Introduction

ICS432

Concurrent and High-Performance Programming

Henri Casanova (henric@hawaii.edu)

Course Goal

- A “hands-on” course on concurrency and high performance programming
  - There is a lot of theory that we could go into
  - Instead we’ll take a pragmatic approach and experience general principles
- By the end of the class you will be able to:
  - Write correct and efficient multi-threaded code in Java and C/C++
  - Understand how to accelerate code, including via multi-threading
  - Have notions of advanced topics on concurrency and high performance: locality, lock-free programming, transactional memories, etc.

Course Website

- Located at:
  - http://courses.ics.hawaii.edu/ics432_fall2022/
  - Linked from my homepage
    - Google for “Henri Casanova”
- Organized as Modules
  - All lecture notes as PDF files, including some screencasts
    - No textbook in this course!
  - Pointers to useful on-line material
  - All assignments
  - Announcements
  - A link to the Syllabus
    - Which we’ll go over now in these slides
- Let’s look at the Web site...

Lectures

- Lecture notes are posted on the course's Web site regularly
  - You can read them before or after the lecture, up to you
  - I am notorious for spacing out on putting the notes up on the site, so just drop me a one-line message
- In general, if you note "something weird" on the Web site or in the lecture notes, do not hesitate to write me a one-liner
  - Broken links, typos, weird "due date", etc.
- Some modules in the course are a bit independent from the rest of the course, and we may cover them out-of-order
  - e.g., so that assignments are more spaced out

What is this course about?

- Focus #1: **Concurrency:** programs that do multiple things at once
  - Very mainstream and important in industry
  - Known to be pretty difficult…
- Focus #2: **High Performance:** programs that go fast
  - Always a good idea to know something about this
  - Concurrency is one of the ways to achieve high performance
- Focus #3: **Software Engineering**
  - You'll write quite a bit of code in this course, and will use software engineering tools/approaches

Concurrency is Hard

- Writing concurrent programs is difficult when one has never been exposed to concurrency
  - And difficult even when exposed to concurrency!
- After taking this course you will be “””””proficient””””” in concurrent programming
  - Correctness, Responsiveness, Performance
  - Java and C/C++
- Why so many quotes around proficient?

Herb Sutter, chair of the ISO C++ standards committee: *“Everybody who learns concurrency thinks they understand it .. and discovers that they didn’t actually understand it yet after all.”*

Relationship to ICS Curriculum

- ICS332: Operating Systems
  - Provides an introduction to concurrency concepts
    - We'll have a review of these concepts (part of it screencast)
    - We will expand **massively** on the "Synchronization" module of ICS332
- ICS443: Parallel Algorithms
  - About some of the theoretical aspects that are relevant in this course, but that we not got into since this is more "hands on"
    - And there is a distinction between "concurrent" and "parallel", which we'll come to…

Shows of hands

- Who has written multi-threaded code (outside of ICS 332)?
  - If so, which language/framework?
- Who has ever tried to make a piece of code faster before (outside of ICS332) ?
- Who would have difficulties bringing a laptop to class for some of the lectures?

Homework Assignments

- The class will have **both** programming assignment and non-programming assignments
- All programming assignments are in the context of a team project
  - A Java App to which we add features throughout the semester
- Non-programming assignments will be all be individual and cover the same concepts as the programming assignments
  - As “I have written code that happens to work each time I run it” does not equate “I have understood everything correctly” for concurrent programming

Assignment Specifications

- Each assignment has clear specifications with examples
  - File names, class names, behaviors, etc.
  - If you find them unclear, let me know right away
- Not conforming with specification makes our life (i.e., grading) very difficult
  - And it will make you absolutely hated by your co-workers in your professional lives
- **You will lose points for not matching the specs**

Command-Line

- Use of the command-line in the terminal is a vital skill
- We are not necessarily doing anything fancy in this class on the command-line, but if you're not comfortable with it it's high-time you do something about it
  - A lot of graduates contact us back saying "I wish I had learned more command-line stuff while in college because my first month on the job was rough! I thought it was just a professor thing…"
- There is a Reading in this module about the command-line….

Software for ICS432

- Operating System that runs Java (whatever one)
- We need Java 11 and Maven 3.6.3
- We need a C compiler that supports OpenMP (gcc), provided via a Docker container

- We'll look at Assignment #0 at the end of this lecture…

Team Project

- The course has “pencil-and-paper” **individual** assignments and **team** programming assignments
- All programming assignments are part of a semester-long team project
  - We start with a Java application I have developed, and add features/capabilities to it throughout the semester
- Let’s talk about:
  - How teams will be formed
  - How teams are supposed to work on assignments
  - How assignments are turned in
  - How grading is done for these assignments

Forming Teams

- Each assignment is done by **3 students**
  - Perhaps a few teams of 4 because we have to do integer division
- I have created one private GitHub repo for each team
- For the first assignment teams are picked randomly and students are added to the repos
  - In the real world you don't get to pick who you work with at all
  - So this is a good experience for you, even though I realize some of you would have loved to be in a team with your friends the whole semester
- For each new assignment, 2 students on each team are moved to 2 different new teams
  - Yes, you will have to work with code written by other students
- Rationale:
  - you will have worked with every other students by the end of the semester
  - you will not get "stuck" with a team member with whom things don't go well
  - you will experience what it's like to work with other people's code as starting point (but one student will have written part of that code in each team)

Working as Teams (1)

- All team work is done in private GitHub repos
- I am the owner of all these repos
- I will invite and uninvite students for each new assignment

Working as Teams (2)

- Every assignment will require you to create and address GitHub issues
- You can work as you want on these issues (e.g., each team member tackles an issue, all team members collaborate on all issues)
- But it's **really** a good idea to know what your team-mates are doing or have done
  - Don't just say "John did all the multithreading stuff"!
  - We have exams, and team assignment grading takes participation into account
- You should review each other's code
  - The code will evolve, and bad design decisions will come back to haunt you
  - So team code review sessions are a must
- Always a great idea to use branches
  - e.g., each team member does an attempt in their branch, and then team members compare/discuss/decide
  - Gets around the problem of "Mary has done everything in 10 minutes because she did a Java concurrency internship last summer and I didn't have time to learn/practice anything"
- Only the main branch will be graded

Working as Teams (3)

- I've set up a Discord server for this course
- I've set up one text channel per team (to which everybody's invited since teams change)
- Feel free to use this as your main way to work in teams
  - But of course each team can self organize and do what they want to collaborate
- We will have in-class, hands-on development sessions
  - During which you can use Discord for screen share, or whatever you want...

Grading Team Assignments

- This course is a challenging elective
- Our basic assumption is that every student takes it because they really wants to learn this stuff, and will work hard
- So, we expect to give the same grade to all students in a team
- However, each team member will submit an anonymous questionnaire in which they evaluate the % effort of each team member
- When these evaluations point to a problem, we will check the git logs and make executive decisions regarding grades
  - In the past, this hasn't been a problem

Homework Assignments Policies

- All assignments are to be turned in using Laulima by 11:55PM on the day the assignment is due
- Late Assignments
  - 10% penalty for up to 24 hours of lateness
  - A grade of *zero* for more than 24 hours of lateness
- Solutions will always be discussed in class, and available upon request by e-mail to me
- Read the syllabus' statement about "academic dishonesty"

Grading on 1000 points

- Two Exams
  - One midterm exam (200 points)
  - One cumulative final exam (200 points)
- Eleven graded Homework Assignments (600 points):
  - Homework #2: 50 points (team)
  - Homework #3: 50 points (individual)
  - Homework #4: 40 points (team)
  - Homework #5: 60 points (individual)
  - Homework #6: 70 points (team)
  - Homework #7: 40 points (individual)
  - Homework #8: 80 points (team)
  - Homework #9: 80 points (team)
  - Homework #10: 50 points (individual)
  - Homework #11: 80 points (team)
  - Homework #12: 30 points (individual)

Professor Advice #1

- Come to class

- I've been told my slides are nice and clear, and so no need coming to lectures
- That is true for about 5% of students, and a horrible mistake for the rest
- Concurrency is known for many "I get it", "wait, not I don't", "Oh, I got it!", "What??? No, I didn't!"
- Flipping through slides without the in-class explanations/digressions/livecoding lures most students into a false sense of "getting it"

Professor Advice #2

- Ask questions, including “can you explain this again?”

- 95% of students at UH think: “I must be the only student who’s confused and it’s so embarrassing to speak up”
- But concurrent is confusing to most people! (see previous slide)
- I really want to avoid is what has happened so often:
  - 5% “vocal” students were getting the material so well they only asked questions that went beyond the course material
  - 95% “mute” students were not getting the material, and never asked a question (intimated by the “crazy” questions from the vocal students)
- Not sure what do do about this, but I want the “crazy” **and** the “I didn’t understand any of this” questions

Professor Advice #3

- Don't start late on assignments

- Every professor tells you this, and it's typically ignored
  - Because most of you have a lot on your plate, we get it
- But at the 400-level, courses become challenging and assignments can be tricky/long, so starting late becomes more and more dangerous
  - Some of you can still blast through assignments in a day, but the fraction who can do so decreases dramatically at the 400-level
- **One good approach:** Read through the assignment on the day it is posted
  - So that you subconsciously think about it right away
  - I found this extremely useful as a student
- We will not answer any questions (including via e-mail) on the day the assignment is due

Questions?

- Any questions on the syllabus?
- Any questions on the course in general?

What's Next

- Do Homework #0 and Homework #1 by the end of next week
- You may also want to:
  - Brush up on your Java?
  - Brush up on git if needed
    - ICS314 Review Site
  - Learn the basics of Maven if needed
    - http://maven.apache.org/guides/getting-started/
    - https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html
- Let's look at Homework #0 and #1 right now…