Teaching Statement

Shuai Mu
Department of Computer Science
Stony Brook University (SBU)

I am very excited by the teaching component of faculty jobs. This statement describes my teaching experiences and philosophy at Stony Brook University. I mainly have two courses (excluding seminars) at Stony Brook, CSE-535 Distributed Systems for graduate students, and CSE-416 Software Engineering for undergraduate students. I also enjoy mentoring a diverse group of excellent students of different levels.

1 Graduate Teaching: CSE-535 Distributed Systems

CSE-535 is the main course I am teaching at Stony Brook. The course topic closely aligns with my research direction; it is a great pleasure to teach the course for 5 times. The course used to be given under the name of Asynchronous Systems. Before I joined Stony Brook, our faculty had been trying to remodel this class to focus on practical distributed systems topics. I am happy to participate in this effort and help finish the transition to make it a formal Distributed Systems course. The course is one of the most popular courses among CS students. The 2023 section has a size of 76. The recent student evaluation rating is 4.6 (2023, CS mean 4.3), and 4.9 (2022, CS mean 4.2).

This course includes the important topics of distributed systems, including distributed replication consistency, consensus, distributed transactions, and blockchains. The course is very challenging as it requires not only a good understanding of the textbook distributed algorithms and protocols, but also a deep understanding of the systems and their history, which is not usually available by simply reading papers. In my experience, providing enough background and context is the best way to clarify the complex concepts. For example, the students were confused by the term "consistency". The reason for this confusion is that "consistency" is heavily overloaded. It could refer to a certain isolation level (for example, snapshot isolation), or a distributed consistency model (linearizability), or user-defined constraints in a database (C in ACID). I was really glad to have the chance to clarify this issue; and the reaction from the students further convinced me that providing sufficient background knowledge not only helps in undergraduate teaching but also helps—perhaps even more—in teaching advanced topics. Another lesson I learned is that complex concepts can be made easier to learn through examples. One lecture was about a state-of-art technique—developing formally certified storage systems. Since the audience had only limited background on the topic, my lecture was designed to provide enough context before jumping into the details. For example, I introduced a real-world bug that this work tries to avoid and a code sample of certification. The experience helped me get acquainted with how to teach content that is not close to my line of research (but still within the scope of computer systems).

Here I include three quotes from the student evaluation:

"This is one of the best and rare courses available at SBU."

"The organization of the class and the curriculum is highly commendable."

"Professor Shuai Mu is a well-known profession and have board and deep knowledge in this field. For students who have system knowledge but new to distributed system, this course provide essential startup and some advanced knowledge. It is useful for students who want to perform research or engineering in distributed system."

For this course, I develop a set of distributed systems labs in C++. These labs provide skeleton and bootstrap code for a distributed storage systems. The labs require the students to fill in essential parts to

complete the system's function, then an automatic testing component will run and test the functionality of students' implementations. The students get to exercise their distributed systems knowledge by implementing it in real code. For example, the labs include a part where the students are asked to implement Raft, a consensus algorithm that is well known to be complex and difficult. In these labs the students get the chance to experience not only the complexity from the algorithm, but also from the other source—the implementation, including all kinds of systems programming components like threading, networking, asynchronous IO, etc. These labs are well liked by the students. Here are a few quotes from the student evaluations:

> *"The labs were very difficult, but very practical and rewarding to complete. ... I was finally able to create the right abstractions and have my code readable to the point that I could easily reason about its correctness. The coding tasks that initially seemed impossible now seem much easier, and I am now equipped to tackle even harder coding challenges in my future career."*
>
> *"CPP labs were excellent. Professor has put a lot of effort in implementing the labs. It definitively enhances logical thinking and debugging skills."*
>
> *"The assignments are challenging, but are a good way to prove understanding of certain course materials. ... Furthermore, the professor always acts in a fair manner. He only assigns homework after it has been bug-checked and works correctly ..."*

2 Undergraduate Teaching: CSE-416 Software Engineering

I have taught an undergrad course at Stony Brook, CSE-416 Software Engineering. Despite its name, this course is more of practice course for senior undergraduate students. This course usually has about 60 students. In this course, every 3-6 students team up, and do a web-focused project. The course was structured during the time when J2EE is dominating the software engineering world. Instructors can use J2EE-like tech stacks as a good training platform for teaching concepts of UML diagrams, use case documents, MVC, unit testing, and OO programming patterns.

Although J2EE is still an important software engineering technique (with a different name) now, in today's world a "software engineer" has a very different range of responsibilities. They work on very different tech stacks and work in very different culture. Very few students will find themselves working with the classic J2EE concepts, nor the UML-driven development style. Therefore, I try to organize the class in a slightly different format. I developed a new course curriculum that discusses topics of blockchain and cryptocurrencies. The students are teamed to work a project on this topic. I am also trying to simulate a "Silicon Valley" culture for developing a project. For example, we value meeting notes over having a well-defined contract document; we value fast iterations over careful planning; we value whiteboard analysis than having carefully drawn diagrams.

In this process, a great help is that there are many public documents, papers, and books in discussing how successful companies are organized. In particular, we try to simulate Google's culture following its published materials. Google has released blogs and papers explaining in details how the company organized and how things run internally, such as how they do hiring (so we train students accordingly for their interviews), how they do code views (the code review is set mainly for knowledge transfer rather than quality control), how Google evaluates their employees (we use the same evaluation process).

I organized the class with these perhaps unusual arrangements, aiming to help students transition from their student role to an engineer role. In my own experience, this is the biggest challenge after graduating college and I hope my class can help ease that challenge. For example, as students we were very used to an objective grading standard: how much marks we can get from the exam directly becomes our grade. Life as an engineer, however, is very different. There is no more exams. Our performance

review come from our supervisors but more from our peers. In this class, as we do not use exam but peer reviews to evaluate students, I can see this transition is indeed surprising and challenging for students. In the first half semester they are often not ready for the new standards, but as things go on and they become used to the new standard, they can become very productive and own their projects very quickly.

Here I include two quotes from the students evaluation:

> *"Shuai is a good professor who really wants to see his students learn and grow. When bringing questions to Shuai about details of our project he will always provide a well- thought out and easy to understand answer. The articles / topics we discussed in class were also quite interesting."*

> *"I liked the open source nature of the class where we could all collaborate on one project, and learning important technology fundamental to real world jobs such as making pull requests. "*

3 Student Mentoring

At Stony Brook I have had the pleasure to work with many brilliant students. The experiences were enjoyable and fruitful. The most valuable lesson I learned from these experiences is that the key to achieving a productive relationship is effective communication and careful planning. In addition, I need to apply higher standards to myself than to the students, so that I can be a role model for them. I believe that one of the greatest advantages of a faculty position is the chance to advise students. It is always a great pleasure to see how the creativity of two individuals can be combined. I very much look forward to advising future undergraduate and graduate students.

Ph.D. Students and Postdoc. I am advising five Ph.D. students (including two co-advised students). I am also advising a postdoc. My first Ph.D. student, Weihai Shen, will propose his Ph.D. thesis in Fall 2024. I design a research plan for each student that fits their background and experience. New students with less research practice get involved in existing projects. This lets them learn from and work alongside more experienced students. Once students are ready to take the lead, I help them design their own projects, find a team, and guide them throughout the research process.

M.S. Students. At SBU, M.S. program includes pursuing an advanced project with a faculty member over 2 semesters. I have advised / I am advising 13 M.S. students with advanced projects. I also advised two M.S. student with thesis, including co-advising an UIUC M.S. student (Andrew B. Yoo), whose thesis work was recognized by Siebel Scholar award. I use the same supportive approach for M.S. students as for Ph.D. students. However, because M.S. programs are shorter, I assign research projects that are clearer and more focused. This ensures they can be completed within the program timeframe.

Undergraduate and High School Students. I have worked with five undergraduate students through the BSMS program in SBU, SBU-BNL program, and an Indonesian government funded collaboration program. I also worked with four high school students through SBU's CSIRE summer program. Through participation in my research projects, students gain valuable research skills while experiencing the thrill of real-world computer science. They find the collaborative environment of a research lab to be particularly rewarding. Just like with my graduate students, I ensure consistent support by meeting with all my advisees weekly. These regular interactions keep their research journeys productive and enriching.