What Makes Quantum Computing So Hard to Explain?

Originalartikel

Backup

<html> Quantum computers, you might have heard, are magical uber-machines that will soon cure cancer and global warming by trying all possible answers in different parallel universes. For 15 years, on my blog and elsewhere, I’ve railed against this cartoonish vision, trying to explain what I see as the subtler but ironically even more fascinating truth. I approach this as a public service and almost my moral duty as a quantum computing researcher. Alas, the work feels Sisyphean: The cringeworthy hype about quantum computers has only increased over the years, as corporations and governments have invested billions, and as the technology has progressed to programmable 50-qubit devices that (on certain contrived benchmarks) really can give the world’s biggest supercomputers a run for their money. And just as in cryptocurrency, machine learning and other trendy fields, with money have come hucksters. In reflective moments, though, I get it. The reality is that even if you removed all the bad incentives and the greed, quantum computing would still be hard to explain briefly and honestly without math. As the quantum computing pioneer Richard Feynman once said about the quantum electrodynamics work that won him the Nobel Prize, if it were possible to describe it in a few sentences, it wouldn’t have been worth a Nobel Prize. Not that that’s stopped people from trying. Ever since Peter Shor discovered in 1994 that a quantum computer could break most of the encryption that protects transactions on the internet, excitement about the technology has been driven by more than just intellectual curiosity. Indeed, developments in the field typically get covered as business or technology stories rather than as science ones. That would be fine if a business or technology reporter could truthfully tell readers, “Look, there’s all this deep quantum stuff under the hood, but all you need to understand is the bottom line: Physicists are on the verge of building faster computers that will revolutionize everything.” The trouble is that quantum computers will not revolutionize everything. Yes, they might someday solve a few specific problems in minutes that (we think) would take longer than the age of the universe on classical computers. But there are many other important problems for which most experts think quantum computers will help only modestly, if at all. Also, while Google and others recently made credible claims that they had achieved contrived quantum speedups, this was only for specific, esoteric benchmarks (ones that I helped develop). A quantum computer that’s big and reliable enough to outperform classical computers at practical applications like breaking cryptographic codes and simulating chemistry is likely still a long way off. But how could a programmable computer be faster for only some problems? Do we know which ones? And what does a “big and reliable” quantum computer even mean in this context? To answer these questions we have to get into the deep stuff. Let’s start with quantum mechanics. (What could be deeper?) The concept of superposition is infamously hard to render in everyday words. So, not surprisingly, many writers opt for an easy way out: They say that superposition means “both at once,” so that a quantum bit, or qubit, is just a bit that can be “both 0 and 1 at the same time,” while a classical bit can be only one or the other. They go on to say that a quantum computer would achieve its speed by using qubits to try all possible solutions in superposition — that is, at the same time, or in parallel. This is what I’ve come to think of as the fundamental misstep of quantum computing popularization, the one that

leads to all the rest. From here it’s just a short hop to quantum computers quickly solving something like the traveling salesperson problem by trying all possible answers at once — something almost all experts believe they won’t be able to do. The thing is, for a computer to be useful, at some point you need to look at it and read an output. But if you look at an equal superposition of all possible answers, the rules of quantum mechanics say you’ll just see and read a random answer. And if that’s all you wanted, you could’ve picked one yourself. What superposition really means is “complex linear combination.” Here, we mean “complex” not in the sense of “complicated” but in the sense of a real plus an imaginary number, while “linear combination” means we add together different multiples of states. So a qubit is a bit that has a complex number called an amplitude attached to the possibility that it’s 0, and a different amplitude attached to the possibility that it’s 1. These amplitudes are closely related to probabilities, in that the further some outcome’s amplitude is from zero, the larger the chance of seeing that outcome; more precisely, the probability equals the distance squared. But amplitudes are not probabilities. They follow different rules. For example, if some contributions to an amplitude are positive and others are negative, then the contributions can interfere destructively and cancel each other out, so that the amplitude is zero and the corresponding outcome is never observed; likewise, they can interfere constructively and increase the likelihood of a given outcome. The goal in devising an algorithm for a quantum computer is to choreograph a pattern of constructive and destructive interference so that for each wrong answer the contributions to its amplitude cancel each other out, whereas for the right answer the contributions reinforce each other. If, and only if, you can arrange that, you’ll see the right answer with a large probability when you look. The tricky part is to do this without knowing the answer in advance, and faster than you could do it with a classical computer. Twenty-seven years ago, Shor showed how to do all this for the problem of factoring integers, which breaks the widely used cryptographic codes underlying much of online commerce. We now know how to do it for some other problems, too, but only by exploiting the special mathematical structures in those problems. It’s not just a matter of trying all possible answers at once. Compounding the difficulty is that, if you want to talk honestly about quantum computing, then you also need the conceptual vocabulary of theoretical computer science. I’m often asked how many times faster a quantum computer will be than today’s computers. A million times? A billion? This question misses the point of quantum computers, which is to achieve better “scaling behavior,” or running time as a function of n, the number of bits of input data. This could mean taking a problem where the best classical algorithm needs a number of steps that grows exponentially with n, and solving it using a number of steps that grows only as n^2. In such cases, for small n, solving the problem with a quantum computer will actually be slower and more expensive than solving it classically. It’s only as n grows that the quantum speedup first appears and then eventually comes to dominate. But how can we know that there’s no classical shortcut — a conventional algorithm that would have similar scaling behavior to the quantum algorithm’s? Though typically ignored in popular accounts, this question is central to quantum algorithms research, where often the difficulty is not so much proving that a quantum computer can do something quickly, but convincingly arguing that a classical computer can’t. Alas, it turns out to be staggeringly hard to prove that problems are hard, as illustrated by the famous P versus NP problem (which asks, roughly, whether every problem with quickly checkable solutions can also be quickly solved). This is not just an academic issue, a matter of dotting i’s: Over the past few decades, conjectured quantum speedups have repeatedly gone away when classical algorithms were found with similar

performance. Note that, after explaining all this, I still haven’t said a word about the practical difficulty of building quantum computers. The problem, in a word, is decoherence, which means unwanted interaction between a quantum computer and its environment — nearby electric fields, warm objects, and other things that can record information about the qubits. This can result in premature “measurement” of the qubits, which collapses them down to classical bits that are either definitely 0 or definitely 1. The only known solution to this problem is quantum error correction: a scheme, proposed in the mid-1990s, that cleverly encodes each qubit of the quantum computation into the collective state of dozens or even thousands of physical qubits. But researchers are only now starting to make such error correction work in the real world, and actually putting it to use will take much longer. When you read about the latest experiment with 50 or 60 physical qubits, it’s important to understand that the qubits aren’t error-corrected. Until they are, we don’t expect to be able to scale beyond a few hundred qubits. Once someone understands these concepts, I’d say they’re ready to start reading — or possibly even writing — an article on the latest claimed advance in quantum computing. They’ll know which questions to ask in the constant struggle to distinguish reality from hype. Understanding this stuff really is possible — after all, it isn’t rocket science; it’s just quantum computing! </html>