Teaching Statement

Yifan Sun

1. Teaching

Teaching has been an important theme of my Ph.D. work and will grow in importance as I pursue my academic career. During my Ph.D. I taught three undergraduate courses as an instructor, including Embedded Design: Enabling Robotics, Fundamentals of Engineering Algorithms, and Fundamental Digital Design (as a co-instructor). With the permission of the Department of Electrical and Computing Engineering, I also restructured and redesigned the Fundamentals of Engineering Algorithms course materials, including the syllabus, lectures, assignments, and exams. My teaching effort and accomplishments were recognized by Northeastern University's College of Engineering, which awarded me the College of Engineering Teaching Award in 2018. Upon reflection of teaching these courses, I will summarize three guiding principles that have benefitted the students in my classes: i) providing timely feedback, ii) integrating industry-informed practices, and iii) solving real-world problems.

I believe deliberate practice, combined with timely feedback, leads to improved learning experiences. One key principle for instructors and TAs should be to provide students with prompt and accurate feedback. To give feedback as soon as possible, I created automated testing scripts for every single assignment. Since we used Github for assignment submission, I contacted Travis CI, a continuous integration solution vendor, to provide us with free CPU cycles to run student assignments. Every time a student committed and pushed code to Github, the student would know whether the solution was correct or not in a couple of minutes, without waiting for days to weeks before a TA could grade their assignment. In addition, I realized that automated tests can tell the students if they are solving the problems correctly, but cannot tell why they are right or wrong. To provide customized guidance, I use instant messaging tools (e.g., Slack) and online discussion boards (e.g., Piazza) to bring students, TAs, and the instructor closer. In Fundamentals of Engineering Algorithms, we reached an average response time as low as 12 minutes on Piazza. In a relatively small class (≈20 students), I would help students debug their code and comment on how to improve the code. In larger classrooms, I mentored TAs to help students and guided students to help each other. I am excited to receive acknowledgments from the students in my course evaluations: "[Yifan is] always available for help: whether during the day for office hours or 11 pm over email. Super helpful."

As computer technologies are developing fast, educators should **integrate industry-informed practices** into their classrooms so that students can be equipped with state-of-the-art tools and techniques. Thanks to my internship experiences at Dell EMC and AMD, and my experience leading large open-source software projects, I was able to adopt best practices from the open-source community and industry to my classroom. In both the Embedded Design: Enabling Robotics and the Fundamentals of Engineering Algorithms courses, I used Github as the assignment submission platform. I taught students Git because version control system knowledge is an essential part of programming. Similar to how they would submit changes to a company's codebase, students created pull requests to submit their solutions. When TAs graded submissions, they would review the pull requests, simulating the code reviewing process used in industry and open-source communities.

In my teaching, I emphasized how technologies could be used to **solve real-world problems**. I prioritized teaching students how industry and academia could use the knowledge they had learned in a classroom. In homework assignments, students should not only practice skills but should also establish an understanding of how to use these skills in large projects. One example is the design of the assignments for the Fundamentals of Engineering Algorithms course. With all the assignments, each student developed a miniature in-memory database and used the database to perform data analytics. Throughout the assignments, I prepared a real-world dataset from *meetup.com* so that students could understand how tech-companies could analyze data with the algorithms taught in classes. Students completed their final homework assignments using the database they developed and the *meetup.com* data to verify the 6-degree separation theorem---any two people can be connected with at most 6 connections (a connection represents two people in the same meetup). They not only verified the theorem with the data provided but also discovered that a few key people who participate in both east-coast and

west-coast meetups could significantly reduce the average number of connections needed to connect any two people.

2. Mentoring

I enjoyed working with talented high school students, undergraduate students from Northeastern University, and Research Experiences for Undergraduates (REU) funded by NSF. As a senior Ph.D. student, I also mentored junior Ph.D. students. From these mentoring experiences, I derived three mentoring principles: i) personalization, ii) cultivating independence, and iii) reconsidering fundamental questions. First, the **personalization** of mentoring styles should be based on each individual's characteristics, goals, and talents. For those students who were primarily interested in industry positions that typically require solid programming basics, I guided them to develop good programming practices and helped them improve programming skills. For those who were interested in research positions, I met them regularly to help them incubate research ideas. Second, I emphasize **cultivating mentee independence** as a researcher or developer. I took an approach in which I gave my mentees easy tasks at the beginning and supervised them closely on these tasks. I would ask them to write programs or paragraphs in a certain way so that they could learn best practices in programming or writing. As my mentees gained more experience, I would give them more freedom until they could conduct a study on their own. This is an effective mentoring approach because, rather than solving problems for them, I was able to accelerate their intellectual growth and develop independence. Finally, I encouraged my mentees to keep **reconsidering fundamental questions** when they got confused, such as: What is your research question? What is your hypothesis? You reached a conclusion, but so what? These questions encouraged my mentees to dig deep into the roots of their challenges and conduct high-impact research.

3. Teaching Interests

I have accumulated rich experience in both software- and hardware-oriented courses. In the future, I would like to teach digital design, embedded design, computer architecture, compilers, operating systems, algorithms, and software engineering courses at both the undergraduate and graduate levels. I am especially interested in teaching freshman students their first programming course to motivate them to get more familiar with coding. Given my expertise in computer architecture and simulator development, I would also welcome the opportunity to plan a computer architecture course based on simulator development, where students would learn by implementing important computer components in a simulator and analyzing performance with the simulator. I would also like to establish a graduate-level GPU architecture course, covering emerging technologies in massively-parallel chip design. I would love to follow my teaching philosophy, providing students with timely feedback and help them apply their knowledge to solve real-world problems.