Teaching Statement
Abhinav Jangda

I am an educator in computer science because I have really enjoyed both teaching and mentoring students and want to continue on this path. I have co-instructed an undergraduate course, was a Teaching Assistant for a graduate level course, and have mentored several undergraduate and graduate students in research projects. I am interested in teaching a variety of undergraduate and graduate courses including both core computing courses and specialized topics. I am also interested in teaching topics that are at the intersection of computer science and other fields of science, such as high performance scientific computations.

Teaching Philosophy  Computer science is a rapidly growing field with new significant advancements coming every year. After graduation, our students will have to learn these new developments themselves. My goal as an educator is to equip students with the skills they need to grow. I believe that a deep understanding of computer science requires learning concepts and then practicing these concepts by solving assignments and building group projects. Solving assignments helps students fill holes in their understanding. Building group projects gives them a platform to understand the real life usage of different concepts, understand why good software engineering approaches matter, and improves their confidence in their abilities.

In my teaching, students use concepts taught in the class to solve several assignments and work on group projects. I begin each lecture with a problem that cannot be solved by previously taught concepts and then show how new concepts can solve this problem. I structure the teaching material of each concept into smaller sub-concepts. After teaching a sub-concept I ask students about questions from their side. To keep students engaged in the lecture, I direct these questions to other students and I design the lectures to create opportunities where I can ask questions to students about already taught concepts. I use this technique to keep track of students that are not able to follow the lecture.

Teaching  I co-taught Introduction to Python Programming course as a senior undergrad at IIT (BHU) Varanasi. The course included teaching around 20 first-year undergraduates. I was responsible for delivering half of the lectures, creating and grading assignments, and creating and grading the final examination. In each lecture, I would teach concepts of Python like control flow, loops, and objects, and then ask students to solve a small in-class assignment. The small class size allowed me to provide individual attention to students. In each lecture, I took an example that could not be solved by previously taught concepts and showed how it can be solved by new concepts. I found that this approach works well in revising what has been taught already and show the utility of new concepts. Moreover, after I have taught several concepts I would give an assignment that utilizes all these concepts. I realized that students were interested in learning about different kinds of algorithms, so, I gave challenging assignments that could be implemented within the class time. I found that these challenging but digestible algorithmic questions kept them excited and engaged to learn new concepts.

During my PhD, I served as a teaching assistant for a graduate level course, Systems, that had around 30 students. This course discusses several aspects of computer systems, such as operating systems, compilers, computer architecture, and parallel programming, and involves writing paper reviews and building project assignments. The project assignments are different in every course iteration and are formed around a small research idea. This is why it is considered as one of the most time-intensive and difficult computer science graduate courses at UMass Amherst. I tried to make this course accessible to all students. I would hold regular weekly office hours for students to discuss their questions and hold extra office hours a day before the assignment submission deadline. Furthermore, I would implement the reference solutions before releasing the project assignments. Developing the solution helped me to assess the feasibility and difficulty of assignments and tailor the assignment to ensure that students will be able to complete it within the deadline.

Mentoring  My work on new domain specific abstractions for Graphics Processing Units (GPUs) has given me valuable experience in GPU programming which other graduate students at UMass Amherst do not have. My experience in GPU programming has given me the opportunities to teach and mentor students from other research groups, who are interested in leveraging GPUs in their own work. My mentoring approach is flexible and changes as the student gains experience working on the project. In the beginning, I do more hands-on mentoring which may involve teaching necessary concepts, reviewing code, and collaborating on the implementation. When the student has gained experience working on the project, I adopt a more hands off approach where I discuss new techniques and help them in the implementation if needed. I always meet them if they are stuck on a problem and work with them to find solutions. I believe that this kind of mentoring style helps students first get good research experience and then they can grow as an independent researchers with their own research style. Since research seems very hard in times especially when receiving paper rejections or when we are not getting desired results, I share how I kept myself motivated when I faced similar situations. I also share mistakes in my projects that have lead to incorrect code, so that, they do not do similar mistakes. I believe sharing these vulnerable stories
with the students helps them realize that failure and mistakes are part of research. I am mentoring two junior graduate students, Sandeep Polisetty and Hanmei Yang.

Sandeep made significant contributions to one of my PhD projects, NextDoor. He had significant past technical research experience, so, I took a hands off approach to mentoring. I taught him the latest and obscure features of GPU programming and then guided him to develop a benchmarking infrastructure for several graph neural networks as baselines in CUDA and PyTorch. His contributions were instrumental to support a wide range of graph neural networks in NextDoor. The experience gained in working on NextDoor has helped him to lead other projects related to efficient graph neural networks on GPUs.

Hanmei is working on optimizing neural network inference on GPUs and this is her first research project. In the beginning of the project Hanmei and I worked closely together, where I taught her concepts of GPU programming, brainstormed new techniques for the project with her, and created a detailed plan for her to implement these techniques. I was helping her grow as a GPU programmer by programming the infrastructure needed around new techniques and letting her implement key parts of the technique. This process helped her to focus on implementing particular techniques and not the infrastructure. She is now implementing new techniques on her own and also presenting her new ideas.

It brings me immense joy when students start making progress on their project independently. I’ve really enjoyed both teaching and mentoring students and look forward to growing as a mentor.

In addition to research mentoring, I have been significantly involved in carving out future plans of several graduated students. I have helped MS and PhD applicants in deciding potential universities and provided guidance in writing their statement of purpose. I followed the strategy of using examples to show how existing sentences can be written using good writing principles. This strategy helped them learn about good writing principles and they were able to coherently describe their technical experience in the statement. The applicants got MS and PhD admission offers from UMass Amherst, Georgia Tech, and Texas A&M. I am mentoring a PhD applicant through the SIGPLAN-M program in which we meet every two weeks. Currently, we are trying to pinpoint his interest within Programming Languages and Systems and then work on his applications.

Teaching Interests

I am interested in teaching introductory computer science courses and upper undergraduate level courses, such as Programming Languages, Software Engineering, Compilers, and Operating Systems. I can also teach undergraduate courses on other core computer science topics like Machine Learning and Computer Architecture.

I am interested in teaching several specialized courses related to my research area at the graduate level. For example, I can teach courses related to Functional Programming, Advanced Compilers, High Performance Computing, and Systems for Machine Learning. I will teach about the current research landscape of these courses and assign students in groups to work on project assignments. Furthermore, I would love to connect the concepts in these courses to my research and would share the insights from my research in the respective courses. Additionally, I will collaborate with other departments to organize seminar courses on interdisciplinary topics, such as, efficient computational chemistry and computational physics workloads.