The opportunity to teach and mentor students is one of the main reasons I seek an academic career. As a faculty member, I am deeply committed to teaching and consider it my responsibility not only to educate but also to inspire students. I also greatly enjoy working with students on research projects and, more importantly, making them independent researchers. In this process, I am committed to creating a diverse and inclusive environment both in the classroom and in my research group.

Teaching Experience and Philosophy

My teaching experience comes mainly from being the co-instructor of CS 739, a graduate-level distributed systems course at UW Madison. I have also served as a teaching assistant for CS 537 (introduction to operating systems) and CS 302 (introduction to programming). In addition, I have delivered several guest lectures in systems courses in the computer sciences department at UW Madison. These experiences have largely shaped my teaching philosophy. My approach towards teaching is grounded in the idea that learning concepts in the classroom, and practically applying or implementing them go hand-in-hand. From my experience, I am convinced this philosophy leads to more effective learning, particularly so in my field of computer systems.

I practice the above philosophy in many aspects: projects, activities, and assessments. For example, in CS 739, I first taught about remote procedure calls (RPCs) and failure handling in the class. Then, I designed a mini-project where students implemented the same ideas in a real system. The goal of this project was to build a highly durable key-value service using the gRPC library. By building a real system, students effectively learn and often internalize the concepts. This process, however, is not without challenges: sometimes, working on real systems can make it hard for students to relate to the core ideas taught in the class. This is because, in a real system, ideas are often tangled with intricate implementation details. To help students discern ideas from details, I ask them specific questions as a part of the assignment. For example, in the above project, I asked, “how is failure handling in gRPC different from the Xerox RPC paper that we read?” Such questions nudge students to focus on specific parts of the code that underpin key ideas. I also designed exams based on the content that students were supposed to learn from practical assignments (in addition to the material taught in the class). This motivates students to do well on the assignments, leading to a more holistic way of learning.

This philosophy has had three main positive effects on students in my experience with CS 739. First, while being challenged, students enjoyed implementing the ideas learned in the class in real systems. Comments and reviews from the course evaluations (by the students) support this claim. For example, one student said “Extremely fun/challenging project and final project, which are tied to the theories we learned in papers/lectures.” Second, with the hands-on experience, students felt more confident about their understanding of concepts and their abilities. For example, one review said “...after doing P1, I am now extremely confident about how to make a durable kstore that can sustain failure.” Finally, I noticed that by working on and learning from practical systems, students gained the courage to tackle problems in even more complex systems. In particular, in the final project, students were able to understand and modify real systems (e.g., ZooKeeper, HDFS, Cassandra) to implement their ideas. This is a crucial skill regardless of whether a student intends to pursue a PhD or join the industry to build large-scale software systems.

I noticed that a similar hands-on approach had a positive influence when I was a teaching assistant for CS 537 (undergraduate OS course at UW Madison). In this course, the lectures covered basic topics such as virtual memory, CPU scheduling, concurrency, and file systems. Students would then implement the ideas in XV6 (a research OS kernel built for teaching operating systems to undergraduates). As a TA, I had the opportunity to teach lectures on condition variables, RAID, and file systems. I then also helped students in the lab with assignments on the same topics. I could see that when students tried to relate parts of the real system to the concepts learned in the class, their understanding was much more concrete.

Teaching Evaluations. I have been evaluated by 32 students who took CS 739 in Spring 2020 at UW Madison. As an instructor, I was rated on a scale of 1 (lowest) to 7 (highest). I received a rating of 6.42 (ranked second among graduate-course instructors). I was also nominated for the SACM CoW award, a teaching award given to CS professors. The course as a whole was given a rating of 6.50; it was ranked first among all courses offered that semester with student reviews such as “Best graduate-level course I had in UW-Madison CS”. As mentioned above, I also received several encouraging comments, which have given me confidence that my teaching philosophy has positively influenced students.
Courses I can Teach

As a new faculty member, I am committed to teaching both undergraduate and graduate courses. At the undergraduate level, I can teach operating systems, distributed systems, systems programming, introductory data structures, etc. I am also willing to teach courses outside my immediate area of expertise. For example, I can teach databases, networks, data science, computer organization, etc. I am open to building upon the existing expertise within the department in running these courses, but I am also enthusiastic about designing new courses and materials. At the graduate level, I am interested in teaching operating systems, distributed systems, and storage systems. I am also interested in creating new courses on topics like cloud computing, big data systems, and emerging areas such as edge computing and IoT. I intend to run these courses based on paper reading and a significant research project, potentially leading to publications at top conferences.

Mentoring Experience and Philosophy

My advising experience comes from mentoring more than 30 graduate students and undergraduates at UW Madison. I have also had an intern work with me at VMware Research. I have also mentored students from other institutions, including Princeton, Penn State, and UC Berkeley, via mentoring programs at OSDI, SOSP, and EuroDW. My advising philosophy is grounded in two main ideas.

First, one size doesn't fit all: I understand that every student is different. Some students need more help communicating their ideas, while others might need hands-on guidance navigating large code bases. I intend to adapt my advising style by understanding individual needs and appropriately channelling my effort in the most needed directions. I have seen the value of this practice when mentoring undergraduate researchers who often make meaningful progress once the low-level tasks and goals are precisely defined. For example, Neil Perry (now a graduate student at Stanford), whom I mentored, made significant progress independently once we discussed in detail the tasks and goals in analyzing the resiliency of distributed ledgers. On the other hand, senior graduate students excel when they are left to explore on their own.

Second, amplify your strengths and learn from peers. I realize that students are motivated in research if they can play to their strengths. For example, my strength was in examining/building practical systems (given my software engineering experience). Thus, I was enthusiastic about working on reliability problems these systems face, where I could leverage my skills. At the same time, as an advisor, it is my responsibility to address weaknesses and help students improve upon them. In addition to an advisor's help, I have come to realize that students greatly benefit by working with other (senior) students. Such collaborations can also alleviate the loneliness that research can sometimes bring. When I started my PhD, I worked with two senior graduate students (on an OSDI '14 paper). From them, I learned a lot about the particular research problem and, more broadly, how to conduct research, develop good research taste, and write effectively. On the other side, I mentored two junior graduate students Yifan Dai and Yien Xu, on building a learned index for LSM trees. I closely worked with these students, mentoring them on the design, experiments, and writing. We published this paper at OSDI '20. I also mentored Konstantinos Kanellis on using machine learning to optimize storage configurations. At VMware Research, I am currently mentoring Yi Xu (a graduate student at UCSD) on exploiting persistent memory devices in a production key-value storage system.

I plan to organize my research group based on the above principles. I will work closely with each student when they start, adjusting my style as I better understand their needs. I will then organize projects so that 2-3 students can work with and learn from each other.

Commitment to Creating a Diverse and Inclusive Environment

In both teaching and research, I intend to create inclusive and diverse environments where every student would feel welcome regardless of their gender, socio-economic status, race, ethnicity, and color. In my experience, even seemingly innocuous questions or comments can sometimes make students feel unwelcome. For instance, asking how many people have had prior programming experience in an introductory course can make students who did not have the privilege of computers feel like they cannot succeed. Coming from a socio-economically disadvantaged setting, I experienced such issues and was intimidated. I was fortunate to have some teachers who broke this barrier for me. I consider it my responsibility to do the same for the next-generation students.

In my research group, I am committed to recruiting and nurturing students from diverse backgrounds. Once a researcher I admire said at a conference “next time bring along at least one person who doesn’t look like you”. This thought has stuck with me since. The first step to realizing this goal is to fix the entry point: when students are recruited to research groups. I will make conscious efforts to recruit students from diverse backgrounds and create an inclusive environment in my research group where all students can succeed.