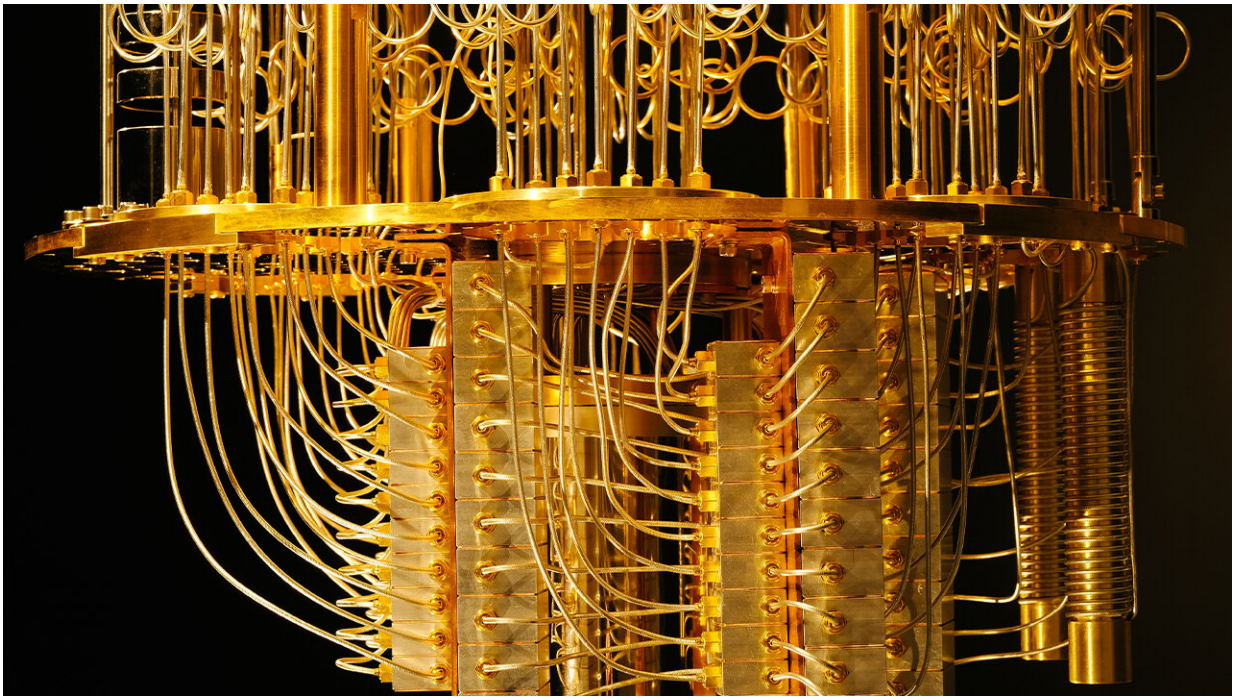
A language for quantum computing

January 20 2022, by Rachel Gordon



Credit: MIT Computer Science & Artificial Intelligence Lab

Time crystals. Microwaves. Diamonds. What do these three disparate things have in common?

Quantum computing. Unlike traditional computers that use bits, quantum computers use qubits to encode information as zeros or ones, or both at the same time. Coupled with a cocktail of forces from quantum physics, these fridge-sized machines can process a whole lot of information—but

they're far from flawless. Just like our regular computers, we need to have the right programming languages to properly compute on quantum computers.

Programming quantum computers requires awareness of something called "entanglement," a computational multiplier for qubits of sorts, which translates to a lot of power. When two qubits are entangled, actions on one qubit can change the value of the other even when they are physically separated, giving rise to Einstein's characterization of "spooky action at a distance." But that potency is equal parts a source of weakness. When programming, discarding one qubit without being mindful of its entanglement with another qubit can destroy the data stored in the other, jeopardizing the correctness of the program.

Scientists from MIT's Computer Science and Artificial Intelligence (CSAIL) aimed to do some unraveling by creating their own programming language for quantum computing called Twist. Twist can describe and verify which pieces of data are entangled in a quantum program, through a language a classical programmer can understand. The language uses a concept called purity, which enforces the absence of entanglement and results in more intuitive programs, with ideally fewer bugs. For example, a programmer can use Twist to say that the temporary data generated as garbage by a program is not entangled with the program's answer, making it safe to throw away.

While the nascent field can feel a little flashy and futuristic, with images of mammoth wiry gold machines coming to mind, quantum computers have potential for computational breakthroughs in classically unsolvable tasks, like cryptographic and communication protocols, search, and computational physics and chemistry. One of the key challenges in computational sciences is dealing with the complexity of the problem and the amount of computation needed. Whereas a classical digital computer would need a very large exponential number of bits to be able

to process such a simulation, a quantum computer could do it, potentially, using a very small number of qubits—if the right programs are there.

"Our language Twist allows a developer to write safer quantum programs by explicitly stating when a qubit must not be entangled with another," said MIT Ph.D. student Charles Yuan, the lead author on a paper about Twist. "Because understanding quantum programs requires understanding entanglement, we hope that Twist paves the way to languages that make the unique challenges of quantum computing more accessible to programmers."

Untangling quantum entanglement

Imagine a wooden box that has a thousand cables protruding out from one side. You can pull any cable all the way out of the box, or push it all the way in.

After you do this for a while, the cables form a pattern of bits—zeros and ones—depending on whether they're in or out. This box represents the memory of a classical computer. A program for this computer is a sequence of instructions for when and how to pull on the cables.

Now imagine a second, identical looking box. This time, you tug on a cable, and see that as it emerges, a couple of other cables are pulled back inside. Clearly, inside the box, these cables are somehow entangled with each other.

The second box is an analogy for a quantum computer, and understanding the meaning of a quantum program requires understanding the entanglement present in its data. But detecting entanglement is not straightforward. You can't see into the wooden box, so the best you can do is try pulling on cables and carefully reason about

which are entangled. In the same way, quantum programmers today have to reason about entanglement by hand. This is where the design of Twist helps massage some of those interlaced pieces.

The scientists designed Twist to be expressive enough to write out programs for well-known quantum algorithms and identify bugs in their implementations. To evaluate Twist's design, they modified the programs to introduce some kind of bug that would be relatively subtle for a human programmer to detect, and showed that Twist could automatically identify the bugs and reject the programs.

They also measured how well the programs performed in practice in terms of runtime, which had less than four percent overhead over existing quantum programming techniques.

For those wary of quantum's "seedy" reputation in its potential to break encryption systems, Yuan says it's still not very well known to what extent quantum computers will actually be able to reach their performance promises in practice. "There's a lot of research that's going on in post-quantum cryptography, which exists because even quantum computing is not all-powerful. So far, there's a very specific set of applications in which people have developed algorithms and techniques where a quantum computer can outperform classical computers."

An important next step is using Twist to create higher-level quantum programming languages. Most quantum programming languages today still resemble assembly language, stringing together low-level operations, without mindfulness towards things like data types and functions, and what's typical in classical software engineering.

"Quantum computers are error-prone and difficult to program. By introducing and reasoning about the 'purity' of program code, Twist takes a big step toward making quantum programming easier by

guaranteeing that the quantum bits in a pure piece of code cannot be altered by bits not in that code," says Fred Chong, the Seymour Goodman Professor of Computer Science at the University of Chicago and Chief Scientist at Super.tech.

More information: Charles Yuan, Christopher McNally, Michael Carbin, Twist: Sound Reasoning for Purity and Entanglement in Quantum Programs. popl22.sigplan.org/details/POP … -in-Quantum-Programs

Provided by Massachusetts Institute of Technology