Yue Cheng — Teaching Statement

I request consideration for promotion to Associate Professor with tenure in the School of Data Science at the University of Virginia (UVA). This statement summarizes my teaching philosophy, teaching experience, and future teaching plan.

I am passionate about teaching and advising. In many aspects, the opportunity to teach and mentor students is what had driven me to pursue an academic career.

During the past five years at GMU, I have taught and mentored students at different levels. I have found these experiences quite fulfilling in different ways. Teaching helps me reflect on my own research: an independent study project sometimes sparks a new research idea and an exciting research project. More importantly, teaching and mentoring helps students succeed in their future career. Perhaps the most rewarding moment was when I received the university recognition that:

"students identify you as someone who has made a positive impact on their career goals, employment plans, or graduate school preparation".

1 Teaching Philosophy

1.1 Learning by Doing

Systems topics often involve hard-to-understand, abstract concepts, principles, and designs. I believe that the best way to understand these hard systems problems is through *"learning by doing"*. Textbook-oriented lecturing is far from enough: students gain rather superficial understanding if they only read the provided materials. Solving problems through hands-on programming labs / projects is key to a deep understanding. Take concurrent programming as an example—`lock()` and `unlock()` should be used cautiously, and a common design pattern is not to hold a lock while waiting, e.g., for an RPC (remote procedure call) to return; students best understand these strategies to avoid common anti-patterns only when they have seen and debugged issues such as deadlocks themselves.

1.2 Learning by Examples

I believe example-driven lecturing is critical to smoothing students' learning curves when digesting abstract protocols and designs. I use visual and demo examples extensively to make materials easy to digest. To explain CPU scheduling policies, I demo'ed the difference of CPU scheduling policies using visualizations provided by KernelShark; students can then better relate textbook knowledge to practical systems and use the right tools for problem solving.

1.3 Learning/Improving through Feedback

The key to successful teaching is effective communication; providing and receiving feedback helps build effective two-way communication. For instance, before class I ask questions in order to help students better prepare for the lecture. Asking questions, however, is not the purpose. I always provide feedback to students' answers. This not only helps students but also helps me readjust my teaching to better address confusions in class. On the other end, I proactively seek feedback from students. For example, I create midterm surveys to seek student feedback mid-way through the semester; based on the feedback, I fine-tune my teaching for improved learning experience.

2 Classroom Teaching

2.1 Undergraduate Courses

At GMU, I have taught two undergraduate CS courses, including a core course *operating systems* (for three semesters with a total enrollment of above 200 students from Fall 2017 to Fall 2019) and an elective course *concurrent & distributed systems* (with an enrollment of 60 students in Fall 2021). In teaching operating systems, I have significantly improved the course by modernizing all materials from slides to assignments. I have switched to a state-of-the-art, open-source OS textbook Operating Systems Three Easy Pieces (OSTEP). I have also developed a set of engaging delivery methods and tools—e.g., live demos and modularized lab assignments—which help students easily relate low-level OS implementations to high-level design principles.

In teaching concurrent & distributed systems, I use examples of real distributed systems that come from my first-hand experience as a systems researcher. Traditionally, this course taught at GMU had a strong flavor of concurrent and parallel programming but lacked emphasis on distributed systems. I refactored the syllabus of this course and developed a completely new set of materials drawing from courses at MIT 6.824 and Princeton COS-418. Beyond covering classic and modern distributed systems designs and principles, it features real-world concurrent and distributed programming with notoriously challenging algorithms such as Raft. Students are asked to fill in key missing components such as leader election and log replication in provided skeleton framework. By the end of the semester, students should have implemented a strongly-consistent, fault-tolerant, distributed key-value store. The goal is to help students gain first-hand experience in debugging challenging concurrency bugs and building workable distributed systems from abstract algorithms and specification.

2.2 Graduate Courses

At graduate level, I have taught *operating systems*, *distributed systems*, and have created a new course *cloud computing*. My graduate courses are tailored to introduce and train graduate students in systems areas. In teaching cloud computing, I built a new project suite that allows students to experiment with distributed systems principles in the context of serverless computing. Students use Go and its builtin `plugin` package to extend the core modules. Lab 1 and Lab 2 ask students to experiment with RPC library and build a serverless dataflow framework; in Lab 3, students would expand on Lab 2 by creating serverless microservice apps where a microservice is a Go plugin Lambda that could communicate with others using RPC or Go channels. This course was very successful and resulted in two Master's students joining my research group.

In addition, I proactively seek feedback from students. During the early pandemic in Spring 2020, I was teaching two graduate-level courses. Unfortunately, formal teaching evaluation was cancelled university-wise for that semester due to COVID. Therefore, I created my own informal teaching evaluation forms and asked students for feedback on how different teaching strategies used in pre- and post-COVID period would affect the learning experience—I adopted two different strategies for these two courses. I built on these valuable feedback to improve my teaching methodology and material. (See the next paragraph for example teaching evaluation feedback that I have received during and before the COVID-19 pandemic.)

2.3 Course Evaluation

My average course evaluation at GMU has been 4.35 (out of 5) with a median of 4.88. This places me above the department- and college-level ratings. In those evaluations, students acknowledged my effort and practices in engaging students and cultivating concepts in the classroom. Some of the notable comments from students:

- **(1)** "*I quite liked your lecture style, especially how you did not simply create a rehashing of the reading in your slides but delved into specific examples and asked questions of the class as you went. I don't think you ever just read a slide verbatim and moved on, which I really appreciated. You elaborated and provided context and examples...*"
- **(2)** "*Your tutorials/introduction videos for the programming assignments were INVALUABLE. It was so incredibly helpful to see you walk through the logic of the problems we were trying to solve and especially where to look in the code, how to properly invoke make and modify makefiles, etc. I suspect that required significant effort on your part beyond the required curriculum, but the payoff for students (at least me) was huge...*"
- **(3)** "*Of all my classes, your solution to the online lectures was unique and, in my opinion, the best way to handle the changing circumstances...*"

3 Future Teaching Plan

I strongly believe that obtaining necessary background in computer systems and Computer Science in general is necessary for all students interested in Data Science or compute-related careers. I am looking forward to teaching Data Science courses on big data systems, cloud computing, and data management & storage systems. I am also enthusiastic about designing new courses such as data processing/data storage/data management, serverless computing, tooling support for data processing and data wrangling, and data-intensive computing. In my future advising, I will reflect on my experiences and strive to build a nurturing research team that promotes student success.

4 Concluding Remarks

This statement summarizes my teaching philosophy and experience as an assistant professor for the last five years. With a carefully planned teaching agenda towards the joint domains of Data Science, Computer Science, and domain-specific computing. I am committed to continued excellence in teaching and education for years to come.