Stainless Verification System Tutorial

Viktor Kunčak and Jad Hamza

FMCAD 2021, 19 October 2021

https://stainless.epfl.ch

For More Information

ASPLOS'22 conference at EPFL:
`https://asplos-conference.org/`

Scala programming language (developed at EPFL):
`https://www.scala-lang.org/`

Stainless verifier for Scala subset:
`https://stainless.epfl.ch`

Information for This Tutorial

`https://github.com/epfl-lara/fmcad2021tutorial/`

First example: MaxBug.scala

Example: Lists of Differences

$$\begin{tikzcd} \begin{array}{|c|c|c|}\hline 100 & 101 & 95 \\ \hline\end{array} \arrow[r, shift left, "\textit{diffs}"] & \begin{array}{|c|c|c|}\hline 100 & +1 & \text{-6} \\ \hline\end{array} \arrow[l, shift left, "\textit{undiff}"] \end{tikzcd}$$

monthly account balances

initial value, monthly gains

More Tutorial Examples

Interfaces: purely functional amortized queue

Modeling: lists of bits

Termination: binary search

Other Reported Case Studies

Verified 14k lines of Scala code

- 5.8k verification conditions
- 6.5 minutes

Verified examples include

- Monad laws. Sorting algorithms. Graph reachability. Dynamic programming.
- Complex functional data structures (amortized queues)
- Lazy and concurrent data-structures. Basic distributed algorithms.
- LZW Compression. Model of key server. Smart contracts.
- Number theory properties (Gödel numbering).

Uses of Stainless

Teaching “Formal Verification” course at EPFL

Collaboration with Informal.Systems:

- Tendermint Light Client in Scala
- Rust front end for Stainless
  (Georg S Schmid, Yann Bolliger, Romain Ruetschi)

Joint Project with Ateleris

- verifying parts of STIX Flight Software by ESA (C code generator)

Foundations of Stainless: System FR

A type system foundation for Stainless verifier:

- Extends System F with refinements and recursive types
- Dependently typed system with Π, Σ, $\bigcap, \bigcup, \{\_\}$
- Type refinements capture preconditions, postconditions
- Support for contravariant data types and streams: indices and $\bigcap$
- Type checking ensures termination using measures (not restricted to structural recursion for function definitions)

Proven correct by interpreting types as sets of untyped terms.

Proofs formalized in (20k lines of) Coq by Jad Hamza.

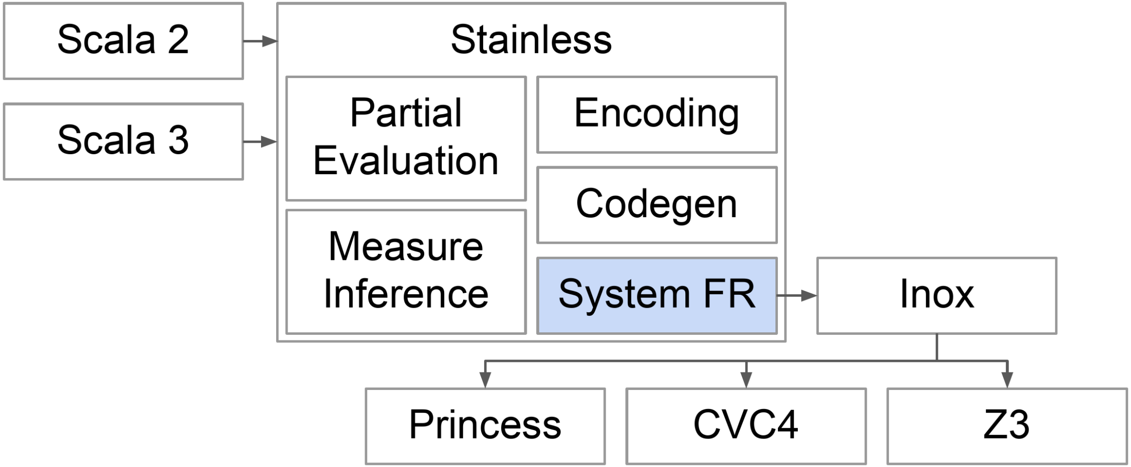
Verification condition generator adapted to follow the type system.

J. Hamza, N. Voirol, V. Kuncak

System FR: Formalized Foundations for the Stainless Verifier

Proc. ACM Program. Lang. OOPSLA. 2019.

Under the Hood of Stainless



Not shown: type encoding, full-imperative (VMCAI'22), C code generation, equivivalence checking of functions

Inspiration

ACL2: `https://www.cs.utexas.edu/users/moore/acl2/`

Alloy Analyzer: `https://alloytools.org/`

Rustan Leino's languages, including
`https://github.com/dafny-lang/dafny`

Coq, F*, Isabelle, Lean, PlusCal and TLA+

Conclusions: Stainless

A software verifier for a widely used functional language.

`https://stainless.epfl.ch`