Lab for Automated Reasoning and Analysis

Viktor Kunčak

lara.epfl.ch
lara.epfl.ch/w/fv
wish

human effort

Command\(\text{(program)}\)

compilation

11011001 01011101
11011001 01011101
11011001 01011101
11011001 01011101
This method shows cat picture, then sends your phone no. to zxq39j.com

Specifications and code in Scala

NL generation (talk to Nataliia Stulova)
Leon demo

https://leon.epfl.ch
Testing vs Verification

From http://stainless.epfl.ch/faq.html

**Testing** can establish the **presence of errors**, but **not their absence**. Basic forms of testing can be easy to deploy, because running a program on a given input is a minimum requirement for software, but such testing is extremely limited. Suppose that we wish to test whether a program running on a smartphone performs **multiplication of two machine numbers** correctly. If we could check one test per nanosecond, we would need $10^{31}$ **billions of years** to enumerate all cases. This also illustrates how minuscule of a region of space around a given test a fuzzer can ever explore. Formal software verification can cover all these cases with a single proof because it uses algebraic properties that are independent of the size of the data that software manipulates.
“What is Formal Verification? - Video by Galois, a formal methods company”

https://www.youtube.com/watch?v=-CTNS2D-kbY
Stainless

Stainless is a tool for verifying Scala programs. Stainless is developed by LARA at EPFL’s School of Computer and Communication Sciences. Stainless can verify that your program is correct for all inputs, it can report inputs for which your program fails when they exist, and it can prove that functions do not loop. Using Stainless before running or deploying the software can eliminate crashes, logical errors, security flaws, and other defects.

https://github.com/epfl-lara/stainless
Smart contracts, with https://www.c4dt.org/

See http://stainless.epfl.ch/smart-contracts.html

Axoni:

https://youtu.be/xO7B4WYKVZE?t=243

https://medium.com/axoni/axlang-formally-verifiable-smart-contracts-for-the-ethereum-ecosystem-6201203be4e8
Verification of distributed systems

Formal model of Scala actors

Starting collaboration with Interchain foundation on verification of parts of their blockchain infrastructure

Other users: IOHK
More information

Main paper for current research:

**System FR: Formalized Foundations for the Stainless Verifier**

http://lara.epfl.ch/~kuncak/papers/HamzaETAL19SystemFR.html

**Formal Verification** MSc/EDIC course:

https://lara.epfl.ch/fv