AISHWARYA GANESAN – Teaching Statement

Teaching Experience and Philosophy

Experience. I have enjoyed teaching since when I was an undergraduate student in India. As part of an outreach program, I regularly tutored middle and high school kids from a residential group home on various topics in math and science during my undergraduate studies. During my Masters at IIT Bombay, I was a teaching assistant for an undergraduate algorithms course and a graduate databases course. At UW Madison, I have delivered guest lectures in graduate-level distributed systems courses.

However, my significant teaching experience stems from my role as a *co-instructor* for the *graduate distributed systems* course (CS 739) at UW Madison in Spring 2020. The goal of this course is to expose students to fundamental problems and ideas in distributed systems. I designed the course based on two pillars: (i) reading papers and (ii) doing a sizable research project. Many students who took the class had not taken an undergraduate-level distributed systems course. Thus, I started each lecture by providing background and context before diving deep into the paper. Students began with a (well-defined) warm-up project and then proceeded to do an open-ended research project. I mentored nine groups of students (a total of 32 students) in the semester-long research projects. These prior teaching experiences inform my teaching philosophy.

Philosophy. I believe teaching is not just about depositing facts to students. Instead, my approach to teaching is based on *collaborative learning*, where I create opportunities for students to critically think and engage with me during the lectures to arrive at solutions. I believe this approach has led to increased student participation in the classroom and consequently resulted in effective learning.

I implemented this teaching approach in CS 739 in three main ways: material preparation, quizzes, and student meetings. First, when developing lecture materials such as presentation slides, I deliberately leave blank spaces. Then, throughout the lecture, I create opportunities for students to think and work together to fill in the missing pieces. For instance, when teaching about Byzantine fault tolerance, I did not present the entire final protocol right away. Instead, I created and asked questions for every key step in the algorithm (e.g., what is the problem with the primary ordering the client requests?). Answering each question and understanding the implications got the students one step closer to the final solution (e.g., the primary cannot be trusted, and thus one needs an additional phase to ensure correct ordering of requests). The interactive nature of my lectures was also crucial in providing a classroom-like experience when we had to move our lectures online mid-semester due to the pandemic.

Next, I designed quizzes to make students think of scenarios/cases/implications not taught in the lecture. I gave these questions either during or after the class. I encouraged students to work in small groups to find the answers. For example, when I taught Raft and consensus protocols, I presented different Raft logs pictorially and asked: "is this a valid log state?" and "can a node with this log become the leader?" These questions encouraged students to think critically and helped fill gaps in their understanding. Finally, I followed the collaborative learning approach when interacting with students during meetings. For example, when advising a project group, instead of prescribing a design, I asked questions to make the students think about the pros and cons of their design. Answering these questions often led students to propose better designs on their own.

The collaborative-learning methods enable students to critically think and analyze the pros and cons of a design. However, in computer systems, a myriad of practical problems arise during implementation. To this end, I compound the collaborative learning approach with two other methods. First, I teach how ideas learned in the class are implemented in real systems. I often do so by including case studies on practical systems (e.g., Zookeeper coordination service when teaching consensus in CS 739). Second, I follow a hands-on approach to learning through assignments and projects. For example, in CS 739, students built a durable and consistent key-value store to apply the concepts learned in class. Additionally, as part of the course projects, students implemented their research ideas by extending popular open-source distributed systems.

Effectiveness and Evaluations. From the course evaluation, I am assured that the collaborative learning approach has yielded benefits. Students overwhelmingly agreed that the collaborative and interactive nature of the classes were helpful. In an anonymous course evaluation, students were asked: *What did the instructor do that helped you learn the material?* Here are a few comments from students: (i) "*Interactive lectures were crucial in retaining attention. Post-class questions were challenging, helped fill in gaps.*" (ii) "*Questions asked in class encourage thinking critically about the papers.*" (iii) "*They (the instructors) encouraged questions and discussions and made the class super interactive.*"

Overall, when students were asked to rate the instructor on a scale of one to seven (seven being the highest), students gave an average rating of **6.42/7**. When asked to rate the course, students gave an average rating of **6.5/7**;

this score was the *highest* among all the courses (both undergraduate and graduate) in the CS department in Spring 2020. Students also felt that our class adapted well to the online setting, giving us a score of **6.38/7**; I firmly believe this is due to the collaborative nature of the lectures. Finally, an anonymous student noted: "*The course was very well structured and probably the best course I have taken so far at UW Madison.*" For my teaching, I was awarded the **Graduate Student Instructor Award**, a department-wide award at UW Madison.

Courses I can Teach. As a faculty member, I would be thrilled to teach undergraduate and graduate classes in distributed systems and operating systems. Outside my immediate area of expertise, I would be excited to teach computer networks, databases, and introductory courses such as computer organization and data structures. In addition, I would love to teach seminar courses focused on special topics such as distributed storage, consensus protocols, big data systems, and edge computing. I would also like to create hands-on courses where students can build systems for emerging network and storage devices. As a part of such a class, students will read and discuss recent papers and get research experience through sizable course projects.

Mentoring Experience and Philosophy

Mentoring students is a very fulfilling and rewarding experience. I have mentored several students on research projects during my Ph.D. at UW Madison and also my postdoctoral stint at VMware Research. First, I have mentored more than 30 graduate students on semester-long research projects as part of CS 739. I first created high-level research ideas for the groups to work on. Then, I regularly met with the students and advised them on system design, experimental setup, and presentation. One of the projects built a learned index for LSM trees, led by two graduate students (Yifan Dai and Yien Xu). I continued to closely work with these students after the course and eventually published this work at OSDI 2020. Another project on using prediction-based techniques for efficient request ordering showed promising results and informed my future research. I have also mentored undergraduate students. At UW Madison, I co-advised Neil Perry, who is now a Ph.D. student at Stanford University. Continuing our prior work on storage faults, Neil analyzed the effects of storage corruptions in the Ethereum blockchain. Finally, at VMware Research, I am mentoring Yi Xu (a graduate student at UCSD) on exploiting persistent memory technologies in an in-house key-value store. This work is currently under preparation.

As a new faculty member, I am eager to nurture students (both graduate and undergraduate) in my research group. My mentoring philosophy is based on two main ideas. First, I will train my students to produce the best-possible research while enabling them to become independent researchers. In my experience, my best ideas have come from brainstorming sessions with my co-authors. Therefore, as an advisor, I intend to closely work with students (especially junior ones) to brainstorm ideas, evaluate design choices, and discuss experiments. At the same time, I will give students enough space and time to grow as independent researchers. For instance, once a student has gained research experience, I will encourage them to explore ideas on their own and convince their co-authors (including myself) of the idea's merits. My goal as an advisor is to *(i)* empower students to believe in their abilities to conduct and communicate cutting-edge research, and *(ii)* enable students to thrive as independent members of the research community.

Second, I firmly believe that for effective advising, a mentor must be truly invested in the growth and well-being of their mentees. To enable career growth, I will encourage my students to apply for fellowships and awards. In addition, I would also introduce them to mentors who would complement me (e.g., a mentor from an industrial lab), exposing them to opportunities outside the university. Finally, I am committed to the well-being of my students. As an advisor, I consider it my responsibility to create a safe and non-toxic yet also productive workplace for my students to succeed.