Saikat Dutta — Teaching Statement

Various teachers and mentors have positively influenced me throughout my life. They have helped me learn technical skills and how to think creatively and communicate effectively. During my PhD, I have mentored **19 students**: 16 undergraduate students (including four women undergraduates), one masters student, and two junior PhD students. Also, I was a teaching assistant for a graduate course and delivered several guest lectures across multiple courses. I have thoroughly enjoyed teaching and mentoring students in these roles. Hence, I am excited by the opportunity to be a teacher and mentor for the future generation of students. I intend to replicate my learning experience and foster an environment where students can learn, grow, and achieve their full potential.

Teaching

Philosophy. My experiences in education have motivated me to develop a teaching philosophy based on the following aspects: 1) lectures should engage students through illustrative examples, problem-solving, and insightful discussions to maximize learning, 2) the classroom should be a collaborative environment that allows every student to freely participate and learn irrespective of their backgrounds and skills, 3) teachers should establish a positive teaching relationship with students through effective communication, one-on-one conversations (e.g., during office hours), and by being approachable.

Experience. I was a teaching assistant (TA) for Advanced Compiler Construction Class (CS 526) at UIUC. As a TA, I was responsible for preparing and grading assignments, grading examinations, holding office hours to help students with any questions, and answering student questions on Piazza. I delivered two lectures on *Dependence Analysis* and *Control Flow Analysis* for the course. I developed a diverse set of test cases for the course assignment to evaluate student solutions (code submissions). I implemented an automated tool that compiles each submission, evaluates it on all test cases, and scores student submissions. My tool was also used in subsequent course offerings in the following years. Due to COVID, we had to move the course entirely online. During this uncertain period, I helped students adapt to the online version of the course by setting up online assignment submissions, conducting online office hours, and managing online exams and grading. As a faculty, I will leverage these experiences while designing my classes and incorporate them into my teaching approach, as outlined later.

Additionally, I delivered two guest lectures in *Approximate and Probabilistic Computing across the system stack* (CS 598sm-Fall 2020), one in *Topics in Software Engineering* (CS 527-Fall 2021), and one in *Topics in Programming Languages: Approximate And Probabilistic Programming Systems* (CS 521-Spring 2022). I have also co-organized the Software Engineering research seminar at UIUC for a semester, which involved organizing student research presentations and leading research discussions.

Approach. Students have a better learning experience when they have interactive engagement during the lecture, connect theory to practice, and are placed in a collaborative classroom environment. I intend to build such an experience through methods of active learning, practical hands-on projects, and classroom activities.

- **Active Learning:** Active Learning methods allow students to engage with the subject by giving them opportunities for thinking, discussing, and investigating. Research has shown that Active Learning can significantly improve students' learning experience [1]. In my courses, I intend to design activities such as posing a question related to a topic after introducing it during a lecture (e.g., pros/cons of an algorithm). Alternatively, I may present them with a problem to solve. I will then ask them to form small groups of two or three to come up with a response. Finally, I will ask each group to explain the solutions and provide them with appropriate feedback. Such activities allow students to apply critical thinking to new concepts, correct misconceptions immediately, and effectively learn complex concepts. This approach will also help me determine if students understand the material and adapt accordingly in future lectures. During my lecture on the *Dependence Analysis* topic in CS 526, I introduced various algorithms for analyzing data dependence across nested "for" loops. After each algorithm, I presented students with problems to solve. Listening to students' solutions revealed whether they correctly understood the topic or not and helped me clarify potential confusion. However, active learning can also be time-consuming and limit the coverage of topics. To alleviate these problems, I will incorporate active learning in smaller steps to find the right balance for maximizing learning.
- **Practical Hands-on Projects:** To obtain a deeper understanding of any topic, students need to explore beyond the course material presented in class. One common technique to provide such an experience is through practical hands-on class projects. When I TA-ed CS 526, students were required to implement a compiler transformation in the LLVM compiler based on given specifications and write test cases that

checked for the correctness of their implementation. Through this project, the students could understand the lower-level details of how a compiler pass is implemented in an industry-standard compiler and connect theory with practice.

In my future courses, I will design similar practical projects. A significant challenge in such projects is to ensure students make steady progress throughout the semester and have the necessary resources to succeed. To achieve these goals, I will 1) group students with different skills into small teams so that the project can benefit from their collaboration, 2) meet periodically with each team at regular intervals to checkpoint their progress and advise them when they require guidance, and 3) ask students to clearly define project goals, periodic deliverables, and individual responsibilities in the team.

Additionally, in a post-COVID world, it is essential to recognize the challenges while adapting a hybrid (offline+online) style of learning. A hybrid approach allows one to mitigate any interruptions to learning and makes the course more accessible to everyone.

Mentoring

Experience. As a PhD student, I have mentored 16 undergraduate students, one masters student, and two junior PhD students, each with different background and technical expertise. I mentored these students on various short- and long-term projects. These projects explored topics at the intersection of Software Testing, Program Analysis, and Machine Learning. Overall, my collaboration with these students has been a productive and rewarding experience. I have also published six research papers at top-tier software engineering conferences with my mentees. Two of the undergraduate students that I mentored are currently pursuing doctoral studies in the same research area; two students pursued masters degrees, while several others have accepted jobs in industry. Watching my mentees grow and succeed in their lives has made me very proud of their achievements and further motivated me to pursue the role of an advisor as a faculty.

Approach. From my mentoring experiences, I have learned that every student has different needs and responds differently to any given mentoring approach. Hence, I believe in tailoring my mentoring style to each student's needs, personality, and research experience. During the first meeting, I discuss with the student to define the project and its goals. Typically, at an early stage, I give students concrete tasks to solve. This approach allows them to learn a new research area and build confidence. I meet with the students regularly to check their progress, give them feedback, and guide them toward the next steps. I ensure the student understands their tasks' relevance in the project context and its goals. This approach gives them proper motivation while working on the project. As the students mature, I provide them with more abstract tasks so that they can grow their research skills and become independent researchers themselves.

Teaching Interests

Based on my experiences as a researcher and teaching assistant, I am excited to teach various topics such as Software Engineering, Compilers, and Program Analysis. I would be interested in offering graduate and undergraduate-level courses in these areas. At the graduate level, I want to enable students to be familiar with active research topics in these areas. I would also like to offer two new courses: 1) *Software Testing*: This course will discuss fundamental concepts related to testing, such as test generation, regression testing, and debugging. This course will contain assignments and a final project related to testing. This course will target a diverse student body containing undergraduate and graduate students and help them gain software skills that are crucial in both research and industry, and 2) *Software Engineering for Machine Learning and Emerging Areas*: This will be a graduate-level seminar course discussing the latest literature on applications of software engineering techniques for Machine Learning and other emerging areas. For this course, the students will be expected to work on a semester-long research project on a topic related to the area.

Additionally, I can also teach undergraduate-level courses on other topics, such as Data Structures, Algorithms, Operating Systems, and Database Systems. I would also like to organize or join a reading group in the department on software engineering, programming languages, and related topics. In the seminar, students will have the opportunity to lead discussions on various research topics, critically analyze research papers, and identify techniques to improve their research projects.

References

[1] Scott Freeman et al. "Active learning increases student performance in science, engineering, and mathematics". *Proceedings of the national academy of sciences* (2014).