Moshe Y. Vardi

DOI:10.1145/3306448

Lost in Math?

WHEN I WAS 10 years old, my math teacher started a Math Club. It was not popular enough to last more than a few weeks, but that was long enough for me to learn about matrices and determinants. When I came home, my mother asked me how the club had been. “Beautiful,” I answered. “Do you mean, ‘interesting?’” she inquired. “No,” I said, “Beautiful!” While some people find mathematics befuddling, others find it elegant and beautiful. The mathematician Paul Erdős often referred to “The Book” in which God keeps the most beautiful proofs of each mathematical theorem. The philosopher Bertrand Russell said, “Mathematics, rightly viewed, possesses not only truth, but supreme beauty.” The beauty can be compelling; something so beautiful must be true!

But the seductive power of mathematical beauty has come under criticism lately. In *Lost in Math*, a book published earlier this year, the theoretical physicist Sabine Hossenfelder asserts that mathematical elegance led physics astray. Specifically, she argues that several branches of physics, including string theory and quantum gravity, have come to view mathematical beauty as a truth criterion, in the absence of experimental data to confirm or refute these theories. The theoretical physics community, she argues, is falling victim to group thinking and cognitive bias, seduced by mathematical beauty. About 10 years ago, in the wake of the 2008 financial crisis, the Nobel Laureate economist Paul Krugman made the same point with respect to economics and mathematics in an influential article titled “How Did Economists Get It So Wrong?” His main answer was: mistaking mathematical beauty for truth. “As I see it,” wrote Krugman, “the economics profession went astray because econo-

mists, as a group, mistook beauty, clad in impressive-looking mathematics, for truth.”

So both physics and economics have, arguably, been lost in math. What about computer science? Specifically, what about theoretical computer science (TCS)? TCS is surely blessed with mathematical beauty. As a graduate student a long time ago, it was mathematical beauty that attracted me to TCS, and continued to lead my research for many years. I find computational complexity theory (or complexity theory, for short), with its theorems (for example, the time-hierarchy and space-hierarchy theorems) and its open questions (for example, P vs NP), to be hauntingly beautiful. Beauty, yes; but what about truth?

Physical theories describe the physical world, and by their “truth” we refer to the fidelity in which they describe this world. Economic theories describe economic systems, but by their “truth” we refer not only to the fidelity in which they describe such systems but also to the quality of the guidance they offer to business people and policymakers. I believe complexity theory is similar to economic theories in that respect. It should not only provide a theoretical framework in which we can study the performance of algorithms, but it should also offer sound guidance to algorithm designers and system developers who use algorithms. So how good a theory is complexity theory from that perspective?

It is clear what it means to measure the performance of a specific algorithm A on a specific problem instance I . But complexity theory aims at describing the performance of A over the space of *all* problem instances and it does so by abstracting away from individual problem instances. The typical way in which we do this abstraction is by considering

all problem instances of length n , and asking for upper and lower bounds on algorithmic performance (usually in terms of time and memory utilization) as a function of n . This approach is referred to as the *worst-case* approach, as it focused on the most challenging problem instance of each length n . If the upper bound that we can prove is one of a slow-growing function, for example, $cn \log n$, for a small constant c , then we have a guarantee of good performance on *all* problem instances. But, in general, most upper and lower bound are much less useful. For example, an exponential lower bound just says some problem instances are hard, but says nothing about the practical significance of such instances.

In previous columns I have discussed this gap between theory and practice in specific settings. As I pointed out, program termination may be unsolvable in theory but solvable in practice,^a while Boolean satisfiability may be intractable in theory but tractable in practice.^b In both cases, the worst-case approach is simply too pessimistic and tells us too little about algorithmic performance in practice. Beauty does not necessarily entail truth. Going beyond worst-case complexity is a key challenge in complexity theory and is the subject of much current research. (See Tim Roughgarden’s Review Article on p. 88).

Follow me on Facebook, Google+, and Twitter.

- a <https://cacm.acm.org/magazines/2011/7/109895-solving-the-unsolvable/fulltext>
- b <https://cacm.acm.org/magazines/2014/3/172516-boolean-satisfiability/fulltext>

Moshe Y. Vardi (vardi@cs.rice.edu) is the Karen Ostrum George Distinguished Service Professor in Computational Engineering and Director of the Ken Kennedy Institute for Information Technology at Rice University, Houston, TX, USA. He is the former Editor-in-Chief of *Communications*.

Copyright held by author.