15-418/618: Parallel Computer Architecture and Programming
Fall 2020 Syllabus

1 Course Details at a Glance

Lectures:	MWF, 8:00am-9:20am, via Zoom
Instructors:	Todd C. Mowry, GHC 9113, 268-3725, `tcm@cs.cmu.edu`
	Brian P. Railing, GHC 6005, `bpr@cs.cmu.edu`
TAs:	Abhijith Anilkumar, `aanilkum@andrew.cmu.edu`
	Arthur Dzieniszewski, `adzienis@andrew.cmu.edu`
	Andrew Yang, `atyang@andrew.cmu.edu`
	Oscar Dadfar, `odadfar@andrew.cmu.edu`
	Ziqi Wang, `ziqiw@andrew.cmu.edu`
Web Page:	http://www.cs.cmu.edu/afs/cs/academic/class/15418-f20/www/
Discussion:	http://www.piazza.com/cmu/fall2020/1541815618

2 Course Description

From smart phones, to multi-core CPUs and GPUs, to the world's largest supercomputers and web sites, parallel processing is ubiquitous in modern computing. The goal of this course is to provide a deep understanding of the fundamental principles and engineering trade-offs involved in designing modern parallel computing systems as well as to teach parallel programming techniques necessary to effectively utilize these machines. Because writing good parallel programs requires an understanding of key machine performance characteristics, this course will cover both parallel hardware and software design.

Course themes include: (i) designing and writing parallel programs that scale effectively to large numbers of processors, (ii) understanding how parallel computers work (since this is very important in order to write fast and efficient parallel software), and (iii) thinking about efficiency (which is not equivalent to speed).

Here are the *learning objectives* for the course:

1. Design efficient and scalable parallel programs by applying algorithmic and architectural knowledge to utilize parallel computing resources;

2. Analyze and contrast the different parallel programming models (e.g., shared memory, heterogeneous, data-parallel, task-oriented, message-passing) and frameworks (e.g., OpenMP, Cilk, Cuda, MPI);

3. Investigate the common parallel and heterogeneous computing technologies and the costs associated with each;

4. Demonstrate appropriate measurement of parallel computing execution;

5. Understand the basics of ubiquitous computing workloads driving modern parallel systems design (e.g., graph processing, deep learning, imaging and graphics) and describe the models and frameworks best suited to each;

6. Explain in a clear language solutions to parallel computing problems.

3 Prerequisites

15-213 (*Intro to Computer Systems*) is a strict prerequisite for this course. We will build directly upon the material presented in 15-213, including memory hierarchies, memory management, basic networking, etc. While 18-447 (*Intro to Computer Architecture*) would be helpful for understanding the material in this course, it is not a prerequisite. Students are expected to be strong C/C++ programmers as there will be exposure to a variety of "C-like" parallel programming languages in this course. Previous experience with operating systems (15-410), distributed systems (15-440), or compilers (15-411) may prove to be very helpful.

4 Impact of COVID-19 on Class Logistics

Fortunately there is very little change in our class logistics compared with previous years. The main difference is that we will not be meeting in person for the lectures.

Here is how we will be handling lecture delivery and our lecture time slot. Lectures will be pre-recorded and posted in advance of the date when they officially appear on the lecture schedule, so that you can view them at your convenience. We will use our official lecture slot as a time to review the lecture material for that day (with the assumption that students have already viewed the pre-recorded lecture), giving students a chance to ask questions, review concepts further, etc. These interactive discussions during the official lecture slot will be recorded via Zoom, and made available via Canvas.

5 Textbook

There is no required textbook for 15-418/618. If you are looking for additional (optional) reference material beyond the lecture notes, you might want to consider the following book:

- John L. Hennessy and David A. Patterson. *Computer Architecture, Sixth Edition: A Quantitative Approach*. Morgan Kaufmann, 2017

6 Course Work

Here is what you will be doing in this class.

Programming assignments: Students will complete four programming assignments. Assignment 1 will be performed individually. The remaining three assignments will be performed in groups of two.

Exams: There will be two in-class exams. Each exam will cover roughly half of the course material (i.e. the second exam is not cumulative). There is no final exam in the course.

Written assignments: There will be two or three written assignments. These will be pencil-and-paper style questions, and they will not involve any programming.

Final project: Over the last six weeks of the course, students will propose and complete a self-selected final project. By default, the final project will be performed in groups of two. Each team will present the project during a poster session (held during the normal final exam time) and produce a detailed write-up describing their work and results.

Class participation quizzes: On the scheduled day of a lecture, we will post a simple quiz on Canvas. These quizzes should be easy: the goal is to just to demonstrate that you are keeping up with the lecture material. (The quizzes also help give us feedback on what the class is understanding.)

6.1 Grading Policy

Your overall grade will be determined as follows:

Programming Assignments (4):	40% (exact weighting is TBD)
Exams (2):	20% (10% each)
Written Assignments:	10%
Final Project:	25 %
Participation:	5 %

The grading cutoff points are: 90% (A), 80% (B), 70% (C), 60% (D). We will selectively consider raising individual grades for students just below the cutoffs based on factors such as attendance, class participation, improvement throughout the course, exam performance, project performance, and special circumstances.

Late hand-in policy. Each student is allotted a total of five late-day points for the semester. Late-day points are for use on programming assignments only. (They cannot be used for quizzes or final projects) Late-day points work as follows:

- A one-person team can extend a programming assignment deadline by one day using one point.
- A two-person team can extend a programming assignment deadline by one day using two points. (e.g., one point from each student)
- If a team does not have remaining late day points, late hand-ins will incur a 10% penalty per day (up to three days per assignment).
- No assignments will be accepted more than three days after the deadline.

6.2 Collaboration Policy

Students in 15-418/618 are absolutely encouraged to talk to each other, to the TAs, to the instructors, or to anyone else about course assignments. Any assistance, though, must be limited to discussion of the problems and sketching general approaches to a solution. Each programming project team must write their own code and produce their own writeup. **Consulting another student's or team's solution, or solutions from the internet, is prohibited. These and any other form of collaboration on assignments constitute cheating.** If you have any question about whether some activity would constitute cheating, just be cautious and ask the instructors before proceeding!

You may not supply code, assignment writeups, or exams you complete during 15-418/618 to other students in future instances of this course or make these items available (e.g., on the web) for use in future instances of this course (just as you may not use work completed by students who've taken the course previously). Make sure to make repositories private if you use public source control hosts like github.

6.3 Student Wellness

Take care of yourself. Do your best to maintain a healthy lifestyle this semester by eating well, exercising, avoiding drugs and alcohol, getting enough sleep and taking some time to relax. This will help you achieve your goals and cope with stress.

All of us benefit from support during times of struggle. You are not alone. There are many helpful resources available on campus and an important part of the college experience is learning how to ask for help. Asking for support sooner rather than later is often helpful.

If you or anyone you know experiences any academic stress, difficult life events, or feelings like anxiety or depression, we strongly encourage you to seek support. Counseling and Psychological Services (CaPS) is here to help: call 412-268-2922 and visit their website at `http://www.cmu.edu/counseling/`. Consider reaching out to a friend, faculty or family member you trust for help getting connected to the support that can help.

If you or someone you know is feeling suicidal or in danger of self-harm, call someone immediately, day or night:

CaPS: 412-268-2922

Re:solve Crisis Network: 888-796-8226

If the situation is life threatening, call the police:

On campus: CMU Police: 412-268-2323

Off campus: 911

If you have questions about this or your coursework, please let us know.

7 Schedule

Table 1 shows the tentative schedule. The idea is to cover the lecture material in roughly the first $\frac{2}{3}$ of the semester (by meeting three rather than two days a week), so that you will have more time to devote to the class project in the last $\frac{1}{3}$ of the semester, and so that you can take advantage of all of the course lecture material in your projects.

Table 1: 15-418/15-618, Fall 2020, tentative schedule (subject to change).

Class	Date	Day	Topic	Assignments
1	8/31	Mon	Why Parallelism?	
2	9/2	Wed	A Modern Multi-Core Processor	
3	9/4	Fri	Parallel Programming Models	L1 Out
	9/7	Mon	*No Lecture: Labor Day*	
4	9/9	Wed	Parallel Programming Basics	
5	9/11	Fri	Performance Optimization I	
6	9/14	Mon	GPU Architecture and CUDA Programming	L1 Due, L2 Out
7	9/16	Wed	Performance Optimization II	
8	9/18	Fri	Parallel Application Case Studies	
9	9/21	Mon	Workload-Driven Performance Evaluation	
10	9/23	Wed	Snooping-Based Cache Coherence	
11	9/25	Fri	Directory-Based Cache Coherence	
12	9/28	Mon	Snooping-Based Multiprocessor Design	L2 Due, L3 Out
13	9/30	Wed	Performance Monitoring Tools	
14	10/2	Fri	Memory Consistency	
15	10/5	Mon	Scaling a Web Site	
	10/7	Wed	**Exam I**	
16	10/9	Fri	Interconnection Networks	
17	10/12	Mon	Implementing Synchronization	
18	10/14	Wed	Fine-Grained Sync, Lock-Free Programming	L3 Due, L4 Out
	10/16	Fri	*No Lecture: Day for Community Engagement*	
19	10/19	Mon	Transactional Memory	
20	10/21	Wed	Earthquake Simulation Case Study	
	10/23	Fri	*No Lecture: Fall Break*	
21	10/26	Mon	Heterogenous Paralelism, HW Specialization	
22	10/28	Wed	Tolerating Latency through Prefetching	L4 Due
23	10/30	Fri	Domain-Specific Parallel Programming	
24	11/2	Mon	*Meetings to discuss project ideas*	
25	11/4	Wed	Domain-Specific Programming on Graphs	Project Proposal Due
26	11/6	Fri	Parallel Deep Neural Networks	
27	11/9	Mon	A Look Under the Hood	
	11/18	Wed	**Exam II**	
	11/23	Mon	*Meeting regarding Milestone Progress*	Project Milestone Due
	TBD		**Project Poster Session (during Final Exam slot)**	