Teaching Statement

Sorelle Friedler

I am passionate about teaching and view my classes as an exciting opportunity for me to continue to learn with and from the students while also exploring new teaching strategies and class topics. Two of my favorite teaching strategies are group learning and the discovery method. Both of these allow students at varying levels of ability to understand the material and be challenged. I also believe that these methods help make my classes and the material more engaging and interactive. Research confirms that group learning helps raise student achievement and interest in the sciences.[1] Group learning also helps to prepare students for work environments in which they will be expected to collaborate. I have found these strategies useful in my experiences as both a middle school teacher and a college instructor.

Before graduate school, in addition to teaching middle school math, I worked for the Math Forum (`http://www.mathforum.org`) and ran a summer camp called Co/Motion. At the Math Forum, along with working more broadly on the site, I answered questions written to the section of the site known as "Ask Dr. Math." These correspondences are designed to lead students to discover answers to their math questions for themselves, without directly giving those answers away. At Co/Motion, I did basic science and computer science activities with teenagers including building boats out of recycled materials, making basic web pages, and taking apart computers. These experiences as well as my own teachers inspired my interest in teaching and specifically in group learning and discovery method techniques.

While a graduate student I have sought out the opportunity to teach two summer classes: a junior-level programming languages class covering topics including functional programming, scripting languages, regular expressions, and context free grammars, and a senior level algorithms class. There were 37 students in the programming languages class; I oversaw one teaching assistant. The class is one of the standard prerequisites for upper-level classes at Maryland, and as such has a set curriculum on which later classes rely. I frequently had the students work in groups and present problems on the board. I knew that lambda calculus, one of the more abstract concepts the class covers, would be complicated for many of my students and decided to approach it in a hands-on way. To introduce the topic, I brought in paper alligators and eggs.[2] Students worked in groups to reduce small lambda calculus expressions in the form of puzzles, with alligators eating other alligator families (lambda calculus expressions) and eggs (variables) hatching into alligator families that had been eaten. Not only did we as a class have fun on that day, I'm certain that some students were only able to understand lambda calculus and retain the knowledge because of this exercise (a few final exams featured alligator drawings).

The algorithms class had 10 students enrolled. The curriculum for this class is flexible, but generally consists of graph algorithms, dynamic programming, divide and conquer strategies, network flow problems, and intractability. I emphasized approximation algorithms and other strategies for dealing with known NP-hard problems, since these are often the types of problems that need to be solved in real-world situations. Along with encouraging the students to work in groups to solve problems during class time, I assigned a group project to allow my students to explore algorithm design strategies and issues using the discovery method. The problem they tackled is an NP-hard scheduling problem, known as the registrar's problem. The purpose is to create a schedule that handles room size constraints, time conflicts, and teacher time conflicts while maximizing the number of requested classes for which students can be scheduled. I was very pleased with how this project turned out: my

[1] Leonard Springer, Mary Elizabeth Stanne, and Samuel S. Donovan. Effects of small-group learning on undergraduates in science, mathematics, engineering, and technology: A meta-analysis. *Review of Educational Research*, 69(1):21 – 51, 1999.

[2] Bret Victor. Alligator eggs. `http://worrydream.com/AlligatorEggs/`.

students wrote excellent papers about their algorithm design and experiments and it was a useful reference point for later topics in the class. The students discovered informally what an upper bound on the optimal solution meant, in this problem's context it would be all students getting to take all the classes they want, and this made a general discussion of solution bounds more accessible. As is often common for summer classes, a few of my students had failed the class the previous semester. The project gave these students a chance to succeed, and also provided a challenge for the better students.

I believe that teaching doesn't end at the end of class. One-on-one time with students can be very important, and I frequently spend time with students in office hours explaining material and discussing its broader context. When discussing approximation algorithms and the results from the experiments they ran for their project, some students in my algorithms class were concerned by how "bad" the worst-case theoretical analysis made their algorithms look, even though the experimental analysis showed that in most cases they were able to create a near-optimal schedule. Office hours gave us the time to discuss the analyses specific to their algorithm while making sure they understood the underlying evaluative concepts. Out-of-class time also allows for off-topic but important related conversations; one of the students from my summer algorithms class is considering going to graduate school, and office hours and continuing email interactions gave us time to discuss his options. These examples demonstrate that students find me approachable in and out of class; I am always available to help them over email or in person.

I have taught or been the teaching assistant for classes on programming languages, algorithms, low-level programming concepts, and computer organization. In addition, I have a strong background in introductory programming and Computational Geometry. Thus, I am prepared to teach a variety of classes. I would also be interested in developing and teaching new interdisciplinary classes and writing intensive classes; for example, Bioinformatics or a class on computers and education. I am excited to continue teaching theoretical and other computer science classes from beginning through advanced levels, learning and growing as a teacher along with my colleagues and students.