TEACHING STATEMENT

GEOFFREY CHALLEN

12/16/2016

I believe that teaching is as important as research. As more students are exposed to computer science and learn our powerful tools, they will use those capabilities to change the world. Teaching also provides opportunities for computer scientists to build and evaluate more effective education tools using our domain knowledge. So just as my research group designs, builds and evaluates novel computer systems, we also design, build, and evaluate novel instructional approaches. Please peruse my courses page for more information about the courses I have taught, including syllabi, online resources, and student evaluations.

1 — COURSES TAUGHT

At UB my primary teaching responsibilities are a large flipped introductory course on the internet (CSE 199, §1.1), an introduction to operating systems (CSE 421/521, §1.2), and a graduate seminar on mobile systems (CSE 72x, §1.3). For two years I co-taught a graduate course on advanced computer systems (CSE 622) which, in the interest of space, is not described. **In all my courses I focus on giving students the opportunity to build and use computer systems.** While I enjoy lecturing, I know that challenging projects inspire them to learn far more outside the classroom than they can inside.

1.1 — CSE 199: How the Internet Works (`internet-class.org`)

In Fall 2016 I taught CSE 199, a new freshman course on "How the Internet Works". The internet is the most significant thing computer scientists have ever built. It's development is rich in history, design principles, exciting engineering, societal relevance, and deep human implications. I cannot imagine a more exciting topic or a better way to introduce students to computer science.

But my interest in developing the course is not only due to the material. The course is part of UB's new Freshman Seminar program, which is intended to allow freshman to interact with faculty in small groups of up to 25 students. However, the "seminar" that I am designing will be taught by multiple faculty to groups of 100 students. Swamped by growing interest, my department does not have enough faculty to offer 25-student seminars. Instead, we used a combination of online learning and flipped classroom activities to provide personalized learning. My challenge was to use technology to scale the small-group seminar experience to large groups.

In Fall 2016 I supervised four teaching faculty and 25 undergraduate TAs in offering this course for the first time. All primary instruction happens online. Content is broken into 292 short videos of roughly five minutes, with approximately 100 minutes of content assigned each week. However, in contrast with many other online courses using this approach, I will not be the only author. Instead, we are building `internet-class.org` to allow anyone with knowledge about the internet to contribute. By making it easy to add content and invite others to do so, I anticipate that we will eventually have hundreds of academic and industry contributors. Students were also required to provide their own explanations as a course assignment, and we plan to use strong student submissions during subsequent offerings.

Multiple explanations for each concept allow `internet-class.org` to adapt to students. Choosing the best content and reinforcements for students will help them learn the material. If my explanation is best, then students may see it first, but be offered multiple other explanations by other faculty and contributors as needed. In other cases, we may be able to learn what types of explanations work well for each student and steer them in the correct direction. This approach utilizes what I call learning webs and it forms the cornerstone of the educational plan in my funded CAREER proposal. Modularity inherent to our approach also allows other faculty to rearrange content to suit their own courses on the internet.

1.2 — CSE 421/521: Introduction to Operating Systems (`ops-class.org`)

I'm a professor because I took a course on operating systems using the OS/161 instructional operating system from Margo Seltzer in 2001. While serving as a TA for the course six times as an undergraduate and Ph.D. student, I repeatedly witnessed the transformative effect it had on students. I was determined to

bring this course with me to UB to train new generations of systems programmers. But this effort immediately ran into a major hurdle. At Harvard, five experienced TAs would be assigned to a class of 30 students (1:6). At UB I was provided two inexperienced TAs for 100 students (1:50).

I worked for several years to adapt the course for UB and continue to improve it incrementally. The results are visible at `ops-class.org` where class lecture videos and slides, assignments, exams, an online forum, and testing resources are available. To help students learn, I videotape all my lectures and post them on YouTube. As of July 2016 my lectures have been viewed 177,120 times and watched for a total of 1,343,806 minutes by 1901 subscribers. This allows students to review the material and prepare for exams. My lectures are also being viewed by people across the world, increasing the visibility of UB's CSE program.

My course staff and I have also made major advances in grading automation for the OS/161 assignments. Automated grading was introduced in the first year that I taught the course. Last spring, we improved the process further through a new `test161` tool. `test161` represents a novel form of distributed automated grading. Secret injection during compilation secures testing output from manipulation by students. This allows us to share tests with students and let them test on their own machines, shifting load away from our servers. Final submissions are made through the `test161.ops-class.org` website. An interactive web interface allows students to watch online grading. Scores are posted on public leader boards with student permission, encouraging friendly competition. An equally important benefit of automated grading is allowing students to improve their answers. Students may submit as many answers as they want until the deadline and use testing results to iteratively improve their submissions.

Automating grading has achieved its goal—allowing TAs to spend more time helping students. In Spring 2015, my TAs and volunteers held 27 TA-hours of office hours per week—approximately the same number per student as at Harvard. It is also more support than any other course in my department.

Despite both being extremely challenging and required for undergraduates, my OS course consistently ranks as one of the top CSE courses. Students have affectionately nicknamed it the "Challen challenge." Note also while other CSE classes struggle with low course evaluation participation, my course is evaluated by almost 100% of students each year. I accomplish this by releasing selected final exam questions early once the class meets course evaluation participation targets. I consider this a reasonable tradeoff. The course benefits each year from students' feedback, and high participation rates ensure that it is representative. And statistics show that students perform no better on the questions released early.

The success of my OS course has led students into my research group and to continued involvement with the course. In Spring 2014, we began a volunteer TA program. Three previous 421/521 students volunteered to serve as course "ninjas" and hold several office hours a week. Based on the success of that program we expanded it to 10 volunteers in 2015 and and 11 in 2016, primarily undergraduates from prior years. The willingness of previous students to serve as volunteers speaks strongly to the course's success.

1.3 — CSE 72x

Since 2013 each fall I have taught a small seminar on a current topic in mobile systems. Course titles have included "Smartphone Sustainability" (2013), "Personal Cloud Computing" (2014), "Using Uncertainty to Program Mobile Systems" (2015), and " Improving Smartphone Quality of Experience" (2016). My seminar offers me an opportunity to work with students that are considering a Ph.D. Several previous enrollees, including Ali Ben Ali and Scott Haseley, have joined my research group. Due to the novelty of these topics, few readings are available that directly address the subject matter. However, this allows us to explore research in related areas such as sensor networking, OS design, energy management, and mobile systems. Class time is used for discussions. Randomly selected students summarize the paper and lead discussion, requiring all students to read and understand all assigned papers. During the course of each semester I usually assign implementation projects related to the specific course content. As the culmination of the course students work together to write a paper summarizing our findings. Papers produced by previous seminars have appeared twice at HotMobile (2014, 2015), the top workshop on mobile computing.

2 — COURSE DEVELOPMENT

Since 2015 I have been actively involved in improving our CS courses. I developed the new internet course described previously (§1.1), and am leading the design of a new introductory programming series (§2.1).

2.1 — CSE 101: Introduction to Computer Science

Since Fall 2015 I have been helping redesign our department's introductory courses. These courses have a huge impact on education in the department. They introduce students to computer science and provide a chance for us to attract a large and diverse group of students. They should provide a solid programming foundation and overview of CS that will support students in later courses.

My experience teaching OS caused me to realize that our current introductory courses were not meeting these goals. Students emerged with weak programming skills, unprepared for downstream courses or internships. This weakness persisted even until senior year, where it made it difficult for them to succeed at my OS assignments. As a member of a committee devoted to redesigning these courses, I worked with Jesse Hartloff to propose a new introductory series. We made our proposal public and received a great deal of useful feedback from current students and alumni. With help from Carl Alphonce and Luke Ziarek, our course proposal was approved and we are preparing to teach it in future years.

3 — CURRICULUM DEVELOPMENT

Starting in Fall 2015 I began chairing an Undergraduate Advisory Committee subcommittee charged with revising our entire CS curriculum. The last review was many years ago, so an update was overdue. Several other faculty and a number of student volunteers participated in the process.

We approached the project in two phases. First, we identified places where we can increase the flexibility in our current curriculum by eliminating unnecessary or inappropriate requirements. A key part of this effort was coding our curriculum and the curriculum of many other schools for the purposes of comparison. Over the years, a number of these requirements have crept into our curriculum. Over time they have reduced the flexibility of the computer science program to a degree approaching the computer engineering program. This eliminates one of the benefits of computer science compared with computer engineering. Unnecessary requirements prevent undergraduates from specializing in different areas and from taking useful advanced courses. The output of the first phase is a set of changes to our curriculum that can be implemented and will have a large impact quickly. Department faculty approved these changes in Fall 2016.

In the second phase taking place in 2016–2017 we will prepare more sweeping recommendations about how to modernize our entire program. We anticipate that these changes will be implemented over a longer time horizon and involve new courses and hiring. During the second phase we are seeking input from current and former students, other faculty, area K–12 educators, and companies that hire our students.

4 — OUTREACH

Complementary to my teaching are ongoing efforts at improving diversity within my field and exposing undergraduates to research. My diversity efforts are described in more detail in my service statement.

4.1 — Undergraduate Research

I love introducing undergraduates to research. I have had the privilege of working with 18 undergraduates on a variety of projects, several of which have produced publications. Nick DiRienzo built a system called PocketMocker allowing smartphone users to conceal their true activities by injecting fake data. Nick presented a poster at HotMobile'14 which won the best poster award and published a PocketMocker paper at MobiCASE'14. Frank Rossi assisted in the development of the PocketLocker system which creates personal storage clouds from multiple personal devices. John Cherry worked on file system analysis as part of the SUNY Louis Stokes Alliance for Minority Participation (LSAMP) summer program.

My group currently includes 9 undergraduates working on projects including improving user quality of experience (Brijesh Rakholia), improving smartphone energy management (Kyle Schoener), investigating the interaction between Android and "modder" communities (Edwin Santos), benchmarking Android databases (Grant Wrazen and Lakshmi Ethiraj), using crowdsourcing to optimize Wifi networks (Grant Wrazen and Vighnesh Iyer), and `internet-class.org` content and infrastructure (Greg Bunyea, Aishani Bhalla, and Wesley Csendom). I integrate undergraduates into my group by having them work alongside us in the lab, giving them tasks appropriate to their training, and providing graduate student mentors.