ECE459: Programming for Performance, version 1.1*
Winter 2014

Patrick Lam

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 (ECE354/SE350/CS350) and concurrency (CS343), this course gives you tools to make code run faster. The focus in OS 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.

Objectives. More specifically, after this course you will be able to:

- investigate a software architecture, propose ways to improve its performance, and estimate the impact of your changes;
- understand what the compiler is doing to automatically improve your code's performance, both in terms of regular optimizations and automatic parallelization;
- use common mechanisms to manually exploit parallelism: manual specification of parallelism (e.g. OpenMP); multithreading (and locks); asynchronous I/O; vectorization (including GPU programming); and distributing the problem over many systems (e.g. via MapReduce).

General Information

Course Website: `http://patricklam.ca/p4p`

Schedule:

Lectures	T	2:30 – 3:50 PM	RCH 302
	Th	2:30-3:50 PM	QNC 2502
Midterm	W	February 26	DC1350
		19:00–20:20	

Although there is a time slot for tutorials, I do not anticipate using it.

*r1.1: added midterm date/time

Instructor:

Patrick Lam
Office: DC2597D or DC2534
Office Hours: Established in consultation with the class, or by appointment
Email: p.lam@ece.uwaterloo.ca

Teaching Assistants:

Xavier Noumbissi
Office: DC 2553
Email: xnoumbis@uwaterloo.ca

Morteza Nabavi
Office: E5-4131
Email: mnabavi@uwaterloo.ca

Saeed Nejati
Office: E5-4132
Email: snejati@uwaterloo.ca

Course Description

Topics. Here is a detailed list of topics and estimated lecture hours for each topic.

Introduction	1.5
Processor architecture and hardware threading	1.5
Threads, races, dependencies, locks, semaphores	3
Asynchronous I/O	1.5
Speculation, SIMD, parallelization patterns	1.5
Compilers, Optimizations and Automatic Parallelization	3
OpenMP	3
Memory ordering, fences, atomics, lock granularity, inlining	3
Cache coherence	1
Profiling for Performance	1.5
Distributed programming: MPI	1.5
Programming GPUs: OpenCL	3
MapReduce & Hadoop	1.5
Software Transactional Memory	1.5
Performance tweaks, loop perforation	1.5
High-performance Languages	1.5
Big Data	1.5

Reference Material

You might find this book useful.

> Darryl Gove. Multicore Application Programming for Windows, Linux and Solaris. Addison-Wesley, 2010.

I will post reasonably complete lecture notes for the material that I'm lecturing on.

Evaluation

This course includes assignments, a midterm, and a final examination. Too many fourth-year courses have projects, so I'm not going to pile on another project here.

Assignments	40%
Midterm	10%
Final exam	50%

Assignments. Since this course has "programming" in the title, you will be expected to write code for these assignments. Here is a projected list of assignments for this course. I plan to have 4 assignments.

1. implementing manual parallelization for servers with Pthreads and asynchronous I/O;
2. using compiler-provided automatic parallelization and OpenMP;
3. identifying opportunities for optimization (e.g. parallelization) and implementing them; and
4. GPU programming with OpenCL.

Assignment handin will be done via git.

Exams. Exams will be open-book, open-notes, no electronics.

Schedule. Here are target assignment dates:

January 13	A1 out
January 27	A1 due, A2 out
February 10	A2 due, A3 out
February 24	A3 out
late February	Midterm
March 10	A3 due, A4 out
April 4	A4 due
Exam period	Final exam

Even if I'm late getting an assignment out, you'll still have two weeks to do the assignment.

Group work. You may discuss assignments with others, but I expect each of you to do each assignment independently (or with your assignment partner only). 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. I will follow UW's Policy 71 if I discover any cases of plagiarism (and I have). I will not use turnitin, but I may use other plagiarism detection software.

Lateness. You have 4 days of lateness to use on submissions throughout the term. Each day you hand in something late consumes one of the days of lateness. The fifth day of lateness causes your lowest assignment mark to be halved, while the sixth day causes both assignment marks to be halved. If you hand in something and you have more than 6 days of lateness, I'll start converting marks to 0 and dropping the associated late days. You can only hand in a submission up to the time I return all of the submissions. You don't get any credit for unused late days.

For example, you may hand in A1 one day late, A2 two days late, A3 1 one day late, and everything else on time. Or you can hand A2 four days late, if you hand in everything else on time. Finally, if you hand in A1 3 days late, A2 1 day late, A3 3 days late, and everything else on time, I will either give you a 0 for A1, leaving you with 4 late days, or give you a 0 for A2, leaving you with 5 late days and causing your mark for A1 to be halved. I'll choose the option which gives you more marks.

Required inclusions

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 his/her university life has been unfair or unreasonable may have grounds for initiating a grievance. Read Policy 70, Student Petitions and Grievances, Section 4, `www.adm.uwaterloo.ca/infosec/Policies/policy70.htm`. When in doubt please be certain to contact the departments administrative assistant who will provide further assistance.

Discipline: A student is expected to know what constitutes academic integrity [check `www.uwaterloo.ca/academicintegrity/`] to avoid committing an academic offence, and to take responsibility for his/her 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, 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`. Note for Students with Disabilities: The Office for Persons with Disabilities (OPD), located in Needles Hall, Room 1132, 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 OPD at the beginning of each academic term.