Teaching Statement

Kostis Kaffes

I am excited to mentor, advise, and teach students as part of my faculty job. My approach to teaching and mentoring is grounded in core principles that I have identified over the course of my teaching, research, and mentoring experience.

Teaching Experience

Cloud Computing Technologies. In Fall 2021, I was the Teaching Assistant (TA) for CS349d: Cloud Computing Technology at Stanford University. CS349d is a Ph.D.-level course where students present and discuss seminal papers on cloud computing and work in groups in a quarter-length research project with the goal of producing a publication. As a TA, I helped design the course curriculum, helped students with their projects, held office hours, and generally answered any questions raised by the students. I particularly enjoyed discussing research ideas with students and guiding them to solve impactful yet tractable problems.

Computer Systems Architecture. Throughout my Ph.D., I have assisted my advisor and instructor, Professor Christos Kozyrakis, in teaching the EE282: Computer Systems Architecture class at Stanford University. This includes guest lecturing and helping with grading midterm and final exams. For example, in Winter 2020, I did a guest lecture on hardware multi-threading, had a discussion with students on the topic, and answered their questions.

Introduction to Programming & Operating Systems. My teaching experience also includes classes at the National Technical University of Athens. As a lab assistant for the first-year Introduction to Programming and the more advanced Operating Systems courses, I helped students with their lab exercises throughout the semester. Assisting in an introductory course gave me a valuable lesson. The student background is very diverse in such classes, and it is crucial to provide an inclusive learning experience for everyone.

Teaching Philosophy

My teaching philosophy has been shaped by my own experiences over the eleven years I have spent in higher education, as well as by the great teachers I had the fortune to come across.

Adopting the Socratic Method. Time spent in lectures is a very small part of students’ lives. Thus, my goal is to provoke the students’ curiosity and teach them how to learn regardless of the setting. This can be achieved through the Socratic Method, which uses argumentative dialogue between teachers and students to stimulate critical thinking. In a traditional university classroom, this approach translates to using participatory learning. As I did in CS349d, I gently push students to drill into problems, raise questions, and think about answers. This approach increases student engagement and improves the ingestion of class material. It is much easier for a student to remember something they discovered themselves than to recall dry pieces of information.

Being Inclusive. Students have different starting points, coming from diverse backgrounds, but I always strive to ensure that they all get equal opportunities and reach the same end goals. I make my classroom a supportive and inclusive environment so that all students can feel confident to succeed. For example, having a speech impediment myself and being a non-native speaker, I understand that it is often hard for students to participate in classroom discussions. Thus, I offer different participation opportunities to students with similar difficulties, e.g., submitting questions before class or recording videos instead of doing in-class presentations.
Learning by Building. Disappointed by the lack of hand-on experience in the classes I have taken, I will design the courses I will teach around the maker mindset. Students will pick systems apart, understand how they work, and build new ones. Homework assignments and class projects will be open-ended, stimulating perseverance, problem-solving, and creativity. The satisfaction of building something useful motivates students to keep innovating in the future, even after they graduate.

Course Plans
Given my background in computer science and electrical engineering, I would be happy to teach classes related to operating systems, networking, computer architecture, and distributed systems at a graduate or undergraduate level. Following my teaching philosophy, when possible, I will structure my classes around class-long projects that allow students to dive deep into the material. For example, for an introductory networking class, I will have students re-implement the TCP stack. At a Ph.D. level, I would like to develop research-oriented, project-based courses on topics such as:

– Scheduling Across the Stack: a course inspired by my Ph.D. research covering different scheduling policies and mechanisms used at various layers of the stack ranging from global load balancers to programmable network devices to operating systems. The course will focus on identifying common patterns and requirements across seemingly different workloads and examining how changing performance requirements affect policy design and mechanism selection.

– Programmable Systems: The end of Moore’s law makes it imperative to add specialization across the stack. However, it is hard to find the right balance between customization and programmability. To address this challenge, I will design a course with the goal of making computer systems more easily programmable and tunable by application developers. Each iteration of the course will focus on a different system component and work towards defining an API that will easily and safely transfer control over its behavior to users.

Mentoring
Mentoring experience. Throughout my Ph.D. I am proud to have worked closely with and mentored several graduate and undergraduate students. Jack Humphries has worked with me on several research projects while an undergraduate student, an employee at Google, and eventually a Ph.D. student in our lab. Initially, I assigned Jack tractable problems that tied into my own research. Over time, my role changed as I provided feedback and guidance on Jack’s research ideas. Jack has been very successful, having co-authored NSDI and HotNets papers as an undergraduate student, while just this year, he got two papers published in SOSP and one in HotOS. During the summer of 2021, I mentored Clemente Farias Canepa and Yale Wang. Clemente integrated an open-source serverless platform, vHive, with the gg execution framework. Yale built serverless functions that can replay publicly available execution traces. Clemente’s and Yale’s work is very impactful as they created infrastructure that enables better experimentation over serverless platforms facilitating future research. I have also informally mentored junior Ph.D. students in our lab by giving them feedback on their research ideas and talks and general advice about navigating the Ph.D. process.

Mentoring philosophy. One of the main advantages of being in academia is the ability to advise and mentor students. I believe that advisors should teach their advisees to be independent thinkers, form their own research questions, and be confident in their ideas. My ultimate goal is to be able to learn more from my students about their area of interest than what I can teach them. Thus, I will always give my students the freedom to pursue the topics they are most passionate about, and I will adjust my advising style based on the individual needs of each student. Motivated by my own experiences, I am also particularly excited about giving research opportunities to undergraduate students and offering additional support and mentoring to students with disabilities and students from underrepresented backgrounds.