

# Timothy Wood

## Teaching Statement

<http://cs.umass.edu/~twood>

I love computer science because it not only includes many puzzling challenges, but also provides tools that foster creative and innovative solutions. My teaching philosophy captures this spirit by teaching students problem solving skills through hands-on projects that emphasize exploration and creativity. My experiences mentoring students, developing the curriculum, and teaching a new course in my department have prepared me to teach students in a range of settings and reaffirmed my desire to be a mentor for students.

**Teaching Experience:** As a graduate student I have actively sought out opportunities to teach and mentor other students. I have taken a course on teaching in scientific disciplines, designed and taught a new course for undergraduates in my department, been a teaching assistant for an undergraduate operating systems course, and acted as a mentor for both undergraduate researchers and new graduate students in my lab.

Much of my teaching experience comes from developing and teaching the Introduction to Linux and Unix course that my department began offering in Fall 2009. As the course had never been offered before, I had the responsibility of planning the initial curriculum and have now taught the course three semesters. I have refined the material and my presentation style each semester. This experience has taught me a lot about crafting interesting lectures, presenting them in an engaging way, and designing useful assignments. I learned after the first semester that students felt the class was too easy. I revised the course to make it more in depth, and was pleased to find that this increased student satisfaction according to their course evaluations. This has been a lot of work, but it has been very rewarding for me to introduce students to Linux for the first time, particularly since a number of beginner students have since told me that they now use Linux exclusively on their own computers. While I am not particularly concerned about what operating system students choose to use, I am glad that I have inspired them to keep expanding their knowledge in an area even after the course has finished.

Teaching both the Linux class and making guest lectures for an undergraduate operating systems course have given me experience teaching classes of 20-45 students, but I have also had extensive one-on-one mentoring experiences. The Linux course included hour-long lab sessions where students worked on their assignments while I was available to help them. This allowed me to tailor my explanations to each student's needs and help guide them through the problem solving process. Over the past few summers I have also acted as a mentor for four undergraduate researchers. I helped them with both planning their research directions and overcoming technical obstacles as they arose. I am a strong believer in encouraging undergraduate research, as I know from my own experience that it is one of the best ways for students to become truly engaged with the material they are learning in their courses.

**Teaching Philosophy:** The goal of my teaching is to emphasize how Computer Science is about problem solving and creativity. This keeps students interested in learning the material since they are made aware of the many interesting interdisciplinary ways that Computer Science can be used.

A common misconception with Computer Science is that it is only about learning how to program. Many students are turned off by early CS courses because they find too much emphasis on programming syntax and structure. To prevent this, my teaching focuses on learning problem solving skills, where programming is often just one of the tools used to reach a solution. While I think it is important for students to learn strong, low level programming skills, the many advances in high level programming languages and frameworks means that students can first be exposed to the fundamental concepts and algorithms in our field in friendlier environments that emphasize problem solving or creativity rather than precise syntax. Helping students learn how to analyze problems and develop creative solutions better prepares them for the diverse settings they will later go into and helps get them excited about our field.

Clearly explaining the relations between topics students are learning and their importance in the real world motivates students and helps them retain knowledge. Students need to see this at two levels, first to understand how the advanced algorithm they learned today is more efficient or useful than the simpler one from last class, and next to appreciate how the algorithm or technique is used in the software systems they interact with on a daily basis. Computers have become so pervasive in our society that this should be increasingly easy: social networks provide the perfect data sets for graph algorithms, the crash in Microsoft Office that made you lose

your English essay could have been the result of a deadlock bug, and did you realize that the iPhone in your pocket is faster than the first supercomputer? It is all too easy for students to lose sight of the big picture, and explicitly showing them these relations not only helps them build connections between material from different courses, but encourages them to pursue further study into the algorithms and techniques that relate to their fields of interest.

**Teaching Style:** My teaching style is based around a mixture of lectures, demonstrations, and hands on projects that reinforce course material in multiple ways so that students with diverse backgrounds and learning needs can all be engaged with the material.

Computers are complex systems, but in reality they are just made up of many levels of simple components that combine to create more complicated behavior. The assignments and projects in my classes will reflect this by starting with a simple project that is then incrementally added to, or combined with other assignments to create something more complex. This not only teaches students a fundamental concept in Computer Science, but also shows them how even the simple things which they are learning can be evolved and expanded on to create useful tools for real world problems. Our field has many excellent opportunities for interesting hands on projects, and these projects are what truly reinforce course material and provide students with the skills they need for their future jobs.

I applied this concept while teaching Linux by introducing students to some basic concepts and then having them use those simple tools to produce more complex behavior. Many students were surprised that after learning about a few fundamental commands, pipes, and how to use “man pages” they could then figure out how to do much more complicated things like analyzing data files or managing system processes. Subsequent assignments taught students both new tools and new ways to combine the building blocks they’d already learned, so that by the end of the course students were comfortable applying the individual concepts, as well as combining them together in advanced scripts.

Different students learn in different ways, and it is important to present material from multiple perspectives to ensure that all students are able to keep up. In the Linux course, I presented concepts with powerpoint slides, but made frequent demonstrations by switching to a command line. Using this approach allowed me to present each concept from two perspectives, and helped me recognize when students were confused. The additional lab sessions also allowed me to teach with an even more hands on approach. This was particularly important since the class had students ranging from freshman to seniors; my lecture material needed to be targeted at the lowest level students while still containing enough advanced nuggets to entice the more experienced students. This taught me the importance of having a flexible lesson plan so that I could adapt the material and presentation style on the fly based on students’ needs.

**Teaching Interests:** I am prepared to teach a range of undergraduate and graduate computer science courses, including introductory programming courses and fundamental systems courses such as networking, operating systems, and distributed systems. In addition, I am familiar with a range of technologies such as web, graphics, game, and mobile application development. I believe that targeted, cutting edge classes on these types of topics are a great way to illustrate the potential for creativity in computer science and entice students to learn practical skills that build on the fundamentals from their other courses; I would be excited to develop and teach these types of courses. My research area is also ripe with graduate seminar possibilities for anyone with an interest in high scalability, large amounts of data, distributed systems, or reliability.

### **Student Quotes from Teaching Evaluations**

“He was a great teacher. Clear, concise, and to the point. Knows what he’s talking about and is always prepared.”

“Whenever I needed help, Professor Wood always gave useful help to complete assignments. He knew the answers to all questions that I had.”

“Very knowledgeable about the subject matter and interested in sharing that knowledge.”

What did you like the most? “The amount of help and attention to students.”

“He did a really good job!!! Awesome class”