Teaching Statement

Tianyin Xu

One of the key reasons that I decide to pursue academia is the privilege of being an educator—to teach, mentor, and work with students. My philosophy as an educator is to **focus on the students**: helping them learn, improve, and grow into computer scientists—independent thinkers and researchers with solid knowledge and skills. All my teaching and mentoring approaches are aligned with this overarching philosophy, and I truly believe Nico Habermann's philosophy of working with students: "focus on the students, since graduating great students means you will produce great research, while focusing on the research may or may not produce great students."

Teaching. I have prepared myself through many teaching experiences during graduate school. I have been a teaching assistant (TA) for courses at both undergraduate and graduate levels for seven quarters. I also gave three lectures as a student instructor, and served as a "meta-TA" to train first-time TAs. In my recent three TA quarters, I was recommended by 98.6% of the students (75.0% gave strong recommendations) for being "well prepared," "very experienced," "with ability of clearly illustrating teaching content," and "good constructive criticism" [1–3]. These experiences have shaped my teaching approach, centered around the following goals:

First and foremost, motivating students is always my primary goal. My approach is to make the course **relevant**. Relevance is the key to engaging students in learning. For example, not all students will become OS kernel developers, but many likely have applied or will use OS concepts such as synchronization, scheduling, and resource management in their study or work. Therefore, I always make effort to relate OS concepts to real-world, everyday problems beyond OS kernels. I talk about how Google manages resources and schedules jobs at a massive scale, how Spark drives its huge success based on the disk-memory trade-off, and why Android does not swap but kills app processes when running out of memory. Students enjoy such discussions and become more engaged.

I also seek to make the course **interesting**. I tell stories like the unhandled hardware interrupt that almost failed MINIX, the Tanenbaum-Torvalds debate on OS kernel architectures, and the law case centered around the argument: is the web browser an essential part of the OS? I find these stories intrigue students and stimulate their curiosity.

Second, I like to guide students to **think independently and differently**. One message I always deliver to students is that there is no "standard answers" for building systems, and that existing design and implementation may not be the best solution. I encourage students to think independently and explore different design alternatives. For example, to explain a specific system, I would start from scratch, describe the design goals, and ask students to explore solutions; then, for each proposed solution, we discuss the feasibility and potential pitfalls, gradually refine the solutions, and eventually reach consensus. By leading such thinking processes, I want to help students think from the perspective of system architects to understand the opportunities and constraints of different design choices.

Third, I want to **cultivate skills and practices**. Systems projects often closely simulate real-world software development: students work in teams to build system components on top of relatively large and complex code base. This is a great opportunity to help them develop good engineering skills and practices, such as coding, testing, and debugging. In the undergraduate OS class led by Geoff Voelker, we teach students how to test and debug complex systems code during lab hours—systems development is much more than writing code that compiles, but is also about assuring the correctness of the code and reasoning out defects through debugging. I believe that teaching these skills and practices can greatly benefit students and prepare them for their future careers.

In my experience, the key to achieving the above goals is to be **interactive** and **hands-on**, both in class and after class. In order to help students, I always listen to them first to understand their confusion and difficulties. This helps me address the "root causes" and makes my teaching more efficient. I tend to be very hands-on in the beginning of the projects—explaining existing code base, walking through sample tasks, and demonstrating testing and debugging methods. I believe these can help students learn from examples and gain confidence to tackle the projects. Once they are on the right track, I encourage them to explore and solve problems independently.

Given my background and experience, I am prepared to teach operating systems, software engineering, and program analysis. I would make hands-on projects an important part of these courses. I would also be very happy to teach advanced graduate courses and seminars related to my research areas.

Mentoring. I greatly enjoy mentoring and working with students. I have mentored two undergraduate students, a Master's student, and a junior Ph.D. student. It has been highly rewarding to see the students learn new knowledge

and skills, develop insights on real-world systems, and apply what they have learned in their careers. Specifically, the students I mentored, Tao Cai and Han M. Naing, continue working on system configuration management in industry, while Raphael Lu chooses to pursue academia in the same topic.

I mentored Tao on his Master's thesis which is a part of the PCheck project. After graduation, Tao joined LinkedIn as a site reliability engineer working on LinkedIn's configuration systems. He told me that the experiences of working with me in the research project helped him develop understanding on building reliable systems in the face of faults and errors. I also mentored Han and Raphael for their undergraduate independent study; both of them later joined my research project on security misconfigurations and co-authored a research paper which will appear in ACM CHI'17. Han joined Webroot Inc., and has helped the company build the configuration management system that deploys and monitors hundreds of cloud instances. Raphael decides to pursue a Ph.D. because "research is interesting, exciting, and also achievable."

My mentoring philosophy is similar to my teaching philosophy (e.g., motivating students, guiding independent thinking, and being hands-on) and focuses more on **discovering** the student's interests and passion, and **matching** them with well-scoped, fruitful projects. I believe these to be key to their happiness and productivity. For Tao's thesis, I first designed a study-based project on characterizing misconfiguration manifestations in real-world systems. Later, I observed that Tao was more interested in hacking system code; hence I re-designed his project to be extracting and analyzing configuration usage code from Java bytecode of the studied programs, a project he was so excited about and accomplished excellently. I like to foster an open, collaborative environment and treat students as respectful colleagues, which allows me to know them better, communicate with them better, and build trust in the process.

As a mentor, my goal is to help students grow into independent researchers. Besides providing technical training, I also want to help students develop important research capabilities and skill sets, including critical thinking, writing, presentation, and collaboration. These are what I learned from my Ph.D. program, and I am excited to provide my students with similar experiences to help them succeed in their future careers—I look forward to seeing their success.

References

[1] Student ASE Evaluation for Tianyin Xu: CSE 101 - Design & Analysis of Algorithm (Summer Session 1 2014). *University of California San Diego*, Summer 2014. http://cseweb.ucsd.edu/~tixu/ta_eval/Xu_Tianyin_Student_ASE_Evaluation_CSE_101_S114.pdf.

[2] Student ASE Evaluation for Tianyin Xu: CSE 221 - Operating Systems (Fall 2014). *University of California San Diego*, Fall 2014. http://cseweb.ucsd.edu/~tixu/ta_eval/Xu_Tianyin_Student_ASE_Evaluation_CSE_221_FA14.pdf.

[3] Student ASE Evaluation for Tianyin Xu: CSE 221 - Operating Systems (Spring 2016). *University of California San Diego*, Spring 2016. http://cseweb.ucsd.edu/~tixu/ta_eval/Xu_Tianyin_Student_ASE_Evaluation_CSE_221_SP16.pdf.