A Verified Certificate Checker for Finite-Precision Error Bounds in Coq and HOL4

FMCAD 2018, 02 November 2018

Heiko Becker, Nikita Zyuzin, Eva Darulova, Magnus O. Myreen

Raphael Monat, Anthony Fox

MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS

CHALMERS UNIVERSITY OF TECHNOLOGY

ENS DE LYON

ARM

Finite-Precision Computations have errors

$0.2 + 0.1$ $\quad\quad$ $\widetilde{0.2} \mathbin{\tilde{+}} \widetilde{0.1}$

$0.3 \neq 0.30000000000000004$

roundoff error

Does it matter?

Finite-Precision Computations have errors

$f(x) = 0.2 + x$

$\tilde{f}(\tilde{x}) = \widetilde{0.2} \mathbin{\tilde{+}} \tilde{x}$

roundoff error

$$\boxed{\max_{x \in [a,b]} | f(x) - \tilde{f}(\tilde{x}) | \leq \varepsilon}$$

Did Daisy compute
a correct roundoff error ε?

$$\max_{x \in [a,b]} |\, f(x) - \tilde{f}(\tilde{x}) \,| \leq \varepsilon$$

f: real valued function
$P(x)$: input constraints

$\tilde{f}$: finite-precision function
ε: roundoff error

>8k Lines of Scala Code

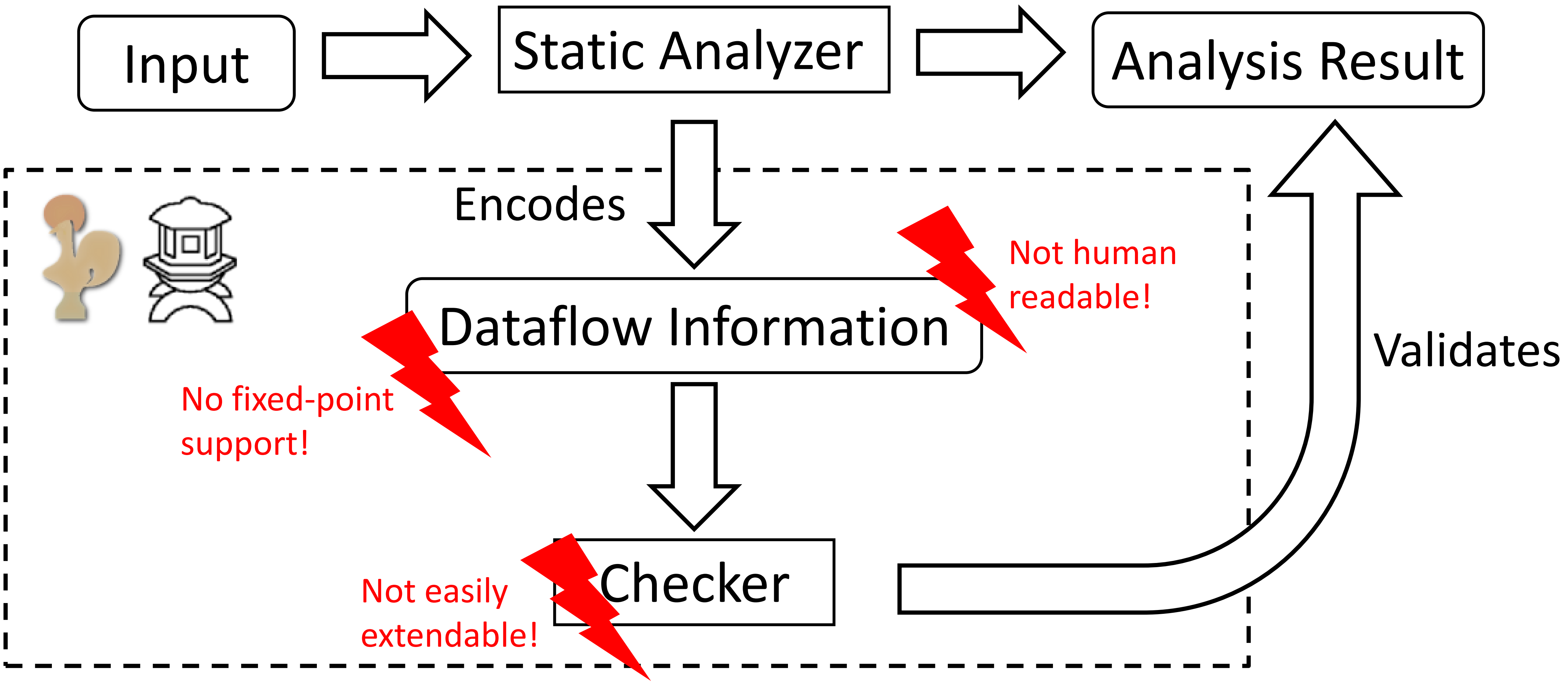
Did Daisy compute
a correct roundoff error ε?

$$\max_{x \in [a,b]} |\, f(x) - \tilde{f}(\tilde{x}) \,| \leq \varepsilon$$

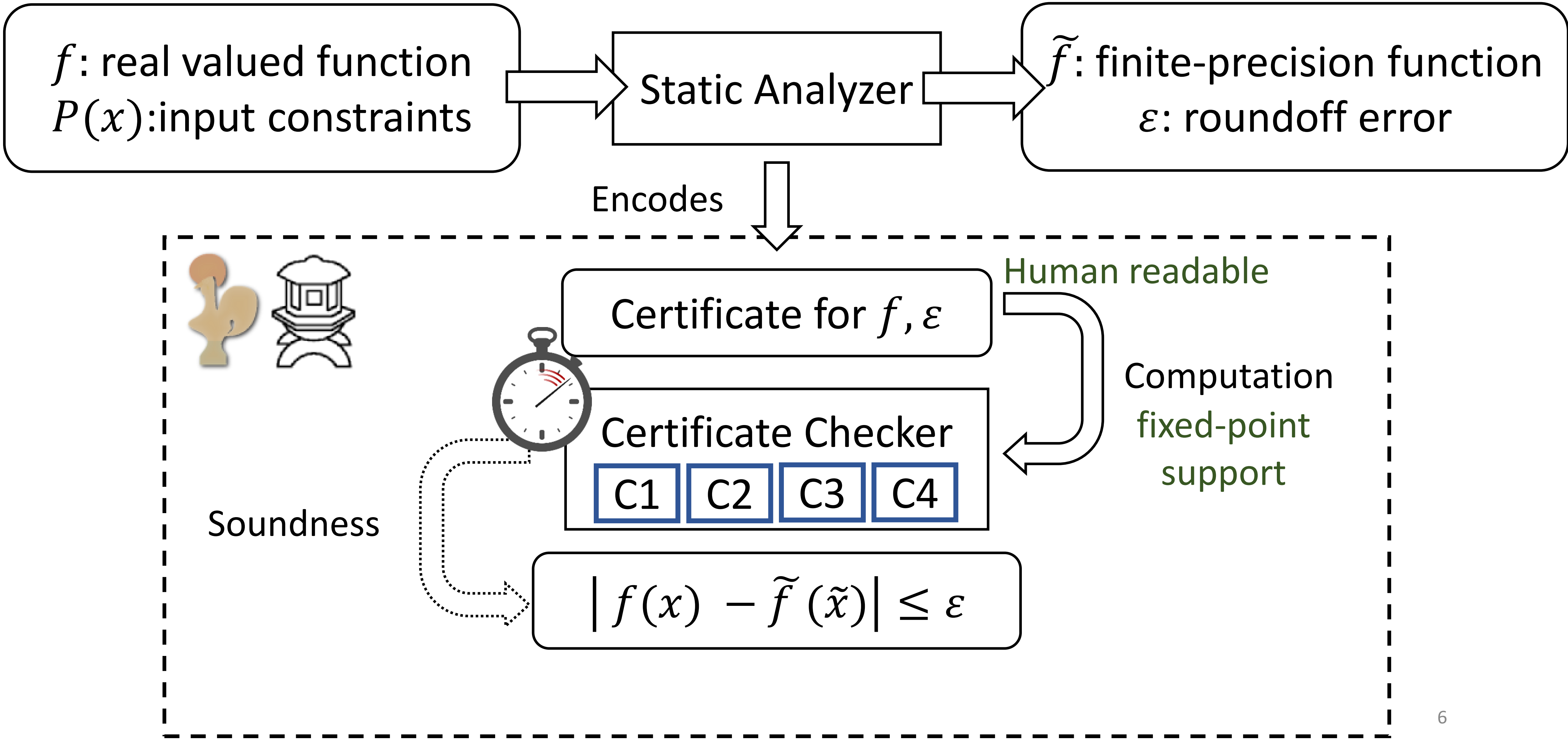
Verification

Validation

Certificate Checking



FloVer: A Checker For Finite-Precision Error Bounds



Contributions

FloVer, a certificate checker for finite-precision error bounds

- checks floating-point and fixed-point error bounds

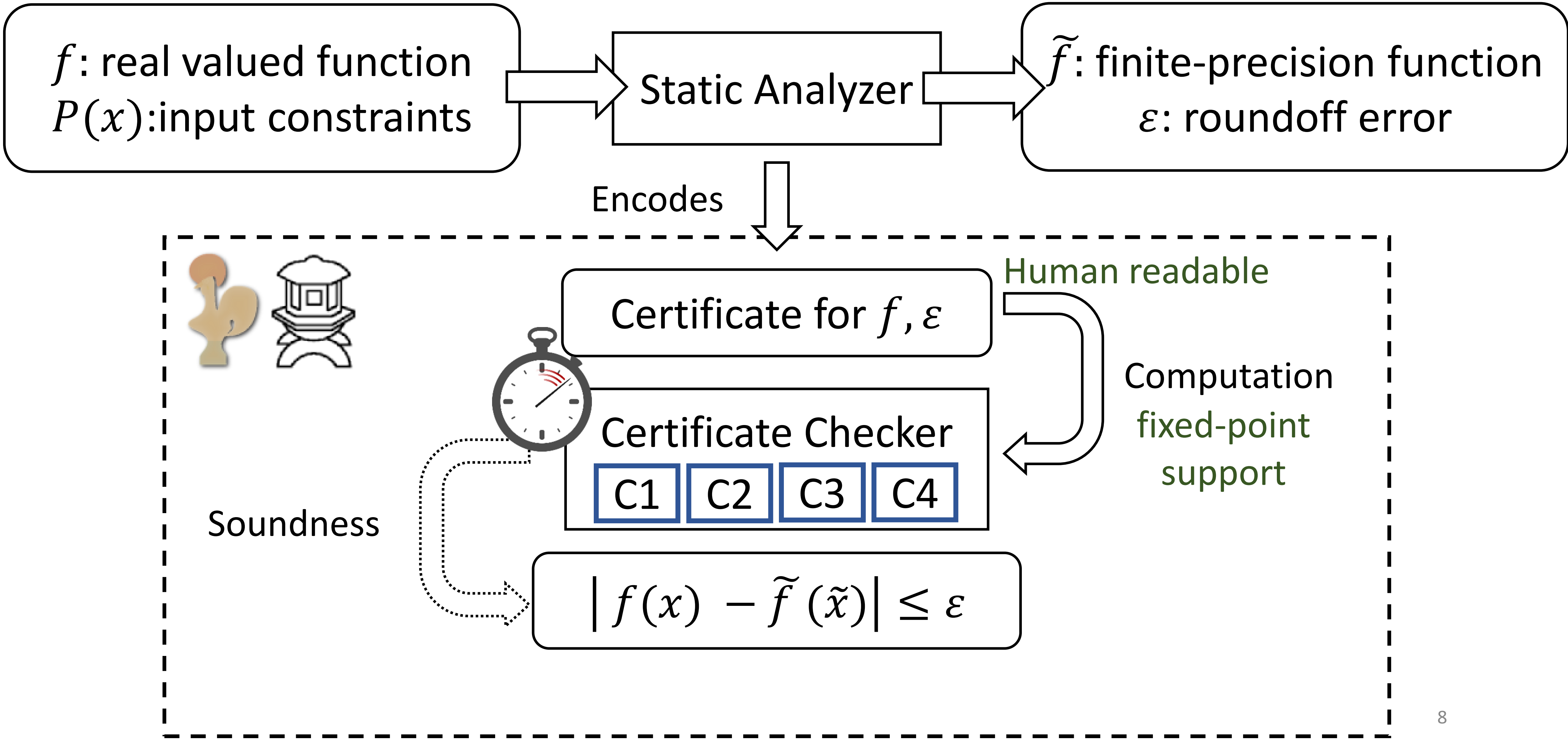
  →soundness proof with respect to IEEE754 semantics

- build modular with separate validators

  →uses interval and affine arithmetic

- supports arithmetic (+,-,*,/), let-bindings and fused-multiply-add
- supports mixed-precision (different types for operations)
- runs fast with verified extracted binary

FloVer: A Checker For Finite-Precision Error Bounds



f: real valued function
$P(x)$: input constraints

Certificate
f: real valued function
$P(x)$: input constraints
R: range analysis result
E: error analysis result

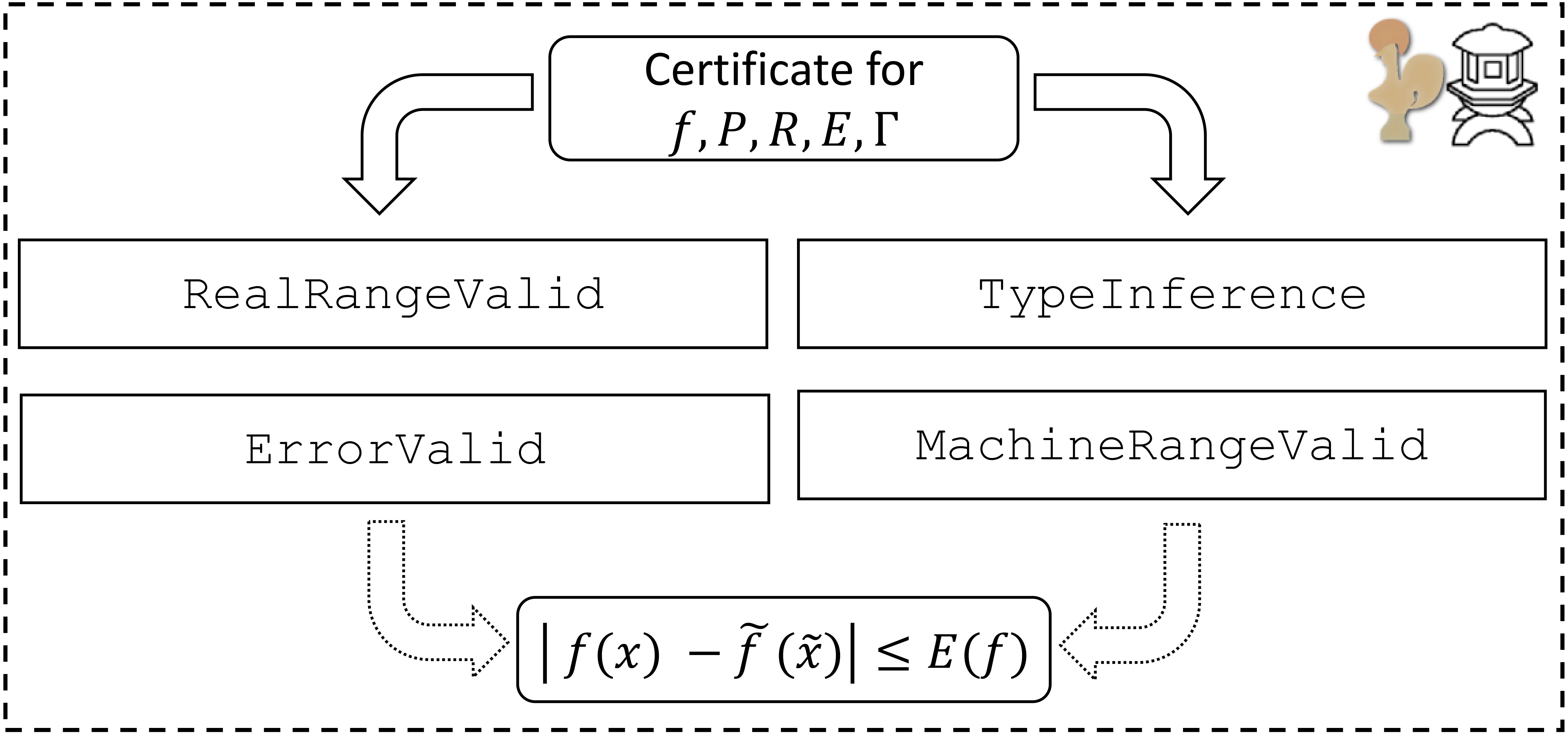
Daisy

real-valued
range analysis R

finite-precision
error analysis E

$\tilde{f}$: finite-precision function
ε: roundoff error

FloVer: A Checker For Finite-Precision Error Bounds



Checking Range Analysis

$$\texttt{RealRangeValid}(0.2, P, R) = 0.2 \subseteq R(0.2)$$
$$\texttt{RealRangeValid}(x, P, R) = P(x) \subseteq R(x)$$
$$\texttt{RealRangeValid}(0.2 + x, P, R) = R(0.2) +^{RA} R\ (x) \subseteq R(0.2 + x)$$

Certificate

$f\ (x) = 0.2 + x$

$P(x)$: input constraints

R: range analysis

E: error analysis

Γ: variable types

Soundness theorem:

$$\texttt{RealRangeValid}(f, P, R) \wedge$$
$$\forall x, (E\ x) \in P\ (x) \Longrightarrow$$
$$(f, E) \downarrow v \wedge v \in R(f)$$

FloVer: A Checker For Finite-Precision Error Bounds

Certificate for
f, P, R, E, Γ

`RealRangeValid` ✓

`TypeInference`

`ErrorValid`

`MachineRangeValid`

$$|f(x) - \tilde{f}(\tilde{x})| \leq E(f)$$

An Abstraction for Floating-Point Computations

$$\tilde{f}(\tilde{x}) = \widetilde{0.2} \,\tilde{+}\, \tilde{x}$$

$$= \left(\widetilde{0.2} + \tilde{x}\right) * (1 + \delta)$$

$$= \left(\widetilde{0.2} + \tilde{x}\right) + \underbrace{\left(\widetilde{0.2} + \tilde{x}\right) * \delta}_{\text{error for + operation}}$$

more errors contributed by operands

IEEE754 abstraction

$$\exists\, \delta .\, a \,\tilde{+}\, b = (a + b) * (1 + \delta)$$

where $|\delta| \leq \varepsilon$

$$\tilde{x} = x * (1 + \delta)$$

Computing a Roundoff Error

We want to bound

$$\left|(0.2 + x) - \left(\widetilde{0.2} \mathbin{\tilde{+}} \tilde{x}\right)\right|$$

IEEE754 abstraction

$$\boxed{\exists \delta .\, a \mathbin{\tilde{+}} b = (a + b) * (1 + \delta)}$$

where $|\delta| \leq \varepsilon$

$$\leq \left|\left(0.2 - \widetilde{0.2}\right) + (x - \tilde{x}) + \left(\widetilde{0.2} + \tilde{x}\right) * \delta\right|$$

$$\leq \underbrace{err_1 + err_2}_{\text{propagation error}} + \underbrace{\left|\left(\widetilde{0.2} + \tilde{x}\right) * \delta\right|}_{\text{roundoff error } (e_{new})}$$

operand errors:

$$\left|0.2 - \widetilde{0.2}\right| \leq err_1$$

$$|x - \tilde{x}| \leq err_2$$

Checking Roundoff Error Bounds

$$\texttt{ErrorValid}(0.2, R, \Gamma, E) = 0.2 * \varepsilon \leq E(0.2)$$

$$\texttt{ErrorValid}(x, R, \Gamma, E) = maxAbs\ (R(x)) * \varepsilon \leq E(x)$$

$$\texttt{ErrorValid}(0.2 + x, R, \Gamma, E) = E(0.2) + E(x) + e_{new} \leq E(0.2 + x)$$

Certificate

$f\ (x) = 0.2 + x$

$P(x)$: input constraints

R: range analysis

E: error analysis

Γ: variable types

Soundness Theorem:

$$\textbf{ErrorValid}(f, E, R) \land$$

$$\textbf{RangeValid}(f, P, R) \land$$

$$(\forall x, E\ x \in P(x)) \Longrightarrow$$

$$(f, E) \downarrow v \land \left(\tilde{f}, E\right) \downarrow \tilde{v} \land\ |v - \tilde{v}| \leq E\ (f)$$

Supported abstract domains

Interval Arithmetic

e $\in [a, b]$ where $a \leq e \leq b$

non-relational, easy to implement

Affine Arithmetic

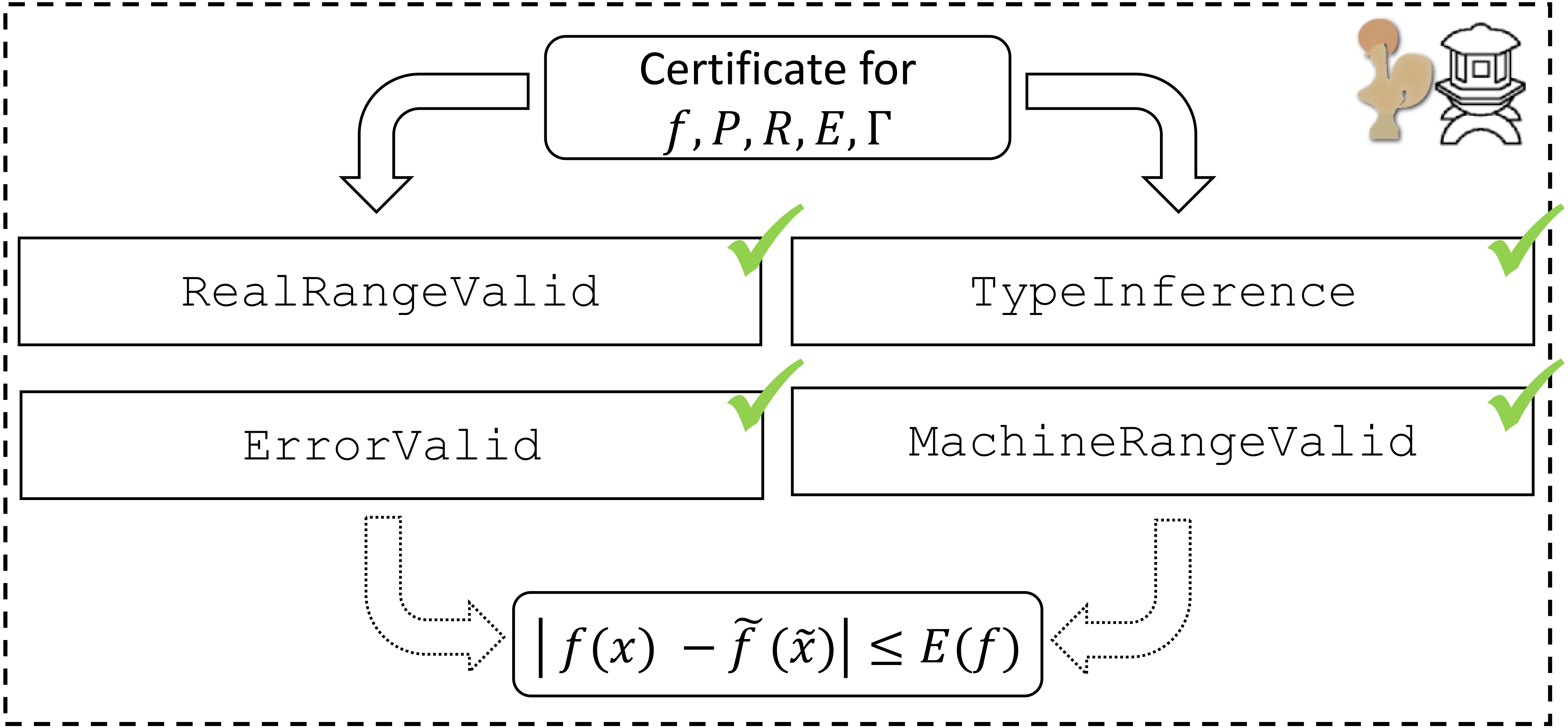
e $\in x_0 + x_1 * \varepsilon_1 + \cdots + x_n * \varepsilon_n$
where $\mathrm{x}_0 + \sum_{i=1} x_i * -1 \leq e \leq x_0 + \sum_{i=1}^{n} x_i$

relational, complex to implement
Track linear, approx. non-linear

Monotonicity of operations:

$$e_1 \in rep_1 \wedge e_2 \in rep_2 \Longrightarrow e_1 \circ e_2 \in (rep_1 \circ^{RA} rep_2)$$

FloVer: A Checker For Finite-Precision Error Bounds



Type inference in FloVer

uniform precision

mixed precision

Annotate variables and constants

Unused

Certificate

f: real valued function

$P(x)$: input constraints

R: range analysis

E: error analysis

Γ: variable types

But we want $\tilde{f}$ to be

```
f(x:single) =
  let y:double = x + 3.0
  in y * 1.3
```

result type ?

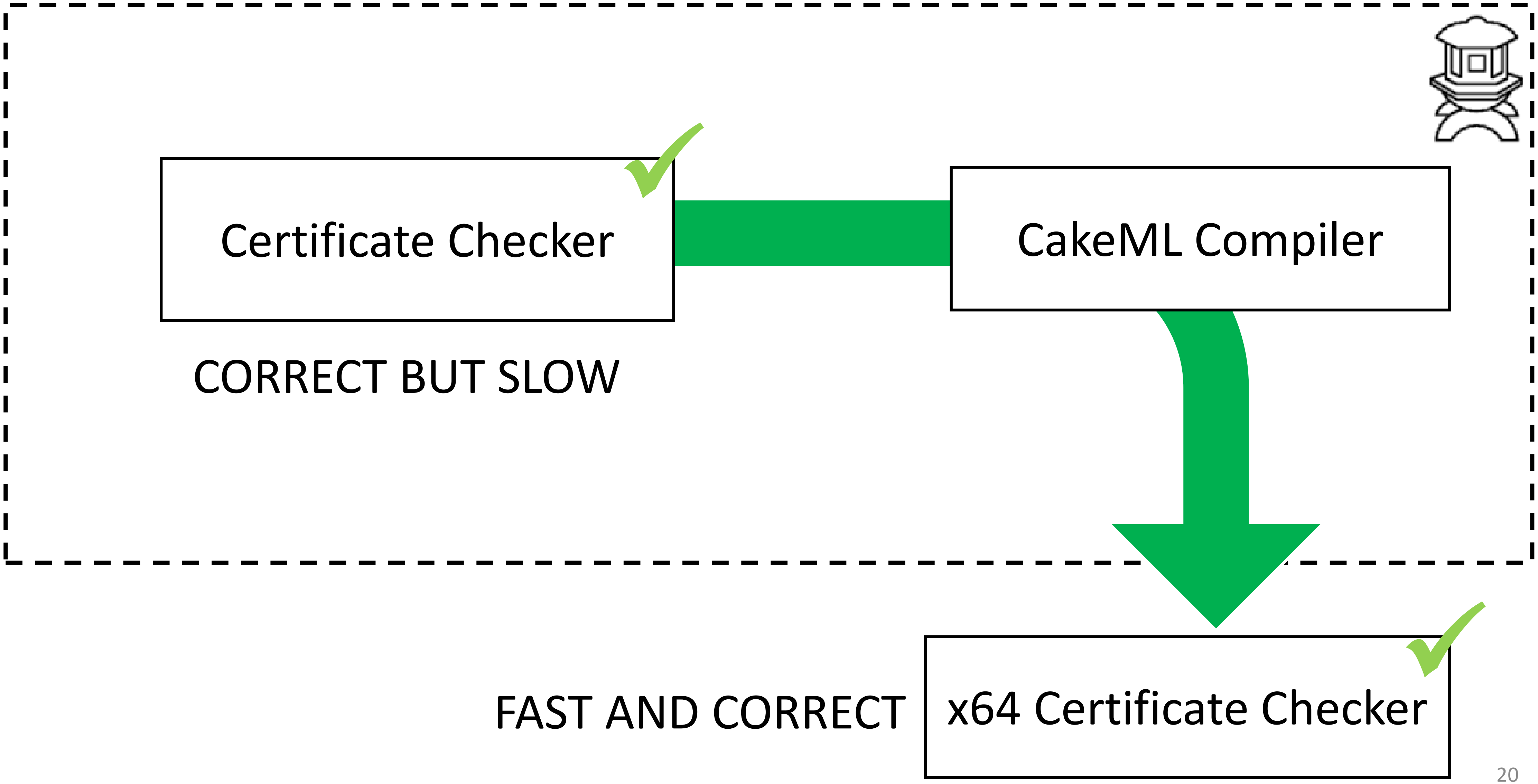
`TypeInference`(f, Γ) infers a global type map M_T for f

Soundness of FloVer

Let f be a real-valued function, E a real-valued environment, $\tilde{E}$ its finite-precision counterpart, P a precondition constraining the free variables of f, Γ a map from all free variables of f to a precision, R a range analysis result, Γ a type-map and E an error analysis result. Then

$$
\begin{gathered}
E \sim_{(E,\mathcal{V},\mathcal{D},\Gamma)} \tilde{E} \land \\
\texttt{TypeInference}(\Gamma, f) = \Gamma \land \texttt{RealRangeValid}(f, P, R) \land \\
\texttt{MachineRangeValid}(f, \Gamma, R, E) \land \\
\texttt{ErrorValid}(f, \Gamma, R, E) \Rightarrow \\
\exists\, v\, \tilde{v}_1\, m_1 .\ (f, E, \Gamma) \Downarrow (v, \infty) \land (\tilde{f}, \tilde{E}, \Gamma) \Downarrow (\tilde{v}_1, m_1) \land \\
(\forall \tilde{v}_2\, m_2 .\ (\tilde{f}, \tilde{E}, \Gamma) \Downarrow (v_2, m_2) \Rightarrow |v - \tilde{v}_2| \leq E(f))
\end{gathered}
$$

Extraction using the CakeML compiler toolchain



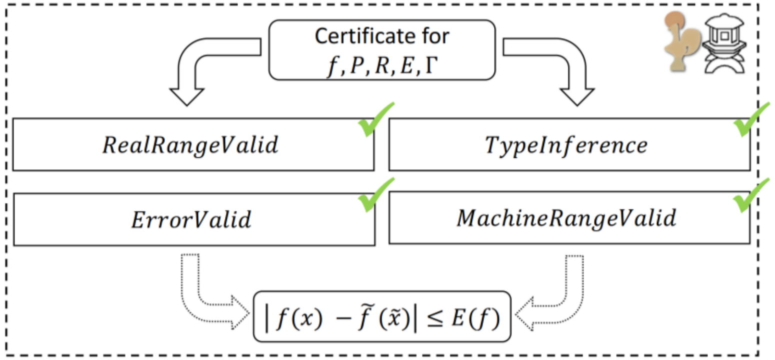
Experiments

Measure end-to-end running times for **18 benchmarks** from literature

Benchmark	# ops.	Daisy	Coq		HOL4	Binary Interval	
			Interval	Affine	Interval	CakeML	OCaml
ballBeam	7	4.62	3.50	3.26	89.04	<0.01	0.02
doppler	17	4.86	5.28	12.21	610.67	0.05	0.02
bspline	28	4.21	4.61	4.07	298.44	0.03	0.08
…	…	…	…	…	…	…	…
traincar1	36	4.85	10.87	9.84	932.93	0.07	0.11
floudas	99	7.76	13.99	12.76	565.68	0.14	0.27
Traincar4	269	10.60	116.94	115.38	17429.30	1.10	0.77

Complexity

FloVer



- Supporting **floating-point** and **fixed-point** arithmetic
- **Runs fully automatically** in Coq and HOL4
- Mixed-precision
- Extract a **verified binary** with CakeML
- **Proven sound** for IEEE754 floating-point semantics
- Code on https://gitlab.mpi-sws.org/AVA/Flover

Questions?