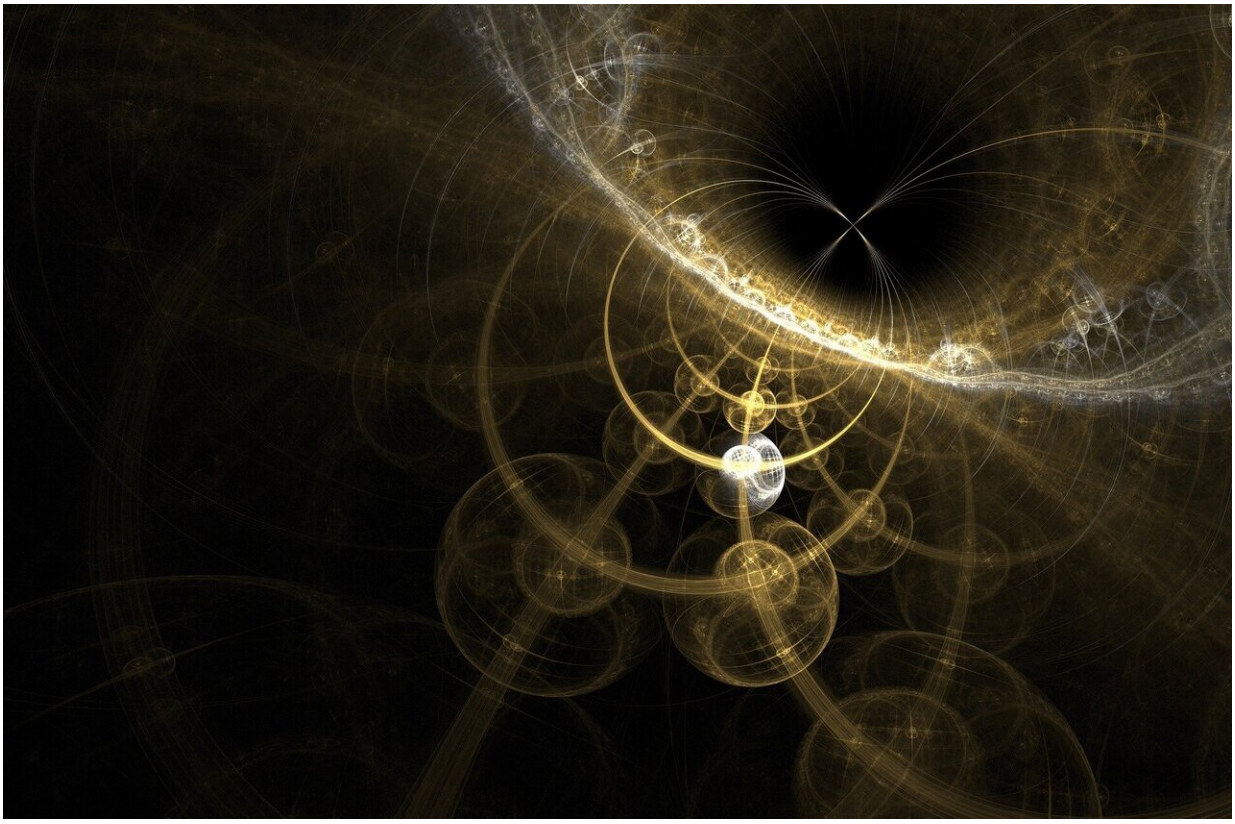
Researchers make steps toward debugging tools for quantum computers

June 21 2019, by Yipeng Huang, Margaret Martonosi



Credit: CC0 Public Domain

In classical computing, debugging programs is one of the most time-consuming tasks in software development. Successful debugging relies on software development tools and also on the experience of the

programmer. In quantum computing, researchers predict debugging will be an even greater challenge. In a paper soon to appear at the ACM/IEEE 46th Annual International Symposium for Computer Architecture (as part of ACM's 2019 Federated Computing Research Conferences), researchers at Princeton University present debugging tools based on statistical tests, with a goal of aiding programmers in building correct quantum programs for near-term quantum computers.

Quantum computing promises to change the computing world by offering capabilities beyond any classical computer. Those capabilities come from quantum algorithms—sequences of instructions that tell a quantum computer what to do in order to calculate some result, much like software for classical computers today. These algorithms cover a wide range of applications. For example, quantum chemistry algorithms would allow scientists to calculate properties of chemical compounds directly from the governing equations of quantum mechanics, a formidable task beyond the reach of modern computers for all but the simplest molecules. Other algorithms promise to speed up searching inside databases and to enable secure communications resistant to eavesdropping.

For about two decades, these quantum algorithms existed only as abstract equations and specifications, and have never actually been run on real quantum computers. That research landscape has changed rapidly. In the past couple of years, researchers have built the first prototype quantum computers capable of running quantum programs. Notably, IBM has made small-scale quantum computers available for the public to run code and see results. With this burgeoning interest in quantum computing experimentation, a new and urgent challenge lies in helping programmers translate those abstract algorithms into correctly functioning quantum program code.

"We were finding that even researchers who specialize in quantum

computing are making subtle mistakes in quantum program code, preventing the programs from giving correct results," Yipeng Huang, postdoc at Princeton University and an author of the paper, said. "If it is so tricky for experienced quantum programmers, how can students new to quantum computing write correct programs, without the aid of tools such as debuggers?"

In the paper titled "Statistical Assertions for Validating Patterns and Finding Bugs in Quantum Programs," Huang and Margaret Martonosi, a professor of Computer Science at Princeton, identify three key difficulties in debugging quantum programs, and evaluate their solutions in addressing those difficulties.

The first difficulty is that programmers cannot easily examine the values of variables of a quantum program, while the program is running. This limitation makes debugging difficult, considering that one of the go-to moves in debugging programs is to inspect the values of variables step-by-step in the course of a program. Quantum programmers cannot do this kind of debugging because reading quantum variables would involve measuring and "collapsing" the delicate quantum states inside quantum computers. Once a quantum state is collapsed, any observations would not be a complete description of the state of the program.

In their paper, Huang and Martonosi address this challenge by finding ways to debug quantum programs using only the information about the collapsed quantum states. They consider debugging programs in two different kinds of settings; in one setting the quantum programs run in simulation on a classical computer, and in the other setting the programs run on real prototype quantum computers. In both settings, they use multiple runs of the quantum program in order find the distribution of the states inside the quantum program.

The second difficulty is that even when observations or simulations are

available, quantum states are in general high-dimensional and difficult to interpret, limiting their usefulness for programmers to debug misbehaving quantum programs.

Huang and Martonosi's solution to this challenge is to use statistical tests on measurement results, in order to help programmers decide if the results are consistent with three types of states. They use the chi-square statistical test to decide if the observed states belong to one of classical, superposition, or entangled states. "We focus our attention on these three types of states because they occur throughout a quantum program, and are easier for programmers to identify," Huang said. "If the states don't match what the programmer expects, the statistical tests help the programmer zoom in and find mistakes in the program code."

The third difficulty is that programmers do not yet have any guidelines for where and what to check when debugging quantum programs. Until recently, quantum algorithms existed mainly as equations; occasionally, the algorithms would be more fleshed-out in the form of quantum circuit diagrams. The task of coding quantum programs entails translating these quantum circuit diagrams into program code. "The state-of-the-art in quantum programming is akin to programming classical computers 50 years ago," Huang said. "For the time being, researchers are writing quantum programs operation-by-operation, on very low-level bits of information. One contribution of our paper is to discuss how the patterns and structures inside quantum algorithms guide programmers to know what to check."

In their paper, Huang and Martonosi use their debugging tools to test several benchmark quantum programs, including one for factoring integers, one for searching for data, and one in the area of quantum chemistry. Program patterns common inside these algorithms, such as looping operations, nesting operations, and mirroring operations, serve as guides for quantum programmers to know where to use the debugging

tools.

Supported by the National Science Foundation through the EPiQC Expedition project, Huang and Martonosi's work in debugging tools is a pragmatic approach to the problem of writing correct quantum programs. It joins a growing field of related approaches, many which are based on formal proofs. "We are finding that writing correct quantum programs relies on a mix of techniques," Huang said. "Just like the case in classical programming, quantum programmers will rely on a mix of pragmatic and formal techniques."

More information: Statistical Assertions for Validating Patterns and Finding Bugs in Quantum Programs, arXiv:1905.09721 [quant-ph] arxiv.org/abs/1905.09721

Provided by Princeton University