Viewpoint

Sanjeev Arora and Bernard Chazelle

Is the Thrill Gone?

Computer science is nothing less than a new way of thinking; explaining it to the wider world is critical to the future of the field.

LISA HANEY

The irony is hard to miss: While computing technology is thriving and extending its reach further into our everyday lives, computer science is facing a crisis in the U.S., including falling undergraduate enrollment [2, 9] and reduced research funding [8]. Part of the reason may be our collective failure as educators, researchers, and practitioners to articulate a cogent, compelling narrative about the science of computing (as opposed to just the technology).

Computer science is one of the most exciting scientific endeavors in recent history. Too bad so few have had the opportunity to share the thrill. For computer science to thrive, its story needs to be told to the outside world (especially high school students and their parents and teachers, as well as policymakers and the popular media) in a way that keeps the science and the ideas center stage.

Many ongoing efforts aim to develop new high school and college curricula that would help lead to an IT-literate work force. While supporting them, we wish to raise another enrollment issue critical to the field: attracting bright high school students and undergraduates who represent the next generation of IT researchers and educators. Peter Lee, associate dean for undergraduate education at Carnegie Mellon University, has documented serious problems in this regard [6], finding they are part of a long-term trend with little connection to either outsourcing or the dot-com bust. He notes, for instance, that the number of computer science finalists at U.S. science competitions, including Siemens Westinghouse and Intel, has dropped over the past 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 awareness in the public at large as to what computer science is. For example, the advanced placement test is mostly about Java, which actually hurts the field by reducing it to programming [3]. High school students know that the wild, exotic beasts of physics (such as black holes, antimatter, the Big Bang) all roam the land of a deep science. But who among them is aware that the Internet and Google also arose from an underlying science? Their list of computing "greats" likely 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, encompassing elements of engineering, psychology, the arts, and more. However, physics and biology seem unhurt by their own heterogeneity.

We think that computer science has a compelling story to tell, going 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 was one of the greatest scientific and technological discoveries of the 20th century. Not only has it produced huge societal and commercial effects, its conceptual significance is increasingly felt in other sciences.

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 Guggenheim Museum in Bilbao, Spain; it's quite another to be Ludwig Van Beethoven or Frank Gehry. Likewise, verifying the correctness of math homework is far easier than actually doing it. Trying to formalize this phenomenon (especially the last one) in terms of computational effort, one arrives at the famous P vs. NP question, an important open scientific question in computing. One of the seven "millennium open problems" chosen by the Clay Mathematics Institute (www.claymath.org/millennium) for million-dollar prizes, P vs. NP is the only one whose significance can be understood by an average high school student.

Cryptography represents another example of the conceptual impact of computer science. Though the field is thousands of years old, cryptography truly flowered only after the formulation of the P vs. NP question and related theory of NP completeness. Public-key cryptography and the RSA cryptosystem were directly inspired by these developments. Subsequent innovation in cryptography even challenged our basic understanding of such everyday concepts 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 results in their mutual knowledge of who is the richer, all the while revealing no information whatsoever about their respective wealth. Or one of them can convince the other that he (or she) has a password to some account without revealing anything about the password (zero knowledge proofs). Or both of them can flip coins over the phone to, say, play a game, while ensuring that neither lies about his coin flips without being caught. (No, this does not require a camera phone.) All these ideas are nearly as magical as the famous "paradoxes" of special relativity yet concern familiar settings and everyday human interaction.

Computer science has also caused other sciences to think differently, a statement often illustrated by pointing to the need for computational approaches in many scientific disciplines. However, more computing cycles represent only a small part of the story.

Consider quantum computing. In the 1990s, following the lead of such physicists as Richard Feynman and David Deutsch, computer scientists used quantum mechanics to define a new computational model and demonstrate its interesting properties. Soon thereafter, computer scientist Peter Shor stunned the world with the announcement of a fast algorithm for factoring integers on this model. This was shocking because security in e-commerce relies on our presumed 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.

Too often, policymakers, and hence funding agencies, treat computer science as a provider of services and infrastructure rather than as an exciting discipline worth studying on its own.

Whole-genome sequencing is yet 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, involving assembly of the overall picture from local (and noisy) snapshots. Their contribution was essential and conceptual rather than just "coding up." Saving years from the overall effort, it considerably advanced the field of genomics. Today, computer science techniques (such as Hidden Markov Models) are widely used in biology.

Along similar lines, a new computational understanding of economics might soon arise from ongoing research into the economic interactions among computationally limited agents (a feature of e-commerce). A better understanding of the brain might emerge from 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. This concept is often viewed as referring to the Arpanet and TCP/IP, causing lawmakers and policymakers to view the entire phenomenon as "building infrastructure." It is therefore important to point out that even if the enabling networking technologies had been in place 30 years ago, companies like Google, Yahoo, and Amazon would still have been impossible. Scientific advances in a host of computer science areas, including operating systems, compilers, databases, and machine learning, were necessary, too, together with the invention of a body of techniques for storing, manipulating, searching, and understanding large amounts of data. For instance, Google's PageRank algorithm exploits the semantic content of Web links, viewing links to a Web page as "votes" about the importance of that page. Such ideas are easy for high school students to conceptualize and might provide a more accurate impression of computer science.

Motivate and Thrill

Lenore Blum and Carol Frieze of Carnegie Mellon University are documenting [1] an ongoing effort to attract bright students, including women, to the computer science major. That success (such as in the Andrew's Leap project) suggests it is indeed possible to motivate and thrill 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 projects elsewhere run into trouble due to the lack of introductory texts (with 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 *Journal of Economic Perspectives* should be another inspiration for all disciplines.) Such texts play a vital role in disseminating ideas to frontline educators. Popular accounts should also be written. Few computer scientists bother to do this, whereas world-class physicists (Feynman, Steven Weinberg, Stephen Hawking, and Brian Greene) have mastered the art of storytelling. The National Science Foundation Physics Division even lists "working toward early inspiration of the young" as one of its three main goals [7]. 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 U.S. Too often, policymakers, and hence funding agencies, treat computer science as a provider of services and infrastructure rather than as an exciting discipline worth studying on its own. Promises of future innovation and related scientific advances will be more credible to them if they actually understand that past and current breakthroughs arose from an underlying science rather than from a one-time investment in "infrastructure."

It is high time the computer science community began to reveal to the public its best kept secret: its work is exciting science—and indispensable to society. C

For information on ACM's Computer Science Teachers Association, visit csta.acm.org/index.html.

References

1. Blum, L. and Frieze, C. In a more balanced computer science environment, similarity is the difference, and computer science is the winner. *CRA News 17,* 3 (May 2005).
2. CNET. College freshmen less interested in tech. (Apr. 22, 2005); news.com.com/College+freshmen+less+interested+in+tech/2100-1022_3-5681438.html.
3. Denning, P. The field of programmers myth. *Commun. ACM 47,* 7 (July 2004).
4. Gross, D. *The Future of Physics*. White paper, Jan. 2005; agenda.cern.ch/fullAgenda.php?ida=a05302#2005-01-26.
5. Harel, D. *Computers Ltd. What They Really Can't Do.* Oxford University Press, 2000.
6. Lee, P. *Shrinking Pipeline.* White paper, Aug. 2004; www.wiki.cs.cmu.edu/public/ pmwiki.php?pagename=Main.ShrinkingPipeline.
7. National Science Foundation. *FY 2006 Budget Request to Congress, Mathematical and Physical Sciences*; www.nsf.gov/about/budget/fy2006/pdf/4-ResearchandRelatedActivities/5-MathematicalandPhysicalSciences/20-FY2006.pdf.
8. Patterson, D. The state of funding for new initiatives in computer science and engineering. *Commun. ACM 48,* 4 (Apr. 2005).
9. Vegso, J. Interest in CS as a major drops among incoming freshmen. *Comput. Res. News 17* (May 2005).

Sanjeev Arora (arora@cs.princeton.edu) is a professor of computer science at Princeton University, Princeton, NJ.
Bernard Chazelle (chazelle@cs.princeton.edu) is a professor of computer science at Princeton University, Princeton, NJ.