Teaching Statement

Zu-Ming Jiang
Department of Computer Science, ETH Zurich

There is an ancient Chinese proverb: *it is better to teach someone how to fish than to give them fish*, which can be applied to the education of computer science. Computer science is rapidly evolving, with various techniques being invented, advancing, or disappearing at an extremely fast pace. Compared to teaching specific techniques to students, it is more important to teach the core philosophy, *i.e.*, *independent thinking* and *pioneering mind*, which fits in general computer science. Following these principles, I was happy and rewarded to witness the growth and success of my students on their professional paths.

Teaching Experience. I have served as a teaching assistant for three core undergraduate-level courses: Compiler Design (over 200 students), Data Modelling and Databases (over 300 students), and Computer Science II (over 300 students); and one graduate-level course: Automated Software Testing (around 70 students). I assisted with Compiler Design for four semesters and was responsible for setting up compiler projects and tutoring students with their projects, presenting exercise sessions, and co-designing the final exams. In 2024, I was the head TA for this course, where I additionally set up the infrastructure, coordinated 7 TAs on designing compiler projects and exercise sessions, addressed students' concerns. For the Data Modelling and Databases course, I designed exercise slides according to the latest lecture material and presented them to the students. In addition, I was responsible for designing new questions for the final exam and validating the overall exam. In Computer Science II, I designed exercise slides for 3 exercise sessions and constructed questions related to computational complexity for the final exam. I had good experiences in the Automated Software Testing course over two semesters, where I gave guest lectures about database testing to the students and led 5-6 groups of students with their proposals and completion of their research projects.

Teaching Philosophy. When teaching students, in addition to the techniques introduced, I always encourage students to *think independently* about the design insight behind the techniques. For example, in the Compiler Design course, when introducing the optimization techniques (*e.g.*, analysis for live variables and reaching definitions), I encouraged the students to explore the common characteristics of these techniques and guided them to uncover the intrinsic principles (*e.g.*, the dataflow analysis framework with its fixpoint algorithm). Another important consideration for me, during teaching, is that I am always mindful to introduce the development track of techniques in a domain and let the students *envision possible futures*. In the Automated Software Testing course, after introducing a testing technique, I emphasized its pros and cons and then connected it to a new technique that overcomes the drawbacks. When I introduced the state-of-the-art, I encouraged the students to imagine the possible future work according to its potential drawbacks and discuss the feasibility of their proposed approaches.

Future Course. I very much look forward to teaching at both undergraduate and graduate levels for courses where I have sufficient experience. Specifically, at the undergraduate level, I can teach a wide range of core courses, including but not limited to Databases, Operating Systems, Compilers, and Data Structure and Algorithms. Specifically, my past teaching experience covers Databases, Compilers, and Data Structure and Algorithms, and my research work is related to Databases and Operating Systems. At the graduate level, I am broadly interested in the courses related to my research domains, where I can share the latest knowledge and insights I have gained over the course of my research, including, Advance Databases, Advance Operating Systems, Cybersecurity, and Software Testing and Verification. In addition, I am especially interested in setting up interdisciplinary courses like System Security and Automated Testing for System Software, where I can help students break the obstacles between different domains and integrate them better.

Advising Style. I enjoy mentoring students and feel honored to witness their success. I have co-advised 2 undergraduates and 3 graduates. One graduate student chose to continue as a PhD candidate at ETH Zurich after finishing his Master's thesis with me. Another graduate student has not started his Master's thesis yet but already completed a research paper with me, which is under review at ASPLOS, a top-tier venue of computer systems. When working with students, I emphasize discovering while at the same time respecting their interests. I believe that a successful researcher always enjoys doing research, and their interests motivate them to explore diverse research projects. During discussions, instead of giving instructions, I prefer to ask questions, which could guide students to think deeply about the research problems and stimulate their creativity. On several occasions, I obtained surprising and innovative ideas from my students. In addition, I closely follow students' progress and proactively provide help without being asked, because students might not realize that they have been trapped in difficult research problems or proceed with their research in the wrong direction.