In the area of computer systems, I believe students need: (1) a strong foundation in computer system knowledge (2) curiosity and ability to explore new directions in computer systems. To this end, I am excited to teach students in two ways. First, I want to provide a strong curriculum in computer systems. I want to teach foundational system courses on hardware/software interfaces, machine organization, and system programming. I also want to teach standard undergraduate- and graduate-level operating systems, distributed systems, and networking courses. I want this stream of courses to provide students a strong foundation in the design and implementation of complex system software. Second, I want to inspire students in new directions by teaching them more advanced and emerging topics in operating systems (e.g., virtualization) and networking (e.g., data center networking). In this stream of courses, my main focus is to help students embrace lifelong learning and to prepare students with the abilities to adapt to technology changes. At the end of the day, computer systems change rapidly, and students need to have the tools to learn on their own after they graduate.

Teaching I have been a teaching assistant for two undergraduate courses. One was on operating systems and the other was on networking. For both, I led a discussion section every week. For the course on operating systems, I also developed the programming assignments.

I believe it is important to motivate students to think deeply about tradeoffs in system design. For problems in computer systems, there are often multiple approaches to realize the same goal, each with its own tradeoffs. When I am teaching operating systems and networking, I always want the students to lay out the design space for their programming assignments first. Then I ask them to come up with multiple approaches to their programming assignment and tradeoffs for each of them. This inspires them to think about the approach at high-level before they jump into coding. Also, this approach helps students learn how to approach problems. Future problems they will need to solve aren't the ones we necessarily can anticipate today, but the concepts we provide can provide a basis for understanding and framing those problems.

Hands-on programming projects are often the most important aspect of teaching computer systems. I plan to develop my own programming assignments or use widely-used ones to keep the curriculum up to date with changing technologies in the industry. I want to lead by example, becoming very familiar with programming assignments so that I can help students even at detailed implementation. I will make sure programming assignments have clear milestones to help students self-correct, catch things early if students are falling behind and need more help.

Mentoring Over the past year, I have mentored two Ph.D. students (Kaiyuan Zhang, Samantha Miller) and two undergraduate students (Matthew Rockett, Shibin Xu). I have worked with them on multiple research projects and a new set of programming assignments for operating system course.

When I mentor Ph.D. students, I want them to become project leaders. I make sure they can

see the big picture of the research. This motivates the students by letting them understand the broader impact of the research. At the same time, this helps students understand challenges and steps in carrying out different parts of the research. I meet with my mentees weekly to discuss system design, related work, and how to split the work between the team. After a while, they have become pretty good at carrying out research independently.

For undergraduate or master students, I believe it is important to motivate them and also hand-hold them on detail implementations. I keep my mentees in the loop about work in our community that is related to our project. I do a lot of code walk-through to help them understand how each piece of the project functions. I also encourage them to hack on some components of the projects. They implemented several parts of our projects, and I found them increasingly capable of relating implementation details to the high-level system design goals.

The teamwork with my mentees has been successful. We have together submitted two research papers. Our efficient container virtualization research recently won the Madrona Prize Runner-up, an award given out by local venture capitalist for student-led CS projects with commercialization potential. The teaching operating system has been used by our department for the introduction to operating system course for two quarters. I always feel happy about their growth, and I am also thankful for their contribution to our research.