Tool checks computer architectures, reveals flaws in emerging design

April 12 2017

Researchers including Professor Margaret Martonosi (center) and graduate students Yatin Manerkar and Caroline Trippel have developed a tool that eliminates bugs by checking computer processor designs for memory issues. The tool is already leading to improvements in a major open-source chip project. Credit: David Kelly Crow

With backing from some of the largest technology companies, a major project called RISC-V seeks to facilitate open-source design for computer chips, offering the possibility of opening chip designs beyond the few firms that dominate the space. As the project moves toward a formal release, researchers at Princeton University have discovered a series of errors in the RISC-V instruction specification that now are leading to changes in the new system.

The researchers, testing a technique they created for analyzing computer memory use, found over 100 errors involving incorrect orderings in the storage and retrieval of information from memory in variations of the RISC-V processor architecture. The researchers warned that, if uncorrected, the problems could cause errors in software running on RISC-V chips. Officials at the RISC-V Foundation said the errors would not affect most versions of RISC-V but would have caused problems for higher-performance systems.

"Incorrect memory access orderings can result in software performing calculations using the wrong values," said Margaret Martonosi, the Hugh Trumbull Adams '35 Professor of Computer Science at Princeton and the leader of the Princeton team that also includes Ph.D. students Caroline Trippel and Yatin Manerkar. "These in turn can lead to hard-to-debug software errors that either cause the software to crash or to be vulnerable to security exploits. With RISC-V processors often envisioned as control processors for real-world physical devices (i.e., internet of things devices) these errors can cause unreliability or security vulnerabilities affecting the overall safety of the systems."

Krste Asanović, the chair of the RISC-V Foundation, welcomed the researchers' contributions. He said the RISC-V Foundation has formed a working group, headed by Martonosi's former graduate student and co-researcher Daniel Lustig, to solve the memory-ordering problems. Asanović, a professor of electrical engineering and computer science at

the University of California-Berkeley, said the RISC-V project was looking for input from the design community to "fill the gaps and the holes and getting a spec that everyone can agree on."

"The goal is to ratify the spec in 2017," he said. "The memory model is part of that."

Lustig, a co-author of Martonosi's recent paper and now a research scientist at NVIDIA, said work was underway to improve the RISC-V memory model.

"RISC-V is in the fortunate position of being able to look back on decades' worth of industry and academic experience," he said. "It will be able to learn from all of the insights and mistakes made by previous attempts."

The RISC-V project essentially offers specifications that guide the hardware and software design of RISC-V processors and software applications. These specifications, known generally as an Instruction Set Architecture, describe the most basic functions of the processor including its arithmetic and logic operations, as well as the way programs use the computer's memory. Hardware designers use the instruction sets when building new chips, and computer programmers rely on it when writing new software.

The vast majority of computers in use today, including the millions of PC and Apple systems, use microprocessors based on instruction sets created by the chip-making giants Intel or ARM (Intel's, for example, is commonly known as x86). By contrast, the RISC-V instruction set has been open source from the start. It was first developed at UC-Berkeley, with the idea that any designer could use the instruction set to create chips, and also the software compilers that translate software applications from high-level programming languages in order to run on

them. The project is now run by the RISC-V Foundation, whose membership includes a roster of universities, nonprofit organizations and top technology companies, including Google, IBM, Microsoft, NVIDIA and Oracle.

Martonosi's team discovered the problems when testing their new system to check memory operations across any computer architecture. The system, called TriCheck, allows designers and others interested in working with a design, to detect memory ordering errors before they become a problem. The name TriCheck derives from three general levels of computing: the high-level programs that create modern applications from web browsers to word processors; the instruction set architecture that functions as a basic language of the machine; and the underlying hardware implementation, a particular microprocessor designed to execute the instruction set.

"We call it TriCheck because it is making sure that the memory-ordering guarantees made by these three levels are in alignment," Martonosi said.

In a paper presented April 10 at the ACM International Conference on Architectural Support for Programming Languages and Operating Systems, the researchers describe the TriCheck system. The paper also details how TriCheck identified potential memory issues in high-performance hardware implementations of the RISC-V instruction set. (Similar errors were not uncovered in other implementations of the architecture.) The authors describe how they ran tests of the RISC-V instruction set using a high-level program written in the C programming language. On one particular RISC-V-compliant design, TriCheck found 144 errant programs out of 1,701 test programs.

The memory ordering challenge stems from the complexity of modern computers. As designers squeeze more performance out of computer systems, they rely on many concurrent operations sharing the same

sections of computer memory. This parallel, shared-memory operation is extremely efficient, both for speed and power usage, but it puts a heavy demand on the computer's ability to interleave and properly order memory usage. If, for example, several processes are using the same section of memory, the computer needs to make sure that operations are applied to memory in the correct order, which may not always be the order in which they arrive from different concurrently running processors.

Subtle changes in any of the three computing levels—the machine level, the compiler and the high-level programming languages—can have unintended effects on the other layers. All three have to work together seamlessly to make sure memory errors don't crop up. One advantage of TriCheck is that it allows experts in one of these layers to avoid conflicts with the other two layers, even if they do not have expertise in them.

"If I write a program in C, it makes some assumptions about memory ordering," Martonosi said. "Subsequently, a different set of memory ordering rules are defined by the instruction-set architecture. We need to check that the high-level program's assumptions are accurately supported by the underlying instruction set and processor design."

However, the researchers said the TriCheck's greatest strength is its ability to give designers a broad view of memory usage. Although designers have long been interested in this perspective, previous attempts to comprehensively analyze memory operations have been too slow to be practical.

TriCheck is able to check memory ordering efficiently by using succinct formal specifications of memory ordering rules, known as axioms. For a given program, compiler, instruction set and hardware implementation, TriCheck can enumerate many ordering possibilities from these axioms, and then check for errors. By expressing the memory-ordering

possibilities as connected graphs, TriCheck can identify potential errors by looking for cycles in the graphs. These checks can be done very efficiently on modern high-performance computers, and TriCheck's speed has allowed it to explore larger and more complex designs than prior work.

The TriCheck project culminates four years of work by Martonosi's group of developing checks across various layers of hardware, memory and software. A recent project, PipeCheck, was aimed at allowing the teams who were developing the physical microchip to alter their designs to avoid memory-ordering problems. Typically, chip designers focus detailed verification of the chip's correctness in the last stage of development. But problems can be expensive to fix that late in design. PipeCheck allows designers to check the memory usage much earlier in the design pipeline and to correct errors sooner at this cheaper stage.

"PipeCheck helps determine whether the central processing unit keeps the promises it makes about memory usage," Martonosi said.

Lustig worked on the PipeCheck and TriCheck projects and joined NVIDIA after receiving his doctorate in 2015. Now at NVIDIA, which makes processors that specialize in computer graphics, Lustig said he appreciates the ability to closely analyze designs for bugs.

"Tools like PipeCheck and TriCheck are extremely useful when building new architectures, particularly for designs that really try to push the envelope," Lustig said. He added that identifying bugs can be difficult, but even when they are uncovered, making sure that solutions don't cause problems in other areas of the system poses new challenges. "TriCheck puts all the pieces together by verifying the software, the compiler, the architecture and the microarchitecture in one single package."

In addition to Martonosi and Lustig, the paper's authors include:

Caroline Trippel and Yatin Manerkar, of Princeton; and Michael Pellauer of NVIDIA. Support for the work was provided in part by the National Science Foundation and by CFAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by the Microelectronics Advanced Research Corporation and the Defense Advanced Research Projects Agency.

Martonosi said the goal for the TriCheck project is to stop bugs before they create problems for users.

"TriCheck is an important step in our overall goal of verifying correct memory orderings comprehensively across complex hardware and software systems," she said. "Given the increased reliance on computer systems everywhere—including finance, automobiles and industrial control systems—moving towards verifiably correct operation is important for their reliability and safety."

Provided by Princeton University