Teaching Statement

Ennan Zhai

Growing up in a family of educators, I have an inherent passion for teaching, and greatly enjoy mentoring and working with students. This is one of the major reasons in my decision to pursue an academic position.

Classroom teaching. My philosophy of classroom teaching can be summarized in three points: 1) to present students with a clear big picture for each topic covered in the class; 2) to throughly describe one of the most representative work in each topic; and 3) to deepen students' understanding of the knowledge via interesting assignments and labs.

I will use my experience as an instructor for the "Building Distributed Systems" course at Yale in Fall 2017 to demonstrate my teaching philosophy. In my class, there were 18 undergraduate students, 16 master students and 2 PhD students—a relatively large class in the computer science department at Yale. This course was coding-intensive, and aimed to teach students the underlying principles of distributed systems and how to build workable distributed systems.

First, I organized lecture topics based on the history of distributed systems development: starting with the first Internet messaging system, UseNet, then describing Peer-to-Peer (P2P) network-based systems, next bridging the P2P time and the Cloud Computing time with knowledge of NAT, virtualization, and MapReduce, and finally teaching knowledge for cloud-scale systems. The above schedule not only offers a clear big picture view of the history, but also enables students to understand the motivation, the state-of-the-arts, and the techniques involved behind each phase in the evolution of distributed systems.

Second, before going through a representative work in each topic, I elaborated on the importance of the topic in order to attract and motivate my students. Then I used an illustrative case study example to throughly explain the working principle of this representative work.

Third, I offered my students interesting practice lab assignments for each topic to deepen their understanding of the knowledge learned in my class. My lab assignments employed a novel "RFC" mode. More specifically, I asked my students to build their own systems that meet a set of specifications I posted, so that each of their implementations should speak the same protocol and is able to interoperate with any other implementations built by their classmates. I encouraged my students to test the interoperability of their systems with each other, and I also ran a reference implementation for them to test against during their development and debugging process. This lab teaching design has proven to be a great success. It not only facilitated interactions among students, but also provided them with hands on experience with the implementation and test flow of distributed systems. I felt my students had a genuine interest in these assignments and had learned a lot in the process.

Mentoring. The responsibility of a mentor is to empower students with the ability to discover problems of their own interest and to develop solutions. One challenge of mentoring is to ensure students make progress. I like to provide students with concrete stepwise achievable goals in addition to maintaining and communicating a bigger vision or open-ended problems that they can work towards. Specifically, what I have been doing is to first give my students a big picture background of the target problem, as well as a summary of the state-of-the-arts. Then I work with them to design a workable first-step solution but with unsolved challenges. And finally, I ask them to improve upon the solution by solving these remaining challenges step by step, guiding them through this process. In the past two years, I have advised two undergraduate students, five master students, and two PhD students in the lab of my postdoctoral advisor Ruzica Piskac. Most of our projects either have been accepted by top-tier conferences, or are under review.

Future teaching plan. Based on my past experience, I believe that I am well-suited for teaching many courses, including distributed systems, system security, operating systems, databases, software engineering and software verification. First, I have taught an entire course on distributed systems as an instructor at Yale. Second, I have designed assignments and labs with the instructors when I served as a teaching assistant for a variety of courses including operating systems, database systems, software engineering, and software analysis and verification. Moreover, my research background and publications also cover these areas.

In addition, I am interested in offering a "holistic" system course that integrates and examines the relationship between distributed systems, security, and verification, along with the introduction of cutting edge technologies. As an example, in this course I would like to instruct students first on using the blockchain technology to decentralize certain centralized Internet systems (*e.g.*, Airbnb and DNS), then on constructing cryptographic protocols to ensure the security and privacy of these blockchain-based systems, and finally on verifying the security properties of these systems with machine-checkable proofs. A course like this can provide students with a complete experience of secure system development and validation, as well as expose students to the current hot technologies.