COMPLEXITY

Machines of the Infinite

Whether or not machines can quickly answer yes-or-no questions could affect everything from national security to the limits of human knowledge

By John Pavlus

ON A SNOWY DAY IN PRINCETON, N.J., IN MARCH 1956, a short, owlish-looking man named Kurt Gödel wrote his last letter to a dying friend. Gödel addressed John von Neumann formally even though the two had known each other for decades as colleagues at the Institute for Advanced Study in Princeton. Both men were mathematical geniuses, instrumental in establishing the U.S.'s scientific and military supremacy in the years after World War II. Now, however, von Neumann had cancer, and there was little that even a genius like Gödel could do except express a few overoptimistic pleasantries and then change the subject:

Dear Mr. von Neumann:

With the greatest sorrow I have learned of your illness.... As I hear, in the last months you have undergone a radical treatment and I am happy that this treatment was successful as desired, and that you are now doing better....

Since you now, as I hear, are feeling stronger, I would like to allow myself to write you about a mathematical problem, of which your opinion would very much interest me....

Gödel's description of this problem is utterly unintelligible to nonmathematicians. (Indeed, he may simply have been trying to take von Neumann's mind off of his illness by engaging in an acutely specialized version of small talk.) He wondered how long it would take for a hypothetical machine to spit out answers to a problem. What he concluded sounds like something out of science fiction:

If there really were [such] a machine ... this would have consequences of the greatest importance. Namely, it would obviously mean that ... the mental work of a mathematician concerning Yes-or-No questions could be completely replaced by a machine.

By "mental work," Gödel didn't mean trivial calculations like adding 2 and 2. He was talking about the intuitive leaps that mathematicians take to illuminate entirely new areas of knowledge. Twenty-five years earlier Gödel's now famous incompleteness theorems had forever transformed mathematics. Could a machine be made to churn out similar world-changing insights on demand?

A few weeks after Gödel sent his letter, von Neumann checked into Walter Reed Army Medical Center in Washington, D.C., where he died less than a year later, never having answered his friend. But the problem would outlive both of them. Now

SET DESIGN: KYLE BEAN; PHOTOGRAPHY: RYAN HOPKINSON

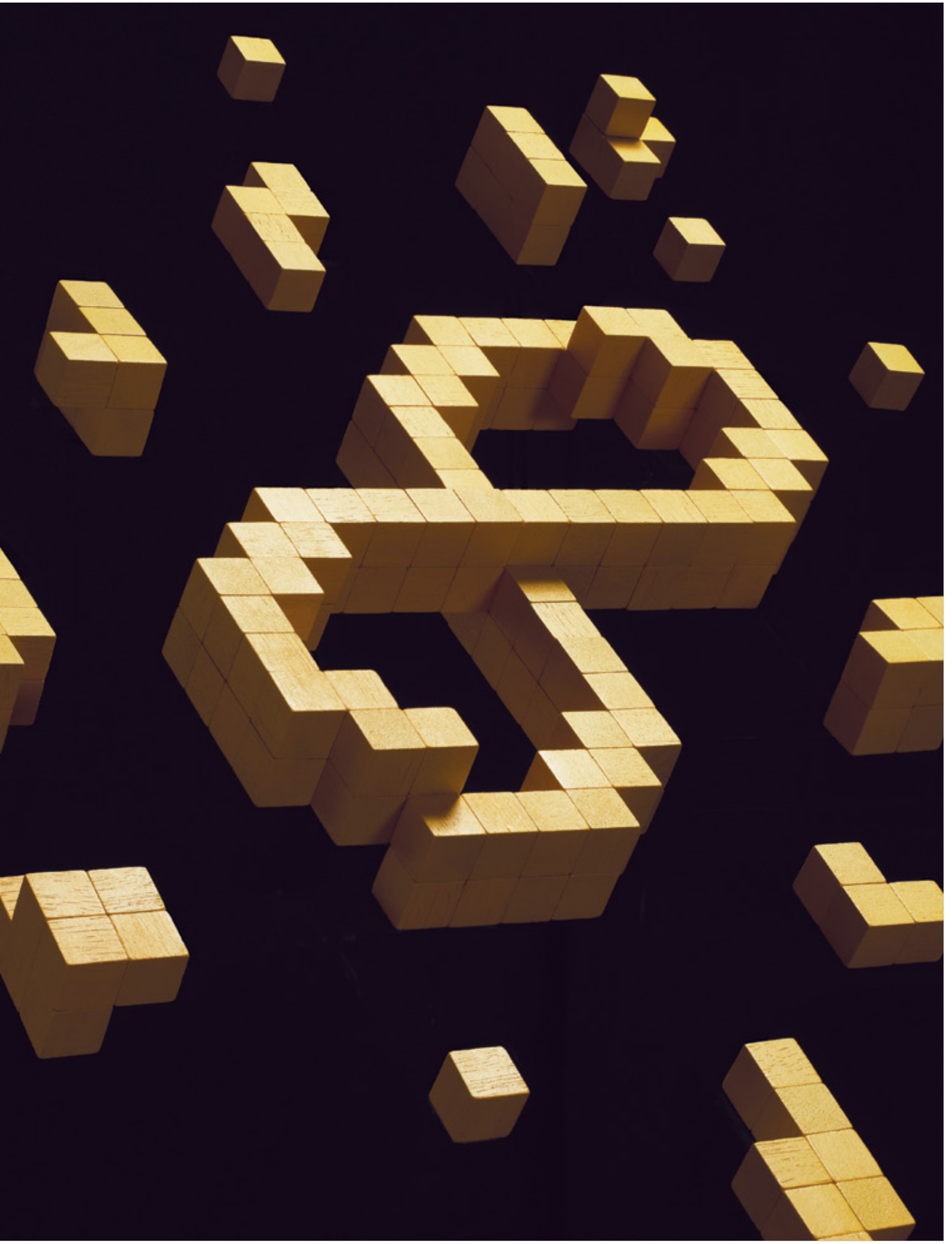


Illustration by Kyle Bean

John Pavlus is a writer and filmmaker focusing on science, technology and design topics. His work has appeared in *Wired, Nature, Technology Review* and other outlets.

known as P versus NP, Gödel's question went on to become an organizing principle of modern computer science. It has spawned an entirely new area of research called computational complexity theory—a fusion of mathematics, science and engineering that seeks to prove, with total certainty, what computers can and cannot do under realistic conditions.

But P versus NP is about much more than just the plastic-and-silicon contraptions we call computers. The problem has practical implications for physics and molecular biology, cryptography, national security, evolution, the limits of mathematics and perhaps even the nature of reality. This one question sets the boundaries for what, in theory, we will ever be able to compute. And in the 21st century the limits of computation look more and more like the limits of human knowledge itself.

THE BET

MICHAEL SIPSER was only a graduate student, but he knew someone would solve the P versus NP problem soon. He even thought he might be the one to do it. It was the fall of 1975, and he was discussing the problem with Leonard Adleman, a fellow graduate student in the computer science department at the University of California, Berkeley. "I had a fascination with P versus NP, had this feeling that I was somehow able to understand it in a way that went beyond the way everyone else seemed to be approaching it," says Sipser, who is now head of the mathematics department at the Massachusetts Institute of Technology. He was so sure of himself that he made a wager that day with Adleman: P versus NP would be solved by the end of the 20th century, if not sooner. The terms: one ounce of pure gold.

Sipser's bet made a kind of poetic sense because P versus NP is itself a problem about how quickly other problems can be solved. Sometimes simply following a checklist of steps will get you to the end result in relatively short order. Think of grocery shopping: you tick off the items one by one until you reach the end of the list. Complexity theorists label these problems P, for "polynomial time," which is a mathematically precise way of saying that no matter how long the grocery list becomes, the amount of time that it will take to tick off all the items will never grow at an unmanageable rate.

In contrast, many more problems may or may not be practical to solve by simply ticking off items on a list, but checking the solution is easy. A jigsaw puzzle is a good example: even though it may take effort to put together, you can recognize the right solution just by looking at it. Complexity theorists call these quickly checkable, "jigsaw puzzle–like" problems NP.

Four years before Sipser made his bet, a mathematician named Stephen Cook had proved that these two kinds of problems are related: every quickly solvable P problem is also a quickly checkable NP problem. The P versus NP question that emerged from Cook's insight—and that has hung over the field ever since—asks if the reverse is also true: Are all quickly checkable problems quickly solvable as well? Intuitively speaking, the answer seems to be no. Recognizing a solved jigsaw puzzle ("Hey, you got it!") is hardly the same thing as doing all the work to find the solution. In other words, P does not seem to equal NP.

What fascinated Sipser was that nobody had been able to mathematically *prove* this seemingly obvious observation. And without a proof, a chance remained, however unlikely or strange, that all NP problems might actually be P problems in disguise. P and NP might be equal—and because computers can make short work of any problem in P, P equals NP would imply that computers' problem-solving powers are vastly greater than we ever imagined. They would be exactly what Gödel described in his letter to von Neumann: mechanical oracles that could efficiently answer just about any question put to them, so long as they could be programmed to verify the solution.

Sipser knew this outcome was vanishingly improbable. Yet proving the opposite, much likelier, case—that P is not equal to NP—would be just as groundbreaking.

Like Gödel's incompleteness theorems, which revealed that mathematics must contain true but unprovable propositions, a proof showing that P does not equal NP would expose an objective truth concerning the limitations of knowledge. Solving a jigsaw puzzle and recognizing that one is solved are two fundamentally different things, and there are no shortcuts to knowledge, no matter how powerful our computers get.

Proving a negative is always difficult, but Gödel had done it. So to Sipser, making his bet with Adleman, 25 years

IN BRIEF

The "P versus NP" question asks whether tough problems whose solutions can be quickly checked (like a jigsaw puzzle) are, at heart, easily solvable as well.
Despite decades of investigation, no one has been able to prove that the two categories are different. If they were not, machines would acquire enormous power.
The problem does not just affect code breakers and Web searches. It suggests a fundamental limitation for biological evolution, physical laws and the nature of knowledge.

seemed like more than enough time to get the job done. If he couldn't prove that P did not equal NP himself, someone else would. And he would still be one ounce of gold richer.

COMPLICATED FAST

ADLEMAN SHARED Sipser's fascination, if not his confidence, because of one cryptic mathematical clue. Cook's paper establishing that P problems are all NP had also proved the existence of a special kind of quickly checkable type of problem called NP-complete. These problems act like a set of magic keys: if you find a fast algorithm for solving one of them, that algorithm will also unlock the solution to every other NP problem and prove that P equals NP.

There was just one catch: NP-complete problems are among the hardest anyone in computer science had ever seen. And once discovered, they began turning up everywhere. Soon after Cook's paper appeared, one of Adleman's mentors at Berkeley, Richard M. Karp, published a landmark study showing that 21 classic computational problems were all NP-complete. Dozens, then hundreds, soon followed. "It was like pulling a finger out of a dike," Adleman says. Scheduling air travel, packing moving boxes into a truck, solving a Sudoku puzzle, designing a computer chip, seating guests at a wedding reception, playing Tetris and thousands of other practical, real-world problems have been proved to be NP-complete.

How could this tantalizing key to solving P versus NP seem so commonplace and so uncrackable at the same time? "That's why I was interested in studying the P versus NP problem," says Adleman, who is now a professor at the University of Southern California. "The power and breadth of these computational questions just seemed deeply awesome. But we certainly didn't understand them. And it didn't seem like we would be understanding them anytime soon." (Adleman's pessimism about P versus NP led to a world-changing invention: a few years after making his bet, Adleman and his colleagues Ronald Rivest and Adi Shamir exploited the seeming incommensurability of P and NP to create their eponymous RSA encryption algorithm, which remains in wide use for online banking, communications and national security applications.)

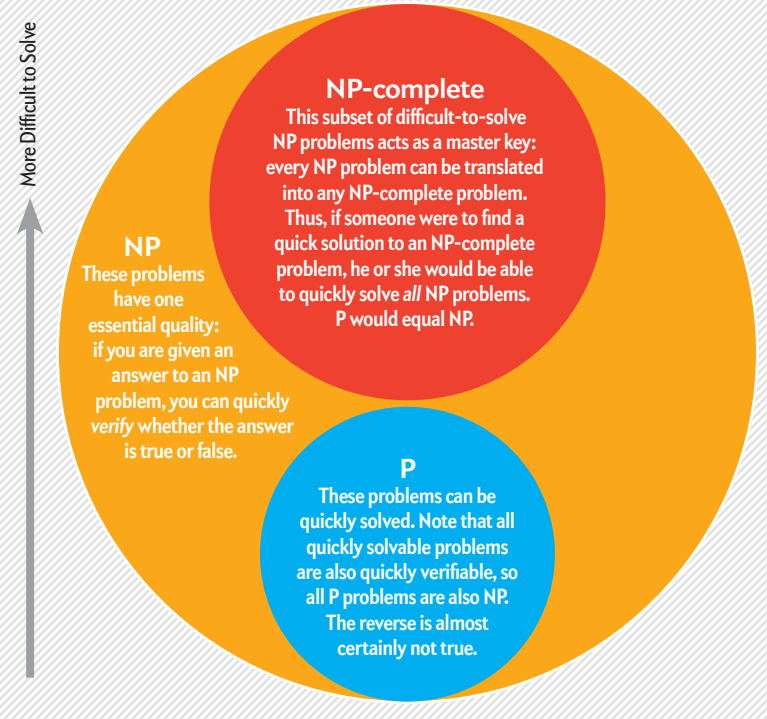
NP-complete problems are hard because they get complicated fast. Imagine you are a backpacker planning a trip through a number of cities in Europe, and you want a route that takes you through each city while minimizing the total distance you will need to travel. How do you find the best route? The simplest method is just to try out each possibility. With five cities to visit, you need to check only 12 possible routes. With 10 cities, the number of possible routes mushrooms to more than 180,000. At 60 cities, the number of paths exceeds the number of atoms in the known universe. This computational nightmare is known as the traveling salesman problem, and in over 80 years of intense study, no one has ever found a general way to solve it that works better than trying every possibility one at a time.

That is the perverse essence of NP-completeness—and of P versus NP: not only are all NP-complete problems equally impossible to solve except in the simplest cases—even if your computer has more memory than God and the entire lifetime of the universe to work with—they seem to pop up everywhere. In fact, these NP-com-

The Basics of Complexity

How long will it take to solve that problem? That's the question that researchers ask as they classify problems into computational classes. As an example, consider a simple sorting task: put a list of random numbers in order from smallest to largest. As the list gets bigger, the time it takes to sort the list increases at a manageable rate—as the square of the size of the list, perhaps. This puts it in class "P" because it can be solved in polynomial time. Harder questions, such as the "traveling salesman" problem [*see box on next page*], require exponentially more time to solve as they grow more complex. These "NP-complete" problems will quickly get so unwieldy that not even billions of processors working for billions of years can crack them.

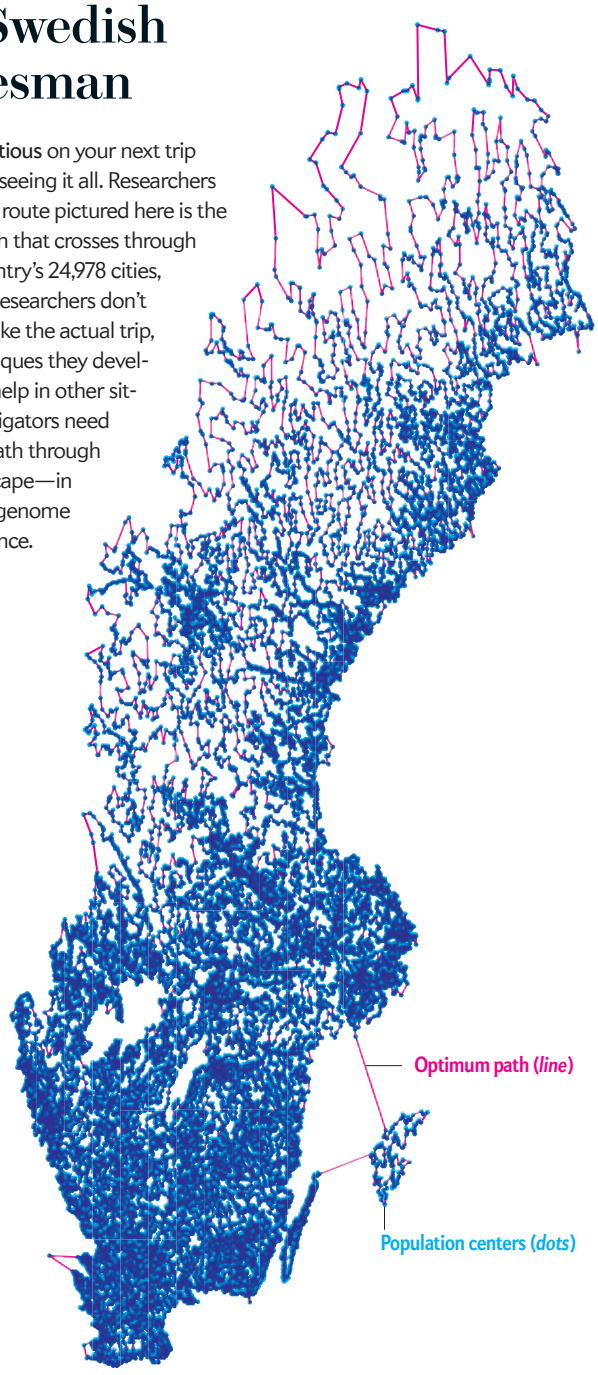
What Kind of Problem Is It?



APPLICATIONS

The Swedish Salesman

If you're feeling ambitious on your next trip to Sweden, consider seeing it all. Researchers have proved that the route pictured here is the shortest possible path that crosses through every one of the country's 24,978 cities, towns and villages. Researchers don't expect anyone to make the actual trip, but the search techniques they developed to solve it will help in other situations where investigators need to find the optimal path through a complicated landscape—in microchip design or genome sequencing, for instance.



DAVID APPLEGATE AT&T Labs, ROBERT BIXBY Rice University, VASEK CHVATAL Concordia University AND WILLIAM J. COOK Georgia Institute of Technology

plete problems don't just frustrate computer scientists. They seem to put limits on the capabilities of nature itself.

NATURE'S CODE

THE PIONEERING Dutch programmer Edsger Dijkstra understood that computational questions have implications beyond mathematics. He once remarked that "computer science is no more about computers than astronomy is about telescopes." In other words, computation is a behavior exhibited by many systems besides those made by Google and Intel. Indeed, any system that transforms inputs into outputs by a set of discrete rules—including those studied by biologists and physicists—can be said to be computing.

In 1994 mathematician Peter Shor proved that cleverly arranged subatomic particles could break modern encryption schemes. In 2002 Adleman used strands of DNA to find an optimal solution to an instance of the traveling salesman problem. And in 2005 Scott Aaronson, an expert in quantum computing who is now at M.I.T.'s Computer Science and Artificial Intelligence Laboratory, used soap bubbles, of all things, to efficiently compute optimal solutions to a problem known as the Steiner tree. These are all exactly the kinds of NP problems that computers should choke their circuit boards on. Do these natural systems know something about P versus NP that computers don't?

"Of course not," Aaronson says. His soap bubble experiment was actually a reductio ad absurdum of the claim that simple physical systems can somehow transcend the differences between P and NP problems. Although the soap bubbles did "compute" perfect solutions to the minimum Steiner tree in a few instances, they quickly failed as the size of the problem increased, just like a computer would. Adleman's DNA-strand experiment hit the same wall. Shor's quantum algorithm does work in all instances, but the factoring problem that it cracks is almost certainly not NP-complete. Therefore, the algorithm doesn't provide the key that would unlock every other NP problem. Biology, classical physics and quantum systems all seem to support the idea that NP-complete problems have no shortcuts. And that would only be true if P did not equal NP.

"Of course, we still can't prove it with airtight certainty," Aaronson says. "But if we were physicists instead of complexity theorists, 'P does not equal NP' would have been declared a law of nature long ago—just like the fact that nothing can go faster than the speed of light." Indeed, some physical theories about the fundamental nature of the universe—such as the holographic principle, suggested by Stephen Hawking's work on black holes—imply that the fabric of reality itself is not continuous but made of discrete bits, just like a computer [see "Is Space Digital?" by Michael Moyer; SCIENTIFIC AMERICAN, February]. Therefore, the apparent

intractability of NP problems—and the limitations on knowledge that this implies—may be baked into the universe at the most fundamental level.

BRAIN MACHINE

SO IF THE VERY UNIVERSE itself is beholden to the computational limits imposed by P versus NP, how can it be that NP-complete problems seem to get solved all the time—even in instances where finding these solutions should take trillions of years or more?

For example, as a human fetus gestates in the womb, its brain wires itself up out of billions of individual neurons. Finding the best arrangement of these cells is an NP-complete problem—one that evolution appears to have solved. "When a neuron reaches out from one point to get to a whole bunch of other synapse points, it's basically a graph-optimization problem, which is NP-hard," says evolutionary neurobiologist Mark Changizi. Yet the brain doesn't actually solve the problem—it makes a close approximation. (In practice, the neurons consistently get within 3 percent of the optimal arrangement.) The *Caenorhabditis elegans* worm, which has only 302 neurons, still doesn't have a perfectly optimal neural-wiring diagram, despite billions on billions of generations of natural selection acting on the problem. "Evolution is constrained by P versus NP," Changizi says, "but it works anyway because life doesn't always require perfection to function well."

And neither, it turns out, do computers. That modern computers can do anything useful at all—much less achieve the wondrous feats we all take for granted on our video-game consoles and smartphones—is proof that the problems in P encompass a great many of our computing needs. For the rest, often an imperfect approximating algorithm is good enough. In fact, these "good enough" algorithms can solve immensely complex search and pattern-matching problems, many of which are technically NP-complete. These solutions are not always mathematically optimal in every case, but that doesn't mean they aren't useful.

Take Google, for instance. Many complexity researchers consider NP problems to be, in essence, search problems. But according to Google's director of research Peter Norvig, the company takes pains to avoid dealing with NP problems altogether. "Our users care about speed more than perfection," he says. Instead Google researchers optimize their algorithms for an even faster computational complexity category than P (referred to as linear time) so that search results appear nearly instantaneously. And if a problem comes up that cannot be solved in this way? "We either reframe it to be easier, or we don't bother," Norvig says.

That is the legacy and the irony of P versus NP. Writing to von Neumann in 1956, Gödel thought the problem held the promise of a future filled with infallible reasoning machines capable of replacing "the mental work of a mathematician" and churning out bold new truths at the push of a button. Instead decades of studying P versus NP have helped build a world in which we extend our machines' problem-solving powers by embracing their limitations. Lifelike approximation, not mechanical perfection, is how Google's autonomous cars can drive themselves on crowded Las Vegas freeways and IBM's Watson can guess its way to victory on *Jeopardy*.

GOLD RUSH

THE YEAR 2000 came and went, and Sipser mailed Adleman his ounce of gold. "I think he wanted it to be embedded in a cube of Lucite, so he could put it on his desk or something," Sipser says. "I didn't do that." That same year the Clay Mathematics Institute in Cambridge, Mass., offered a new bounty for solving P versus NP: $1 million. The prize helped to raise the problem's profile, but it also attracted the attention of amateurs and cranks; nowadays, like many prominent complexity theorists, Sipser says, he regularly receives unsolicited e-mails asking him to review some new attempt to prove that P does not equal NP—or worse, the opposite.

Although P versus NP remains unsolved, many complexity researchers still think it will yield someday. "I never really gave up on it," Sipser says. He claims to still pull out pencil and paper from time to time and work on it—almost for recreation, like a dog chewing on a favorite bone. P versus NP is, after all, an NP problem itself: the only way to find the answer is to keep searching. And while that answer may never come, if it does, we will know it when we see it. SA

MORE TO EXPLORE

The Efficiency of Algorithms. Harry R. Lewis and Christos H. Papadimitriou in *Scientific American*, Vol. 238, No. 1, pages 96–109; January 1978.

The History and Status of the P versus NP Question. Michael Sipser in *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, pages 603–618; 1992.

The Limits of Reason. Gregory Chaitin in *Scientific American*, Vol. 294, No. 3, pages 74–81; March 2006.

The Limits of Quantum Computers. Scott Aaronson in *Scientific American*, Vol. 298, No. 3, pages 62–69; March 2008.

SCIENTIFIC AMERICAN ONLINE

Read a chapter of William J. Cook's new book *In Pursuit of the Traveling Salesman* at ScientificAmerican.com/sep2012/salesman