Technique significantly boosts the speeds of programs that run in the Unix shell

June 7 2022, by Adam Zewe

Researchers have created a technique that boosts the speeds of programs that run in the Unix shell, a ubiquitous programming environment created 50 years ago, by parallelizing the programs. Credit: Christine Daniloff, MIT

Researchers have pioneered a technique that can dramatically accelerate

certain types of computer programs automatically, while ensuring program results remain accurate.

Their system boosts the speeds of programs that run in the Unix shell, a ubiquitous programming environment created 50 years ago that is still widely used today. Their method parallelizes these programs, which means that it splits program components into pieces that can be run simultaneously on multiple computer processors.

This enables programs to execute tasks like web indexing, natural language processing, or analyzing data in a fraction of their original runtime.

"There are so many people who use these types of programs, like data scientists, biologists, engineers, and economists. Now they can automatically accelerate their programs without fear that they will get incorrect results," says Nikos Vasilakis, research scientist in the Computer Science and Artificial Intelligence Laboratory (CSAIL) at MIT.

The system also makes it easy for the programmers who develop tools that data scientists, biologists, engineers, and others use. They don't need to make any special adjustments to their program commands to enable this automatic, error-free parallelization, adds Vasilakis, who chairs a committee of researchers from around the world who have been working on this system for nearly two years.

Vasilakis is senior author of the group's latest research paper, which includes MIT co-author and CSAIL graduate student Tammam Mustafa and will be presented at the USENIX Symposium on Operating Systems Design and Implementation. Co-authors include lead author Konstantinos Kallas, a graduate student at the University of Pennsylvania; Jan Bielak, a student at Warsaw Staszic High School;

Dimitris Karnikis, a software engineer at Aarno Labs; Thurston H.Y. Dang, a former MIT postdoc who is now a software engineer at Google; and Michael Greenberg, assistant professor of computer science at the Stevens Institute of Technology.

A decades-old problem

This new system, known as PaSh, focuses on program, or scripts, that run in the Unix shell. A script is a sequence of commands that instructs a computer to perform a calculation. Correct and automatic parallelization of shell scripts is a thorny problem that researchers have grappled with for decades.

The Unix shell remains popular, in part, because it is the only programming environment that enables one script to be composed of functions written in multiple programming languages. Different programming languages are better suited for specific tasks or types of data; if a developer uses the right language, solving a problem can be much easier.

"People also enjoy developing in different programming languages, so composing all these components into a single program is something that happens very frequently," Vasilakis adds.

While the Unix shell enables multilanguage scripts, its flexible and dynamic structure makes these scripts difficult to parallelize using traditional methods.

Parallelizing a program is usually tricky because some parts of the program are dependent on others. This determines the order in which components must run; get the order wrong and the program fails.

When a program is written in a single language, developers have explicit

information about its features and the language that helps them determine which components can be parallelized. But those tools don't exist for scripts in the Unix shell. Users can't easily see what is happening inside the components or extract information that would aid in parallelization.

A just-in-time solution

To overcome this problem, PaSh uses a preprocessing step that inserts simple annotations onto program components that it thinks could be parallelizable. Then PaSh attempts to parallelize those parts of the script while the program is running, at the exact moment it reaches each component.

This avoids another problem in shell programming—it is impossible to predict the behavior of a program ahead of time.

By parallelizing program components "just in time," the system avoids this issue. It is able to effectively speed up many more components than traditional methods that try to perform parallelization in advance.

Just-in-time parallelization also ensures the accelerated program still returns accurate results. If PaSh arrives at a program component that cannot be parallelized (perhaps it is dependent on a component that has not run yet), it simply runs the original version and avoids causing an error.

"No matter the performance benefits—if you promise to make something run in a second instead of a year—if there is any chance of returning incorrect results, no one is going to use your method," Vasilakis says.

Users don't need to make any modifications to use PaSh; they can just

add the tool to their existing Unix shell and tell their scripts to use it.

Acceleration and accuracy

The researchers tested PaSh on hundreds of scripts, from classical to modern programs, and it did not break a single one. The system was able to run programs six times faster, on average, when compared to unparallelized scripts, and it achieved a maximum speedup of nearly 34 times.

It also boosted the speeds of scripts that other approaches were not able to parallelize.

"Our system is the first that shows this type of fully correct transformation, but there is an indirect benefit, too. The way our system is designed allows other researchers and users in industry to build on top of this work," Vasilakis says.

He is excited to get additional feedback from users and see how they enhance the system. The open-source project joined the Linux Foundation last year, making it widely available for users in industry and academia.

Moving forward, Vasilakis wants to use PaSh to tackle the problem of distribution—dividing a program to run on many computers, rather than many processors within one computer. He is also looking to improve the annotation scheme so it is more user-friendly and can better describe complex program components.

"Unix shell scripts play a key role in data analytics and software engineering tasks. These scripts could run faster by making the diverse programs they invoke utilize the multiple processing units available in modern CPUs. However, the shell's dynamic nature makes it difficult to

devise parallel execution plans ahead of time," says Diomidis Spinellis, a professor of software engineering at Athens University of Economics and Business and professor of software analytics at Delft Technical University, who was not involved with this research. "Through just-in-time analysis, PaSh-JIT succeeds in conquering the shell's dynamic complexity and thus reduces script execution times while maintaining the correctness of the corresponding results."

"As a drop-in replacement for an ordinary shell that orchestrates steps, but does not reorder or split them, PaSh provides a no-hassle way to improve the performance of big data-processing jobs," adds Douglas McIlroy, adjunct professor in the Department of Computer Science at Dartmouth College, who previously led the Computing Techniques Research Department at Bell Laboratories (which was the birthplace of the Unix operating system). "Hand optimization to exploit parallelism must be done at a level for which ordinary programming languages (including shells) don't offer clean abstractions. The resulting code intermixes matters of logic and efficiency. It's hard to read and hard to maintain in the face of evolving requirements. PaSh cleverly steps in at this level, preserving the original logic on the surface while achieving efficiency when the program is run."

More information: Practically Correct, Just-in-Time Shell Script Parallelization: nikos.vasilak.is/p/pash:osdi:2022.pdf

This story is republished courtesy of MIT News (web.mit.edu/newsoffice/), a popular site that covers news about MIT research, innovation and teaching.

Provided by Massachusetts Institute of Technology

Citation: Technique significantly boosts the speeds of programs that run in the Unix shell (2022,

June 7) retrieved 20 March 2024 from https://techxplore.com/news/2022-06-technique-significantly-boosts-unix-shell.html