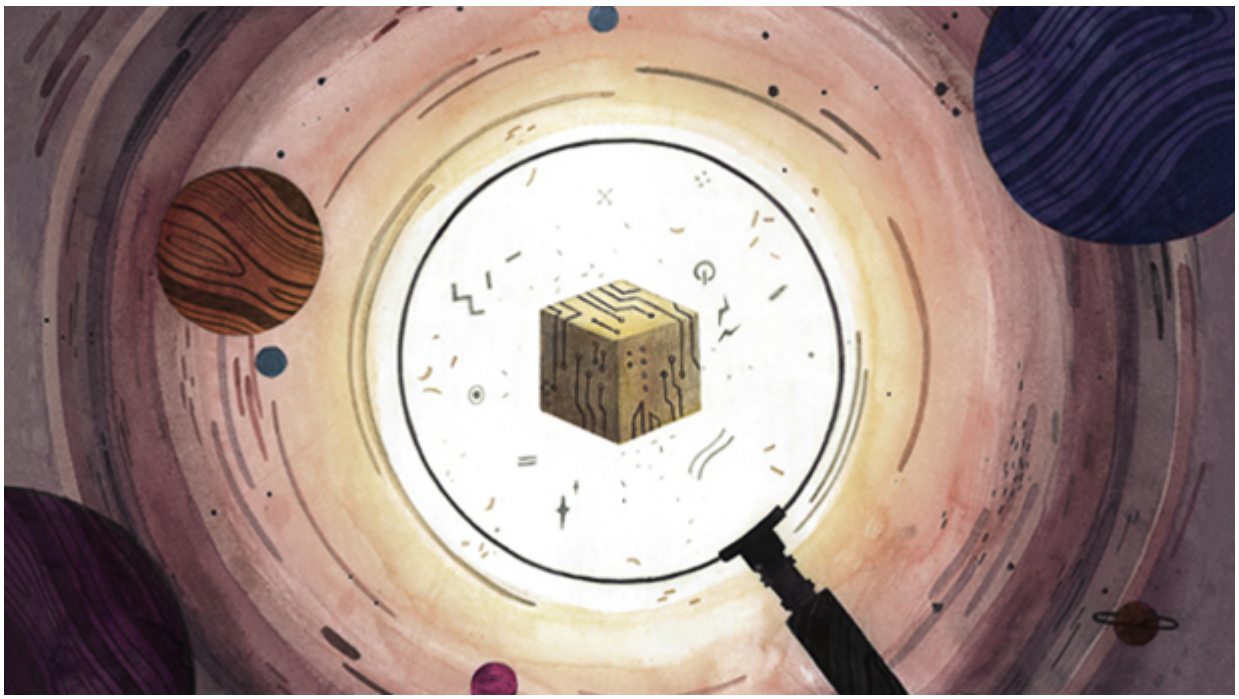
# The Physical Origin of Universal Computing

The physical nature of computers might reveal deep truths about their uniquely powerful abstract abilities.

*By Michael Nielsen*



Imagine you're shopping for a new car, and the salesperson says, "Did you know, this car doesn't just drive on the road."

"Oh?" you reply.

"Yeah, you can also use it to do other things. For instance, it folds up to make a pretty good bicycle. And it folds out to make a first-rate airplane. Oh, and when submerged it works as a submarine. And it's a spaceship too!"

> **Quantized**
>
> *A monthly column in which top researchers explore the process of discovery. This month's columnist, Michael Nielsen, is a computer scientist and author of three books.*

You'd assume the salesperson was joking. But we take a comparable flexibility for granted in our computers. We can use the same machine to fly past the Statue of Liberty with a flight simulator,

make financial projections using a spreadsheet, chat with friends on Facebook, and do many other things. It's very nearly as astonishing as a single machine that works as a car, bicycle and spaceship.

Two characteristics of computers make this flexibility possible. First, computers are programmable. That is, by inputting an appropriate sequence of instructions, we can change a computer's behavior. Second, computers are universal. That is, with the right program we can make a computer perform any algorithmic process whatsoever, as long as the machine has enough memory and time.

These ideas of programmability and universality have become so embedded in our culture that they're familiar even to many children. But historically they were remarkable breakthroughs. They were crystallized in a 1937 paper by Alan Turing, who argued that any algorithmic process whatsoever could be computed by a single universal, programmable computer. The machine Turing described — often known as a Turing machine — was the ancestor of modern computers.

> If you had a complete understanding of the machine, you'd understand all physical processes.

To make his argument, Turing needed to show that his universal computer could perform any conceivable algorithmic process. This wasn't easy. Until Turing's time, the notion of an algorithm was informal, not something with a rigorous, mathematical definition. Mathematicians had, of course, previously discovered many specific algorithms for tasks such as addition, multiplication and determining whether a number is prime. It was pretty straightforward for Turing to show that those known algorithms could be performed on his universal computer. But that wasn't enough. Turing also needed to convincingly argue that his universal computer could compute any algorithm whatsoever, including all algorithms that might be discovered in the future. To do this, Turing developed several lines of thought, each giving an informal justification for the idea that his machine could compute any algorithmic process. Yet he was ultimately uncomfortable with the informal nature of his arguments, saying "All arguments which can be given are bound to be, fundamentally, appeals to intuition, and for this reason rather unsatisfactory mathematically."

In 1985, the physicist David Deutsch took another important step toward understanding the nature of algorithms. He made the observation that algorithmic processes are necessarily carried out by physical systems. These processes can occur in many different ways: A human being using an abacus to multiply two numbers is obviously profoundly different from a silicon chip running a flight simulator. But both are examples of physical systems, and as such they are governed by the same underlying laws of physics. With this in mind, Deutsch stated the following principle. I'll use his words — although the language is specialized, it's actually pretty accessible, and fun to see in the original form:

> Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means.

In other words, take any physical process at all, and you should be able to simulate it using a universal computer. It's an amazing, *Inception*-like idea, that one machine can effectively contain within itself everything conceivable within the laws of physics. Want to simulate a supernova? Or the formation of a black hole? Or even the Big Bang? Deutsch's principle tells you that the universal computer can simulate all of these. In a sense, if you had a complete understanding of the machine, you'd understand all physical processes.

Deutsch's principle goes well beyond Turing's earlier informal arguments. If the principle is true, then it automatically follows that the universal computer can simulate any algorithmic process, since algorithmic processes are ultimately physical processes. You can use the universal computer to simulate addition on an abacus, run a flight simulator on a silicon chip, or do anything else you choose.


Michael Nielsen

Furthermore, unlike Turing's informal arguments, Deutsch's principle is amenable to proof. In particular, we can imagine using the laws of physics to deduce the truth of the principle. That would ground Turing's informal arguments in the laws of physics and provide a firmer basis for our ideas of what an algorithm is.

In attempting this, it helps to modify Deutsch's principle in two ways. First, we must expand our notion of a computer to include quantum computers. This doesn't change the class of physical processes that can be simulated in principle, but it does allow us to quickly and efficiently simulate quantum processes. This matters because quantum processes are often so slow to simulate on conventional computers that they may as well be impossible. Second, we must relax Deutsch's principle so that instead of requiring perfect simulation, we allow simulation to an arbitrary degree of approximation. That's a weaker idea of what it means to simulate a system, but it is likely necessary for the principle to hold.

With these two modifications, Deutsch's principle becomes:

> Every finitely realizable physical system can be simulated efficiently and to an arbitrary degree of approximation by a universal model (quantum) computing machine operating by finite means.

No one has yet managed to deduce this form of Deutsch's principle from the laws of physics. Part of the reason is that we don't yet know what the laws of physics are! In particular, we don't yet know how to combine quantum mechanics with general relativity. And so it's not clear that we can use computers to simulate processes likely to involve quantum gravity, such as the evaporation of black holes.

But even without a quantum theory of gravity, we can ask whether computers can efficiently simulate the best theories of modern physics — the Standard Model of particle physics, and general

relativity.

Researchers are actively working to answer these questions. Over the past few years, the physicist John Preskill and his collaborators have shown how to use quantum computers to efficiently simulate several simple quantum field theories. You can think of these as prototypes of the Standard Model of particle physics. They do not contain the full complexity of the Standard Model, but they have many of its basic ideas. While Preskill and his collaborators haven't yet succeeded in explaining how to simulate the full Standard Model, they have overcome many technical obstacles to doing so. It's plausible that a proof of Deutsch's principle for the Standard Model will be found in the next few years.

The case for general relativity is murkier. General relativity allows for strange singularities that rip and tear space-time in ways that are not yet fully understood. While numerical relativists have developed many techniques for simulating specific physical situations, to my knowledge no complete, systematic analysis of how to efficiently simulate general relativity has yet been done. It's an intriguing open problem.

In his book *The Sciences of the Artificial*, the polymath Herbert Simon distinguished between the sciences of the natural — such as physics and biology, in which we study naturally occurring systems — and sciences of the artificial, like computer science and economics, in which we study systems created by human beings.

At first glance, it seems that the artificial sciences should be special cases of the natural sciences. But as Deutsch's principle suggests, the properties of artificial systems such as computers may be just as rich as those of naturally occurring physical systems. We can imagine using computers to simulate not only our own laws of physics, but maybe even alternate physical realities. In the words of the computer scientist Alan Kay: "In natural science, Nature has given us a world and we're just to discover its laws. In computers, we can stuff laws into it and create a world." Deutsch's principle provides a bridge uniting the sciences of the natural and the artificial. It's exciting that we're nearing proof of this fundamental scientific principle.

*This article was reprinted on ScientificAmerican.com.*