Teaching Statement

Timothy L. Hinrichs

Teaching Philosophy

In my opinion, there is no better way to learn Computer Science than to be taught its theoretical underpinnings and to gain hands-on experience building systems. Learning both forces students to think critically about the two—assessing the utility of theory in terms of system-building and evaluating a system's design and implementation theoretically. The proper balance between systems and theory varies from course to course, but generally speaking I believe that most systems classes today would benefit from more theory and most theory classes would benefit from more systems.

For example, when I was a TA for Logic and Automated Reasoning I noticed that because the focus was theoretical, students left the course thinking logic is only an intellectual curiosity instead of a tool used in practical settings such as databases, formal methods, programming languages, and security. At the same time, a course like Web Programming that focuses on system building, without also teaching the principles underlying client-server architectures and their benefits and limitations, will leave students without the intellectual tools to adapt when technology evolves.

Undergraduate Courses

Undergraduate courses are a joy to teach, both because students are eager to learn and because they bring fresh, innovative ideas. I am willing to teach a wide variety of Computer Science courses, though my main interest is in Computer Security and Logic and Automated Reasoning.

- **Introduction to Computer Security.** For each of several application areas (*e.g.*, embedded systems, networks, databases, web applications), we discuss security vulnerabilities and defenses at several levels of abstraction: the theoretical model of the application, the programming language implementation of the application, and the binary implementation of the application. Students will participate in programming projects at each of the three levels. The mix of theory and systems in this class occurs throughout both lectures and programming projects.
- **Introduction to Logic and Automated Reasoning.** This course helps students mature mathematically while learning technology that is widely applicable. Instead of teaching logic by considering one of the most general and theoretically interesting examples available (first-order logic) the class focuses on one that is more intuitive and boasts of industrially successful applications (Herbrand Logic). In this course, the theory of logic and automated reasoning is motivated by real-world problems, and through programming projects the students get hands-on experience solving those problems.

Graduate Courses

At the graduate level, my teaching interests mirror my research interests.

- **Web security**: A survey of security issues on the World Wide Web and how to address them, including attacks (SQL injection, Cross Site Scripting, Cross Site Request Forgery), techniques for finding vulnerabilities (at the level of binary code, programming languages, and mathematical models), techniques for defending vulnerabilities against attacks in existing systems (*e.g.*, program analysis), and approaches that prevent vulnerabilities (synthesis and frameworks).
- **Formal Aspects of Computer Security**: A course focusing on tools for understanding the security of systems formally (logic, automated reasoning, formal methods) and the application areas for which those tools are well-suited (access control, encryption, digital protocols).