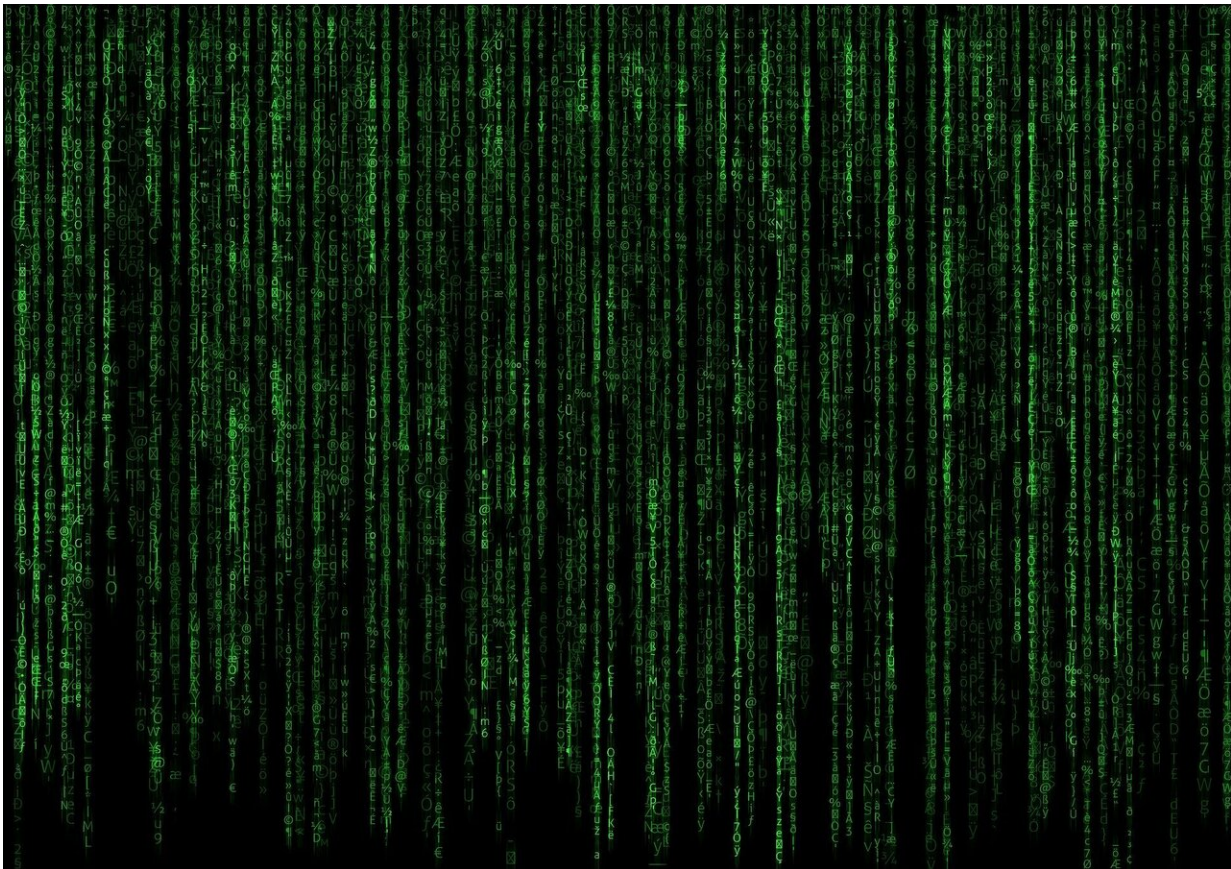# New programming language and tool ensures code will compute as intended

July 30 2020, by Daniel Tkacik

Credit: Pixabay/CC0 Public Domain

Not long ago, people using Microsoft Word would check for spelling errors by specifically telling the software to run "Spell Check." The

check took a few seconds to do, and users could then go in and fix their typos. Nowadays, Spell Check runs automatically as users write—as I write this story.

Microsoft Word and its constant running of Spell Check is a basic example of "concurrent" programming—a form of computing in which an executable runs simultaneously with other programs and computations. Most programs today are concurrent programs, ranging from your operating system to the many applications, from word processing to web browsing, that people use on a daily basis.

"When you have multiple things happening at the same time, you need some way of coordinating between them to make sure they're not stomping on each other," says CyLab's Bryan Parno, an associate professor in Electrical and Computer Engineering and the Computer Science Department. "Historically, this has been a very buggy process."

Parno and a team of researchers recently published a new coding language and tool for high-performance concurrent programs that ensures that programs are provably-correct—that is, that the code is mathematically proven to compute correctly. The language and tool, named Armada, was presented at this year's Conference on Programming Language Design and Implementation, and the paper received a Distinguished Paper award.

"What's novel about Armada is that it's designed to be extremely flexible so you can write the code the way you want so it'll go as fast as it can," says Parno. "But you'll still get strong assurance that it's going to do the right thing and not mess anything up on the back end."

Parno likens the complexity of concurrent programs—and their susceptibility to bugs—to an auction. Typically, one auctioneer receives bids from lots of people. It may take a long time to get to the highest bid

with so many people and one auctioneer. If you split everyone up into, say, ten rooms, each with its own auctioneer, that would speed things up, but it would be very difficult for the auctioneers to stay coordinated; there would be lots of room for error.

"There needs to be a way for all of those auctioneers to talk to each other while simultaneously working towards the highest bid amongst all of the rooms," says Parno. "It can get very complicated, which is why you don't usually see auctions run in this way."

Parno believes that Armada will benefit anyone writing concurrent programs, which span a huge range in applications.

"From payroll systems to hospital records to any kind of e-commerce—they are all backed by databases, and databases are always going to be backed by concurrent software," says Parno. "Aside from simple programs, these days almost everything has some sort of concurrency to it."

Provided by Carnegie Mellon University