The Thrill Is Gone?

(To appear in Communications of the ACM, August 2005)

Sanjeev Arora Bernard Chazelle

June 4, 2005

The irony would be hard to miss: While computing technology is thriving and extending its reach into our everyday lives, computer science is facing a serious crisis in the United States (falling undergraduate enrollments [2, 8], low research funding [7]). Why? Part of the answer is, we believe, our collective failure as educators, researchers, and practitioners of the field to articulate a cogent, compelling narrative about the science of computing (as opposed to the technology). Computer science is one of the most exciting scientific endeavors in recent history. Too bad so few have been exposed to the thrill. We argue below that, in order for computer science to thrive, its "story" needs to be told to the outside world (especially high schoolers, parents, teachers, policymakers, and the popular media) in a manner that keeps the science and the exciting ideas at center stage.

There are many ongoing efforts to develop new high school and college curricula that will lead to an IT-literate workforce. While supporting those efforts, we wish to address here another enrollment issue critical to our field: attracting bright high schoolers and undergraduates who will form the next generation of IT researchers and educators. Professor Peter Lee [6], associate dean for undergraduate education at Carnegie-Mellon, has documented serious problems in this regard, and found them to be part of a long-term trend with little connection to either outsourcing or the dotcom bust. For instance, he notes that the number of CS finalists at national science competitions, such as Siemens Westinghouse and Intel, has continuously dropped in the last decade and now approaches zero. This is in sharp contrast to the early 1980s, when an entire generation of bright students flocked to the field.

Part of the problem is the lack of consensus in the public at large on what computer science actually is. The Advanced Placement test is mostly about Java, which hurts the field by reducing it to programming [3]. High school students know that the wild, exotic beasts of physics (black holes, antimatter, Big Bang) all roam the land of a deep science. But who among them are even aware that the Internet and Google also arose from an underlying science? Their list of computing "Greats" probably begins with Bill Gates and ends with Steve Jobs.

Some observers have suggested that the identity of computer science is blurred by its heterogeneous nature, which encompasses elements of science, engineering, psychology, the arts, etc. But physics and biology seem unhurt by *their* heterogeneity.

We feel that computer science has a compelling story to tell, which goes far beyond spreadsheets, java applets, and the joy of mouse clicking (or even Artificial Intelligence and robots). Universality, the duality between program and data, abstraction, recursion, tractability, virtualization, and fault tolerance are among its basic principles. No one would dispute that the very idea of computing is one of the greatest scientific and technological discoveries of the 20th century. Not only has it had huge societal and commercial impact but its conceptual significance is increasingly being felt in other sciences. Computer science is a new way of thinking.

Consider our everyday experience that appreciating creativity is far easier than being creative. It's one thing to enjoy the Moonlight Sonata or be taken in by the view of the Bilbao museum; it's quite another to be Beethoven or Frank Gehry. Likewise, to verify the correctness of a math homework is far easier than actually doing the homework. When one tries to formalize this phenomenon (especially the last one) in terms of computational effort, one arrives at nothing but the famous P versus NP question, an important open scientific question in computing today. One of the seven "Millenium open problems" chosen by the Clay Math Institute for million-dollar prizes, P vs NP is also the only one of the seven whose significance can be understood by an average high schooler.

The next example of conceptual impact is cryptography. This field is, of course, thousands of years old. But it truly flowered only after the formulation of the P vs NP question and the related theory of NP-completeness. Public-key cryptography and the famous RSA cryptosystem were directly inspired by these developments. Subsequent innovations in cryptography even challenged our basic understanding of everyday concepts such as randomness and knowledge. For example, there is the "millionaire's paradox" (also called *Yao's millionaire's problem*): It's possible for two people to engage in a dialogue that will result in their mutual knowledge of who is the richer, all the while revealing no information whatsoever about their respective wealths. Or for one of them to convince the other that he has a password to some account without revealing anything about this password (zero knowledge proofs). Or for both of them to flip coins over the phone, say to play a game, while ensuring that neither can lie about his coin flips without being caught. (No, this does not require a camera phone!) All of these facts are nearly as magical as the famous "paradoxes" of special relativity, yet they concern familiar settings and everyday human interaction.

Computer science is also causing other sciences to think differently. Often this statement is illustrated by pointing to the need for computational approaches in many scientific disciplines, but one needs to emphasize that more computing cycles is only a small part of the story.

Consider quantum computing. In the 1990s, following the lead of physicists (Richard Feynman, David Deutsch, and others), computer scientists used quantum mechanics to define a new computational model and demonstrated its interesting properties. Soon thereafter, a computer scientist, Peter Shor, stunned the world with the announcement of a fast algorithm for factoring integers on this model. The stunning had much to do with the fact that e-commerce security relies crucially on our inability to factor large numbers. But it also raised important questions about quantum reality relevant to physics. So much so that Nobel laureate David Gross [4] includes a better understanding of quantum computation among his top 25 physics challenges.

Whole-genome sequencing is another example of the integration of computer science thinking into the natural sciences. The Human Genome Project, completed in 2003, was greatly aided by computer scientists who realized that the underlying sequencing problem –including the wet lab portion–was primarily algorithmic: how to assemble the overall picture from local (and noisy) snapshots. Their contribution was essential and conceptual rather than just "coding up." It saved a few years from the entire effort and considerably advanced the field of genomics. Today CS techniques such as Hidden Markov Models are widely used in genetics.

Along similar lines, a new computational understanding of economics might soon arise from ongoing research into economic interactions among computationally limited agents (a feature of e-commerce). A better understanding of the brain might grow out of a marriage of experimental neuroscience and algorithmic thinking.

Finally, any telling of the computer science story must include its singular contribution to society: "bringing the world to our fingertips." Often this is taken to refer to Arpanet, TCP/IP etc., causing lawmakers to think of this entire phenomenon as "building infrastructure." It is therefore important to point out that even if the enabling networking technologies had been in place thirty years ago, companies such as Google, Yahoo, and Amazon would still have been impossible. Scientific advances in a host of CS areas such as operating systems, compilers, databases, machine learning, etc., were necessary too, together with the invention of a body of techniques for storing, manipulating, searching, and understanding large amounts of data. The story is worth telling. For instance Google's Page Rank algorithm exploits the semantic content of Web links. It views links to a web page as "votes" about the importance of that page. Such ideas are easy to conceptualize for high school students and might provide a more accurate impression of the wonders of computer science.

Motivate and Thrill

Lenore Blum and Carol Frieze [1] of CMU have been documenting an ongoing successful effort to attract bright students, including women, to the CS major. The success of CMU programs such as Andrew's Leap suggests that it is, indeed, possible to motivate and thrill bright college students by focusing on the big ideas, and on the notion that computer science is a new way of thinking.

Unfortunately, efforts to replicate these efforts elsewhere run into trouble because there are no introductory texts (with a few exceptions such as [5]) that provide an exciting overview of the "computer science story"—let alone something analogous to the classic "Feynman Lectures in Physics." (The excellent *Journal of Economic Perspectives* should be another inspiration for all disciplines.) Such texts play a vital role in disseminating ideas to frontline educators. There is also a need to write popular accounts. Few computer scientists bother to do this, whereas world-class physicists (Feynman, Weinberg, Hawking, Greene) have mastered the art of story telling. The NSF Physics Division even lists "working toward early inspiration of the young" as one of its three main goals [9]. Within computer science there is evidence that D.R. Hofstadter's Pulitzer prize-winning 1979 book "*Gödel, Escher, Bach – An eternal golden braid*" attracted an entire generation of students to the field, especially to artificial intelligence.

One wonders if the failure of computer scientists to articulate the intellectual excitement of their field is not one of the causes of their current funding crisis in the US. Too often policymakers, and hence funding agencies, treat computer science as a provider of services and infrastructure rather than an exciting discipline worth studying on its own. Our promises of future technological innovations and scientific advances will be more credible to them if they actually understand that past and current breakthroughs arose from an underlying science rather than a one-time investment in "infrastructure."

We think it is high time that the computer science community should reveal to the public our best kept secret: our work is exciting science—and indispensable to the nation.

Sanjeev Arora and **Bernard Chazelle** are Professors of Computer Science at Princeton University.

References

[1] *In a More Balanced Computer Science Environment, Similarity is the Difference and Computer Science is the Winner.* L. Blum and C. Frieze. *CRA News*, 17(3), May 2005.

[2] *College freshmen less interested in tech,* CNET (April 2005).

[3] *The field of programmers myth.* Peter Denning, *CACM*, July 2004.

[4] *The Future of Physics*, by David Gross (Jan 2005).
http://agenda.cern.ch/

[5] *Computers Ltd.: What They Really Can't Do,* by David Harel, Oxford University Press, 2000.

[6] *Shrinking Pipeline,* by Peter Lee (August 2004).
https://www.wiki.cs.cmu.edu/public/pmwiki.php?

[7] *President's letter: The state of funding for new initiatives in computer science and engineering.* D. A. Patterson, CACM, April 2005.

[8] *Interest in CS as a Major Drops Among Incoming Freshmen,* by Jay Vegso, Computing Research News 17 (May 2005).

[9] *FY 2006 Budget Request to Congress, Mathematical and Physical Sciences, NSF.*