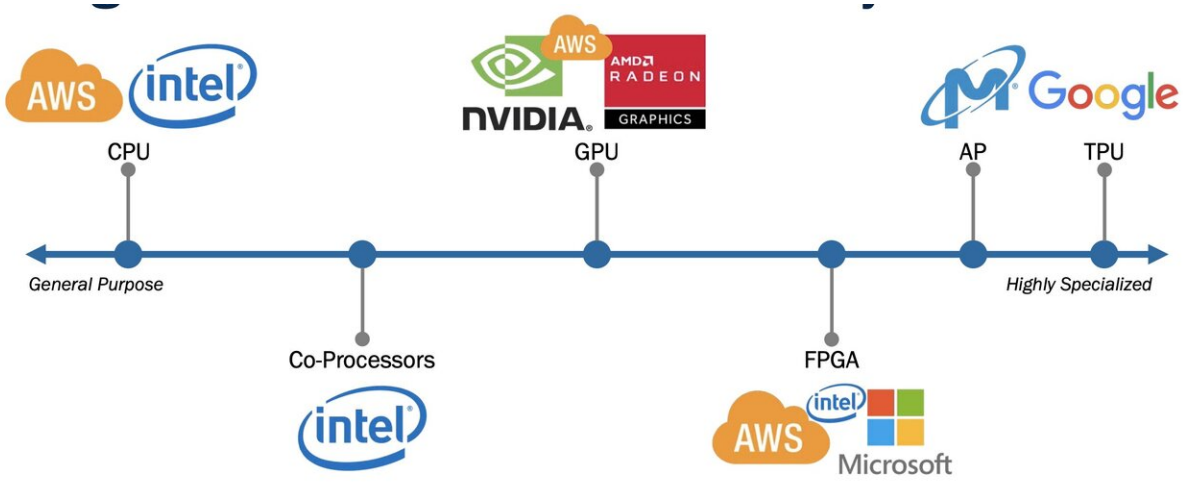
Putting hardware accelerators to work with automatic code translation

March 27 2020



Companies like Google, Amazon, and Intel have broadly adopted different kinds of hardware accelerators in their cloud computing pipelines, but many programs still can't take full advantage of them. Credit: University of Michigan

A new technique developed by researchers at the University of Michigan could enable broader adoption of post-Moore's Law computing components through automatic code translation. The system, called AutomataSynth, allows software engineers to tap into the power of hardware accelerators like FPGAs without specialized programming knowledge or needing to rewrite old, CPU-centric code.

With Moore's Law nearing its end, companies and designers rely on a

number of hardware techniques to circumvent the diminishing returns provided by new CPUs. Among the most viable short-term candidates have been hardware accelerators like field-programmable gate arrays (FPGAs), which can be dedicated to rapidly executing particular common functions and eliminating bottlenecks in larger applications.

While their adoption by companies like Microsoft and Amazon Web Services is already well underway, FPGAs are limited in their use by programming requirements that are foreign to many software developers . These requirements also limit their use on pre-existing legacy software, which was typically written to work specifically with CPUs.

Most programs in use today have to be completely rewritten at a very low level to reap the benefits of hardware acceleration. Because of this, the components are being installed more rapidly than they're actually being utilized.

"Companies are taking steps to try to make [FPGAs] more approachable for people," says Kevin Angstadt, a Ph.D. candidate leading the project, "but when it comes to writing new programs, the process is still very primitive."

The team, which includes Profs. Westley Weimer and Jean-Baptiste Jeannin, sought to break through those adoption barriers by automatically rewriting certain low-level functions used by many larger applications. Implemented at scale, this would mean developers could make full use of FPGAs by simply adding a few lines to their existing code—no rewriting required.

Angstadt compares the process to the adaptation of graphics processing units (GPUs) to general purpose computing, initially enabled by NVIDIA's CUDA platform. CUDA provided developers with an interface through which they could tap into a GPU's processing power

on non-graphics tasks.

"The goal of this work was to do something similar with FPGAs," Angstadt says. "You can't just write in the same language when you're using them, but we can give you the illusion of writing in the same language."

To achieve this, the researchers made use of a technique called bounded automata learning. The technique uses a combination of different program analyses to produce a state machine that is both equivalent in function to the original code and amenable to acceleration with FPGAs. Essentially, their system reads the code its given, learns the functionality of the code, and writes a hardware description of this behavior to send to the accelerator.

In a paper presented on the work, they demonstrated their technique on one class of functions, string kernels, which handle search and comparison operations performed on text.

"Anytime you search through a text document, you're using one of the most primitive forms of these string kernels," says Angstadt. But they can be far more complex—the functions are fundamental to such diverse applications as spam detection, genetic analysis, product recommendations, and particle physics.

Running AutomataSynth on a benchmark suite of real-world string functions written to work with CPUs, the team found that it was able to learn fully-equivalent hardware designs in 72% of cases and close approximations in another 11%.

Application of this technique to other classes of functions remains an open problem for further work, but Angstadt is confident it can be put to use broadly.

"Our research indicates that even though many kinds of functions exist other than the ones we support, there are more applications than we realize that can be rewritten as string kernels," says Angstadt. "So we might still be able to apply similar techniques to support further kinds of code."

If successful, we could see FPGAs become a standard component for general-purpose computing.

The researchers presented their findings in the paper "Accelerating Legacy String Kernels via Bounded Automata Learning," accepted by the 2020 Architectural Support for Programming Languages and Operating Systems (ASPLOS) Conference.

More information: Accelerating Legacy String Kernels via Bounded Automata Learning: web.eecs.umich.edu/~weimerw/p/ … eimer-asplos2020.pdf

Provided by University of Michigan

Citation: Putting hardware accelerators to work with automatic code translation (2020, March 27) retrieved 30 April 2024 from https://techxplore.com/news/2020-03-hardware-automatic-code.html