Teaching Statement

Paulina Varshavskaya

The primary role of a teacher in college is to teach mental skills. My teaching philosophy is to lead students in their quest to acquire the skills of logical and especially algorithmic thinking and computational problem-solving. This naturally concerns computer science majors, whose ability to think computationally will become their livelihood. But it is also an important skill in the toolbox of a modern person and citizen whose relationship to computer science may be otherwise tangential. Algorithmic thinking helps you to decompose complex problems into simple, solvable ones, or to plan a sequence of steps that will lead to a desired goal. It can be a tool with which to optimize your workday or to understand a controversial public issue such as online privacy or electronic voting.

Today's undergraduates will be more computer-savvy and familiar with the use of technology, but they might not be any more well-versed in algorithmic thinking than the earlier generations. To understand the science in computer science is to understand the complexity and decomposition of a problem, the relationships between problems and between the framing of a problem and possible solutions, which parallels the interplay between data structures and algorithms. This essential notion governs my approach to the classroom, fostering the development of the computational thinking skills first, but importantly, also guiding students in the practical use of their skills, drawing examples from existing scholarship and art of CS.

This academic year, I sought out an opportunity to develop and hone my teaching skills by working as a part-time Lecturer at Tufts University. In the Fall semester of 2008 I taught Comp131 Artificial Intelligence to a mixed class of upper-level undergraduates as well as graduate students. This challenging course was demanding of my students; so it also required me to adapt quickly as I discovered the students' varied backgrounds (from undergraduate psychology to graduate electrical engineering) and learning styles. I developed a typical class structure where using prepared digital slides and demos was balanced by solving problems on the blackboard with student input. In this class, I taught the mental skills of abstraction (e.g., from a navigation problem or game description to a graph search algorithm), heuristics and the often counter-intuitive probabilistic reasoning. But the biggest lesson I drew from this experience was how much of an art student motivation can be. The teacher really needs to tie the material into her students' outside interests, which in this case ranged from professional gambling to theories of human moral judgment. So I added a substantial final project component to the class, and the students impressed me with the creativity, the skill, and the effort they put into their projects.

Currently, I am developing a new course which I will be teaching at Tufts in the coming Spring 2009 semester — Comp150-07 Intro to Intelligent Robotics. I am very excited to have this opportunity to really put my "learning by doing" philosophy to the test: in this course, students will spend at least half of the allotted time in practical labs, building and programming robots of their own design in a fun environment. Preparing to teach robotics has driven me to reach across departments to faculty in mechanical engineering and the Tufts University Center for Engineering Education and Outreach, whose amazing resources I am graciously allowed to use. As a graduate student at MIT, I was a teaching assistant for Prof. Rodney Brooks's alternative AI class (6.836 Embodied Intelligence) — an upper-level undergraduate and graduate course that combined a thorough introduction to robotics from the computational AI perspective with cutting-edge material that challenged it. I have also closely observed the development of a new introductory robotics course (6.141J Robotics: Science and Systems) at MIT, led in part by Prof. Daniela Rus. I participated in the development and content editing of the RoboticsCourseware.org repository of teaching materials, created

under the auspices of the IEEE Robotics and Automation Society. Through this work, I gained familiarity with the approaches, presentation styles and emphases in teaching introductory robotics at some of the best schools in the US, Singapore and Switzerland. I hope to incorporate what I learned from these experiences with great educators into my new robotics class.

In addition to formal classroom experience, I have a good understanding of the importance of mentorship in undergraduate education. During January and the spring term of 2007, I advised a sophomore EECS student under the MIT Undergraduate Research Opportunity Program. He contributed an asynchronous implementation to my thesis research project, while we concurrently worked on his research and written communication skills. This experience has taught me how to hook onto a student's particular talents to broaden their perspective and address those issues with which they may have less facility. This year, I am applying the lessons learned to advising a senior computer science student under the Tufts University Computer Undergradute Scholars Program. She is building a small-form bipedal robot from the commercially available Bioloid robotic kit and learning to design autonomous, biologically inspired controllers for it. I plan to continue involving undergraduates in my research, and am looking forward to building lasting advising relationships with graduate students on larger-scale projects.

Personal mentorship also has the educational value of working against unfortunate but ingrained stereotypes of computer science and the people who study and practice it. I strive both to dispel the myths by holding the algorithmic lens to exciting scientific, engineering and daily practical problems, and to provide a personal example of, I hope, a well-rounded intellectual person who has chosen computer science as her focus of study. In such an outreach effort, I participated in a "Women in Science and Math" conference for middle-school girls by giving four workshops on the relevance and fun of robotics as a career choice. The workshops included a hands-on robot building and programming experience, and it was extremely satisfying to see on the girls' faces the excitement and recognition of having just built something on their own and of understanding its workings. Several girls told me they would ask for Lego Mindstorms for Christmas.

To create a great classroom experience, the teacher naturally has to master the subject-matter, but also create a connection with the students, and engage them in a discovery of the material that is educational through being fun. It helps if the teacher can become a bit of a performance artist. In that respect, my five-year experience teaching Winter School for the MIT Outing Club during the months of January of 2001 through 2005, was a great source of both skill and confidence. It has also taught me how to design and deliver lectures, practical demonstrations, exercises, and field trips. I am now applying the skills and enthusiasm derived from this experience teaching self-propelled travel in winter wilderness also to my teaching of computer science.

I have been fortunate to have had all of these enjoyable and educational experiences, which have both shaped my teaching philosophy and style, and contributed to a strong desire to teach more. My main expertise lies in the broad field of AI, especially in robotics and machine learning, but I am equally eager to teach general introductory computer science, theory of computation, discrete mathematics and probability, algorithms and data structures and programming languages. I am particularly interested in teaching a first introductory class in computer science. With the current infrastructure and technology available especially for web-based applications, it would be absolutely exciting to see young students go from zero computer science to building, say, a Facebook application which they and their friends can then use and enjoy. A class like that would of necessity make students "learn by doing", while providing them a clear goal, introducing essential CS concepts, and giving the satisfaction of having created something tangible and useful at the end of it all.