Teaching Statement
Benjamin Atkin

As well as being an integral part of academic work, teaching is an activity from which I derive a great deal of satisfaction. In my time at Cornell I have been involved in teaching a number of courses, in several different capacities. I have been a senior teaching assistant with significant responsibilities for course organisation in three systems courses, operating systems, distributed systems, and security. As well as the usual duties of grading and holding office hours, I have also helped set up and administer course projects, and on several occasions I have acted as a stand-in lecturer. In addition, I have twice taught the summer version of Cornell’s regular operating systems course.

From these experiences, I have learnt many lessons in how to run a course and how to effectively communicate computer science concepts to students. By both attending and teaching courses, I have had an opportunity to personally observe which approaches have served the instructor and the students well and which ones have not.

In particular, organising and teaching operating systems and distributed systems courses has convinced me that a clear grasp of the practical issues underlying these subjects is vital for students to understand them. Giving students the opportunity to get their hands dirty, such as by finding out just how badly race conditions can cause things to go wrong, is an excellent safeguard against a critical topic like synchronisation seeming abstract and peripheral to real-world programming. Though computer science instruction rightly focuses on underlying concepts, practical experience in system-building helps students spot where their theoretical knowledge applies to the problems they will encounter. Constructing and measuring the behaviour of operating systems also gives students a valuable insight into why everyday software systems are implemented the way they are. For these reasons, I believe that a large practical project is an essential complement to a computer systems course. This also coincides with my own research focus on building actual systems.

There are several courses that I would very much like to teach and which I have had particular experience with:

**Operating Systems**: I have been heavily involved with the operating systems course at Cornell, having served as a teaching assistant for the course once, and co-taught the six-week summer version of the course on two occasions.

The Cornell operating systems course covers some aspects of networking and mobile computing, in addition to the traditional topics. In my opinion this correctly reflects the trend in computer systems organisation, away from programs on single hosts operating independently, and towards hosts cooperating to form a larger system. Though a thorough coverage of operating systems fundamentals is still imperative, a modern operating systems course should condense the coverage of some areas, such as disk and CPU scheduling, in favour of material on emerging topics in distributed systems. This change in the material covered should ideally be reflected in the course textbook as well. At Cornell we use Silberschatz, Galvin and Gagne's textbook, which I feel gives a solid coverage of traditional topics, but is limited with respect to networking topics and how large systems such as the Internet and the Domain Name System are structured. Before teaching an operating systems course, I intend to consider the question of an appropriate textbook carefully, or whether to supplement the expanded sections of the course with readings from additional sources.
Setting a major project for an operating systems course or practicum requires treading a fine line between omitting “realism” to simplify the system, and exposing too much irrelevant detail. During my work as a teaching assistant, I was one of the principal developers of Cornell’s PortOS instructional operating system. PortOS provides a semester-long project covering various topics in operating systems and networking, and has been used in eight iterations of the course so far, including the summer courses which I have taught. The system has proved a valuable tool for providing practical exposure to operating systems concepts, without requiring students to understand a great deal of unfamiliar source code. I plan to use PortOS or a similar system when I run my own operating systems course. The PortOS operating system is described online at http://www.cs.cornell.edu/People/egs/portos.

**Distributed Systems:** I would like to teach a graduate-level distributed systems course, which would broadly follow the similar course at Cornell. The course would begin by introducing fundamental topics in distributed systems such as the impossibility of distributed consensus, keeping track of the global state of the system, and clock synchronisation. The bulk of the course would then cover central features of distributed systems such as naming, different styles of inter-host communication, transactions, replication and structure of large-scale systems. These concepts can be illustrated with reference to existing real-life distributed systems, such as the Web, server farms and peer-to-peer file sharing.

As a practical component, the course would include a choice of several projects, building on top of existing middleware software, which will allow students to focus on building relatively sophisticated systems without worrying about low-level issues. An example project would be to examine experimentally how a commercial web services system copes with high load or server failure and implement some additional components to ameliorate any problems discovered, using techniques such as caching and replication.

**Advanced Systems:** The final course which I am especially interested in teaching would be a graduate-level seminar-style course covering influential papers in operating systems and distributed systems, the majority of which would be presented by students. Students taking such a course would work on small projects examining the behaviour of these systems or attacking open problems.

This style of course is considerably different from lecture courses in requiring an annual revision the selection of papers to reflect important journal and conference publications. From personal experience at Cornell, I am aware of how important such a seminar is in teaching systems students to present research work in front of an audience. It is the instructor’s responsibility to ensure that students learn good presentation technique and that a meaningful discussion develops around the material. Including papers in mobile computing and large-scale systems will also allow me to introduce systems students to work in my own research area.

Finally, I would also be able to teach courses in computer architecture, data structures or networking, though I would require more time to prepare material for those courses than for the ones listed above.