ECE 459: Programming for Performance (Winter 2022)

Department of Electrical & Computer Engineering
University of Waterloo

About the Course

Undergraduate Calendar Description "Profiling computer systems; bottlenecks, Amdahl's law. Concurrency: threads and locks. Techniques for programming multicore processors; cache consistency. Transactional memory. Streaming architectures, vectorization, and SIMD. High-performance programming languages."

Prerequisites ECE 252, ECE 254, or SE 350 (or equivalent); Level at least 4A Electrical Engineering or Computer Engineering or Software Engineering.

Or, to put that in less formal terms: remember when we talked about semaphores and mutexes and all of that? If not, please review `https://github.com/jzarnett/ece252/tree/master/lectures` and its video form at `https://www.youtube.com/playlist?list=PLFCH6yhq9yAHFaI00FrrgG0dPg8a5SjTJ`. (Or you can look at CS 343 material). Also, I really hope you feel comfortable with programming. This isn't the place for you if you're not feeling confident in programming. Really. We're going to learn a new language (Rust!) so if you don't feel comfortable with the fundamentals of programming and if learning a new language is frightening then flee while you can!

Brief Overview

Many modern software systems must process large amounts of data, either in the form of huge data sets or vast numbers of (concurrent) transactions. This course introduces students to techniques for profiling, rearchitecting, and implementing software systems that can handle industrial-sized inputs. These techniques will enable you to design and build critical software infrastructure, especially in an age of Big Data.

While you may have seen some of these ideas in the context of operating systems (ECE254/SE350/CS350) and concurrency (ECE252/CS343), this course gives you tools to make code run faster. The focus in OS/concurrency is understanding and implementing the primitives; our focus is on using them effectively.

We will sometimes see implementation details that you need to get right to write certain applications, but as with any university-level course, this course focusses more on the concepts than magic invocations, so that you can continue to apply the basic ideas after the technologies inevitably change. For instance, 2021 was the first year that this course is running in Rust (rather than C and C++), but the core content remains the same.

General Information

Course Website The lecture material will be at `https://github.com/jzarnett/ece459` and the videos are at `https://www.youtube.com/playlist?list=PLFCH6yhq9yAHnjKmB9RLA2Qdk3XhphqrN`. There's also videos online. The primary source for course materials is github. We'll also use Piazza. See the section about the piazza policy.

Scheduling This course will be partly a flipped-classroom: some lecture slots will be filled by traditional lecture format, and some will be in-class exercises, which are not for marks but just for practice. When there is an in-class exercise, we'll expect you to watch some videos that week to keep pace. All the lecture materials from the Winter 2021 term will be available in video format to consume at a time that's convenient for you. Whether or not you attend the lectures, you should be getting through the lecture content at the normal pace of about 2.5 hours per week.

The final assessment will happen during what would normally be the final exam period.

Schedule Oddities: Reading Week is scheduled for Feb 21–25. We encourage you to take at least some days out of the reading week pauses to rest and recharge. In principle, 3rd and 4th year classes don't happen on the day of Capstone Project Symposia. There are a lot of programs represented in our course, to the point where no single program accounts for more than half of the students. Therefore we are not cancelling classes. We understand if you are going to miss it. We're also aware that IRS is a thing (hopefully).

Final Exam: The final exam period is Apr 8–26. There will be a Registrar-scheduled take-home exam scheduled somewhere between those dates; same drill as for the other courses. You'll want to have access to your Rust development environment on the exam day: because it is a take-home exam, we can ask you practical questions.

We expect you to solve the final exam on your own: no talking to other people. You may Google information that you might need but you may not post questions on forums.

About Prof. Zarnett. I graduated from the Computer Engineering program at Waterloo (under a previous curriculum), and have since earned my Master's Degree (also at Waterloo) and my P.Eng. license. For the last 8+ years I have also been teaching here at UW and the other courses I have taught and worked on include ECE 150, ECE 155, ECE 252, ECE 254, ECE 290, ECE 356, and MTE 241. I'm now one of the two computer engineering academic advisors, so you can also talk to me about that!

In addition to being your instructor, I work full time in software engineering in industry. This imposes certain schedule constraints, but I do my best. The best way to get questions answered is via Piazza. There's also the possibility of asking questions at (virtual) office hours.

About Prof. Lam. Way back when, I put Computer Engineering on my OUAC form. Then I got the AIF in the mail and decided I didn't want to study Computer Engineering after all. Yet here I am! I'm an Associate Professor of Electrical and Computer Engineering. Some of you will know me as the previous Director of Software Engineering. I studied math/CS yet convinced PEO to license me as a P.Eng.

Professoring is my full time day job. My research interests focus on using compiler and static analysis techniques to ensure program correctness. We'll touch on that at some points in this term. An important part of the reason I chose this day job is to help students. The best way to do that is by talking. I used to like to talk to students when they came by my office, but that's not going to happen this term. You can email me at `patrick.lam@uwaterloo.ca` and schedule a call if you'd like to talk to a 2D image of me on your screen.

About the Teaching Assistants. Teaching assistants can help you with the course material, including assignments and exams. They will do most of the assignment marking.

Teaching Assistants:

Textbook

There are detailed notes. There's no required textbook for the course. The notes cite a number of sources, some of which are useful books.

Evaluation

This course includes assignments and a take-home final examination. Too many fourth-year courses have projects. This tends to make the end of term unmanageable. We won't contribute to that problem too much.

Academic Integrity Exercise	1%	
Assignments	64%	(4 at 16% each)
Final exam	35%	

A grade of INC (Incomplete) is to be assigned if you do not attempt all assignments. If the assignments are not completed within 4 months of the end of the course to the satisfaction of the course instructor, the INC (Incomplete) grade will be converted to FTC (Failure to Complete). If the assignments are completed successfully within the required timeframe, the overall grade will be calculated using the rules for late assignment submission. Furthermore, any submitted assignment must be an honest attempt—submitting an empty project or similar is not adequate.

Exams. Exams will be open-book, open-notes, you can use your tools like the compiler, analysis tools, etc. Don't collaborate with others and do your own work. The University rules say if you do not submit the final exam, without an acceptable reason, your grade in the class will be DNW—Did Not Write. This is very undesirable. Turn in the final exam.

Assignments (aka labs). Since this course has "programming" in the title, you will be expected to write code for these assignments. Here is a list of assignments for this course. We have 4 assignments as below:

1. Manual parallelization of a computation using threads; and use of nonblocking I/O;
2. Inter-thread communication via message passing (channels), and via shared memory;
3. GPU programming with CUDA;
4. Open-ended program improvement via profiling.

Assignments will be posted on LEARN, and assignment hand-in will be done via `git` using the university provided git.uwaterloo.ca service. We've coordinated with other 4th year ECE courses to put our due dates on Wednesdays.

Your assignment code will be checked for plagiarism using MOSS (Measure Of Software Similarity) as well as some manual checks. You may request to opt out of the automatic screening by sending a formal written letter to your instructor explaining why; a meeting will then follow to discuss the subject with the instructor.

You'll have at least two weeks to do each assignment. Trying to place the assignment due dates in the term is hard. We try to avoid FYDP deadlines, not place them on top of midterms, keep them vaguely in sync with the course material covered so far, and finally we're not allowed to make the due date for them during reading week or after the end of term. Oh, and because there are so many students from so many programs, it is effectively impossible to pick a date that works for everyone. Apologies in advance. That's why there are grace days (see below).

Lateness. Also known as "Grace Days". You have five (5) days of lateness to use on submissions throughout the term. Each day you hand in something late consumes one of the days of lateness. Grace days are counted in units of whole days, so if you submit 2 hours late that's 1 grace day, and if you submit 22 hours late that is still 1 grace day. You can use up to two (2) grace days on any assignment.

If you use too many grace days, here's what happens. The sixth day of lateness causes your lowest assignment mark to be halved. The seventh day causes your lowest two assignment marks to be halved. If you have more than seven days of lateness, we'll start converting marks to 0 and dropping the associated late days.

You don't get any credit for unused late days.

Grace days are tracked in Learn for transparency.

Group work. You may discuss assignments with others, but we expect each of you to do each assignment independently. Acceptable collaboration includes discussing ideas and structures with others, as well as helping others debug their code. If your code is too close in structure to someone else's code, you are going to have a problem. The best way to avoid such problems is by (1) not sending your code around; and (2) not writing down anything beyond general notes (pseudocode) about other peoples' code. We will follow UW's Policy 71 if we discover any cases of plagiarism (and we have). Under this policy, the instructor may have follow-up conversations with individual students to ensure that the work submitted was completed on their own. Any follow-up will be conducted remotely (e.g., MS Teams, Skype, phone), as the University of Waterloo has suspended all in-person meetings until further notice.

We want to emphasize that we take the issue of plagiarism very seriously, and so does the University of Waterloo. If you are uncertain about this subject, please seek some guidance. There are many resources available to you. You can check the university policies, talk to the course instructor, ECE/SE undergrad office, et cetera.

Or, let's sum this up in two short instructions:

1. Acknowledge the work of others.
2. If you are uncertain, ask!

Piazza Policy. Piazza is a great tool for collaboration and it allows rapid but asynchronous exchange of information. It almost goes without saying, but we ask you to be respectful and polite when communicating via this medium and to assume the best about others, as tone can be difficult to interpret on a discussion forum.

Before posting your question, please consider whether this question can be answered by looking at either the course notes, lab manual, or documentation available on the internet (e.g., man pages). To help others find answers later, (1) we discourage screenshot-only posts (since the content does not show up in search results) and (2) we encourage you to please restrict a single thread to one question (or a closely-related set of questions).

For the most part, we encourage (but do not require) you to make your question(s) public so that other students may benefit from the answer. If something was non-obvious or unclear to you, it was likely to others as well. If your question includes detailed design information or code excerpts related to a deliverable (e.g., an assignment), then making it a private question is necessary.

Please also keep in mind that course staff may not be able to answer your question immediately as they have many other responsibilities throughout the term (and the frequency of questions tends to spike before due dates or exams). We encourage you to think through the problem before posting, but acknowledge that just like "rubber duck debugging" sometimes the act of writing down the nature of the problem provides insight into solving it.

Re-marking. If you believe that your grade on an assignment is incorrect, you may ask that it be re-marked. To request that a question be re-marked, you will need to submit your request in writing via e-mail to the instructor.

When you submit your request, it should include the following: (1) Your name and student ID number; (2) a clear indication of which question or part of the deliverable is to be re-marked; and (3) an explanation of why you believe the grade assigned was incorrect.

Requests for re-marking may be submitted any time before the final exam. Be forewarned: when a deliverable is being re-marked, your grade could go up, it could stay the same, or it could go down. You will be notified of the outcome.

Extra Credit In this class, there will be no opportunities to earn extra credit. Make-up assignments or examinations will not be offered under any circumstances.

Attendance & Illness There are video lectures. We can't make you watch them. In fact, I [PL] don't know whether I would actually watch all the hours of lectures if I was taking the class (I hate video). The lecture notes are the source material for the video lectures and also contain much of the information that we say. As you've surely noticed by now, at university, there is a lot that you have to learn on your own by doing assignments.

We do hope, though, that you stay on top of the material throughout the term and don't try to cram it all before the final exam/assignments. That's why the assignment deadlines are spread throughout the term. This is your second term of emergency remote teaching and we hope that you've picked up what works and doesn't work from the first one.

If you feel ill, you should seek appropriate medical attention. If you miss an exam for health reasons, you need a verification of illness form (and it has to be rated "severe" on the form). Forms can be completed by the physicians at Health Services or other healthcare providers in the area. If you anticipate missing a deliverable deadline or an examination for a non-medical reason, you should contact your instructor as soon as you are aware of the problem. Given sufficient notice, alternate arrangements may be possible. Alternate arrangements are rare and at instructor discretion. We understand that this term may have a higher than average number of emergencies and pledge to be understanding.

Please, please keep yourselves safe this term and respect the spirit and letter of all pandemic guidelines. You don't want COVID (even though you are unlikely to die from it), and you don't want to share it with the more vulnerable members of our community. There's no way this pandemic is going to be over by the end of this term. I hope that we will be able to celebrate your graduation, although it remains unclear if we can do this in June of this year.

Laptop and Device Policy The human visual system has evolved to perceive saber-toothed tigers in the savannah. Fortunately, tigers are rare in Waterloo, Ontario (Geese, on the other hand...). Unfortunately, your classmates are still human and hence their attention will be drawn to flashing lights (or Facebook, or movies, or video games) in their peripheral vision. We encourage everyone to be respectful of their classmates and to not distract them.

Wise use of computers and the Internet can be helpful for fully engaging in class. You might want to try out some syntax, or you might want to look up constructors, or you might want to verify your instructor's somewhat outrageous-sounding claim.

To support the benefits of the Internet while reducing distractions, we will adopt the following policy in this class. We are asking that the first 2 rows of class be text-oriented: if using a device, use a command prompt or text editor, maximized to the whole screen. Paper is always good, of course. Mac OS X and UNIX command prompts are probably your best bet; for those of you on Windows, you can use the Windows Subsystem for Linux. From the command prompt, you can use compilers and text-mode web browsers (w3m, lynx, links/elinks, etc....) tmux may also be helpful in managing multiple terminal sessions. Being proficient with the terminal is a highly-useful skill for a Software Engineer.

We acknowledge that lectures are not always engaging. Instead of distracting screen content, we recommend non-distracting ways of tuning out, like doodling on paper (while taking notes), or doing homework. (we also recommend passing notes to each other instead of talking).

If you need to sit towards the front of the class and use a GUI program, then please discuss with your instructor to register yourself as an exception. If you sign up for the exception list, we'll ask to you agree to not display games, videos, or social media on your screen (unless it is part of the class).

Enforcement is a sensitive issue, especially given the existence of exceptions. We are primarily asking each of you to respect the policy on your own. But, if you see someone with games, videos, or social media in the terminal zone, you can politely bring it up with them.

tl;dr: paper or text-oriented programs in first 2 rows of class during regular lectures (not in-class exercises).

Contingency Planning for Pandemic-Related Disruptions. The department asks us to plan for either a short-term (e.g., one week) or long-term (e.g., remainder of the term) disruption of in-person activities. In either case, we will cancel all the in-person lectures and in-class exercises and use only the videos for lecture content. Labs

(assignments) and the final assessment will not be affected either way.

University Policies

Academic Integrity In order to maintain a culture of academic integrity, members of the University of Waterloo community are expected to promote honesty, trust, fairness, respect and responsibility. Check `www.uwaterloo.ca/academicintegrity/` for more information.

Grievance A student who believes that a decision affecting some aspect of their university life has been unfair or unreasonable may have grounds for initiating a grievance. Read Policy 70, Student Petitions and Grievances, Section 4, `adm.uwaterloo.ca/infosec/Policies/policy70.htm`
If in doubt, contact the department's administrative assistant, who will provide further assistance.

Discipline A student is expected to know what constitutes academic integrity (see above section) to avoid committing an academic offence, and to take responsibility for their actions. A student who is unsure whether an action constitutes an offence, or who needs help in learning how to avoid offences (e.g., plagiarism, cheating) or about "rules" for group work/collaboration should seek guidance from the course instructor, academic advisor, or the undergraduate Associate Dean. For information on categories of offences and types of penalties, students should refer to Policy 71, Student Discipline, `www.adm.uwaterloo.ca/infosec/Policies/policy71.htm` . For typical penalties check Guidelines for the Assessment of Penalties, see
`www.adm.uwaterloo.ca/infosec/guidelines/penaltyguidelines.htm` .

Appeals A decision made or penalty imposed under Policy 70 (Student Petitions and Grievances) (other than a petition) or Policy 71 (Student Discipline) may be appealed if there is a ground. A student who believes he/she has a ground for an appeal should refer to Policy 72 (Student Appeals)
`www.adm.uwaterloo.ca/infosec/Policies/policy72.htm`.

Privacy Questions about the collection, use, and disclosure of personal information by the University, should be directed to the Freedom of Information and Privacy Coordinator, Secretariat, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, Canada N2L 3G1. The email address of the Freedom of Information and Privacy Coordinator is `fippa@uwaterloo.ca`. See also University of Waterloo Policy 19: Access to and Release of Student Information; Information and Privacy.
`https://uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-19`

Note for Students with Special Needs The AccessAbility Services (formerly known as OPD) located in Needles Hall, collaborates with all academic departments to arrange appropriate accommodations for students with disabilities without compromising the academic integrity of the curriculum. If you require academic accommodations to lessen the impact of your disability, please register with the AccessAbility Services office at the beginning of each academic term.