Expanding the Pipeline of Students in Computer Science

Eric Roberts and Mehran Sahami
Computer Forum Annual Meeting
Stanford University
March 18, 2008
The Crisis in the Computer Science Pipeline

That there is currently a crisis in computing education is not in doubt.
— McGettrick et al., SIGCSE 2007

- As everyone has now been aware for some time, computing enrollments in the United States and most of Europe have plummeted since 2001.

- This drop is of significant economic concern because those same countries are training far fewer people than they need to fill the available positions. In the United States, there are now many more jobs in the IT sector than there were at the height of the dot-com boom, with all projections pointing toward continued growth.

- This decline has been even more rapid among women and minority students, reducing diversity as the pool shrinks.
Outline

1. The severity of the pipeline problem
2. A critical analysis of the underlying reasons for the decline and the most common explanations.
3. What professional societies are doing.
4. What Stanford is doing.
5. What industry, academia, and the professional societies can do together.
The Pipeline Problem in Computer Science

Although there are indications that the decline has bottomed out, the number of computer science majors at research universities has fallen by almost 50 percent since its peak in 2000.

The Problem Starts Early

The UCLA HERI study shows that students have already made their decisions before they reach university.

By 1999, everyone and their dog wanted to major in CS

Source: Higher Education Research Institute at UCLA, 2005

Graphic thanks to Andy Maag
CS is Losing Ground

- The Computer Science exam is the only Advanced Placement exam that has shown declining student numbers in recent years.
CS Is Tiny Compared with Other Sciences

The graph shows the growth in the number of students taking Calculus AB, Biology, and Computer Science (CS) from 1997 to 2006. The number of students taking Calculus AB has increased steadily, with a significant rise in the last few years. The number of students taking Biology has also seen growth, although at a slower pace than Calculus AB. For Computer Science (CS A+AB), the number of students remains relatively stable, with a slight increase over the years.
Degree Production vs. Job Openings

The Data From Stanford

CS major declarations

Graph showing the number of CS major declarations from 1993 to 2007.
A Slightly More Well-Known Graph

Source: Yahoo! finance
The Obvious Correlation

- Normalize both graphs by 1998 values
  - Adjust for a one year lag time in declarations

Correlation = 0.61
By 2003, ... sensational news stories appeared about a supposedly horrific loss of these [computer programming] jobs [due to offshoring].

-- The Washington Times, June 6, 2004

Correlation up to 2003 = 0.88
One Possible Solution

Just give us one more bubble...

Actual bumper sticker seen in Palo Alto circa 2003
The Conventional Wisdom

• Just as pretty much everyone now recognizes the existence of an enrollment crisis, most everyone has a favorite totalizing explanation. The leading theories include:
  – Negative images of those who work and study in the field
  – Fears about job security after the dot-com bust and offshoring
  – A “broken” curriculum that does not appeal to today’s students

• While there is truth behind each of these theories, none of them can serve as a comprehensive explanation of the student behavior we see today. Even when taken together, these theories overlook several important factors that are at least as important as underlying causes for enrollment decline.

• The factors that lead to declining enrollments are complex and highly interconnected. Solving the problems depends on developing a better understanding of those factors and how they interact.
The Image of Computing Remains a Problem

In 1998, sixth-graders in selected California schools were asked to draw their image of a computer professional. The drawings are for the most part aligned with traditional stereotypes, as follows:
Myths of a Jobs Crisis Persist

There is no shortage of evidence that people believe the myths about the lack of jobs and the danger of outsourcing.

All this talk about “Blue Skies” ahead just can’t hide the stark fact that Americans who don’t wish to migrate to India and/or some other off-shore haven are going to have a difficult career.

Why would any smart American undergrad go into IT when companies like IBM and HP are talking of stepping up their off-shoring efforts in the coming years? They want cheap labor, no matter the real cost.

I have been very successful in IT, but I certainly wouldn’t recommend it today to anyone except people who are geeks. . . . I think the latest figures from the U.S. Department of Labor are not correct.
A Thought Experiment about Offshoring

• Suppose that you are Microsoft and that you can hire a software developer from Stanford whose loaded costs will be $200,000 per year. Over in Bangalore, however, you can hire a software developer for $75,000 per year. Both are equally talented and will create $1,000,000 annually in value. What do you do?

• Although the developer in Bangalore has a higher return, the optimal strategy is to hire them both. After all, why throw away $800,000 a year?

• Any elementary economics textbook will explain that one hires as long as the marginal value of the new employee is greater than the marginal cost. The essential point is that companies seek to maximize return, and not simply to minimize cost.
The Truth on Offshoring

• More IT jobs today in US than during boom.
• Employment data suggest that new jobs are being created more quickly than jobs are being moved overseas. Thus, offshoring of software seems so far to have increased the number of jobs, not only in India and China, but in the United States as well.

• Confusion at the Bureau of Labor Statistics
  – Projected Job Growth from 2006 to 2016
    • “Computer programmer”: below average
    • “Computer scientists” & “software engineers”: above average

• Need to create awareness of “CS in the large”
  – CS is increasingly fundamental to work in other fields
The Curriculum Cannot Be the Problem

- The computing curriculum as traditionally implemented has deficiencies and can always be improved.
- As an explanation for declining enrollments, however, the “curriculum is broken” theory has serious shortcomings:
  1. It cannot explain why enrollments have varied so much over time.
  2. It fails to account for the fact that institutions saw a similar loss of enrollment even when their curricula were different. Most of the proposed curriculum improvements were in place somewhere in 2000-01, but declines occurred everywhere. The resurgence of enrollment in the last year also seems independent of curriculum.
  3. Students decide to avoid computing long before they have any idea what the university curriculum is.
  4. Students who take our courses tend to like them but still shy away from the computer science major.
Students Like Our Courses But Go Elsewhere

Computer Science

Management Science & Engineering

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

student flow
How Students Choose Their Majors

For the most part, students do not base their decisions on what they want to study, but instead on what they want to do.

∴

If students are not majoring in computer science, the problem is likely to be that they don’t want to work in the field.
The Real Image Problem

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.
The Reality Is Also a Problem

Students with whom I’ve talked are concerned about:

• Long hours with little chance for a balanced life
• A less pleasant social milieu than other occupations
• A sense that success in programming is possible only for those who are much brighter than they see themselves to be
• Work that is often repetitive and unchallenging, particularly when it involves maintaining legacy technology
• No chance for a lasting impact because of rapid obsolescence
• Fears that employment with an individual company is dicey even though opportunities are good in the industry as a whole
• Frustration at being managed by nontechnical people who make more money but are not as bright (Dilbert’s boss)
• A perception that programmers are definitely on the labor side of the labor/capital divide
Dilbert’s Boss Has More Appeal than Dilbert
Rediscovering the Passion, Beauty, Joy, and Awe

Grady Booch, SIGCSE 2007
The Vilification of Programming

• Those who argue most strongly for the broken curriculum theory often blame programming for the woes of the discipline, decrying the widely held view among students that

\[
\text{computer science} = \text{programming}
\]

This view is indeed too narrow.

• Unfortunately, however, some have started to argue for the far less defensible proposition that

\[
\text{programming} \not\subset \text{computer science}
\]

Adopting this position throws the baby out with the bathwater.
Dangerous Trends

We have met the enemy and he is us.

— Walt Kelly

As an illustration of this trend, consider the following post that appeared on SIGCSE-MEMBERS on August 14, 2006:

I have an idea for a panel that I’d like to organize for SIGCSE’07. I’m asking for volunteers (or nominations of others) to serve on the panel. The panel I’d like to organize would have a title something like:

“Alternative Models for a Programming-lite Computer Science Curriculum”

The theme of the panel would be to share ideas and thoughts on how we might reduce (or eliminate) the emphasis on programming within a computer science curriculum. The basic idea is to cause discussion centered on the knowledge and skills students of tomorrow will need in the global economic workspace and the implications for the CS curriculum. As more and more aspects of software development of “offshored”, what kind of curriculum would allow a student to be successful in the IT field?
Industry Is Not Amused

• *Every* technical person in the industry with whom I’ve spoken is horrified by the prospect of reducing the emphasis on programming in the undergraduate curriculum.

• At the ACM Education Council meeting in September, a panel of technical people from companies like Microsoft, Google, Amazon, and Boeing were united in their concern about the scarcity of competent software developers. I have summarized their position as “the computing curriculum is not nearly as broken as it seems likely to become.”

• Employers in developed countries with high-tech sectors are desperate for more people with programming talent. When Bill Gates visited Stanford in February, he reported that he was very happy with the students coming from Stanford; he only wished “Microsoft could hire three to four times as many.”
Programming Remains Central

• As with many of the popular theories for declining enrollments, the call to “reduce or eliminate” programming from computing curricula arises from some undeniable assumptions:
  – There are more jobs in IT that don’t require programming.
  – Programming is not particularly popular with students today.
  – Offshoring of programming jobs has increased.

• Unfortunately, this analysis ignores the following equally valid propositions:
  – There are more jobs in IT that do require programming.
  – Programming has historically been what attracts students the most.
  – Offshoring exists largely because of a shortfall of skilled employees.
Revising the Undergraduate CS Curriculum

• Field has evolved more significantly than curriculum in last 20 years, and will continue to do so

• Students should be explicitly made aware of the options in Computer Science
  – Diversity of areas within computer science
  – Significant role of computing in inter-disciplinary work
  – Not just trying to “fix” the curriculum

• Provide context for computing
  – Programming is the *means*, not the *ends*
  – Still, should not discount the importance of rigorous software engineering skills
    • *Don’t “water down” the curriculum to just attract more students!*
Increasing the “Footprint” of CS

Editor’s Note: Two-dimensional projection clearly does not capture the relative importance or organizational nuances of the field. Some topics may be closer to you than they appear on this slide.
“Footprint” of CS Students See Today

Editor’s Note: Two-dimensional projection clearly does not capture the relative importance or organizational nuances of the field. Some topics may be closer to you than they appear on this slide.
Tracks Allow More Depth...

Total amount of material covered must remain the same
...in a More Diverse Set of Areas
Total Potential “Footprint” is Larger

Core material everyone sees is streamlined to accommodate
Revised Curricular Structure: *Core*

**Theory Core: 3 Courses**
1. Mathematical Foundations of Computing
2. Probability Theory for Computer Scientists
3. Data Structures and Algorithms

**Systems Core: 3 Courses**
1. Programming Methodology and Abstractions
2. Computer Organization and Systems
3. Principles of Computer Systems
Revised Curricular Structure: *Tracks*

~4 Courses

- Students must complete requirements for any one track
- Developing depth in a specialization
- Provide course/theme options within each track
- Provide multi-disciplinary options
- Modularize curriculum
Why Tracks?

• Explicitly shows available options
  – Broad picture from awareness raising matches curriculum
  – Allows students to focus on areas in which they have the greatest interest, thus increasing appeal of program

• Helps eliminate image of CS as “just programming”
  – Shows diversity of themes in computer science
  – Provides more context for what is possible with CS degree
  – Still provides significant programming education

• Provides organizational infrastructure
  – Easier to evolve major as the field evolves
  – E.g., add/drop/modify tracks (or programs in them)
Initial Set of Track Areas

- Artificial Intelligence
- Theory
- Systems
- Human-Computer Interaction
- Graphics
- Information
  - Management and applications of (un)structured data
- Biocomputation
- Unspecialized
  - Essentially, our current program
- *Individually Designed*
Revised Curricular Structure: *Electives*

~2-4 Courses
- Restricted electives
- Allow pursuing breadth and/or additional depth
- Track-specific elective options allow for interdisciplinary work
Track and Elective Structure

- All tracks have at least 4 (possibly more) required courses
  - Required track courses are generally advanced CS courses
- Elective courses (2-4 courses, depending on track)
  - Set of general CS electives that all students may choose from
  - Additionally, each track specifies track-specific electives that may count as elective courses only by students in that track
  - Track-specific electives allow for additional depth or related interdisciplinary course options
    - Biocomputation track: Genomics, Dynamic Models in Biology
    - Graphics track: Studio Art, Psychology of Vision, Digital Photography
    - HCI track: Needs Finding, Psychology of Perception, Cognition
Revised Curricular Structure: *Capstone*

1 Course

- “Senior project” capstone course
- Developing capstone courses to parallel tracks
- Both application development and research options
Structure Aligns with Broader Context

- **IEEE-ACM Computing Curricula 2001 Report**
  - Supports tracks model
  - Revision committee adopted modular structure to support adaptability

- **ICER: Integrative Computing Education & Research**
  - Change the popular image of computing
  - Encourage curricular experimentation and innovation
  - Make sure introductory students recognize that the field offers many opportunities
  - Strengthen interdisciplinary connections
Broadening the Initiative

• Need for curricular reinvigoration not unique to Stanford
  – Many universities suffering even greater drops in enrollment
  – Many other schools considering possible next steps
• Stanford’s continued leadership in education
  – Have resources to make changes and experiment
  – Actively engage Computer Science community with results
  – Other initiatives: books, material repository, etc.
• Engage both other academic and industry partners on a continuing basis
• The CS Pipeline affects us all…
• …and we can all have an impact on it!
Positive Initiatives

• The National Science Foundation sponsored four regional conferences on Integrated Computing and Research (ICER) and has funded several proposals under a new Computing Pathways (C-PATH) initiative.

• Several ACM Education Board projects are proving helpful:
  – A brochure for high-school students
  – The CC2001 series of curriculum reports
  – The Computer Science Teachers Association
  – A community effort to develop Java tools (the ACM Java Task Force)

• In addition to the Stanford revision, there are many interesting ideas in the community that are showing promise:
  – Mark Guzdial’s “media computation” course at Georgia Tech
  – Stuart Reges’s “back to basics” strategy at the University of Washington
  – Jeannette Wing’s “computational thinking” concepts
  – The Alice Project developed at Carnegie-Mellon
  – Various robot-based introductions
  – Pair-programming strategies at a variety of schools
Some Encouraging Signs

And for those programming jobs, the reason it’s possible to sit in front of a computer for extended periods of time is because in CS we can learn new things, achieve goals, and be creative. Every day! It’s this last point that really drives me, personally. If you ask any passionate person how they can "___ all day long", it’s because that’s their outlet for being creative.

From Dan Garcia’s “Faces of CS” web site.
What We Need To Do

• Recognize that the problems extend well beyond the university.
• Press government and industry to improve computing education at the K-12 level.
• Take creative steps to bolster both the image and the reality of work in the profession.
• Emphasize the fact that programming remains essential to much of the work in the field.
• Encourage research into new software paradigms that can bring back the “passion, beauty, joy, and awe” that can make programming fun again.
The End