Teaching Statement

Michael H. Borkowski

My teaching philosophy is to develop students as critical thinkers by embracing active learning in an interactive classroom while also providing an inclusive and supportive environment. Active learning engages the students in the process of their own learning, allowing them to collaboratively build problem-solving and programming skills during class meetings. Taking the time to embrace students as individuals with diverse needs and backgrounds enables me to create an inclusive and supportive environment in all aspects of the course. I take an approach of iterative improvement to my teaching and course materials, responding to student feedback and incorporating technology to identify which aspects of a course could be improved in the next offering.

I have taught three undergraduate courses at UC San Diego as an instructor of record: an introductory pre-first year summer course (one of three instructors), a discrete mathematics course for first- and second-year students (which I am also scheduled to teach again this winter quarter), and the upper-level programming languages course[1] for juniors and seniors. These courses ranged from 23 to over 80 students. In addition to this I have also served as a teaching assistant 12 times for a variety of courses ranging from under 15 students to nearly 300. I have been recommended by 93% of student TA evaluations across all courses.

Inclusive and Supportive Environments I recently had the opportunity to work as one of the three co-instructors alongside Profs. Niema Moshiri and Curt Schurgers in the Summer Program for Incoming Students[2], offered directly by the CSE department at UCSD. This year's program consisted of 51 students selected mostly from low-income backgrounds with little formal exposure to computer science. Because the students were entering a college classroom for the first time, and because there was a wide range of prior preparation and skills, we approached teaching by combining an inclusive environment with a no-stakes structure. We personally met with every student each week to check in, offer advice, and to try to solve any problems that arose. There were no grades for labs and projects, only constructive and encouraging feedback from undergraduate mentors. No college credit was given, freeing us from exams and letter grades, as well as keeping program fees lower and financial assistance relatively higher, which increased participation of students from low-income backgrounds.

I apply these general principles in all my teaching: to reduce the stakes for homework assignments many of my assignments have an optional "second chance" submission phase after the graded assignment is returned. This allows students to iterate on their work and take feedback into account right away. I also reserve a portion of my office hours for individual one-on-ones, and I ensure that some of my teaching assistants do likewise. In UCSD's large undergraduate program, students feel that office hours can be dominated by questions about the current homework. Other students feel uncomfortable about asking questions on earlier or more basic material in front of several classmates. These one-on-one meetings allow my staff and I to determine how best to

[1] https://canvas.ucsd.edu/courses/44436
[2] https://spis.ucsd.edu/academic.html

support students as individuals. When students express concern about struggling in the course, I assign one of the teaching assistants to meet with the student in a longer one-on-one meeting or do so myself.

Active Learning in Lecture-Based Classes Whether a lecture is scheduled for 50 minutes or three hours, I have broken up lectures into smaller segments so that students don't become passive notetakers. I use polls to both check for understanding as well as pose practice problems for discussion with the student's neighbors. Students often do not want to be the one to break the silence, so it's crucial for me to leave the front of the room and talk to students where they are sitting. In a larger lecture hall, I can facilitate introductions and conversation between students across a few empty seats. The opportunity for students to share knowledge and skills with each other in a low-pressure environment also helps students with "writer's block" when staring at a blank editor or piece of paper. After these exercises, we can work through these examples with live-coding or on a virtual whiteboard. I emphasize a free-flowing exchange of ideas, taking time to explore and show the value in every thoughtful suggestion, correct or incorrect. Ultimately the goal is to emphasize the learning process and struggling with new concepts over memorization and "getting it" quickly.

Similarly, rather than show lecture slides with pre-written proofs in a Discrete Mathematics class, I often use blank slides which enable me to write out a proof while talking through the thought processes that the students will go through while writing such a proof. While the course material teaches specific proof structures and techniques for students to combine and use, students are often unsure which technique to choose and what facts about, say, natural numbers can be used. Modelling the thought processes involved in generating a mathematical proof enables students to better grasp the critical thinking necessary to determine which techniques to use in which situation.

This active learning process is something that I seek to continually improve. After my first experience as an instructor of record, I realized that polls incorporated into lecture slides worked well for checking conceptual understanding but didn't work as well for more open-ended problems. To solve this, in my next class I incorporated more diverse exercises and activities by adapting worksheets used by other instructors to produce handouts containing in-class exercises as well as relevant definitions. These were available both on the course website and as hard copy handouts. Extra problems were provided for students to use as practice and were later covered by the TA in the next discussion section, which provided concrete motivation to attend the extra hour.

Technology in the Classroom I embrace the role of technology in teaching, harnessing it to improve student experiences. As the instructor of the Programming Languages course, I migrated the GitHub homework repos onto GitHub Codespaces, which provides the VSCode IDE and access to the GHC Haskell compiler in the browser with a few clicks and "just works." This freed my staff of three TAs and three undergraduate tutors from having to spend the first month of the quarter primarily providing tech support. Students were much less likely to be blocked by unfamiliar errors and fall behind.

I also intend to incorporate learning technology into a discrete math course. For instance, Proof Blocks[3] is a recent tool built into the open-source PrairieLearn platform that can pose mathematical proofs as Parsons problems, where students drag and drop components of a proof into order. The research[1] on this tool shows comparable gains in learning to writing proofs from scratch. For my scheduled discrete mathematics course next quarter, I will deploy a Prairie Learn

[3] `https://www.proofblocks.org/`

server this year to use Proof Blocks in review/reinforcement problems that I am writing.

Teaching Plans I would love to teach courses across the core computer science curriculum: introductory courses, discrete math, theory, and programming languages. I would also be excited to create or revamp a course on web development/applications or databases. These courses would involve project-based and team-based learning, and give students hands-on experience with technologies they may encounter in future software engineering internships.

Copies of my teaching and TA evaluations are available at `https://michaelborkowski.github.io/files/evaluations.pdf`.

References

[1] Seth Poulsen, Yael Gertner, Benjamin Cosman, Matthew West, and Geoffrey L. Herman. Efficiency of learning from proof blocks versus writing proofs. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSE 2023, page 472–478, New York, NY, USA, 2023. Association for Computing Machinery.