Sound Reasoning about Integral Data Types with a Reusable SMT Solver Interface

Régis Blanc Viktor Kuncak

Laboratory for Automated Reasoning and Analysis
École Polytechnique Fédérale de Lausanne

June 13, 2015

The Leon Verification System

- Verifier for the Scala language.
- Support a well-defined subset of Scala.
  - A functional core language.
  - Many imperative extensions.
  - Some ways to express non-determinism.
- Complete for finding counterexamples.
- Big project from the LARA group at EPFL, with contributions from many present (and past) members.

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Contracts

Specifications can be defined using contracts.

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Contracts

Specifications can be defined using contracts.

- Postconditions

```
def abs(n: Int): Int = {
  if(n <= 0) -n else n
} ensuring(res => res >= 0)
```

Contracts

Specifications can be defined using contracts.

- Postconditions

```
def abs(n: Int): Int = {
  if(n <= 0) -n else n
} ensuring(res => res >= 0)
```

- Preconditions

```
def fact(n: Int): Int = {
  require(n >= 0)
  if(n == 0) 1 else n * fact(n-1)
}
```

Contracts

Specifications can be defined using contracts.

- Postconditions

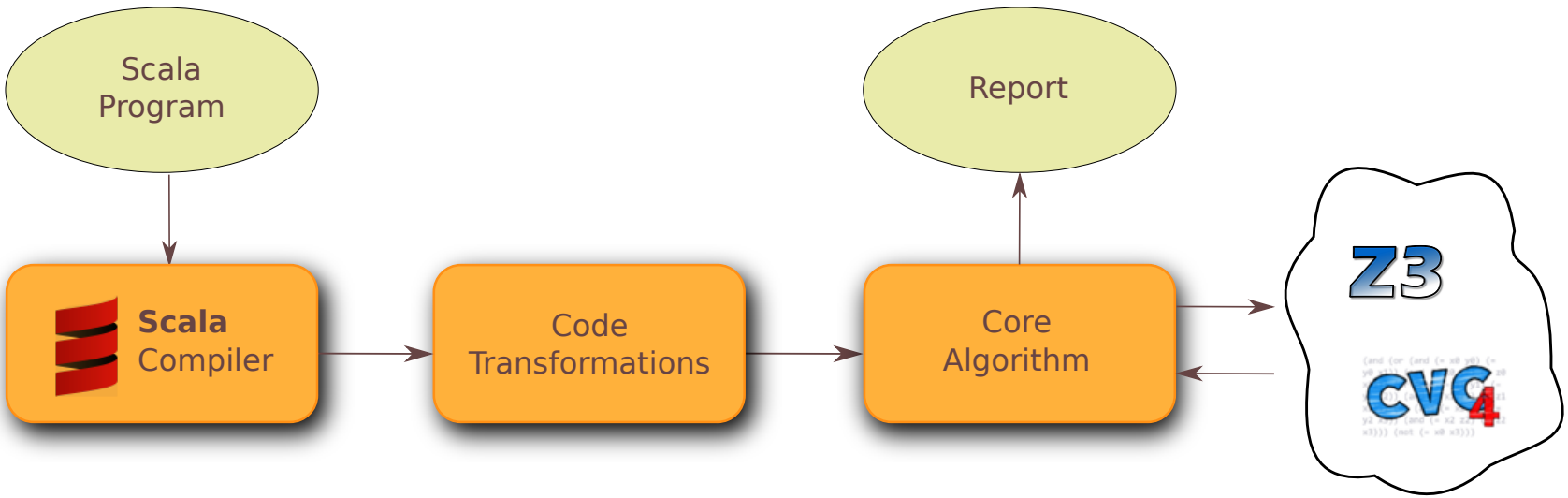
```
def abs(n: Int): Int = {
  if(n <= 0) -n else n
} ensuring(res => res >= 0)
```

- Preconditions

```
def fact(n: Int): Int = {
  require(n >= 0)
  if(n == 0) 1 else n * fact(n-1)
}
```

The implementation and specification languages are the same.

Architecture of Leon



Demo

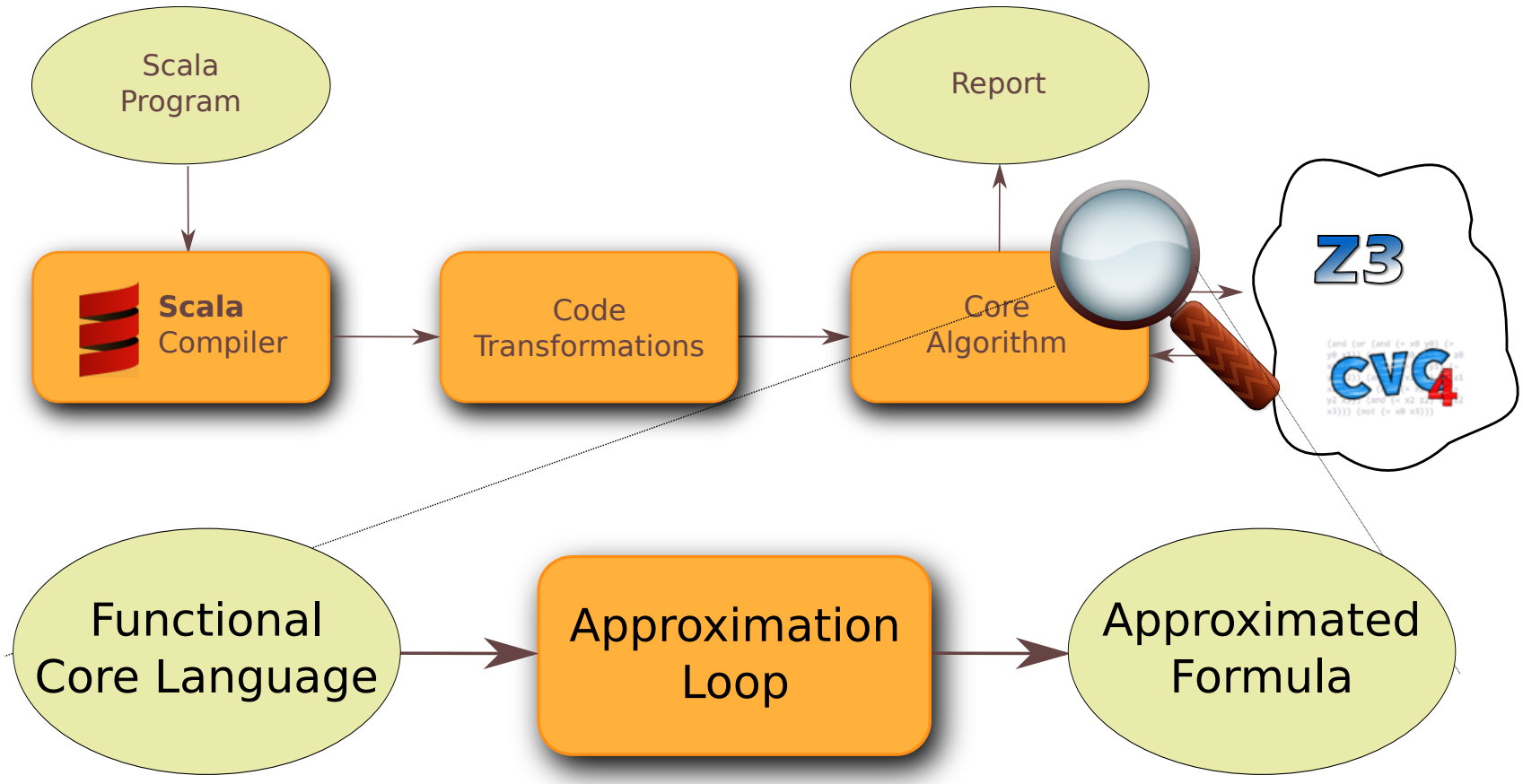
Int and BigInt

Int	Primitive integer type: bit-vector semantics
BigInt	Library type: mathematical integer semantics

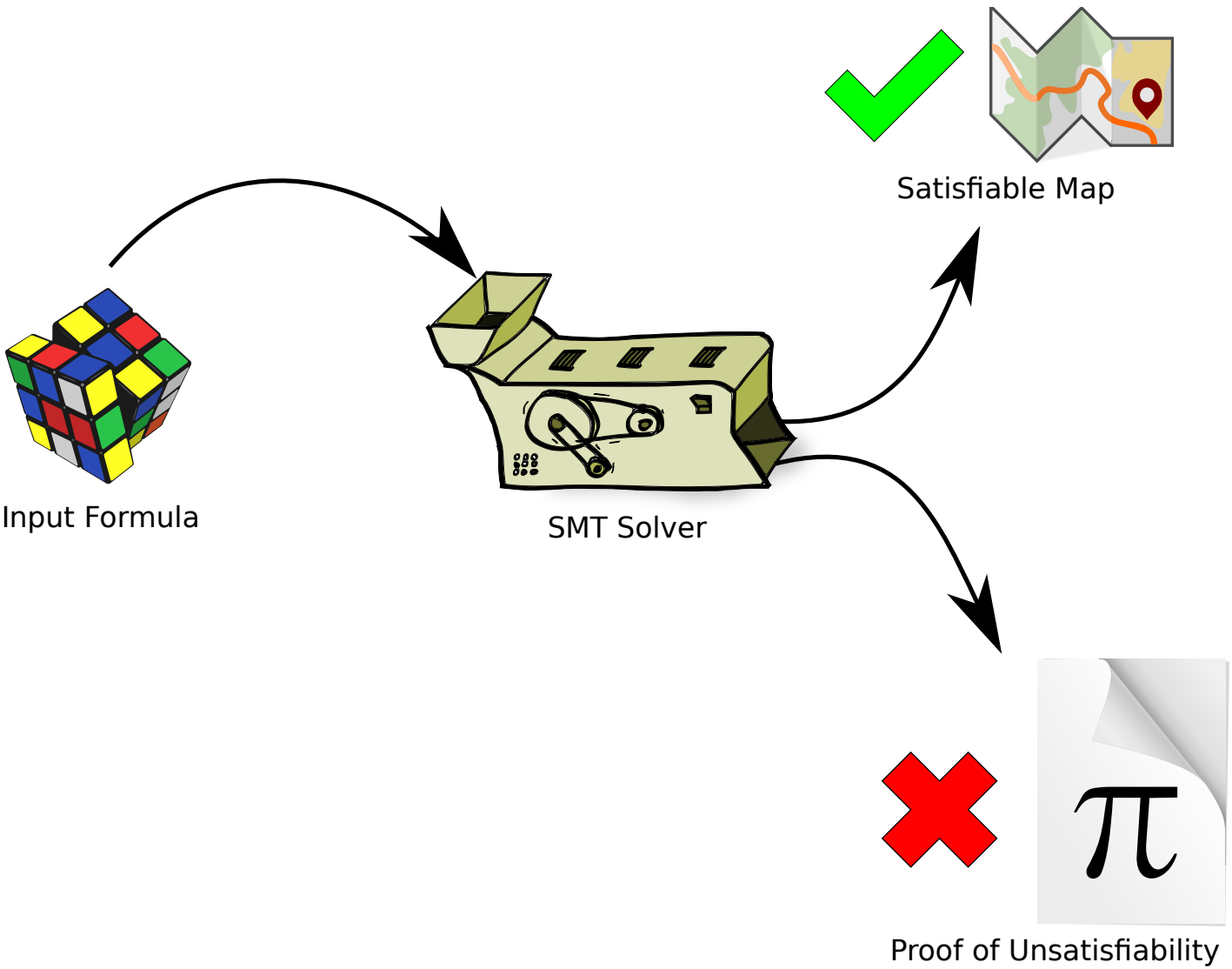
- Mathematical reasoning is usually easier with integers.
- Most programs use Int instead of BigInt.
- Easy to ignore the bounded nature of Int.

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

A Closer Look at Leon Unrolling



SMT Solver


Satisfiable Map
Input Formula
SMT Solver
Proof of Unsatisfiability

SMT Solver Theories

Any mathematical theory with a well defined axiomatization.

Of interest to programming languages:

- **Int** Mathematical, unbounded, integers: Corresponds to Scala `BigInt`.
- **BitVector** Fixed, finite-size, bit-vectors: Correspond to Scala `Int`.
- **ADT** Algebraic data types. Models a subset of case classes functionalities.
- **Array** Map from one type to another. Models Scala `Array` and `Map`.
- **UF** Uninterpreted functions. Helps with abstractions.

Many Alternative Implementations

- With so many theories, support varies from solver to solver.
- State-of-the-art algorithms: ongoing research.
- Good to remain as solver-agnostic as possible.

SMT-LIB Interface

- With so many theories, support varies from solver to solver.
- State-of-the-art algorithms: ongoing research.
- Good to remain as solver-agnostic as possible.

"**SMT-LIB** is an international initiative aimed at facilitating research and development in Satisfiability Modulo Theories (SMT)"

`http://www.smtlib.org`

- Text-based format to standardize communication with SMT solvers.
- Similar to a programming language, but declarative. Syntax based on Lisp.
- Large library of benchmarks. Enable organization of the annual SMT-COMP competition.
- Good support in existing solvers, including Z3 and CVC4.

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Leon: Integration with SMT Solvers

- Leon abstracts away the backend solver.
- One of the implementation generate SMT-LIB commands: get many different solvers essentially for "free"
- The SMT-LIB interface is exposed in a stand-alone Scala module.

Leon: Integration with SMT Solvers

- Leon abstracts away the backend solver.
- One of the implementation generate SMT-LIB commands: get many different solvers essentially for "free"
- The SMT-LIB interface is exposed in a stand-alone Scala module.

`scala-smtlib` is a lightweight abstraction on top of the SMT-LIB standard.

`https://github.com/regb/scala-smtlib`

- Simple, type-safe, communication with SMT solvers.
- Support for the latest SMT-LIB 2.5 standard.
- Include a fully complient parser (not used in Leon) that can help building applications with SMT-LIB as input.

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Conclusion

Extensions to the Leon system

- Sound reasoning about integral data types: `Int` and `BigInt`.
- Solver-agnostic backend with the help of an open-source SMT-LIB Scala library.

Work in progress

Optimization of BigInt

- When writing program, `BigInt` is often closer to the expected meaning than `Int`.
- However can often be two order of magnitude slower.
- Why not proving bounds statically on code using `BigInt` and compiling to equivalent and faster `Int`.

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE