Teaching Statement

Julian Shun

Teaching Experience. My first exposure to teaching was when I worked as a teaching assistant for an introductory computer science course (Structure and Interpretation of Computer Programs) as an undergraduate at UC Berkeley. I remember that during my first week as a teaching assistant, one of my students was struggling with recursion, a fundamental concept in computer science. I spent two hours on the Friday night of that week to help him understand this concept by working through problems with him on the whiteboard and helping him with the thought process. By the end of our meeting, he was able to solve problems by himself and explain his solutions to me. As a teaching assistant, I had an opportunity to hold weekly recitations with a group of students, reviewing materials from lecture or teaching new material. My first time as a teaching assistant was a very rewarding experience for me, as I realized that I could make a positive impact on others through teaching, and that I enjoy doing so in an academic environment.

As a Ph.D. student at Carnegie Mellon University, I had the opportunity to work as a teaching assistant for two courses—Introduction to Computer Systems and Parallel and Sequential Data Structures and Algorithms. I had experience designing a homework assignment on A^* search, and by using real-world examples such as finding directions on maps, students were able to more easily grasp and appreciate the topic. I was also responsible for all of the recitation material and learned to choose the materials to review based on students' interests and their understanding of the material in lecture. I helped design the course exams, and learned how to select problems that could be solved in a short amount of time while also testing the students' understanding of the material. Finally, I gave a guest lecture on sequential and parallel hash tables, which improved my ability to explain technical concepts to a large audience.

In addition to teaching courses, I also had the opportunity to mentor several undergraduate students at CMU in conducting research. Giving students research guidance that led to fruitful results was rewarding. Advising students has also allowed me to become better at identifying interesting problems and important resources to look at. My research collaboration with undergraduate students has led to several conference publications.

Parallel Thinking. With the prevalence of parallel machines, designing parallel solutions for tasks has become very important. I believe that many problems have parallel solutions which seem natural if one has been taught to "think in parallel". "Parallel thinking" should be taught to students early in the undergraduate computer science curriculum. This is crucial in enabling students to approach problems that they see during their studies with an awareness of the inherent parallelism available. To teach parallelism to students, it is important to identify the appropriate abstractions to separate the high-level concepts from the low-level details. This involves providing useful parallel primitives that students can use in designing algorithms, as well as teaching high-level ideas in parallel algorithm design such as prefix sums, contraction, divide-and-conquer, pointer jumping and randomized symmetry breaking. Concurrency issues, such as dealing with race conditions and deadlocks, scheduling, and architectural issues, while extremely important, should be taught later on, after students have grasped the higher-level ideas in parallel thinking. This can be done by providing the appropriate interfaces that hide these issues from the students. Finally, I believe that to teach parallelism, it is important to both introduce theory and also have students write implementations, so that students see real speedups in their code and also understand where the speedups come from.

Curriculum Development. The Parallel and Sequential Data Structures and Algorithms course at CMU for first and second year undergraduates is a relatively new course designed to expose students to parallel thinking by teaching both the theory of parallel algorithms and also their implementations. Major approaches to parallel algorithm design are taught, and abstractions are introduced so that students do not have to worry about low-level implementation details. I have been heavily involved in teaching and developing the material for this course, and would be eager to take part in the development and teaching of similar courses. In addition, with my experience in both the theory and practice of algorithms, I am in a good position to develop and teach courses on real-world algorithms and algorithmic engineering. I believe that these types of courses are important in bridging the gap between theory and practice of parallelism. I also have significant experience in parallel programming, architectures, and systems, and would be eager to teach courses in these areas. Finally, I would be happy to take part in developing and teaching any introductory-level undergraduate course as well as graduate or advanced undergraduate classes on algorithms, high performance computing, and data science.

Development of Teaching Tools. I am interested in developing tools that can help in the classroom and enhance the educational experience. By using the tools I have developed for simplifying parallel programming, students can avoid having to deal with the frustrations of nondeterminism in their programs, and be able to write and reason about simple and efficient parallel programs. The *Problem Based Benchmark Suite* that I have developed contains various parallel solutions to fundamental problems, which are simple enough to be taught in classes. The breadth of solutions can expose students to the advantages and disadvantages of various parallel programming languages, algorithms, architectures, and models. In fact, the benchmark suite is being used in the undergraduate course on parallel algorithms and data structures at CMU. In addition, due to its simplicity, implementing the graph processing framework that I have developed (*Ligra*) has been used as a project assignment in the undergraduate course on parallel programming and architecture at CMU, and has also been taught in courses at other universities. Finally, the simple parallel algorithms for maximal independent set, maximal matching, and graph connectivity that I have developed, which are efficient in both theory and practice, have been taught in graduate-level courses.

Teaching Summary. I intend to spread my knowledge to others through teaching in the classroom, curriculum development, developing teaching tools, and advising students. I have experience in all of these areas and am confident that I will be an effective teacher. I am eager to teach and excited take part in training the next generation of computer scientists and engineers.