CS 517: Computational Complexity

Instructor: Dr. Mike Rosulek, <rosulekm@eecs.oregonstate.edu>

TA: Brent Carmer <carmerb@oregonstate.edu>

Meets: MWF 2-3, in MLM 033

Website: http://eecs.oregonstate.edu/~rosulekm/cs517
Please check often for announcements, homeworks, etc.

Textbook: There is no required text. Lecture notes from various sources and in-class slides will be posted on the course website. You will also be expected to be independently resourceful as needed. If you prefer killing trees, I suggest the following:

- *Computational Complexity: A Conceptual Perspective*, Oded Goldreich
- *Computational Complexity: A Modern Approach*, Sanjeev Arora & Boaz Barak

Challenges, Expectations (Scare Tactics)

In a computational complexity course, we do lots of things that are potentially very alien and conceptually challenging. Most of our time is spent simply **writing algorithms**, but we write algorithms . . .

. . . in exotic, bizarrely constrained, abstract *models of computation* (e.g., a nondeterministic oracle Turing machine with 1 bit of advice)

. . . to solve abstractly specified problems (e.g., solve an arbitrary, unspecified PSPACE-complete problem; disagree with every n^2-time Turing machine)

. . . using weird, abstractly specified assumptions (e.g., assuming that any exponential-time Turing machine has a polynomial-size circuit).

. . . to understand how powerful different abstract resources are in the context of computation (e.g., randomness, interaction, nondeterminism) and to understand what properties make problems inherently hard to solve (e.g., alternating quantifiers).

This is an **advanced & intensive course designed for CS PhD students.** This course is probably going to challenge you no matter what, but you will face an even more significant uphill battle in this course **unless:**

. . . you have mastered the basics of undergraduate algorithms and automata theory.

. . . you are comfortable understanding and producing precise mathematical writing.

. . . you are comfortable with high levels of *abstraction*.

In particular, I suggest that you **drop the course** if:

. . . you have not taken a graduate-level algorithms course (take CS515 first).

. . . you have not taken a course in automata theory (take CS516 first).

Problem Sets

Expect approximately 5 problem sets throughout the quarter. For each problem set, at least one in-class meeting will be reserved for mutual aid on the problem set.

- Attempt to solve the problems (individually or in groups) before the problem-solving session.
- Ensure you leave the problem-solving session with all the details for **correct** solutions to all the problems.
- After the problem-solving session, each student will be randomly assigned a subset of the problems. Students must write up and submit solutions to their assigned problems. These solutions must be written up **individually** and typeset **in LaTeX**.
- We will provide feedback on your solutions.

At the end of the quarter, you will turn in the following:

- First drafts of all your written solutions, with our feedback.
- Revised drafts of all your solutions, taking into account our feedback.
- A `latexdiff` between the two drafts.

Solutions will be graded according to:

- Correctness. The structure of the course leaves no excuse for incorrect or nonsensical solutions. In order to receive a B letter grade in the class, you must submit correct solutions to the problems.
- Improvement in quality of technical/mathematical writing between drafts.

As PhD students in computer science, you are conducting research on technical and abstract concepts. You will be expected to write clearly and fluently about your research. If you can write clearly and fluently about computational complexity then you can handle anything.

Relevance to your Research

One of your assigned problems will be to identify a hardness result (NP-hardness, PSPACE-hardness, etc) from your research area and write a summary. This will be more involved than a typical problem set, and will probably require several pages.

Other Policies

Cheating: Academic dishonesty (including plagiarism and cheating) will not be tolerated. Consult the university's student conduct code for more details. I will follow the guidelines given there, and seek out the maximum allowable penalty for violations that occur in this course.

If you have a question about what constitutes academic dishonesty, please ask me.

Disabilities: Accommodations are collaborative efforts between students, faculty, and Disability Access Services (DAS). Students with accommodations approved through DAS are responsible for contacting the faculty member in charge of the course prior to or during the first week of the term to discuss accommodations. Students who believe they are eligible for accommodations but who have not yet obtained approval through DAS should contact DAS immediately at 737-4098.