Teaching Statement | Milijana Surbatovich, Fall 2022

The opportunity to teach and mentor students is the key reason why I want to become a faculty member.

Teaching Philosophy & Interests

My approach to teaching will center ***actively engaging students*** in a way that develops their ***ability to connect concepts to practice.***

Students perform better when they engage with the material instead of passively listening to a lecture. For students to meaningfully engage, I think they need a variety of ways to participate and build confidence. For example, when I have given guest lectures, I frequently ask the students questions and give them many opportunities to ask questions in return. While questions are one very important way for students to interact more deeply with the lecture content, I tended to get answers from the same handful of students. Directly calling on more quiet ones would force participation but not necessarily build confidence, if they feel they got it wrong. Thus, I will design my classes to use *active-learning methods*, e.g., think-pair-share, giving students in-class opportunities to discuss and grapple with concepts, peer-to-peer. Along with demanding active attention from students, such exercises provide more low-stakes ways to participate.

I believe that a class is successful if students can apply what they have learned to outside contexts, whether another computer science or engineering class, an internship project, or even a class from a different discipline. To accomplish this, I need to cultivate a student's ability to build connections between a concept and its application, beyond the immediate next homework or exam. As an interdisciplinary researcher between programming languages and systems, I frequently have to connect formal methods theory to low-level system details. What I have found most helpful in developing papers and talks is examples. Stepping through examples exposes the process and logic behind a concept and makes connections explicit. I will combine example-based lectures with active learning, e.g., after going through an example, I will modify one aspect and have the students discuss what changes. Additionally, one strategy I think is very important for a student to internalize a concept is debriefing both correct and *in*correct answers, as part of the next lecture after an exam and after every in-class quiz or activity. Explaining why an answer is incorrect allows students to learn from mistakes and identifies wrong intuitions early.

Teaching interests. My research area is applied programming languages and formal methods for non-traditional computing systems. I frequently use language-based security reasoning and plan to move more in the security direction. I like hw/sw interactions. As such, I am interested and equipped to teach undergraduate courses in computer systems/organization, compilers, security, and programming language design. At the graduate level, I can teach courses on computer systems, compilers, software and information security, or formal foundations of security. I would like to create a course about programming languages for hw/sw co-design and run a seminar on hardware verification.

Teaching Experience

My teaching philosophy and goals are informed by my experiences leading workshops for introductory courses, TAing undergraduate and graduate courses, and pedagogical training.

When I was an undergraduate, I was a workshop leader for the introductory programming sequence (one semester of programming and one semester focusing on data structures.) Workshops were a weekly meeting with assigned groups of 6 -12 students, during which I would go over a tiered problem set, focusing on teaching the students problem solving skills and thinking programmatically, rather than simply reviewing the lecture content. This experience taught me how to manage a group with differing initial skill levels, as some students came in with high school programming classes and for others this was their first CS class. A key technique I found useful was stepping through the first problem or two together and then breaking into smaller groups. In the smaller groups, I would have the more advanced students teach the ones who were struggling, peer-to-peer. This encouraged participation from students who were otherwise afraid to speak up in the whole group, and it solidified the material for the

advanced students more than simply rushing to get through the most advanced problems would have. I can adapt these techniques to larger classes through class time activities, e.g., by building in time after going through example problems for students to discuss a more complicated version or application with the people sitting around them.

In graduate schools, I TAed an upper undergraduate course on the formal foundations of security (~60 students, sole TA) and a mixed under/graduate course on software security (~70 students, one other TA). A key responsibility for both of these was holding weekly office hours. Particularly for the undergraduate course, where homeworks were proof based, I tried to help students in creating their own ``scaffolding'', working through an easier version of the problem, so they could figure out the next step to bridge the gap towards the real homework problem. I generally found this to be even more successful if multiple students came in for the same problem, since they could build of off each other.

Additionally, I am on track to complete the Future Faculty Program, given by CMU's Education Center. As part of this program, I have taken seminars on best teaching practices and course design, with an emphasis on centering DEI as a fundamental course principle, and I have worked with teaching fellows to incorporate active learning and foster student engagement in lectures. My key takeaway from this program is to design my courses to be *goal-oriented*, with explicitly articulated learning objectives. Knowing how assignments connect to course goals helps student motivation and explicit objectives help students in applying what they have learnt.

Mentorship Experience & Philosophy

My mentoring goal is to strengthen each student's ability to solve problems independently and formulate research questions, at a level appropriate to the stage of the student's academic career.

I have experience mentoring high school and undergraduate students, and I am excited to continue mentoring students at these stages as a faculty member, along with graduate students. I mentored the high-school student over a semester-long internship, guiding her in learning programming at the introductory college level. We would have a brief check-in each day to track her progress. A key technique I found useful was to have her recap in her own words what she learned in the previous day, to emphasize the concepts learnt, not rote terminology. I would also encourage her to generate at least one question or to discuss what was particularly challenging, to cultivate the ability to identify and articulate gaps in knowledge. I mentored an undergraduate student for several semesters, as an assistant for my ongoing research project. She was already skilled with proof writing but was unfamiliar with computer system design. During our weekly discussions I concentrated on explaining how to connect her specific proof tasks to the big picture of the system we were reasoning about, as a key research skill is being able to define the boundaries of a problem and identify concrete ways to break it down into approachable chunks.

In mentoring graduate students, I think the key challenge is fostering the ability to do independent research while maintaining student motivation. Three concrete components of that goal are learning self-direction, learning how to scope a project, and developing resilience to stay motivated. First, from my own experience, an important way to develop self-direction is to hold weekly meetings with the student, where they set the meeting agenda, identify the challenges from the past week, and come up with concrete next steps, with brainstorming assistance as needed. Second, a technique that I found useful in learning how to scope projects and will emphasize with my students is considering writing as *part* of the research process, not just the final step needed to put a paper together. Writing forces clarity of thought, requires identifying the flow and connection between ideas, and clarifies the overarching purpose of a project. Thus, writing down the story behind a research project early guides the process and helps determine the scope of an idea. Finally, I wish to foster a sense of resilience by creating both a constructive and supportive group environment. One reason graduate school can be grueling is that feedback---whether through paper reviews or more informal group situations---tends towards the negative, as most feedback *is* criticism, usually constructive (and sometimes not). To teach students how to deal with rejection, I will share my own experiences and techniques to move forward with implementing the constructive portions. When giving feedback, e.g., during a practice talk session, I will identify and point out what worked along with areas needing improvement, to help them remain motivated and grow their strengths, as well as to model a positive culture within the group.