended to give students an understanding of many-core (multicore and GPU) architectures and parallel programming models. We recommend using OpenCL or CUDA, but students are free to use other parallel programming languages (e.g., OpenMP, MPI, etc.) as long as they get an appreciation of the problems and solutions researchers have identified in the field of many-cores. Also, students will get experience in presenting research. Finally, students will get a thorough understanding of how to write parallel programs for current many-core architectures.

Programming Environment:

We will be using Linux-based EECIS servers for the programming assignments in this class.

MLB (mlb.acad.ece.udel.edu) will be used for OpenMP and MPI programming
CUDA (cuda.acad.ece.udel.edu) will be used for GPU programming

An EECIS account is **required** for this class. If you don't have an account, please sign up for one here: https://accounts.eecis.udel.edu/register/

To learn about the machine configurations, please consult the following link:
https://www.eecis.udel.edu/wiki/ececis-docs/index.php/FAQ/GeneralInformation

Projects:

Most of your grade will come from two projects, which will total 70% of your grade. Both projects are mandatory. Each student is encouraged to form a team. We are aiming for groups of 2 to 3 members per team. Plan on check pointing your project with the instructor (showing your progress) at various stages during your project. You must check point ***at least once*** for each project. These checkpoints will be part of your grade. You are not required to stay with the same group for both projects.

The **first** project involves getting acquainted with the tools and programming environment of the multicore you will be using for both projects. Project 1 involves taking one small kernel and getting it to run on a GPU or multicore CPU using OpenCL, CUDA, or OpenMP. Then, you will time the parallel and sequential versions of the code and report any speedups obtained. The second part of the project will be to compile, instrument, and run a larger application and collect profile statistics. The goal of this part of the project is to identify the most computationally expensive part(s) of the program. You will write a report of 2 pages per team member describing the parallelization of the program and the performance obtained. You should also report on your profiling experiments with the larger application. We will provide a latex template to help you understand what we want for the report.

For the **second** project, each group will parallelize two applications that you choose from a set of bioinformatics, data mining, financial, or your own applications. There are a handful of milestones associated with doing this project, including a project report (~3 pages per team member) and a short presentation of your project at the end of the semester. You are required to write a report describing the two applications you choose and discussing in detail the critical parts of the code you parallelized. You will also have to report the performance improvement you obtained over serial version of the code and discuss any optimizations (either manual or automatic) you used to improve performance.

Project Report Guidelines:

Papers must be well written and formatted correctly. We expect you to clearly describe the project and experimental results. You are required to present experimental results using graphs. We recommend using the latex template provided which will specify the font size and margins. You are expected to properly reference any related material. That is, part of the project is to go on the Internet (e.g., using Google Scholar or Citeseer) to find papers related to the work you're doing. A negative result is fine (e.g., degradation in performance compared to the sequential version), as long as you demonstrate that you found something interesting along the way.

Expectations:

- Class Participation
- Ask questions
- Challenge speakers
- Not a lecture-type course or passive-learning experience

Policy on Academic Dishonesty

You are allowed and even encouraged to discuss material covered in class, requirements of the assignment, features of the parallel programming models and/or language, tools or programming environment used, and coding and debugging techniques with anyone you desire.

Also, students choosing to work as a team are freely allowed to exchange, help, design, and code with all members of the team. It is fine to study code found on the Internet, but you cannot use substantial parts of code found on the Internet in your project.