Cornell Bowers C·IS
College of Computing
and Information Science

Applied High-Performance and Parallel Computing

Spring 2024

CS 5220

Cornell Bowers CIS Department of Computer Science

Course website: ________________________

Faculty Name(s): Giulia Guidi

Faculty Email: gguidi@cs.cornell.edu

Faculty Office Hours: Gates 437, Wed 11-12 PM or by appointment

Course Staff and Course Staff Office Hours:

Professor: Giulia Guidi

TA: TBD

Credits and Credit Hour Options: 4.0 credits, Letter grade

- 3 credits for the lecture/homework component, and 1 additional credit for independent work on the final project.

Prerequisites/Corequisites: CS3410 or equivalent and be familiar with C/C++, i.e., students should be able to read and write serial programs written in C or a related language. A course in numerical analysis and linear algebra may be helpful.

Time and Location: This course meets Tuesdays and Thursdays, 1:25 – 2:40 PM, for a total of 26 lectures.

Course Description

Overview of computer architecture and memory hierarchy, performance basics, parallel programming models, and survey of parallel machines. Parallel programming languages, vectorizing compilers, parallel libraries and toolboxes, overview of modern parallel algorithms. The course homework and final project focus primarily on hands-on parallel programming and performance tuning and analysis, in-class active learning activities focus primarily on theoretical understanding.

Course Objectives/Student Learning Outcomes

By the end of the course, a student will be able to:

- Describe single processor parallelism and architectural features, especially cache hierarchies.
- Describe parallel machine organizations and programming models (distributed memory and shared memory machines, distributed and shared memory programming using MPI, OpenMP, SYCL, CUDA, and UPC++).
- Recognize and describe standard parallel patterns for data partitioning, synchronization, and load balancing common in scientific computing.
- Optimize the serial performance of existing programs.
- Parallelize and measure the performance of various programs.

Course Materials

The course will be taught from lecture notes available on the course web page**.**

Method of Assessing Student Achievement

- **Basis of Grade Determination:** Class participation (10%), five to six homework (55%), and a final project (35%)
- **Grading Scale:** This class adheres to Cornell's grading scale:
    - A+*: 98-100%*
    - A*: 93-97%*
    - A-*: 90-92%*
    - B+*: 88-89%*
    - B*: 83-87%*
    - B-*: 80-82%*
    - C+*: 78-79%*
    - C*: 73-77%*
    - C-*: 70-72%*
    - D*: 60-69%*
    - F*: Below 60%*

Course Management

<u>Collaboration</u>

An assignment is an academic document, like a journal article. When you turn it in, you are claiming everything in it is your original work, *unless you cite a source for it*.

You are welcome to discuss homework and projects among yourselves in general terms.

However, you should not look at code or writeups from other students or allow other students to see your code or writeup, even if the general solution was worked out together. Unless we explicitly allow it on an assignment, we will not credit code or writeups that are shared between students (or teams, in the case of team assignments).

If you get an idea from a classmate, the TA, a book, or other published source, or elsewhere, please provide an appropriate citation. This is not only critical to maintaining academic integrity, but it is also an important way for you to give credit to those who have helped you out. When in doubt, cite! Code or writeups with appropriate citations will never be considered a violation of academic integrity in this class (though you will not receive credit for code or writeups that were shared when you should have done them yourself).

Academic Integrity

We expect academic integrity from everyone. School is stressful, and you may feel pressure from your coursework or other factors, but that is no reason for dishonesty! If you feel you can't complete the work on your own, come talk to the professor, the TA, or your advisor, and we can help you figure out what to do.

For more information, see Cornell's Code of Academic Integrity.

Emergency procedures

In the event of a major campus emergency, course requirements, deadlines, and grading percentages are subject to changes that may be necessitated by a revised semester calendar or other circumstances. Any such announcements will be posted to Piazza and the course home page.

Sample Course Schedule Chart

Week, date	Topic
1, 1/23	**Lecture 1: Introduction & Performance Basics**
1, 1/25	**Lecture 2: Memory Hierarchies and Matrix Multiplication**
2, 1/30	**Lecture 3: Single Processor Architecture and More MatMul and SIMD**
2, 2/01	**Lecture 4: Roofline Modeling**
3, 2/06	**Lecture 7: Shared Memory Programming**
3, 2/08	**Lecture 8: GPU Programming (Part 1)**
4, 2/13	**Lecture 9: GPU Programming and SYCL (Part 2)**

4, 2/15	**Lecture 10: Distributed Memory Programming (Part 1)**
5, 2/20	**Lecture 11: Distributed Memory Programming (Part 2)**
5, 2/22	**Lecture 12: Parallelism and Locality (Part 1)**
6, 2/29	**Lecture 13: Parallelism and Locality (Part 2)**
7, 3/05	**Lecture 14: Dense Linear Algebra**
7, 3/07	**Lecture 15: Sparse Linear Algebra (Part 1)**
8. 3/12	**Lecture 16: Sparse Linear Algebra (Part 2)**
8, 3/14	**Lecture 17: UPC++ and PGAS**
9, 3/19	**Lecture 18: Machine Learning (Part 1)**
9, 3/21	**Lecture 19: Machine Learning (Part 2)**
10, 3/26	**Lecture 20: Graph Partitioning**
10, 3/28	**Lecture 21: Load Balancing and Work Stealing**
12, 4/09	**Lecture 22: N-Body Simulation**
12, 4/11	**Lecture 23: Parallel Graph Algorithms**
13, 4/16	**Lecture 24: Computational Biology (Part 1)**
13, 4/18	**Lecture 25: Computational Biology (Part 2)**
14, 4/23	**Lecture 26: Cloud Computing (Part 1)**
14, 4/25	**Lecture 27: Cloud Computing (Part 2)**
15, 4/30	**Lecture 28: Guest Lecture**
15, 5/02	**Lecture 29: Guest Lecture**
15, 5/07	**Final Project Lighting Talk Session**

NOTE: This schedule is subject to change as needed during the semester.