TEACHING STATEMENT

Fang-Yi Yu (fayu@umich.edu)

Why do we need to go to school? In "under the tree of mine," the Nobel prize winner Kenzaburo Oe answered: *to understand yourself and to communicate with others.* This simple statement has grown on me as I stay in school longer. In computer science, a solid foundation in math is necessary for us to communicate with other people, computers, and the world. We need to answer questions like, "how efficient is your program," or "why does your program work." To answer these questions, we need to use math to convince others. To me, mathematics is a language. Schools need to teach students how to read, speak, and write in math.

One way to help students learn math is to encourage them to communicate in math. For example, when answering students' questions, I like to use *an interactive proof process.* Instead of giving static answers, I try to understand the students' attempt first. Then I give hints and ask them to think about the connection between the hints and their attempt. Depending on students' ability in math, this interactive process can decompose homework questions into customized exercises. Moreover, in this process, students can use math to communicate with me: How can they phrase their idea in math? Why do these lemmas imply the statement? In addition to this interactive process, I encourage them to help others. Helping others can boost students' confidence in math and reinforce what they have learned.

One common challenge of learning math is jargon. I believe that in undergrad intro-level classes, we should give students broad and tangible material and help students read more and be able to gather information by themselves. I was a teaching assistant for an undergrad-leveled course, the foundation of theoretical computer science, which had more than five hundred students. Instead of going deep in any topic, we covered a broad range of topics, including Chernoff bounds, RSA encryption, and interactive proofs. In addition to the standard lectures, we also have extra discussion sections. I oversaw the section notes of those discussion sections and always stressed that each lecture note should be self-contained and have a similar difficulty as online references like Wikipedia or StackExchange. As the class progressed, I noticed that students had more discussion on class material on Piazza and were able to find more references online by themselves.

Another challenge of learning math is its manifold aspects. With the advance of computer science, students are expected to be proficient in combinatorics and discrete math to understand matchings and shortest path algorithms on graphs. They also need to be familiar with differentiable manifolds and real functional analysis to understand the intuition of gradient descent or validity of empirical risk minimizations in machine learning. With my background in electrical engineering and math, I believe these different modes of mindsets are inevitable for researchers in modern computer science. In a graduate-leveled course of computational complexity, I was the teaching assistant and hosted discussion sections for advanced material. In the section about graph Laplacians, I introduced the Laplacians operator as used in partial differential equations to help them connect the ideas of harmonic analysis in the discrete and continuous aspects.

As the field of computer science develops rapidly, I find it challenging and rewarding to teach classes in computer science. Students have diverse backgrounds and majors. Different students have different needs, and I use different approaches and method to communicate with them. I believe schools should provide broad and solid knowledge for students to communicate in this rapidly evolving world.