Teaching Statement

Ye Yu

I have been taught and advised by many great teachers in the past years of learning. They inspired me to learn from them and practice my teaching skills when I had opportunities to teach. Although it takes tremendous effort to excel in teaching and advising, teaching is also a rewarding process for me, which develops my interpersonal and communication skills and improves time management and multitasking capabilities. I also view teaching as an approach to outreach and to involve students in a research program, driving an active retention of foundational material, and as an opportunity to expand my expertise by presenting the state-of-the-art in my research community to a computer science audience.

1. Teaching Experience and Philosophies

I have served as Teaching Assistant for introductory level Computer Science courses, including *The C Programming Language* at Beihang University, and *Introduction to Computer Science* at the University of Kentucky. In addition to my formal teaching responsibilities, I have been actively participating the student competitive programming contests as the assistant coach for the University of Kentucky student ACM/ICPC programming contest team. These coaching and teaching experiences provided me a valuable insight into the effort involved in many aspects of teaching, including preparing lecture materials, homework assignments and projects, giving lectures, communicating and tutoring, in addition to grading and organizing events.

All in all, my experiences have left the following desiderata in designing classes: i) to develop material and a course structure that promotes student-driven interest in the topic; ii) to enable opportunities for students to both to learn how to work as individuals and as team players; iii) to expose students to both the core foundations and principles of the topic at hand, and their impact and application in society.

While there are some differences between undergraduate teaching, graduate teaching, and student advising, my approaches to helping students are applications of the following philosophies:

- **Respect the goals, needs, and strengths of different students**: I communicate with students and understand their goals. Then I help them tailor their study approaches according to their strengths and weaknesses, and guide them to the best of their potential in course-work or research projects. For example, for the students that are interested in programming contests, their goals may be different. While encouraging them to team up and match their abilities, I ask students who are going to find engineer jobs to practice more on implementation. For students interested in research, I encourage them to explore more algorithms and solve research problems.
- **Equip students with skills to address a wide range of problems**: As a professor, I need to do more than conveying answers for homework and exams. I inspire students to master the core principles and thought processes of problem-solving and critical thinking. For example, after introducing B+ tree and self-balanced binary tree, I encourage students to compare their time and space complexity, and discuss why they are suitable for different problems.
- **Let students learn from hands-on experiences**: A few group projects would help students to better understand course material, for both undergraduate and graduate courses. Based on my experience, for courses such as second-year algorithm/data structure courses, using an online judging system to ask the student to solve questions is an efficient approach to enhance their understanding of the course materials and to improve their programming skills. For upper-grade courses, setting course project would be more suitable. For example, Implementing an existing protocol or system is an ideal starter project for senior students. Other important approaches include motivating students to become self-starters and fitting their work into a grand vision.

2. Teaching Plan

Undergraduate teaching: Given my research background and previous teaching experiences, I am qualified to teach a variety of courses, especially system-related and algorithm-related courses, including *Computer Networks, Database Systems, Computer Architecture, Cloud Computing and Data Centers, Operating systems, Algorithms, Data Structures, Discrete Mathematics.* I am also qualified for all introductory programming courses. From the many years of programming competition coaching, I have developed an appreciation for hands-on implementation in undergraduate education of Computer Science. I am experienced in guiding the students through expanding their knowledge base, strengthing their ability of theoretical analysis, and enhancing their programming skills. One useful technique I used is to let the students solve programming contest tasks with their implementations of data structures and algorithms, and then with the existing libraries. For upper-level courses, in addition to focusing on both the application and theory of the course material, I will also discuss the history and impact of the designs. For example, when I discuss the designs of various systems, I also talk about the evolution of the systems in the history and why people chose the state-of-art designs for current applications. Also, I will keep students updated with the latest breakthroughs in information technologies and demonstrate to them how it is powered by what we learn in class.

Graduate teaching: I can teach both diversity lecture courses for students with various backgrounds such as Advanced Computer Networks, Distributed Systems, and Cloud Computing, and seminar courses for students particularly interested in systems such as Cloud and Big Data Systems. As an instructor of graduate courses, I will introduce the technical background and latest research progress in particular research areas, and encourage meaningful discussions. For course projects, I will first suggest a list of potential topics and provide background materials. I will then talk with students individually and help them to establish research framework as well as developing detailed plans. Following my research style, I will introduce theoretical techniques that provide the foundations of some computer system designs. In the course project, I will encourage students to employ various theoretical tools to address their research topics.

Student advising: My goal as an advisor is to cultivate the new generation of computer scientists. To advise a student who just starts research, I will first provide a variety of problems as options for them to begin with. Students will gradually focus on the topics in which they are most interested and talented. Then I will guide them toward discovering new problems and developing solutions on their own. In detailed presentations of technical papers, students will learn how to organize their explanation of motivation, design innovations, and performance evaluation in a logical and easily understandable way. I will also encourage students to communicate and collaborate with industry experts, and to know the requirements of system research in companies.

For undergraduate students, I believe participation in competitive programming contests is a practical approach to help them to identify the specific area that they are most interested in. The ability to code is a fundamental skill for computer science students, which is especially important for students seeking for opportunities in the industry. I will be happy to continue to contribute to student programming contest training if suitable.

Course Development: For graduate seminars, keeping the class highly interactive and getting all the students involved in the discussion is important. I believe that discussions in class help make the problems clearer and inspire new ideas. As a teacher, I have a particular interest in encouraging students to be more active and ask questions. For example, based on a similar format borrowed from a graduate seminar, I will let each student review one paper before class, and ask them to discuss the merits and problems of these papers. Besides, I will let each student select a paper to present to the class. In this way, each student has a chance to serve as a reviewer and get actively involved. I also believe that graduate seminars should not only use conference and journal papers as course materials, but also standards, industrial technical reports, and white papers. For example, during my experience at Facebook, I found that my textbook knowledge of network protocols such as TCP/IP and BGP is not enough to tackle the real issues in real large-scale cloud systems, and I had to read through the IETF documents to understand the reasons for introducing each BGP features. Incorporating these types of materials into the curriculum helps students to build a solid background about the details of each system, and a better understanding of how real systems are built.

I am also interested in developing a new graduate-level course "Application of Algorithms and Data Structures in Cloud and Distributed Systems". I believe the boundary between theory and practice is an important but unfortunately underexplored area in networking courses in most schools. We have seen several cloud and distributed systems that leverage smart data structures and algorithms developed in the theory community and achieve great performance, scalability. Therefore, it is essential for graduate students to be equipped with a bag of theoretical tools when they approach systems problems. The new course consists of two parts. In the first half of the course, I will focus on building up students' theory background: For each technique (e.g., Bloom filters, cuckoo hashing, distributed hash table, sketches, Rabin fingerprinting, perfect hashing algorithms, etc.). I will ask the students read a paper about the technique and another paper or two about its use in practical networked systems. In the second half, I will focus on practical systems problems so that students can realize the beauty of theories in their research. I will discuss papers that use dierent theoretical techniques to address the same practical issue (e.g., dealing with scalability challenges in networks and distributed systems, in routing, load balancing, packet forwarding, measurement, etc.). I will also design course projects for students to apply the theoretical techniques they learned to systems problems.

E-mail address: ye.yu@uky.edu