Teaching Statement

Amanda M. Holland-Minkley
hollandm@cs.cornell.edu

I have a particular interest in introductory courses and courses for those not specializing in computer science or even necessarily technical fields. If students are going to be equipped to understand their increasingly computerized world – whether as computer experts, those who must work with, manage, or communicate with computer experts, or simply citizens necessarily concerned with the changes computers are causing in society – they must be offered entry-level courses which do not just teach programming syntax but introduce the techniques and results of computer science. These approaches encourage teaching not just facts and techniques, but why we use them, how to understand them, and what needs to be said about them to explain their use or application to others. As a problem-solving field, computer science lends itself well to interactive and constructive teaching techniques. Additionally, a vital part of any computer science curriculum, particularly in entry courses, is requiring students to write about the course concepts and their own problem solving processes. For students who are still developing their programming skills, giving them an alternative technique for working with course material will facilitate ultimate understanding, and the vital skill of technical writing will be better taught if students are expected to describe their technical work throughout their entire curriculum.

At Cornell, I was able to exercise these methodologies and interests. I helped create the first Academic Excellence Workshop for our introductory programming class. In this workshop supplement to the traditional class, students met in small groups once a week to cooperatively work on problems, facilitated by students who had been enrolled in the class the previous semester. The new students had the advantage of advice and guidance from those who had recently struggled with the course material, and the facilitators expanded their technical communication skills and tested the depth of their knowledge through working with the novice students. Instructing Cornell's introduction to engineering course on "Computation, Information, and Intelligence", I presented advanced artificial intelligence topics to freshmen by drawing on students' mathematical background and real-world experience. This gave students context for what they will go on to learn in other programming-focused courses, and motivated some students who may otherwise have dismissed computer science. In a Java programming class in which students were implementing a small compiler, I gave a lecture on the natural language task of parsing English sentences, not only introducing an interesting artificial intelligence problem, but also showing students that what they learn in one classroom will be relevant in others. I hoped to show the students the power of viewing computer science as a problem-solving field, in which a new problem may be solved by an insight from a seemingly unrelated problem.

Ensuring that computer science classrooms incorporate a diversity of learning techniques will have the side effect of encouraging a diversity of students. All students should have the opportunity to see problems both within and without of their intellectual comfort zone. These techniques must be employed from the beginning of the computer science curriculum for there to be hope for attracting and retaining students who traditionally turn away from the field. In my research on the need for robust communities to retaining women in engineering, I saw many indications that one of the powers of cooperative learning for encouraging diversity is the resulting strengthening of the classroom community. Reflecting the spectrum of backgrounds and interests of one's students is another vital component, which can be incorporated in part by the diversification of course topics as discussed above.

My enthusiasm for introducing computer science problems and techniques to novice students, particularly those who may not be attracted to the field through traditional routes, means that I would enjoy teaching introductory courses, whether as a computer science survey or a programming class, or intermediate classes in data structures and algorithms. Of course, I am equipped and enthusiastic about teaching any of the computer science core courses. My research background strongly equips me for teaching upper level courses in artificial intelligence, particularly in natural language processing, as well as logic, proof theory, and related discrete math topics. My goal in any courses I undertook would be to put into practice the methodologies described above, while ensuring a strong integration with the department's computer science curriculum as a whole.