Python-based compiler achieves orders-of-magnitude speedups

March 14 2023, by Rachel Gordon

Codon is a Python-based compiler that aims to democratize high-performance computing. Credit: Alex Shipps/MIT CSAIL via Midjourney

In 2018, The Economist published an in-depth piece on the programming language Python. "In the past 12 months," the article said,

"Google users in America have searched for Python more often than for Kim Kardashian." Reality TV stars, be wary.

The high-level language has earned its popularity, too, with legions of users flocking daily to the language for its ease of use due in part to its simple and easy-to-learn syntax. This led researchers from MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL) and elsewhere to make a tool to help run Python code more efficiently and effectively while allowing for customization and adaptation to different needs and contexts. The compiler, which is a software tool that translates source code into machine code that can be executed by a computer's processor, lets developers create new domain-specific languages (DSLs) within Python—which is typically orders of magnitude slower than languages like C or C++—while still getting the performance benefits of those other languages.

DSLs are specialized languages tailored to specific tasks that can be much easier to work with than general-purpose programming languages. However, creating a new DSL from scratch can be a bit of a headache.

"We realized that people don't necessarily want to learn a new language, or a new tool, especially those who are nontechnical. So we thought, let's take Python syntax, semantics, and libraries and incorporate them into a new system built from the ground up," says Ariya Shajii, Ph.D. , lead author on a new paper about the team's new system, Codon. "The user simply writes Python like they're used to, without having to worry about data types or performance, which we handle automatically—and the result is that their code runs 10 to 100 times faster than regular Python. Codon is already being used commercially in fields like quantitative finance, bioinformatics, and deep learning."

The team put Codon through some rigorous testing, and it punched above its weight. Specifically, they took roughly 10 commonly used

genomics applications written in Python and compiled them using Codon, and achieved five to 10 times speedups over the original hand-optimized implementations. Besides genomics, they explored applications in quantitative finance, which also handles big datasets and uses Python heavily. The Codon platform also has a parallel backend that lets users write Python code that can be explicitly compiled for GPUs or multiple cores, tasks that have traditionally required low-level programming expertise.

Pythons on a plane

Unlike languages like C and C++, which both come with a compiler that optimizes the generated code to improve its performance, Python is an interpreted language. There's been a lot of effort put into trying to make Python faster, which the team says usually comes in the form of a "top-down approach," which means taking the vanilla Python implementation and incorporating various optimizations or "just-in-time" compilation techniques—a method by which performance-critical pieces of the code are compiled during execution. These approaches excel at preserving backwards-compatibility, but drastically limit the kinds of speedups you can attain.

"We took more of a bottom-up approach, where we implemented everything from the ground up, which came with limitations, but a lot more flexibility," says Shajii. "So, for example, we can't support certain dynamic features, but we can play with optimizations and other static compilation techniques that you couldn't do starting with the standard Python implementation. That was the key difference—not much effort had been put into a bottom-up approach, where large parts of the Python infrastructure are built from scratch."

The first piece of the puzzle is feeding the compiler a piece of Python code. One of the critical first steps that is performed is called "type

checking," a process where in your program, you figure out the different data types of each variable or function. For example, some could be integers, some could be strings, and some could be floating-point numbers—that's something that regular Python doesn't do. In regular Python, you have to deal with all that information when running the program, which is one of the factors making it so slow. Part of the innovation with Codon is that the tool does this type checking before running the program. That lets the compiler convert the code to native machine code, which avoids all of the overhead that Python has in dealing with data types at runtime.

"Python is the language of choice for domain experts that are not programming experts. If they write a program that gets popular, and many people start using it and run larger and larger datasets, then the lack of performance of Python becomes a critical barrier to success," says Saman Amarasinghe, MIT professor of electrical engineering and computer science and CSAIL principal investigator. "Instead of needing to rewrite the program using a C-implemented library like NumPy or totally rewrite in a language like C, Codon can use the same Python implementation and give the same performance you'll get by rewriting in C. Thus, I believe Codon is the easiest path forward for successful Python applications that have hit a limit due to lack of performance."

Faster than the speed of C

The other piece of the puzzle is the optimizations in the compiler. Working with the genomics plugin, for example, will perform its own set of optimizations that are specific to that computing domain, which involves working with genomic sequences and other biological data, for example. The result is an executable file that runs at the speed of C or C++, or even faster once domain-specific optimizations are applied.

While Codon currently covers a sizable subset of Python, it still needs to

incorporate several dynamic features and expand its Python library coverage. The Codon team is working hard to close the gap with Python even further, and looks forward to releasing several new features over the coming months. Codon is currently publicly available on GitHub.

In addition to Amarasinghe, Shajii wrote the paper alongside Gabriel Ramirez, a former CSAIL student and current Jump Trading software engineer; Jessica Ray, an associate research staff member at MIT Lincoln Laboratory; Bonnie Berger, MIT professor of mathematics and of electrical engineering and computer science and a CSAIL principal investigator; Haris Smajlović, graduate student at the University of Victoria; and Ibrahim Numanagić, a University of Victoria assistant professor in Computer Science and Canada Research Chair.

The research was presented at the ACM SIGPLAN 2023 International Conference on Compiler Construction, and published as part of the *CC 2023: Proceedings of the 32nd ACM SIGPLAN International Conference on Compiler Construction*.

More information: Ariya Shajii et al, Codon: A Compiler for High-Performance Pythonic Applications and DSLs, *CC 2023: Proceedings of the 32nd ACM SIGPLAN International Conference on Compiler Construction* (2023). DOI: 10.1145/3578360.3580275

https://techxplore.com/news/2023-03-python-based-orders-of-magnitude-speedups.html