# Teaching Statement

Matthew Andrews

During my time at Bell Labs and MIT I have had ample opportunities to act as a teacher. This has been one of the most rewarding parts of my career since it is only by giving presentations and talking about a subject that one can encourage others to appreciate its value and appeal. I am excited about the chance to have a more formal and varied teaching role in an academic position.

## 1 Teaching experience – Bell Labs

My teaching experience at Bell Labs has taken two forms. On the formal side I mentored five summer interns during the years 1999-2001 and 2005-2006. On the less formal side my work in Bell Labs Research has often required me to act as a consultant to business units. In this role I have given numerous presentations to Bell Labs engineers to educate them about the theory of algorithms.

My five summer interns were Kamesh Munagala (Stanford, summers 1999-2000), Milan Vojnović (EPFL, Switzerland, summer 2001), Alex Slivkins (Cornell, summer 2005), Umut Akyol (Rutgers, summer 2006) and Kyomin Jung (MIT, summer 2006). My main goal with summer interns is to try and expose them to practical problems to which they can apply their algorithmic knowledge. I feel that it is important to give them an experience distinct from what they are used to as a graduate student. For each of my first three interns our summer work eventually led to a publication. For my 2006 interns we are currently in the process of submitting our work.

In 1999 Kamesh and I looked at a number of questions that were posed by Lucent's Internet Content Delivery and Distribution business unit. The key problems were cache placement, online caching of multimedia streams and assignment of users to the most appropriate cache. Kamesh developed a number of algorithms using techniques from facility location and online algorithms. In 2000 I asked Kamesh to work on scheduling in fast optical input-queued switches where the amount of data transmitted in each time step is much larger than a typical internet packet size. This problem was directly inspired by a Lucent optical packet switch.

With Milan in 2001 the optical switch project was continuing and so I asked him to look at another related problem. This was the question of how to design a schedule for a fixed set of traffic flows such that the jitter experienced by each flow is minimized. Milan devised a number of algorithms using probabilistic techniques. In July 2003 I served on Milan's Ph.D. thesis committee.

In 2005 Alex and I worked on analyzing how internet congestion control algorithms such as TCP interact with networks of queues. Our study was motivated by a set of results that show that in a simplified model, TCP-like algorithms converge to a "socially optimal" state. We were able to construct an example showing that if queuing dynamics are fully considered, the system can oscillate in a suboptimal state and never converge.

In 2006 I worked with Umut on simulating many of the protocols from the CBMANET project on the Opnet simulation platform. With Kyomin I worked on showing stability of the MaxWeight scheduling algorithm in the context of dynamic graphs with multicommodity traffic.

With regards to my other work at Bell Labs there were a number of projects in which I believe I played a "teaching" role. In one of these I worked with a group that was developing a tool for allocating telecom

repair technicians to tasks. They wished to move from a simple system that allocated tasks in a first-come-first-served manner to a more complicated system that allocated a whole day's worth of tasks at once. This presented a number of complex algorithmic and scheduling problems and for a number of months I gave weekly presentations to the group explaining the basics of complexity theory and describing various options on how the problem could be solved. One of the candidate algorithms was eventually incorporated into the Lucent tool.

Another project where I played a similar role was in the design of the scheduler for Lucent's EVDO wireless data system. I gave presentations on a bi-weekly basis that described basic ideas in queuing theory and also presented a number of possible scheduling schemes. One of these schemes was later adopted by the business unit. Other presentations I gave considered the interactions between the networking layer and the wireless layer. Many of these interactions had not been considered by the group before.

## 2 Teaching experience – Massachusetts Institute of Technology

I worked as a Teaching Assistant at MIT for three semesters. For one semester I was the TA for a graduate advanced algorithms class. I was responsible for writing and organizing lecture notes, helping students with problem sets and grading homework. For two semesters I was a TA for undergraduate multivariable calculus. My duties for this course consisted of giving two recitations per week together with grading exams and homework. This was an intense course in which the professor was unable to cover all the material in lectures. As a result, I was responsible for teaching students many of the difficult aspects of multivariable calculus such as integration using spherical coordinates. A number of students from other recitation sections asked to be transferred into my section.

### Other teaching experience

In the summer of 2000 I was invited to give a tutorial at a DIMACS summer school on foundations of wireless networks and applications. In the summer of 2005 I gave a similar tutorial at an IMA summer workshop on wireless communications. Over the past few years I have given numerous seminars for high school students at the Royal Latin School in Buckingham, UK where my mother is a math teacher. As a professor, I would be happy to be involved in any university events for high school students.

In addition to the specific teaching experience mentioned in this statement, I have regularly given presentations at conferences, workshops, universities and within Bell Labs throughout my research career.

## 3 Teaching Goals

My goal as a teacher will be to give students a broad education in computer science and networking while stressing both mathematical rigor as well as explaining how the theoretical advances in computer science in the last decades can be applied to practical problems.

As an undergraduate studying mathematics at Oxford University I was required to take extremely detailed and rigorous courses on topics such as Fourier Analysis and Lebesgue Integration Theory. However, no effort was made to explain why Fourier Analysis is useful and no attempt was made to train students in techniques for actually evaluating complex integrals. At the other extreme, I believe that papers sometimes appear in the networking literature that "prove" incorrect results due to insufficient mathematical rigor. My goal will be to try and find the middle ground where students know how to make precise statements about algorithms and also know how to apply algorithmic techniques to real-world problems.

I will encourage students to implement all algorithms that they study and develop. My experience at Bell Labs has taught me that deriving the correct theoretical algorithm for a problem is sometimes only a

small fraction of the struggle. There are times when the theoretical solution can be written down succinctly but the implementation details take a great deal of additional effort.

Fortunately, I believe it is easier than ever to build prototypes for testing ideas in networking both within Linux and also via programmable hardware devices such as network processors. In addition I would encourage students to test algorithms through simulation. I feel that there is a real art to writing a useful simulation since it is easy to put in so many details that the true nature of the device one is trying to simulate is obscured. My aim would be to help students learn the best way to design an effective simulation.

For the teaching of network protocols I would encourage students to look at real traces of internet traffic to see how protocols actually perform. For example I believe it is extremely informative to study TCP traces to determine exactly what the protocol is doing at each point in time. Often the behavior is not as simple as it is presented in analytical papers.

One area that I would like to emphasize is collaboration in group projects. In most of the classes I took at university, the project work was to be performed by each member of the class individually. However, my experience in industry is that almost all development is carried out in groups with different people responsible for requirements, implementation and testing. This involves a set of skills, such as the clear writing of specifications, that are not often taught at universities. I would like to make sure that my students acquire these skills.

Although I have emphasized above that I believe that students should "get their hands dirty" when studying algorithms and networking, I do believe it is essential that they are provided with a strong background in theoretical techniques. When one has implemented an algorithm it is usually impossible to test out all possibilities that could arise. Therefore it is often comforting to have a theorem describing how badly an algorithm can "break". Moreover, theoretical analyses regularly give an excellent sense of which algorithms are likely to perform best in practice. Lastly, I believe that in the past few decades the theory of computer science has developed in many beautiful ways. I am looking forward to the opportunity to share these developments with students.

## 4 Courses

I would like to teach courses in:

- **Introductory Networking:** Internet protocols, switch design, queuing theory, network design and optimization. Emphasis on both mathematical rigor and on applicability to real-world networking problems. Course would include simulation and (where possible) building prototypes.
- **Wireless networking:** Physical layer design, modulation and coding schemes, power control, wireless network layouts, wireless data systems (e.g. third generation and WiFi), resource allocation, wireless protocol design, wireless applications.
- **Mathematics for computer science:** Introductory combinatorics, graph theory.
- **Introduction to algorithms:** Asymptotic analysis of running times, graph algorithms, data structures.
- **Advanced algorithms:** Online algorithms, approximation algorithms, randomized algorithms, algorithmic techniques in mathematical programming.
- **Complexity theory:** Basics of complexity classes, randomized complexity classes, reductions, hardness of approximation.

In addition to the above subjects I believe that I could teach most freshman courses in computer science and mathematics.