DOI:10.1145/1467247.1467260

Jeffrey D. Ullman

Viewpoint
Advising Students for Success

Some advice for those doing the advising (and what the advisors can learn from the advisees).

No two doctoral students are the same, and the things an advisor needs to do for each vary accordingly. I can look back over my career and see several approaches that work, and one approach that is popular but doesn't really serve the student well. To begin, the goal of the advisor is to teach someone how to become an independent thinker, inventor, and problem-solver. You must take someone barely out of their teenage years and convince them that they can do something that none of the most experienced people in the field have been able to do. And they must do that not only once, but throughout their professional lifetime. Frankly, when I went off to study for my doctorate, I had no idea what writing a thesis entailed; had I known, I never would have gone to graduate school.

What Not to Do

I was a student, and later faculty member, in an electrical engineering department, where the widely held opinion was that the way you wrote a thesis was to read many papers. Look at the last section, where there were always some "open problems." Pick one, and work on it, until you are able to make a little progress. Then write a paper of your own about your progress, and don't forget to include an "open problems" section, where you put in everything you were unable to do.

Unfortunately this approach, still widely practiced today, encourages

PHOTOGRAPH BY HECTOR GARCIA-MOLINA

Students and colleagues attend Jeff Ullman's retirement celebration in 2003.

mediocrity. It gives the illusion that research is about making small increments to someone else's work. But worse, it almost guarantees that after a while, the work is driven by what *can* be solved, rather than what *needs* to be solved. People write papers, and the papers get accepted because they are reviewed by the people who wrote the papers being improved incrementally, but the influence beyond the world of paper-writing is minimal.

The Early Model: Theoretical Theses

In the first years of computer science as an academic discipline, many theses were "theoretical," in the sense that the contribution was mostly pencil-and-paper: theorems, algorithms and the like, rather than software. While much of this work was vulnerable to the problem just described—paper building on paper—it is quite possible for a theoretical thesis to offer a real contribution. For example, even before I joined the Princeton faculty, I had a summer intern at Bell Labs, Ravi Sethi. At that time Ken Thompson and Dennis Ritchie were involved in the Multics project, an operating system for the GE635 computer. This beast was the first to have more than one register in which arithmetic could be done, and the word passed to Ravi and me that they needed techniques to compile code in a way that made best use of several registers. Ravi's thesis was an algorithm for compiling arithmetic expressions using any given number of registers, in the fewest possible steps. This algorithm actually was put into the C compiler for the PDP-11, a few years later.

While Ravi's thesis was "theoretical"—neither of us wrote any code—the work illustrates how I believe any thesis should develop. The work was not based on what some paper left open, but rather on an expressed need: a way to compile expressions using several registers. The big advantage we had was that we were part of an environment that was pushing the frontiers. Had we not been at Bell Labs, it is doubtful we would have realized the problem was worth addressing. We surely could not have read about it in a paper. Even Andrei Ershov, who had previously published the node-numbering scheme we used, only saw it as a way to compile for a one-register machine, and did not suggest in his paper that someone else should look at machines with multiple registers.

When I went off to study for my doctorate, I had no idea what writing a thesis entailed; had I known, I never would have gone to graduate school.

The Ideal Ph.D. Student

The best scenario is that the student tells me what their thesis should be, and carries it out independently. Moreover, their thesis topic is selected because they perceive a need on the part of some "customer." Sergey Brin came closest to this ideal, since he and Larry Page, with no help from me, saw both the need for a better search engine and the key ways that goal could be reached, while students at Stanford. The one missing element: neither of them got their degree; but more about that later.

A close approximation was George Lueker, who came to visit me one day to ask if I had any ideas for a thesis topic. George was not then my student, being enrolled in the Applied Math Program at Princeton. I happened to be reading about chordal graphs that morning, and suggested an algorithm to detect chordality. A year later, he came back and showed me a thesis he had written on *pq*-trees, a data structure that even today has several important applications beyond chordality testing. Several other students have dragged me kicking and screaming to learn a new area, even if I then got involved in selection of their thesis topic. Matt Hecht had me learn about data-flow analysis; Allen Van Gelder did the same with logic programming.

Why does it matter who suggests the thesis topic? We're trying to get young scientists to the point where they can make independent judgments about what is worth working on. There are several decisions to be made: what is worth doing, what is feasible to do, and how do you do it? While an advisor can help with all these things, it is wonderful to meet a student to whom this comes naturally. Another point that I tried not to forget as I grew older was that young people can often see things that those of us who have become set in our ways cannot. Trusting the technical judgment of the young is not a bad strategy.

What Students Need

To make students successful, we need to be ready to provide many services.

Finding customers. As mentioned at the beginning of this Viewpoint, there needs to be an exposure to problems that are at the frontier, and that are needed by a "customer." Sometimes, they can find a customer in industry, as Ravi Sethi did at Bell Labs. Summer internships can be a great opportunity. However, advisors should encourage students to intern at a strong industrial research group, one where the goals are more than minor tweaks to what exists.

Whether the thesis is theoretical or an implemented solution, students need to be guided to understand who will consume their contribution if they are successful. And the answer cannot be "people will read the paper I will write, and they will use the open problems in it to help form their own theses." Especially when dealing with theoretical theses, the chain of consumption may be long, with idea feeding idea, until the payload is delivered. Yet if we let students ignore the question of whether such a chain and payload plausibly exist, we are doing them a disservice.

Walking before you run. Exposure to problems is not enough. Some, although surely not all, Ph.D. students need to convince themselves that they can do something original. Here are a few ideas that have worked:

- One way to give a beginning student practice with the mechanics of research is to think through a small problem yourself, and then propose

to a beginning doctoral student that they work on the problem. Since you have a path in mind, it is easy to raise questions that will lead them where they should go, until they have worked through to the solution on their own. A single experience like this is usually enough to get them operating independently.

▸ Try getting the student to make an early transition from reading papers to exploring their own ideas. Certainly, you need to read enough to get the concepts of your field, but after a point, the more you read, the closer your mode of thinking becomes to that of the field at large, and "out of the box" thinking becomes harder. If they produce promising ideas, then of course a more careful literature search must be performed. I've seen enough examples to believe that it is a rare case (although sadly not impossible) where the student's ideas are completely subsumed under what has already been done.

▸ My colleague Hector Garcia-Molina often encourages students to start not by looking for the theoretically optimal solution, but for a simple, easily implementable solution that gets you 90% of the way there. The optimality might be studied later and can form an important part of the thesis.

▸ My colleague John Mitchell reminds us that even after getting past the hurdle of believing one can invent, the thesis can be intimidating because of its large scale. He gets students to focus on writing a single paper (preferably for a conference where they will meet people, not for a journal). After they have written a few papers, building a thesis from them will seem much less intimidating.

Expressing ideas. An advisor must make sure that their students can write clearly. There is little point training students to generate great ideas if they cannot communicate them. It is essential that the advisor reads very carefully and checks every sentence of a student's first attempts at writing. A common situation, and one that must be caught early, is writing that goes into a lot of detail on the easy parts, and gets fuzzy or overly terse when it comes to presenting the hard parts: the proof of a key theorem or the details of a complex algorithm, for example. So an advisor must judge what is hard and be sure that the writing does justice to those parts.[a]

We're trying to get young scientists to the point where they can make independent judgments about what is worth working on.

Fear factor. Yet another common job of the advisor is to teach the student to fail cheerfully and without embarrassment. Not every student has a built-in fear of failure, but many assume it is wrong to attempt something they doubt is possible. Often, the student's model of a "problem" comes from homework, where the solution is certainly known. They are ashamed to report "I didn't get anything done this week," even if it was not for lack of effort. You don't want students to spend a lot of time trying to write a program that takes another program as input and removes all bugs (as a fellow student of mine was once advised by *his* advisor to try), but it is OK to encourage a student to do something ambitious and risky, like finding more bugs than anybody else. In these cases, a vital job of the advisor is getting students to risk their time and effort, and to deal with the case where nothing good results.

Group therapy. A popular technique for encouraging and engaging students is the free lunch. Not only do Ph.D. students benefit, but it can be used to attract undergraduates into the research community. For the past 15 years, I have been privileged to be part of the "Database Group" (now "Infolab") at Stanford, consisting of faculty Gio Wiederhold, Hector Garcia-Molina, Jennifer Widom, our students, staff, and visitors. At regular Friday lunches, students take turns presenting informal talks on their work, and good-natured argument from the floor is the norm. Students get over the fear of defending their ideas in public, as well as benefiting from insights of others. Students also may practice for an upcoming conference talk and receive very detailed suggestions from fellow students. Another important function of the lunch discussion is bonding, facilitated by a social committee to run group events, and by regular trip reports, which serve as a vehicle for learning about one another's lives.

A Newer Model: Project-Oriented Theses

It took many years to reach this point, but it is now fairly routine to have substantial software projects carried out in an academic setting. While there will always be the occasional thesis that is purely "pencil-and-paper," a much more productive approach is to introduce beginning Ph.D. students to a project. Often they enjoy "learning by doing," contributing to the software development, while learning the new notions that are being investigated by the project. Senior students often get the opportunity to help, and even to supervise, junior students.

The best example I've seen of how to use this mode effectively comes from my colleague Jennifer Widom. In a series of innovative projects (semistructured data, stream databases, and now uncertain databases), she has perfected a routine, consisting of:

1. Define a general goal for the research, and get a team of doctoral students working together.
2. Spend a substantial period of time,

a (Aside: While it sounds pedantic at first, you get a huge increase in clarity by chasing the "nonreferential this" from students' writing. Many students (and others) use "this" to refer to a whole concept rather than a noun. For example: "If you turn the sproggle left, it will jam, and the glorp will not be able to move. *This* is why we foo the bar." Now the writer of this prose fully understands about sproggles and glorps, so they know whether we foo the bar because glorps do not move, or because the sproggle jammed. It is important for students to put themselves in the place of their readers, who may be a little shaky on how sproggles and glorps work, and need a more carefully written paragraph. Today, it is not that hard to find a "this" that is nonreferential. Almost all begin sentences, so grepping for '`This`' will find them.)

perhaps 6–12 months, in which the theory and models underlying the problem area are developed. (Jennifer says that this step—making the students part of the planning and modeling—is what distinguishes her approach.)

3. Then, start an implementation project. Get the students working on pieces. The goal of each project is a robust, distributable prototype, not something that can be carried intact to commercialization.

4. Allow students to identify their own aspect of the broader problem area on whose difficulties they will focus. Students develop their own ideas, which form the core of their thesis, and are able to validate the ideas by installing them in the larger system.

It is sad that many research-funding agencies, such as DARPA, have become so "mission-oriented" recently. While it may be possible to support a Ph.D. student doing part of a project implementation, Step 4 is left out; there is no room on the project for a student to explore original work outside the boundaries of the project. For example, I have heard from several independent sources that while the European Union has been supporting "research" generously, the support is sufficiently constrained by concrete deliverables that there is no way to support Step 4 on the projects. In countries where Ph.D. support comes from a state source, this arrangement presents no serious impediment. However, in countries where Ph.D. students are dependent on project support, it becomes hard to train first-rate researchers.

Students and Startups

One of the trickiest decisions an advisor has to make is how to deal with the student who wants to found a startup while they are working on their doctorate. Few people agree with me on this point, but I believe that, unless the startup idea is insane, they should go out and do the startup. My theory is that, while getting a doctorate and entering the research arena is a high calling, it is not the highest possible calling. A startup can have more impact on our lives than a thesis. Moreover, if they miss the opportunity to do a successful startup, then they have lost a great deal. If the startup flops, as many do, they have lost only a few years, and can resume work on a doctorate if they wish.

Sergey Brin never asked me whether or not he should quit the Ph.D. program and found Google, but I would have told him to do so had he asked. Another student, Anand Rajaraman, did ask my advice on this matter when he was about half a year from finishing. I told him to leave and be a founder of Junglee. The venture was quite successful. A few years later he returned to Stanford, started an entirely new thesis topic that abstracted some of what he had learned at Junglee, and is now Dr. Rajaraman.

You don't have to be in Silicon Valley to think about startups. Great ideas can develop anywhere, and a responsible advisor will, when appropriate, present to their students the option that their work might form the basis of a commercial venture. I recall an email message from a student at another school asking the question: "can a piece of work be both a thesis and useful?" When I replied in the affirmative, I was then asked to explain this point to their advisor. That advisor was serving the student poorly, although their attitude seems fairly common. Even in the course of reviewing this Viewpoint, I encountered the view that a piece of technical work is more to be admired if it cannot be commercialized.

Afterword

Of the various things I've done in my career, I am most proud of my 53 Ph.D. students and their academic descendants (see infolab.stanford.edu/~ullman/pub/jdutree.txt; also see the photo appearing on the first page of this Viewpoint). Many have done things I could never do myself, and done so remarkably well. Each has brought unique talents to their work, and it has, for me, been an education just to watch them. I'd like to imagine that I contributed to their success, although I'm pretty sure that the only thing I really did was stay out of their way so they could realize their own potential. C

Jeffrey D. Ullman (ullman@cs.stanford.edu) is the Stanford W. Ascherman Professor of Computer Science (Emeritus) at Stanford University.

For this Viewpoint, I have repurposed some of the ideas of Hector Garcia-Molina, John Mitchell, Jennifer Widom, and Gio Wiederhold, for which I thank them. Additional thanks go to Mark Hill for suggesting developing this Viewpoint about Ph.D. advising.

Calendar of Events

March 15–19
The 2009 ACM Symposium on Applied Computing,
Honolulu, HI,
Sponsored: SIGAPP,
Contact: Sung Y. Shin,
Phone: 605-688-6235,
Email: sung.shin@sdstate.edu

March 16–18
10th International Symposium on Quality Electronic Design,
San Jose, CA,
Contact: Tanay Karnik,
Phone: 503-712-4179,
Email: tanay.karnik@intel.com

March 19–22
Fourth International Conference on Intelligent Computing and Information Systems,
Cairo, Egypt,
Contact: Mohamed Essam Khalifa,
Phone: 20127937560,
Email: esskhlalifa@yahoo.com

March 22–25
7th Annual IEEE/ACM International Symposium on Code Generation and Optimization,
Seattle, WA,
Sponsored: SIGMICRO, SIGPLAN,
Contact: David R. Tarditi, Jr.,
Email: dtarditi@microsoft.com

March 22–27
2009 Spring Simulation Conference,
San Diego, CA,
Contact: Gabriel A. Wainer,
Email: gwainer@cse.carleton.ca

March 23–26
International Conference on Web Information Systems and Technologies,
Lisbon, Portugal,
Contact: Joaquim B. Filipe,
Phone: 351-91-983-3996,
Email: jfilipe@insticc.org

March 31–April 1
Second International Workshop on Social Computing, Behavioral Modeling and Prediction,
Phoenix, AZ,
Contact: Huan Liu,
Phone: 480-727-7349,
Email: hliu@asu.edu