The next five years are critical for computer science education. The policy decisions we make during that time will determine whether computer science continues to grow and strengthen in the coming years or whether the field endures another capacity collapse that disrupts the flow of students into a discipline that is essential to the modern economy.

After the dot-com bubble burst a little over a decade ago, student interest in computer science declined significantly, leading to reduced enrollments across the United States. In many institutions, the decline was so precipitous that computer science programs were downsized or, in a few instances, eliminated. In response, professional societies in computing and the National Science Foundation launched programs to rekindle student interest. Those efforts—supported as they were by various independent changes in society—have been enormously successful. At the large research institutions, computer science enrollments have been growing for several years, to the point that they have generally eclipsed any previous highs. The turnaround in lower-tier institutions and in high schools has been slower, but there are indications—most notably the increase in students taking the AP/CS exam last year—that these institutions are likely to follow suit.

I am convinced that the dominant problem for the next several years will be that of building sufficient capacity in our educational institutions to satisfy the increasing demand. Given the extraordinary marketability of computing skills in today’s society and the inertia that comes from having passed a “tipping point” in popular culture, interest in computer science will continue to grow no matter what we do as a field. Moreover, given the need to expand the diversity of the computer science workforce and the almost insatiable demand for the right kinds of talent, we must continue to build interest in computer science, particularly as young people begin to make choices about their field of study. If, however, the universities do not have the capacity to grow along with the demand, the result will be a constriction of the pipeline of the sort that occurred during the 1980s. That result—particularly in light of the fact that we have a historical example from which we could learn—would be a disaster both for our field and for the national economy.

Translating this concern into research directions for the NSF will take some careful thought and planning. In this paper, I argue for additional research around the following questions:

1. What challenges make it difficult for institutions to build the necessary capacity?
2. How can we maximize the production of students with the exceptional levels of talent that the industry demands?
3. What strategies—including those outside traditional academic programs and curricula—will help to expand the pipeline of students in this critical area?
1. Identifying challenges that limit the creation of capacity

One of the most disturbing aspects of the current state of computer science education is the failure of many decision makers, both inside and outside of the field, to recognize the dangers of expanding enrollments butting up against a fixed capacity. As I argue in a recent ACM Inroads editorial entitled “Meeting the challenges of rising enrollments” available at http://cs.stanford.edu/~eroberts/papers/RisingEnrollments.pdf, the collapse of student numbers at the end of the 1980s was not the result of declining interest but instead reflected a failure of capacity. Student numbers grew inexorably, faculty workloads increased, faculty began to leave for greener pastures in industry, graduate students chose not to follow their mentors into academia, and, in the end, computer science programs ended up turning students away.

The most compelling evidence for this perspective on that history, interestingly enough, comes directly from research conducted by the NSF. In 1983, Kent Curtis issued a report (see http://cs.stanford.edu/~eroberts/Curtis-ComputerManpower/) on the problems facing academic computer science. In the following passage, Curtis argues that the term crisis was indeed justified by the extent of the problems:

> We must conclude that the educational institutions of the country cannot obtain the labor they need and have poor prospects of finding it in the near future. They face a real crisis. The migration of student interest is working against them, not in their favor, and the job mobility which allows so many people to enter computer professions in business, industry and government is not effective for educational institutions because of their highly specialized job requirements.

Curtis based his evidence for the existence of a crisis on the following observations:

1. Students are not entering graduate school but are being lured by attractive salaries and professional opportunities at the bachelor’s level.
2. Graduate students are leaving graduate school without completing their Ph.D.s.
3. Faculty are leaving academia for industry.

In the 1980s, academic institutions responded to the problem of growing student demand in the face of faculty shortages by attempting to meet that demand with their existing faculty. As Curtis reports, this strategy had negative long-term effects:

> [An NSF study on the faculty hiring crisis reveals that] 80% of the universities are responding by increasing teaching loads, 50% by decreasing course offerings and concentrating their available faculty on larger but fewer courses, and 66% are using more graduate-student teaching assistants or part-time faculty. . . . In brief, they are using a combination of rational management measures to adjust as well as they can to the severe manpower constraints under which they must operate. These measures make the universities’ environments less attractive for employment and are exactly counterproductive to their need to maintain and expand their labor supply. They are also counterproductive to producing more new faculty since the image graduate students get of academic careers is one of harassment, frustration, and too few rewards.

The field needs a contemporary study that provides authoritative evidence that these problems remain true today. One of the fundamental problems that computer science departments face is that few people in university administrations believe that this sort of capacity collapse can happen. Computer science is unique within the university structure
in having more faculty openings than applicants, at least during boom times. The idea that one might be in a position where a department couldn’t hire additional faculty is not one that most administrators understand. Moreover, the fact that enrollments have been cyclical, makes university administrations leery of increasing staff if those students will later vanish. That concern, however, becomes a self-fulfilling prophecy. When staff levels fail to rise in parallel with the demand, the result is to “make the universities’ environments less attractive for employment [which is] exactly counterproductive to their need to maintain and expand their labor supply.”

NSF needs to fund research that can reestablish those conclusions with current data. This type of research is different from what usually constitutes educational research, but is nonetheless vital to the health of the academic discipline. In the absence of that research, I fear we will be condemned to repeat the patterns of history to the detriment of everyone.

Fortunately, there is time to have an impact in this area. At present, the most significant rates of growth have occurred within institutions that are more able to develop the necessary capacity. If Stanford needs to grow its computer science program to meet the unprecedented student demand, we can do so. The real problem will come when this same wave crests within state universities, liberal arts colleges, community colleges, and high schools. We need to be ready with compelling data, adequate resources, and new ideas to avoid being in the tsunami.

2. Research into educating high-capacity students

Despite the fact that industry executives are unanimous in their concern about a shortage of talent, contrary narratives continue to exist. A recent article in IEEE Spectrum (http://spectrum.ieee.org/at-work/education/the-stem-crisis-is-a-myth/), takes the position that, despite the seemingly universally industry consensus, there are more available workers in STEM fields than available openings. Fifteen years ago, Norman Matloff made a similar point in the New York Times (January 26, 1998):

Readers of recent reports about a shortage of computer programmers would be baffled if they also knew that Microsoft hires only 2 percent of its applicants for software positions. . . . You don’t have to be a “techie” to see that such a low ratio, typical for the industry, contradicts the claims of a software labor shortage. If companies were that desperate, they simply could not be so picky.

Today, Matloff could make an even better-sounding case, given that companies like Google hire significantly less than one percent of their applicant pool.

The problem with such arguments is that they fail to take into account a fundamental characteristic of software developers. Productivity among equivalently trained software engineers varies by several orders of magnitude. In a Wall Street Journal article from November 23, 2005, Google’s Vice President of Engineering, Alan Eustace, asserts that one top-notch engineer is worth “300 times or more than the average.” In a field in which productivity differentials of this sort are commonplace, all companies must be picky. Hiring run-of-the-mill engineers does relatively little good for the bottom line. Software companies rise and fall on their ability to hire the six- and seven-sigma programmers at the high end of the distribution.
Unfortunately, very little serious research has been invested into the phenomenon of productivity variance among programmers. Most authors who try to make this argument go all the way back to the Sackman study of 1968, which was a completely different time. More recent studies exist, but they are few and far between.

More critically for the NSF, there are almost no studies on how these variances affect computer science education. All of us have anecdotal information about the enormous range of capabilities we see in our classes. As Mark Guzdial reported in his blog post that serves as a springboard for this conference, “failure rates worldwide of 30–50% in the first class have been reported for decades.” Those failure rates, however, occur together with a much smaller cohort of students who amaze you at every turn. We know very little about what differentiates those extremely successful students from those who never seem to get it.

My impression is that, as a community, we have a reluctance to address this issue. In part, that reticence comes from a commitment to democratic empowerment. Most of us want all our students to succeed and feel uncomfortable with the idea of focusing disproportionate levels of attention of an elite subgroup of students who will likely succeed with or without our help. I don’t, however, propose to use the research in this way. If we understood more fully why some students are so much more successful than others, we would be better equipped to turn average programmers into the highly productive software developers that industry is so desperate to attract. In the 12,000+ students I’ve taught at Stanford, I’ve seen that transformation occur many times, but am entirely unclear as to how that transformation occurs or what one can do to foster it other than by ensuring that students have ample opportunities to practice their skills and that they are always encouraged to go beyond the minimum requirements.

For the most part, I don’t believe that I have the right research skills to analyze this phenomenon. We need research projects that combine domain expertise in computer science with a deep understanding of psychology and pedagogy. I would like to see NSF fund interdisciplinary projects that will help to understand these issues.

3. Developing innovative strategies to expand the computer science pipeline

Over the last decade, many initiatives have sought to build both the size and diversity of the computer science pipeline. Those efforts must continue. At the same time, it seems to me that we need to promote more experiments in this area, both to test out a greater range of strategies and to avoid having a set of similar initiatives fail in the same ways.

Many creative initiatives can be introduced within the confines of existing academic institutions. Stanford, for example, is about to inaugurate a new program that allows students to declare a joint major integrating computer science with another department. We believe that this program will restore some much-needed balance among the different fields of study that will counteract the increasing pressure for students to choose only majors, like computer science, that are seen as eminently marketable. More importantly, this proposal gives students an opportunity to pursue—and indeed to create—new interdisciplinary programs that integrate disparate fields of study. NSF should fund the research necessary to evaluate programs of this type.
At the same time, I believe that it will also be necessary to experiment with novel strategies for teaching computer science that fall outside the boundaries of traditional institutions. If current trends continue, it seems inevitable that many institutions will be unable to hire sufficient numbers of faculty to offer computer science majors to every student wishing to pursue one. Ensuring that students at those institutions are not frozen out of the field altogether will require some creativity. Some liberal arts colleges, for example, have coordinate arrangements with larger research institutions that allow students to build a foundation at the liberal arts college and then transfer to the research institution to complete their degree. Another possible model is that of an intensive single-term course of study at a research institute, as Stanford currently offers at the Hopkins Marine Station. Thus, instead of having students attend a program like the University of Virginia’s Semester at Sea, one might create a highly focused institute at which one might have a Semester at C++. Some have suggested abandoning the university model entirely, depending instead on teaching practical computer science through intensive mentor-protégé interactions of the sort traditionally used to teach music or creative writing. Dick Gabriel, who is both a computer scientist and a prizewinning poet, has suggested the idea of a Master of Fine Arts degree in software organized along these lines.

The situation is even more problematic at the precollege level. If universities once again start to have difficulties recruiting faculty with the necessary skills to teach computer science effectively, finding those teachers must be harder still for public high schools and middle schools. The essence of the problem is economic. When teachers are given the skills necessary to teach computer science well, they could also use those skills to enter a much more lucrative job market. Some teachers, of course, will stay in schools out of a real commitment to education. Others, however, will leave, making it necessary to train even more teachers than one initially thinks the nation needs.

There are, however, other strategies. In the intensive campaign to promote math and science education that occurred in response to the Cold War fears about Sputnik, young Americans had many opportunities to study math and science outside of schools. The National Science Foundation’s Summer Science Training Programs of the 1960s and 1970s and the resurrected Young Scholars program of the early 1990s were important examples of this approach. Privately run summer programs in computer science are widespread and incredibly popular but are accessible only to students whose families have the wherewithal to pay. Free, government-sponsored programs to which admissions were based on aptitude and potential seem to offer a more inclusive model.

In this area, the NSF would be an obvious candidate to offer funds for developing and running such programs. At the same time, there is a clear research agenda associated with all of these ideas. As with any idea that pushes against existing boundaries, there is a real chance that some of the experiments will fail, along with a hope that some will succeed. Measuring that success and understanding various strategies for improving the effectiveness of successful programs will require educational research that will need funding, encouragement, and support. The NSF seems like the obvious agency to take the lead in this area.