Computer Arithmetic and Formal Proofs

Guillaume Melquiond

Inria, Université Paris-Saclay

2019-01-31 & 2019-02-02

Motivation 1: Formal Verification of Math Libraries

Cody & Waite's algorithm (1980)

```
double cw_exp(double x)
{
  // exception handling and constants
  ...
  // argument reduction
  double k = nearbyint(x * InvLog2);
  double t = x - k * Log2h - k * Log2l;
  // polynomial approximation
  double t2 = t * t;
  double p = 0.25 + t2 * (p1 + t2 * p2);
  double q = 0.5  + t2 * (q1 + t2 * q2);
  double f = t * (p / (q - t * p)) + 0.5;
  // result reconstruction
  return ldexp(f, (int)k + 1);
}
```

This floating-point function accurately approximates exp.

Motivation 2: Numerical Integrals in Modern Math Proofs

Double bubbles minimize (2000)

The proof parameterizes the space of possible solutions by a two-dimensional rectangle [...]. This rectangle is subdivided into 15,016 smaller rectangles which are investigated by calculations involving a total of 51,256 numerical integrals.

Motivation 2: Numerical Integrals in Modern Math Proofs

Double bubbles minimize (2000)

The proof parameterizes the space of possible solutions by a two-dimensional rectangle [...]. This rectangle is subdivided into 15,016 smaller rectangles which are investigated by calculations involving a total of 51,256 numerical integrals.

Major arcs for Goldbach's problem (2013)

$$\int_{-\infty}^{\infty} \frac{(0.5 \cdot \log(\tau^2 + 2.25) + 4.1396 + \log \pi)^2}{0.25 + \tau^2} d\tau$$

We compute the last integral numerically (from -100,000 to 100,000).

mathoverflow

Questions Tags Users Badges Una

Rigorous numerical integration

9

2

I need to evaluate some (one-variable) integrals that neither SAGE nor Mathematica can do symbolically. As far as I can tell, I have two options:

(a) Use GSL (via SAGE), Maxima or Mathematica to do numerical integration. This is really a non-option, since, if I understand correctly, the "error bound" they give is not really a guarantee.

(b) Cobble together my own programs using the trapezoidal rule, Simpson's rule, etc., and get rigorous error bounds using bounds I have for the second (or fourth, or what have you) derivative of the function I am integrating. This is what I have been doing.

Is there a third option? Is there standard software that does (b) for me?

na.numerical-analysis

share cite improve this question

asked Mar 5 '13 at 23:03

H A Helfgott

3,141 • 17 • 61

Outline

Outline

Computational Reflection

Example (How to prove $1 + 2 + 3 + 4 = 10$?)

```
Goal (1 + 2 + 3 + 4 = 10)%R.
Proof.
rewrite <- 3!plus_IZR.
apply eq_refl.
Qed.
```

Computational Reflection: Interpretation

```
Inductive expr :=
  | C (n : Z)
  | P (u v : expr).

Fixpoint interp (e : expr) : R :=
  match e with
  | C n => IZR n
  | P u v => Rplus (interp u) (interp v)
  end.
```

Computational Reflection: Interpretation

```
Inductive expr :=
  | C (n : Z)
  | P (u v : expr).

Fixpoint interp (e : expr) : R :=
  match e with
  | C n => IZR n
  | P u v => Rplus (interp u) (interp v)
  end.
```

Example (How to prove $1 + 2 + 3 + 4 = 10$?)

```
Goal (1 + 2 + 3 + 4 = 10)%R.
Proof.
change
  (interp (P (P (P (C 1) (C 2)) (C 3)) (C 4)) = 10%R).
```

Computational Reflection: Transformation

```
Fixpoint normalize (e : expr) : Z :=
  match e with
  | C n => n
  | P u v => Zplus (normalize u) (normalize v)
  end.

Lemma normalize_correct (e : expr) :
  interp e = IZR (normalize e).
```

Computational Reflection: Transformation

```
Fixpoint normalize (e : expr) : Z :=
  match e with
  | C n => n
  | P u v => Zplus (normalize u) (normalize v)
  end.

Lemma normalize_correct (e : expr) :
  interp e = IZR (normalize e).
```

Example (How to prove $1 + 2 + 3 + 4 = 10$?)

```
rewrite normalize_correct.
apply eq_refl.
Qed.
```

Computational Reflection: Reification

```
Ltac reify t :=
  match t with
  | IZR ?n => constr:(C n)
  | Rplus ?u ?v =>
    let u' := reify u in
    let v' := reify v in
    constr:(P u' v')
  end.
```

Computational Reflection: Reification

```
Ltac reify t :=
  match t with
  | IZR ?n => constr:(C n)
  | Rplus ?u ?v =>
    let u' := reify u in
    let v' := reify v in
    constr:(P u' v')
  end.
```

Example (How to prove $1 + 2 + 3 + 4 = 10$?)

```
Goal (1 + 2 + 3 + 4 = 10)%R.
Proof.
match goal with
| |- ?t = _ =>
  let t' := reify t in
  change t with (interp t')
end.
```

Computational Reflection

Extensions

- Symbolic reasoning

```
Goal forall x:R, ((x + 1) * (x - 1) = x^2 - 1)%R.
Proof. intros x. ring. Qed.
```

Computational Reflection

Extensions

- Symbolic reasoning

```
Goal forall x:R, ((x + 1) * (x - 1) = x^2 - 1)%R.
Proof. intros x. ring. Qed.
```

- Approximate representations

```
Goal (1 <= RInt (fun x => exp (sin x)) 0 1)%R.
Proof. interval. Qed.
```

Outline

Coq Integers

Definition (Peano integers)

```
Inductive nat : Set :=  O : nat | S : nat -> nat.
```

Coq Integers

Definition (Peano integers)

```
Inductive nat : Set := O : nat | S : nat -> nat.
```

Definition (Lists of bits)

```
Inductive positive : Set :=
  | xI : positive -> positive (* xI n == 2*n+1 *)
  | xO : positive -> positive (* xO n == 2*n   *)
  | xH : positive.            (* xH   == 1     *)
```

Coq Integers

Definition (Peano integers)

```
Inductive nat : Set :=  O : nat | S : nat -> nat.
```

Definition (Lists of bits)

```
Inductive positive : Set :=
  | xI : positive -> positive (* xI n == 2*n+1 *)
  | xO : positive -> positive (* xO n == 2*n   *)
  | xH : positive.            (* xH   == 1     *)
```

Definition (Trees of machine integer)

```
Inductive BigN.t : Type :=
  | N0 : int31 -> BigN.t
  | N1 : zn2z int31 -> BigN.t
  | N2 : zn2z (z2nz int31) -> BigN.t
  | ...
  | Nn : forall n : nat, ... -> BigN.t.
```

Outline

Outline

Interval Arithmetic

Definition (Intervals)

Interval: connected closed subset of $\mathbb{R}$, e.g., $\mathbf{x} = [\underline{x}; \overline{x}]$ or $[\underline{x}; +\infty)$.

Interval Arithmetic

Definition (Intervals)

Interval: connected closed subset of $\mathbb{R}$, e.g., $\mathbf{x} = [\underline{x}; \overline{x}]$ or $[\underline{x}; +\infty)$.

Definition (Interval extension)

$\mathbf{f} : \mathbb{I}^n \to \mathbb{I}$ is an interval extension of $f : \mathbb{R}^n \to \mathbb{R}$ if
$\forall \mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{I},\ \forall x_1, \ldots, x_n \in \mathbb{R},$
$x_1 \in \mathbf{x}_1 \wedge \ldots \wedge x_n \in \mathbf{x}_n \Rightarrow f(x_1, \ldots, x_n) \in \mathbf{f}(\mathbf{x}_1, \ldots, \mathbf{x}_n).$

Interval Arithmetic

Definition (Directed rounding)

$$\forall u, v \in \mathbb{F} \subseteq \mathbb{R}, \quad \bigtriangledown(u \diamond v) \leq u \diamond v \leq \bigtriangleup(u \diamond v).$$

Interval Arithmetic

Definition (Directed rounding)

$$\forall u, v \in \mathbb{F} \subseteq \mathbb{R}, \quad \bigtriangledown(u \diamond v) \leq u \diamond v \leq \bigtriangleup(u \diamond v).$$

Definition (Interval extensions of $+$, $-$, $\times$)

If $u \in [\underline{u}, \overline{u}]$ and $v \in [\underline{v}, \overline{v}]$, then

$$\begin{aligned}
u + v &\in [\bigtriangledown(\underline{u} + \underline{v}); \bigtriangleup(\overline{u} + \overline{v})], \\
u - v &\in [\bigtriangledown(\underline{u} - \overline{v}); \bigtriangleup(\overline{u} - \underline{v})], \\
u \cdot v &\in [\min(\bigtriangledown(\underline{u} \cdot \underline{v}), \bigtriangledown(\underline{u} \cdot \overline{v}), \bigtriangledown(\overline{u} \cdot \underline{v}), \bigtriangledown(\overline{u} \cdot \overline{v})); \\
&\qquad \max(\bigtriangleup(\underline{u} \cdot \underline{v}), \bigtriangleup(\underline{u} \cdot \overline{v}), \bigtriangleup(\overline{u} \cdot \underline{v}), \bigtriangleup(\overline{u} \cdot \overline{v}))].
\end{aligned}$$

Dependency Effect

Example ($x - x^2$ when $x \in [0;1]$?)

$x - x^2 \in \mathbf{x} - \mathbf{x}^2 = [0;1] - [0^2;1^2] = [0-1;1-0] = [-1;1].$ ✗

Dependency Effect

Example ($x - x^2$ when $x \in [0;1]$?)

$x - x^2 \in \mathbf{x} - \mathbf{x}^2 = [0;1] - [0^2;1^2] = [0-1;1-0] = [-1;1].$ ✗

Rewriting

$x - x^2 = \frac{1}{4} - (x - \frac{1}{2})^2 \in \frac{1}{4} - [-\frac{1}{2};\frac{1}{2}]^2 = \frac{1}{4} - [0;\frac{1}{4}] = [0;\frac{1}{4}].$ ✓

Dependency Effect

Example ($x - x^2$ when $x \in [0;1]$?)

$x - x^2 \in \mathbf{x} - \mathbf{x}^2 = [0;1] - [0^2;1^2] = [0-1;1-0] = [-1;1]$. ✗

Rewriting

$x - x^2 = \frac{1}{4} - (x - \frac{1}{2})^2 \in \frac{1}{4} - [-\frac{1}{2};\frac{1}{2}]^2 = \frac{1}{4} - [0;\frac{1}{4}] = [0;\frac{1}{4}]$. ✓

Bisection

If $x \in \mathbf{x}_1 \cup \mathbf{x}_2$, then $f(x) \in \mathbf{f}(\mathbf{x}_1) \cup \mathbf{f}(\mathbf{x}_1)$.

Dependency Effect

Example ($x - x^2$ when $x \in [0;1]$?)

$x - x^2 \in \mathbf{x} - \mathbf{x}^2 = [0;1] - [0^2;1^2] = [0-1;1-0] = [-1;1]$. ✗

Rewriting

$x - x^2 = \frac{1}{4} - (x - \frac{1}{2})^2 \in \frac{1}{4} - [-\frac{1}{2};\frac{1}{2}]^2 = \frac{1}{4} - [0;\frac{1}{4}] = [0;\frac{1}{4}]$. ✓

Bisection

If $x \in \mathbf{x}_1 \cup \mathbf{x}_2$, then $f(x) \in \mathbf{f}(\mathbf{x}_1) \cup \mathbf{f}(\mathbf{x}_1)$.

For $x \in [0;\frac{1}{2}]$, $x - x^2 \in [0;\frac{1}{2}] - [0;\frac{1}{4}] = [-\frac{1}{4};\frac{1}{4}]$.
For $x \in [\frac{1}{2};1]$, $x - x^2 \in [\frac{1}{2};1] - [\frac{1}{4};1] = [-\frac{1}{2};\frac{3}{4}]$.
So, $x - x^2 \in [-\frac{1}{2};\frac{3}{4}]$. ✧

Outline

Floating-Point Numbers

Definition (Floating-point numbers)

$\mathbb{F}_\beta = \{m \cdot \beta^e \mid (m, e) \in \mathbb{Z}^2\} \subseteq \mathbb{R}$.

Floating-Point Numbers

Definition (Floating-point numbers)

$\mathbb{F}_\beta = \{m \cdot \beta^e \mid (m, e) \in \mathbb{Z}^2\} \subseteq \mathbb{R}$.

Definition (Formats)

FIX: $e \geq e_{\mathsf{min}}$,
FLX: $|m| < \beta^\varrho$,
FLT: $|m| < \beta^\varrho \wedge e \geq e_{\mathsf{min}}$.

Floating-Point Numbers

Definition (Floating-point numbers)

$\mathbb{F}_\beta = \{m \cdot \beta^e \mid (m, e) \in \mathbb{Z}^2\} \subseteq \mathbb{R}$.

Definition (Formats)

FIX: $e \geq e_{\mathsf{min}}$,
FLX: $|m| < \beta^\varrho$,
FLT: $|m| < \beta^\varrho \wedge e \geq e_{\mathsf{min}}$.

Definition (Generic formats)

$\mathcal{F}_\varphi = \{x \in \mathbb{R} \mid x \cdot \beta^{-\varphi(\mathsf{mag}(x))} \in \mathbb{Z}\}$
with $\mathsf{mag}(x) = \lfloor \log_\beta |x| + 1 \rfloor$.

Rounding

Definition

- $\square(x) = \lfloor x \cdot \beta^{-e} \rceil \cdot \beta^{e} \in \mathcal{F}_{\varphi}$ with $e = \varphi(\mathrm{mag}(x))$.

Rounding

Definition

- $\square(x) = \lfloor x \cdot \beta^{-e} \rceil \cdot \beta^e \in \mathcal{F}_\varphi$ with $e = \varphi(\mathrm{mag}(x))$.
- $\bigtriangledown(x) = \lfloor x \cdot \beta^{-e} \rfloor \cdot \beta^e$, $\bigtriangleup(x) = \lceil x \cdot \beta^{-e} \rceil \cdot \beta^e$.

Rounding

Definition

- $\square(x) = \lfloor x \cdot \beta^{-e} \rceil \cdot \beta^e \in \mathcal{F}_\varphi$ with $e = \varphi(\mathrm{mag}(x))$.
- $\bigtriangledown(x) = \lfloor x \cdot \beta^{-e} \rfloor \cdot \beta^e$, $\bigtriangleup(x) = \lceil x \cdot \beta^{-e} \rceil \cdot \beta^e$.

Properties

- $\forall x \in \mathcal{F}_\varphi, \quad \square(x) = x.$
- $\forall x \in \mathbb{R}, \quad \square(\square(x)) = \square(x).$
- $\forall u, v \in \mathbb{R}, \quad u \leq v \Rightarrow \square(u) \leq \square(v).$

Rounding

Definition

- $\square(x) = \lfloor x \cdot \beta^{-e} \rceil \cdot \beta^e \in \mathcal{F}_\varphi$ with $e = \varphi(\mathrm{mag}(x))$.
- $\bigtriangledown(x) = \lfloor x \cdot \beta^{-e} \rfloor \cdot \beta^e$, $\bigtriangleup(x) = \lceil x \cdot \beta^{-e} \rceil \cdot \beta^e$.

Properties

- $\forall x \in \mathcal{F}_\varphi, \quad \square(x) = x.$
- $\forall x \in \mathbb{R}, \quad \square(\square(x)) = \square(x).$
- $\forall u, v \in \mathbb{R}, \quad u \leq v \Rightarrow \square(u) \leq \square(v).$

$\square(x)$ is computable if x is represented by a pair $\langle m_x, e_x \rangle$. ✓

Addition and Multiplication

Definition (Exact operations)

- $(m_x \cdot \beta^{e_x}) + (m_y \cdot \beta^{e_y}) = (m_x \cdot \beta^{e_x - e} + m_y \cdot \beta^{e_y - e}) \cdot \beta^e$ with $e = \min(e_x, e_y)$.
- $(m_x \cdot \beta^{e_x}) \times (m_y \cdot \beta^{e_y}) = (m_x \cdot m_y) \cdot \beta^{e_x + e_y}$.

Addition and Multiplication

Definition (Exact operations)

- $(m_x \cdot \beta^{e_x}) + (m_y \cdot \beta^{e_y}) = (m_x \cdot \beta^{e_x - e} + m_y \cdot \beta^{e_y - e}) \cdot \beta^e$ with $e = \min(e_x, e_y)$.
- $(m_x \cdot \beta^{e_x}) \times (m_y \cdot \beta^{e_y}) = (m_x \cdot m_y) \cdot \beta^{e_x + e_y}$.

Definition (Rounded operations)

By composition of exact operations and rounded operations.

Addition and Multiplication

Definition (Exact operations)

- $(m_x \cdot \beta^{e_x}) + (m_y \cdot \beta^{e_y}) = (m_x \cdot \beta^{e_x - e} + m_y \cdot \beta^{e_y - e}) \cdot \beta^e$ with $e = \min(e_x, e_y)$.
- $(m_x \cdot \beta^{e_x}) \times (m_y \cdot \beta^{e_y}) = (m_x \cdot m_y) \cdot \beta^{e_x + e_y}$.

Definition (Rounded operations)

By composition of exact operations and rounded operations.

Linear in $|e_x - e_y|$ for addition. ✗

Division and Square Root

Lemma (Square root)

$$\exists e \in \mathbb{Z}, \quad \square(\sqrt{m_x \cdot \beta^{e_x}}) = \square(\lceil \sqrt{m_x \cdot \beta^{e}} \rfloor \cdot \beta^{(e_x - e)/2}).$$

Division and Square Root

Lemma (Square root)

$\exists e \in \mathbb{Z}, \quad \square(\sqrt{m_x \cdot \beta^{e_x}}) = \square(\lceil \sqrt{m_x \cdot \beta^{e}} \rfloor \cdot \beta^{(e_x - e)/2})$.

e can be computed from φ, e_x, and the β-size of m_x.

Division and Square Root

Lemma (Square root)

$$\exists e \in \mathbb{Z}, \quad \square(\sqrt{m_x \cdot \beta^{e_x}}) = \square(\lceil \sqrt{m_x \cdot \beta^{e}} \rfloor \cdot \beta^{(e_x - e)/2}).$$

e can be computed from φ, e_x, and the β-size of m_x.

Theorem (Division)

$$\exists e \in \mathbb{Z}, \quad \square((m_x \cdot \beta^{e_x})/(m_y \cdot \beta^{e_y})) = \square(\lceil m_x \cdot \beta^{e}/m_y \rfloor \cdot \beta^{e_x - e_y - e}).$$

Division and Square Root

Lemma (Square root)

$$\exists e \in \mathbb{Z}, \quad \square(\sqrt{m_x \cdot \beta^{e_x}}) = \square(\lceil \sqrt{m_x \cdot \beta^{e}} \rfloor \cdot \beta^{(e_x - e)/2}).$$

e can be computed from φ, e_x, and the β-size of m_x.

Theorem (Division)

$$\exists e \in \mathbb{Z}, \quad \square((m_x \cdot \beta^{e_x})/(m_y \cdot \beta^{e_y})) = \square(\lceil m_x \cdot \beta^{e} / m_y \rfloor \cdot \beta^{e_x - e_y - e}).$$

$e \simeq \varrho$ in practice. ✓

CoqInterval in a Nutshell

```
Inductive float : Set :=
  Fnan | Float : ... -> float.

Inductive interval : Set :=
  Inan | Ibnd : float -> float -> interval.

Definition sub prec xi yi :=
  match xi, yi with
  | Ibnd xl xu, Ibnd yl yu =>
    Ibnd (F.sub rnd_DN prec xl yu)
         (F.sub rnd_UP prec xu yl)
  | _, _ => Inan
  end.
```

Outline

Elementary Functions

Table-maker dilemma

Devising a computable function f such that $\square(\exp(x)) = \square(f(x))$ is too difficult.

Elementary Functions

Table-maker dilemma

Devising a computable function f such that $\square(\exp(x)) = \square(f(x))$ is too difficult.

Workaround: implement $\exp : \mathbb{F} \to \mathbb{I}$ with no guaranteed tightness.

Power Series

Lemma (Alternated series)

If $\sum_k (-1)^k a_k x^k$ converges toward $f(x)$
and if $k \mapsto a_k x^k$ is positive decreasing, then

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

Power Series

Lemma (Alternated series)

If $\sum_k (-1)^k a_k x^k$ converges toward $f(x)$
and if $k \mapsto a_k x^k$ is positive decreasing, then

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

$$\arctan x = x \cdot \sum_{i=0}^{\infty} (-1)^i \cdot \frac{(x^2)^i}{2i+1}.$$

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

- guess n by estimating the convergence of $a_k x^k$,
  e.g., $n = p/2$ for $f = \arctan$, $\beta = 2$, and $|x| \leq \frac{1}{2}$,

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

- guess n by estimating the convergence of $a_k x^k$,
  e.g., $n = p/2$ for $f = \arctan$, $\beta = 2$, and $|x| \leq \frac{1}{2}$,
- guess ϱ, e.g., $\varrho = p + c_f$,

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

- guess n by estimating the convergence of $a_k x^k$, e.g., $n = p/2$ for $f = \arctan$, $\beta = 2$, and $|x| \leq \frac{1}{2}$,
- guess ϱ, e.g., $\varrho = p + c_f$,
- compute $\sum_{k=0}^{n-1} (-1)^k a_k x^k$ using intervals,

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

- guess n by estimating the convergence of $a_k x^k$,
  e.g., $n = p/2$ for $f = \arctan$, $\beta = 2$, and $|x| \leq \frac{1}{2}$,
- guess ϱ, e.g., $\varrho = p + c_f$,
- compute $\sum_{k=0}^{n-1} (-1)^k a_k x^k$ using intervals,
- stop as soon as $a_k x^k \leq \beta^{-p}$,

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

- guess n by estimating the convergence of $a_k x^k$, e.g., $n = p/2$ for $f = \arctan$, $\beta = 2$, and $|x| \leq \frac{1}{2}$,
- guess ϱ, e.g., $\varrho = p + c_f$,
- compute $\sum_{k=0}^{n-1} (-1)^k a_k x^k$ using intervals,
- stop as soon as $a_k x^k \leq \beta^{-p}$,
- add the remainder.

Power Series

Algorithm

$$0 \leq (-1)^n \left(f(x) - \sum_{k=0}^{n-1} (-1)^k a_k x^k \right) \leq a_n x^n.$$

To approximate $f(x)$ with an accuracy of β^{-p},

- guess n by estimating the convergence of $a_k x^k$,
  e.g., $n = p/2$ for $f = \arctan$, $\beta = 2$, and $|x| \leq \frac{1}{2}$,
- guess ϱ, e.g., $\varrho = p + c_f$,
- compute $\sum_{k=0}^{n-1} (-1)^k a_k x^k$ using intervals,
- stop as soon as $a_k x^k \leq \beta^{-p}$,
- add the remainder.

This approach does not give a usable interval extension **f**
due to the dependency effect (unless **x** is almost a point interval).

Argument Reduction (arctan)

Argument reduction for arctan

To get $|x| \leq \frac{1}{2}$, rewrite $\arctan x$ into

1. $-\arctan(-x)$ for $x < 0$,

Argument Reduction (arctan)

Argument reduction for arctan

To get $|x| \leq \frac{1}{2}$, rewrite $\arctan x$ into

1. $-\arctan(-x)$ for $x < 0$,
2. $\begin{cases} \frac{\pi}{4} + \arctan \frac{x-1}{x+1} & \text{for } x \in [\frac{1}{2}; 2], \\ \frac{\pi}{2} - \arctan \frac{1}{x} & \text{for } x \geq 2. \end{cases}$

Argument Reduction (arctan)

Argument reduction for arctan

To get $|x| \leq \frac{1}{2}$, rewrite $\arctan x$ into

1. $-\arctan(-x)$ for $x < 0$,
2. $\begin{cases} \frac{\pi}{4} + \arctan \frac{x-1}{x+1} & \text{for } x \in [\frac{1}{2}; 2], \\ \frac{\pi}{2} - \arctan \frac{1}{x} & \text{for } x \geq 2. \end{cases}$

Machin's formula

$$\frac{\pi}{4} = 4 \cdot \arctan \frac{1}{5} - \arctan \frac{1}{239}.$$

Argument Reduction (exp, log)

Argument reduction for exp

To get $x \in [-2^{-8}; 0]$, rewrite $\exp x$ into

1. $1/\exp(-x)$ for $x > 0$,

Argument Reduction (exp, log)

Argument reduction for exp

To get $x \in [-2^{-8}; 0]$, rewrite $\exp x$ into

1. $1/\exp(-x)$ for $x > 0$,
2. $(\exp(x/2))^2$ as long as $x < -2^{-8}$.

Argument Reduction (exp, log)

Argument reduction for exp

To get $x \in [-2^{-8}; 0]$, rewrite $\exp x$ into

1. $1/\exp(-x)$ for $x > 0$,
2. $(\exp(x/2))^2$ as long as $x < -2^{-8}$.

ϱ is increased according to the number of reconstruction steps. ✧

Argument Reduction (exp, log)

Argument reduction for exp

To get $x \in [-2^{-8}; 0]$, rewrite $\exp x$ into

1. $1/\exp(-x)$ for $x > 0$,
2. $(\exp(x/2))^2$ as long as $x < -2^{-8}$.

ϱ is increased according to the number of reconstruction steps. ✧

Argument reduction for log

To get $x \in [1; 1 + 2^{-8}]$, rewrite $\log x$ into

1. $-\log(x^{-1})$ for $x < 1$,

Argument Reduction (exp, log)

Argument reduction for exp

To get $x \in [-2^{-8}; 0]$, rewrite $\exp x$ into

1. $1/\exp(-x)$ for $x > 0$,
2. $(\exp(x/2))^2$ as long as $x < -2^{-8}$.

ϱ is increased according to the number of reconstruction steps. ✧

Argument reduction for log

To get $x \in [1; 1 + 2^{-8}]$, rewrite $\log x$ into

1. $-\log(x^{-1})$ for $x < 1$,
2. $2 \log \sqrt{x}$ as long as $x > 1 + 2^{-8}$.

Argument Reduction (exp, log)

Argument reduction for exp

To get $x \in [-2^{-8}; 0]$, rewrite $\exp x$ into

1. $1/\exp(-x)$ for $x > 0$,
2. $(\exp(x/2))^2$ as long as $x < -2^{-8}$.

ϱ is increased according to the number of reconstruction steps. ✦

Argument reduction for log

To get $x \in [1; 1 + 2^{-8}]$, rewrite $\log x$ into

1. $-\log(x^{-1})$ for $x < 1$,
2. $2 \log \sqrt{x}$ as long as $x > 1 + 2^{-8}$.

ϱ is increased by $-\log_\beta(1 - x)$ for x close to 1^-. ✗

Argument Reduction (cos, sin, tan)

Argument reduction for cos, sin, and tan

To get $x \in [-\frac{1}{2}; \frac{1}{2}]$, rewrite

$$\begin{aligned} \sin x &\rightarrow \operatorname{sign}(\sin x) \cdot \sqrt{1 - (\cos x)^2}, \\ \tan x &\rightarrow \operatorname{sign}(\sin x) \cdot \operatorname{sign}(\cos x) \cdot \sqrt{(\cos x)^{-2} - 1}. \end{aligned}$$

Argument Reduction (cos, sin, tan)

Argument reduction for cos, sin, and tan

To get $x \in [-\frac{1}{2}; \frac{1}{2}]$, rewrite

$$\begin{aligned} \sin x &\to \operatorname{sign}(\sin x) \cdot \sqrt{1 - (\cos x)^2}, \\ \tan x &\to \operatorname{sign}(\sin x) \cdot \operatorname{sign}(\cos x) \cdot \sqrt{(\cos x)^{-2} - 1}. \end{aligned}$$

then

$$\begin{aligned} \cos x &\to 2 \cdot (\cos \tfrac{x}{2})^2 - 1, \\ \operatorname{sign}(\sin x) &\to \operatorname{sign}(\sin \tfrac{x}{2}) \cdot \operatorname{sign}(\cos \tfrac{x}{2}). \end{aligned}$$

Interval Operators

$$\exp \mathbf{x} = [\exp \underline{x}; \exp \overline{x}] \qquad \text{by monotony}$$

Interval Operators

$\exp \mathbf{x} = [\exp \underline{x}; \exp \overline{x}]$ by monotony

$\cos \mathbf{x} = [\cos \overline{x}; \cos \underline{x}]$ for $\mathbf{x} \subseteq [0; \pi]$ by monotony

Interval Operators

$$\exp \mathbf{x} = [\exp \underline{x}; \exp \overline{x}] \quad \text{by monotony}$$

$$\cos \mathbf{x} = [\cos \overline{x}; \cos \underline{x}] \quad \text{for } \mathbf{x} \subseteq [0; \pi] \quad \text{by monotony}$$

$$\cos \mathbf{x} = [-1; \max(\cos \underline{x}, \cos \overline{x})] \quad \text{for } \pi \in \mathbf{x} \subseteq [0; 2\pi] \quad \text{by monotony}$$

Interval Operators

$$\exp \mathbf{x} = [\exp \underline{x}; \exp \overline{x}] \qquad \text{by monotony}$$

$$\cos \mathbf{x} = [\cos \overline{x}; \cos \underline{x}] \quad \text{for } \mathbf{x} \subseteq [0; \pi] \qquad \text{by monotony}$$

$$\cos \mathbf{x} = [-1; \max(\cos \underline{x}, \cos \overline{x})] \quad \text{for } \pi \in \mathbf{x} \subseteq [0; 2\pi] \qquad \text{by monotony}$$

$$\cos \mathbf{x} = \left(\cos \tfrac{\underline{x}+\overline{x}}{2} + \tfrac{\overline{x}-\underline{x}}{2} \cdot [-1; 1]\right) \cap [-1; 1]$$

by bounded variations

Interval Operators

$$\exp \mathbf{x} = [\exp \underline{x}; \exp \overline{x}] \qquad \text{by monotony}$$

$$\cos \mathbf{x} = [\cos \overline{x}; \cos \underline{x}] \quad \text{for } \mathbf{x} \subseteq [0; \pi] \qquad \text{by monotony}$$

$$\cos \mathbf{x} = [-1; \max(\cos \underline{x}, \cos \overline{x})] \quad \text{for } \pi \in \mathbf{x} \subseteq [0; 2\pi] \qquad \text{by monotony}$$

$$\cos \mathbf{x} = \left(\cos \tfrac{\underline{x}+\overline{x}}{2} + \tfrac{\overline{x}-\underline{x}}{2} \cdot [-1; 1]\right) \cap [-1; 1]$$

by bounded variations

Interval extension for cos is poor, but isotone. ✧

Memoization and Constants

Example

```
Definition pi prec : interval := ...
```

Memoization and Constants

Example

```
Definition pi prec : interval := ...
```

Coinductive types are lazy

```
CoInductive cell := Cell : t -> cell.

CoInductive stream :=
  Cons : cell -> stream -> stream.

Fixpoint get n s {struct n} : t :=
  match n, s with
  | O, Cons (Cell v) _ => v
  | S n, Cons _ s => get n s
  end.

CoFixpoint create f n : stream :=
  Cons (do f n) (create f (S n)).
```

Outline

Manual Proof

Example ($\forall x \in [3;5],\ 1+\sqrt{x} \leq 4$)

```
Module F := SpecificFloat StdZRadix2.
Module I := FloatIntervalFull F.
Module J := IntervalExt I.
Notation "x \in xi" :=
  (contains (I.convert xi) (Xreal x)) (at level 100).

Lemma foo x (H : 3 <= x <= 5) : 1 + sqrt x <= 4.
Proof.
refine (proj2 (subset_contains (I.convert _)
    (I.convert (I.bnd F.nan (F.fromZ 4)))
    (I.subset_correct _ _ _) (Xreal _) _)) ; cycle 1.
apply (J.add_correct 10%Z).
apply I.fromZ_correct.
apply (J.sqrt_correct 10%Z).
apply (H : x \in I.bnd (F.fromZ 3) (F.fromZ 5)).
vm_compute. apply eq_refl.
Qed.
```

Reifying Terms

Straight-line programs

```
Inductive binary_op : Set := Add | Sub | Mul | Div.

Inductive term : Set :=
  | Forward : nat -> term
  | Unary : unary_op -> nat -> term
  | Binary : binary_op -> nat -> nat -> term.
```

Reifying Terms

Straight-line programs

```
Inductive binary_op : Set := Add | Sub | Mul | Div.

Inductive term : Set :=
  | Forward : nat -> term
  | Unary : unary_op -> nat -> term
  | Binary : binary_op -> nat -> nat -> term.
```

Example $((t+\pi)\sqrt{t}-(t+\pi))$

```
(*                              [t, pi] *)     Binary Add 0 1
(*                        [t+pi, t, pi] *) :: Unary Sqrt 1
(*                  [sqrt t, t+pi, t, pi] *) :: Binary Mul 1 0
(*          [(t+pi)*sqrt t, sqrt t, ...] *) :: Binary Sub 0 2
(* [(t+pi)*sqrt t - (t+pi), (t+pi)*...] *) :: nil
```

Reification

Example (Finding a term in a list)

```
Ltac list_find a l :=
  let rec aux l n :=
    match l with
    | nil         => false
    | cons a _    => n
    | cons _ ?l' => aux l' (S n)
    end in
  aux l 0.
```

Evaluation

Definition (Evaluation)

$[\![p]\!]_T(\vec{x})$: evaluation of an SLP p with an initial stack $\vec{x}$ using operations over T.

Evaluation

Definition (Evaluation)

$[\![p]\!]_T(\vec{x})$: evaluation of an SLP p with an initial stack $\vec{x}$ using operations over T.

Lemma (Containment of evaluation)

For any program p, we have

$$\forall \vec{x} \in \mathbb{R}^n,\ \forall \vec{\mathbf{x}} \in \mathbb{I}^n,\ (\forall i \leq n,\ x_i \in \mathbf{x}_i) \Rightarrow [\![p]\!]_{\mathbb{R}}(\vec{x}) \in [\![p]\!]_{\mathbb{I}}(\vec{\mathbf{x}}).$$

Evaluation

Definition (Evaluation)

$[\![p]\!]_T(\vec{x})$: evaluation of an SLP p with an initial stack $\vec{x}$ using operations over T.

Lemma (Containment of evaluation)

For any program p, we have

$$\forall \vec{x} \in \mathbb{R}^n,\ \forall \vec{\mathbf{x}} \in \mathbb{I}^n,\ (\forall i \leq n,\ x_i \in \mathbf{x}_i) \Rightarrow [\![p]\!]_{\mathbb{R}}(\vec{x}) \in [\![p]\!]_{\mathbb{I}}(\vec{\mathbf{x}}).$$

Corollary

$$\forall \mathbf{y} \in \mathbb{I},\ [\![p]\!]_{\mathbb{I}}(\vec{\mathbf{x}}) \subseteq \mathbf{y} \Rightarrow [\![p]\!]_{\mathbb{R}}(\vec{x}) \in \mathbf{y}.$$

Outline

Dependency Effect

Example

Given $\tilde{y} = 1 + x + x^2/2$, $y = \exp x$, and $x \in [0; 1]$,

$$\begin{aligned} \tilde{y} - y &\in (1 + \mathbf{x} + \mathbf{x}^2/2) - \exp(\mathbf{x}) \\ &\in ([1;1] + [0;1] + [0;0.5]) - [\exp 0; \exp 1] \\ &\in [1; 2.5] - [1; e] \\ &\in [1 - e; 2.5 - 1] \subseteq [-1.72; 1.5]. \end{aligned}$$

Automatic Differentiation

Mean-value theorem

$$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},\ f(x) = f(x_0) + (x - x_0) \cdot f'(\xi).$$

Automatic Differentiation

Mean-value theorem

$$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},\ f(x) = f(x_0) + (x - x_0) \cdot f'(\xi).$$

Corollary

$$\forall x \in \mathbf{x},\ f(x) \in (\mathbf{f}(x_0) + (\mathbf{x} - x_0) \cdot \mathbf{f}'(\mathbf{x})) \cap \mathbf{f}(\mathbf{x}).$$

Automatic Differentiation

Mean-value theorem

$$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},\ f(x) = f(x_0) + (x - x_0) \cdot f'(\xi).$$

Corollary

$$\forall x \in \mathbf{x},\ f(x) \in (\mathbf{f}(x_0) + (\mathbf{x} - x_0) \cdot \mathbf{f}'(\mathbf{x})) \cap \mathbf{f}(\mathbf{x}).$$

Automatic differentiation

$$\begin{aligned}
(\mathbf{u}, \mathbf{u}') + (\mathbf{v}, \mathbf{v}') &= (\mathbf{u} + \mathbf{v}, \mathbf{u}' + \mathbf{v}'), \\
(\mathbf{u}, \mathbf{u}') \times (\mathbf{v}, \mathbf{v}') &= (\mathbf{u} \cdot \mathbf{v}, \mathbf{u}' \cdot \mathbf{v} + \mathbf{u} \cdot \mathbf{v}'), \\
\exp(\mathbf{u}, \mathbf{u}') &= (\exp(\mathbf{u}), \mathbf{u}' \cdot \exp(\mathbf{u})).
\end{aligned}$$

Automatic Differentiation

Mean-value theorem

$$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},\ f(x) = f(x_0) + (x - x_0) \cdot f'(\xi).$$

Corollary

$$\forall x \in \mathbf{x},\ f(x) \in (\mathbf{f}(x_0) + (\mathbf{x} - x_0) \cdot \mathbf{f}'(\mathbf{x})) \cap \mathbf{f}(\mathbf{x}).$$

Automatic differentiation

$$\begin{aligned}
(\mathbf{u}, \mathbf{u}') + (\mathbf{v}, \mathbf{v}') &= (\mathbf{u} + \mathbf{v}, \mathbf{u}' + \mathbf{v}'), \\
(\mathbf{u}, \mathbf{u}') \times (\mathbf{v}, \mathbf{v}') &= (\mathbf{u} \cdot \mathbf{v}, \mathbf{u}' \cdot \mathbf{v} + \mathbf{u} \cdot \mathbf{v}'), \\
\exp(\mathbf{u}, \mathbf{u}') &= (\exp(\mathbf{u}), \mathbf{u}' \cdot \exp(\mathbf{u})).
\end{aligned}$$

If $0 \notin \mathbf{f}'(\mathbf{x})$, then $f(x) \in \mathrm{hull}(\mathbf{f}(\underline{x}) \cup \mathbf{f}(\overline{x}))$. ✓

Polynomial Approximations

Taylor-Lagrange Formula

$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},$

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1}.$$

Polynomial Approximations

Taylor-Lagrange Formula

$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},$

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1}.$$

Polynomial approximation

$(\vec{\mathbf{p}}, \mathbf{\Delta})$ encloses f over $\mathbf{x}$ if

$$\exists p \in \mathbb{R}[X], \quad (\forall i,\ p_i \in \mathbf{p}_i)\ \wedge\ \forall x \in \mathbf{x},\ f(x) - p(x - x_0) \in \mathbf{\Delta}.$$

Polynomial Approximations

Taylor-Lagrange Formula

$\forall x \in \mathbf{x},\ \exists \xi \in \mathbf{x},$

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1}.$$

Polynomial approximation

$(\vec{\mathbf{p}}, \mathbf{\Delta})$ encloses f over $\mathbf{x}$ if

$$\exists p \in \mathbb{R}[X], \quad (\forall i,\ p_i \in \mathbf{p}_i)\ \wedge\ \forall x \in \mathbf{x},\ f(x) - p(x - x_0) \in \mathbf{\Delta}.$$

- Taylor-Lagrange formula for elementary functions,
- polynomial arithmetic for composite expressions.

Dependency Effect

Example ($1 + x + x^2/2 - \exp x$ for $x \in [0; 1]$)

```
Variable x: R.
Hypothesis H: 0 <= x <= 1.

Goal Rabs (1 + x + x*x/2 - exp x) <= 172/100.
Proof. interval. Qed.

Goal Rabs (1 + x + x*x/2 - exp x) <= 171/100.
Proof. Fail interval. Abort.

Goal Rabs (1 + x + x*x/2 - exp x) <= 22/100.
Proof. interval with (i_bisect x, i_depth 12). Qed.

Goal Rabs (1 + x + x*x/2 - exp x) <= 22/100.
Proof. interval with (i_bisect_diff x, i_depth 3). Qed.

Goal Rabs (1 + x + x*x/2 - exp x) <= 22/100.
Proof. interval with (i_bisect_taylor x 2, i_depth 1). Qed.
```

Shortcomings

Missing features

- Monotony on Taylor models.

Shortcomings

Missing features

- Monotony on Taylor models.
- Tight polynomial evaluation.

Shortcomings

Missing features

- Monotony on Taylor models.
- Tight polynomial evaluation.
- Multivariate bisection.

Shortcomings

Missing features

- Monotony on Taylor models.
- Tight polynomial evaluation.
- Multivariate bisection.
- Multivariate approximation.

Shortcomings

Missing features

- Monotony on Taylor models.
- Tight polynomial evaluation.
- Multivariate bisection.
- Multivariate approximation.
- Backward propagation.

Outline

Proper Definite Integrals

Lemma

$$\int_u^v f \in (v-u) \cdot \mathrm{hull}\{f(t) \mid t \in [u;v]\}.$$

Proper Definite Integrals

Lemma

$$\int_u^v f \in (v-u) \cdot \mathrm{hull}\{f(t) \mid t \in [u;v]\}.$$

Corollary

$$\int_u^v f \in (\mathbf{v}-\mathbf{u}) \cdot \mathbf{f}(\mathrm{hull}(\mathbf{u},\mathbf{v})).$$

Proper Definite Integrals

Lemma

$$\int_u^v f \in (v-u) \cdot \mathrm{hull}\{f(t) \mid t \in [u;v]\}.$$

Corollary

$$\int_u^v f \in (\mathbf{v}-\mathbf{u}) \cdot \mathbf{f}(\mathrm{hull}(\mathbf{u},\mathbf{v})).$$

Example

$$\int_0^1 t\,dt \in [0;1].$$

Polynomial Approximation

Lemma

If $(p, \mathbf{\Delta})$ encloses f over $[u; v]$, and if P is a primitive of p, then

$$\int_u^v f \in P(v) - P(u) + (v - u) \cdot \mathbf{\Delta}.$$

Examples

Proper definite integrals

- $\displaystyle\int_0^1 \frac{dt}{1+t^2} = \int_0^1 \sqrt{1-t^2}\,dt = \frac{\pi}{4}.$

Examples

Proper definite integrals

- $\displaystyle\int_0^1 \frac{dt}{1+t^2} = \int_0^1 \sqrt{1-t^2}\,dt = \frac{\pi}{4}.$
- $\displaystyle\int_0^1 \left|\left(x^4+10x^3+19x^2-6x-6\right)\,e^x\right| dx \simeq 11.14731055.$

Examples

Proper definite integrals

- $\displaystyle\int_0^1 \frac{dt}{1+t^2} = \int_0^1 \sqrt{1-t^2}\,dt = \frac{\pi}{4}.$
- $\displaystyle\int_0^1 \left|\left(x^4+10x^3+19x^2-6x-6\right)\,e^x\right| dx \simeq 11.14731055.$
- $\displaystyle\int_0^8 \sin(t+e^t)\,dt \simeq 0.3474$

Improper Definite Integrals

Lemma

Assume that f is bounded, f and g are continuous, and g has a constant sign, over $[u; +\infty)$. Assume that $\int_u^{+\infty} g$ exists.
Then $\int_u^{+\infty} fg$ exists, and

$$\int_u^{+\infty} fg \in \text{hull}\{f(t) \mid t \geq u\} \cdot \int_u^{+\infty} g.$$

Improper Definite Integrals

Lemma

Assume that f is bounded, f and g are continuous, and g has a constant sign, over $[u; +\infty)$. Assume that $\int_u^{+\infty} g$ exists.
Then $\int_u^{+\infty} fg$ exists, and

$$\int_u^{+\infty} fg \in \mathrm{hull}\{f(t) \mid t \geq u\} \cdot \int_u^{+\infty} g.$$

Corollary

If $\mathbf{G}$ is an interval extension of $x \mapsto \int_x^{+\infty} g$, then

$$\int_u^{+\infty} fg \in \mathbf{f}(\mathrm{hull}(\mathbf{u}, +\infty)) \cdot \mathbf{G}(\mathbf{u}).$$

Well-Known Functions

- $\int_u^{+\infty} e^{\gamma x}\,dx = -\frac{e^{\gamma u}}{\gamma}$ for $\gamma < 0$.
- $\int_u^{+\infty} \frac{\log^\beta x}{x}\,dx = -\frac{\log^{\beta+1} u}{\beta+1}$ for $\beta < -1$.
- $\int_u^{+\infty} x^\alpha\,dx = -\frac{u^{\alpha+1}}{\alpha+1}$ for $\alpha < -1$.
- $\int_u^{+\infty} x^\alpha \log^\beta x\,dx =$
$$-\left(\frac{u^{\alpha+1}\log^\beta u}{\alpha+1}\right) - \frac{\beta}{\alpha+1}\int_u^{+\infty} x^\alpha \log^{\beta-1} x\,dx$$
for $\alpha < -1$ and $\beta \in \mathbb{N}$.

Examples

Improper definite integrals

- $\displaystyle\int_1^{+\infty} \cos x \, \frac{\ln x}{x^2} \, dx \simeq -0.1595.$

Examples

Improper definite integrals

- $\displaystyle\int_1^{+\infty} \cos x \, \frac{\ln x}{x^2} \, dx \simeq -0.1595.$
- $\displaystyle\int_{-\infty}^{+\infty} \frac{(0.5 \cdot \log(\tau^2 + 2.25) + 4.1396 + \log \pi)^2}{0.25 + \tau^2} \, d\tau$

$$\in [226.849; 226.850].$$

Outline

Cody & Waite's algorithm (1980)

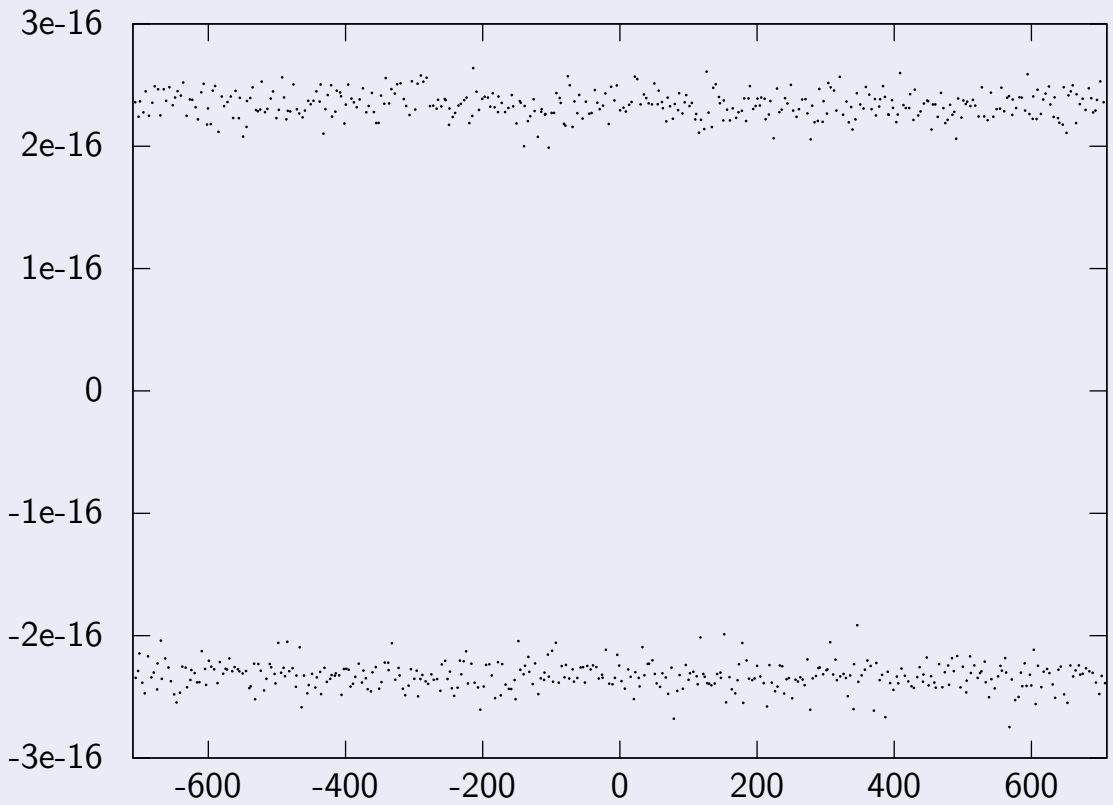
Example

```
double cw_exp(double x)
{
  // exception handling and constants
  ...
  // argument reduction
  double k = nearbyint(x * InvLog2);
  double t = x - k * Log2h - k * Log2l;
  // polynomial approximation
  double t2 = t * t;
  double p = 0.25 + t2 * (p1 + t2 * p2);
  double q = 0.5  + t2 * (q1 + t2 * q2);
  double f = t * (p / (q - t * p)) + 0.5;
  // result reconstruction
  return ldexp(f, (int)k + 1);
}
```

Cody & Waite's algorithm (1980)

Accuracy

$$\forall x \in \mathbb{F}_{b64},\ x \in [-746; 710] \Rightarrow \left| \frac{\texttt{cw\_exp}\, x - \exp x}{\exp x} \right| \leq 2^{-51}.$$



Outline

Formalizing IEEE 754

Floating-point numbers

- Signed zeros: $+0$ and -0.
- Signed infinities: $+\infty$ and $-\infty$.
- Not-A-Number, with a payload.
- Finite numbers: $m \cdot \beta^e$ with $|m| < \beta^p$, $e_{\mathsf{min}} \leq e \leq e_{\mathsf{max}}$.

Formalizing IEEE 754

Floating-point numbers

- Signed zeros: $+0$ and -0.
- Signed infinities: $+\infty$ and $-\infty$.
- Not-A-Number, with a payload.
- Finite numbers: $m \cdot \beta^e$ with $|m| < \beta^p$, $e_{\min} \leq e \leq e_{\max}$.

```
Inductive binary_float (prec emax : Z) : Set :=
  | B754_zero     (s : bool) : binary_float prec emax
  | B754_infinity (s : bool) : binary_float prec emax
  | B754_nan      (s : bool) (pl : positive) :
    nan_pl prec pl = true -> binary_float prec emax
  | B754_finite   (s : bool) (m : positive) (e : Z) :
    bounded prec emax m e = true ->
    binary_float prec emax.
```

Formalizing IEEE 754

Rounding directions

```
Inductive mode : Set :=
  mode_NE | mode_ZR | mode_DN | mode_UP | mode_NA.
```

Formalizing IEEE 754

Rounding directions

```
Inductive mode : Set :=
  mode_NE | mode_ZR | mode_DN | mode_UP | mode_NA.
```

Example (Division)

```
Definition Bdiv :
  forall prec emax : Z,
  (0 < prec)%Z -> (prec < emax)%Z ->
  (binary_float prec emax -> binary_float prec emax ->
     {n : binary_float prec emax | is_nan prec emax n
         = true}) ->
  mode -> binary_float prec emax -> binary_float prec
      emax ->
  binary_float prec emax.
```

Formalizing IEEE 754

Example (Division, exceptions)

```
Definition Bdiv prec emax ... m div_nan x y :=
  match x, y with
  | B754_nan _ _ _, _ | _, B754_nan _ _ _  => build_nan (div_nan x y)
  | B754_infinity sx, B754_infinity sy     => build_nan (div_nan x y)
  | B754_infinity sx, B754_finite sy _ _ _ => B754_infinity (xorb sx sy)
  | B754_finite sx _ _ _, B754_infinity sy => B754_zero (xorb sx sy)
  | B754_infinity sx, B754_zero sy         => B754_infinity (xorb sx sy)
  | B754_zero sx, B754_infinity sy         => B754_zero (xorb sx sy)
  | B754_finite sx _ _ _, B754_zero sy     => B754_infinity (xorb sx sy)
  | B754_zero sx, B754_finite sy _ _ _     => B754_zero (xorb sx sy)
  | B754_zero sx, B754_zero sy             => build_nan (div_nan x y)
  | B754_finite sx mx ex _, B754_finite sy my ey _ => ...
  end.
```

Formalizing IEEE 754

IEEE 754-2008, §4.3

"Each operation shall be performed as if it first produced an intermediate result correct to infinite precision and with unbounded range, and then rounded that result [...]."

Formalizing IEEE 754

IEEE 754-2008, §4.3

"Each operation shall be performed as if it first produced an intermediate result correct to infinite precision and with unbounded range, and then rounded that result [...]."

Example (Division, correction)

```
Theorem Bdiv_correct :
  forall (prec emax : Z) (Hp : (0 < prec)%Z) (Hm : (prec < emax)%Z)
    (div_nan : ...) (m : mode) (x y : binary_float prec emax),
  let emin := (3 - emax - prec)%Z in
  B2R prec emax y <> 0%R ->
  let r := round radix2 (FLT_exp emin prec) (round_mode m)
    (B2R prec emax x / B2R prec emax y) in
  let z := Bdiv prec emax Hp Hm div_nan m x y in
  let sign := xorb (Bsign prec emax x) (Bsign prec emax y) in
  if Rlt_bool (Rabs r) (bpow radix2 emax)
  then  (* finite case *)
    B2R prec emax z = r /\
    is_finite prec emax z = is_finite prec emax x /\
    (is_nan prec emax z = false -> Bsign prec emax z = sign)
  else  (* overflow *)
    B2FF prec emax z = binary_overflow prec emax m sign.
```

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.
- No upper bound on exponent range.

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.
- No upper bound on exponent range.
- $\mathcal{F} = \{m \cdot \beta^e \mid |m| < \beta^\varrho \wedge e \geq e_{\mathsf{min}}\}$.

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.
- No upper bound on exponent range.
- $\mathcal{F} = \{m \cdot \beta^e \mid |m| < \beta^q \wedge e \geq e_{\mathsf{min}}\}$.
- $u \oslash v$ becomes $\circ(u/v)$.

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.
- No upper bound on exponent range.
- $\mathcal{F} = \{m \cdot \beta^e \mid |m| < \beta^\varrho \wedge e \geq e_{\mathsf{min}}\}$.
- $u \oslash v$ becomes $\circ(u/v)$.

Some properties

- $\circ(x) = x + \delta$ with $|\delta| \leq \frac{1}{2}\beta^{-\varphi(\mathsf{mag}(x))}$.

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.
- No upper bound on exponent range.
- $\mathcal{F} = \{m \cdot \beta^e \mid |m| < \beta^\varrho \wedge e \geq e_{\mathsf{min}}\}$.
- $u \oslash v$ becomes $\circ(u/v)$.

Some properties

- $\circ(x) = x + \delta$ with $|\delta| \leq \frac{1}{2}\beta^{-\varphi(\mathsf{mag}(x))}$.
- $\circ(x) = x \cdot (1 + \varepsilon)$ with $|\varepsilon| \leq \frac{1}{2}\beta^{-\varrho+1}$ if $|x| \geq \beta^{e_{\mathsf{min}}+\varrho}$.

Separation of Concerns

Eliminating as many exceptions as possible

- No infinities, no NaNs, an unsigned zero.
- No upper bound on exponent range.
- $\mathcal{F} = \{m \cdot \beta^e \mid |m| < \beta^\varrho \wedge e \geq e_{\mathsf{min}}\}$.
- $u \oslash v$ becomes $\circ(u/v)$.

Some properties

- $\circ(x) = x + \delta$ with $|\delta| \leq \frac{1}{2}\beta^{-\varphi(\mathsf{mag}(x))}$.
- $\circ(x) = x \cdot (1 + \varepsilon)$ with $|\varepsilon| \leq \frac{1}{2}\beta^{-\varrho+1}$ if $|x| \geq \beta^{e_{\mathsf{min}}+\varrho}$.
- $\circ(u - v) = u - v$ if $u, v \in \mathcal{F}$ and $\frac{u}{v} \in [\frac{1}{2}; 2]$.

Back to the Floating-Point Algorithm

Example (Cody & Waite)

```
Notation pow2 := (bpow radix2).
Notation rnd :=
  (round radix2 (FLT_exp (-1074) 53) ZnearestE).
Definition add x y := rnd (x + y).
...
Definition nearbyint x :=
  round radix2 (FIX_exp 0) ZnearestE x.

Definition Log2h := 3048493539143 * pow2 (-42).
...

Definition cw_exp (x : R) :=
  let k := nearbyint (mul x InvLog2) in
  let t := sub (sub x (mul k Log2h)) (mul k Log2l) in
  let t2 := mul t t in
  let p := add p0 (mul t2 (add p1 (mul t2 p2))) in
  let q := add q0 (mul t2 (add q1 (mul t2 q2))) in
  pow2 (Zfloor k + 1) *
    (add (div (mul t p) (sub q (mul t p))) (1/2)).
```

Correctness

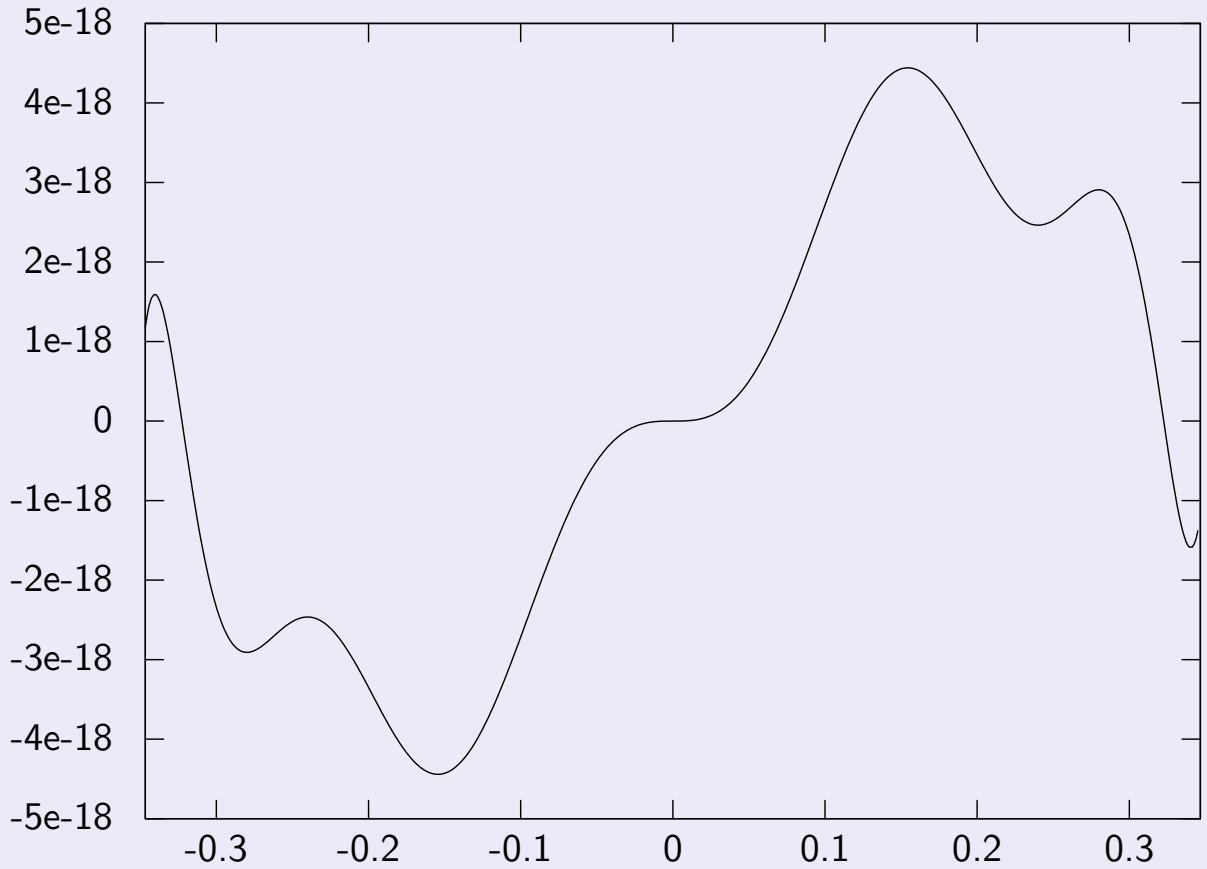
```
Definition cw_exp (x : R) :=
  let k := nearbyint (mul x InvLog2) in
  let t := sub (sub x (mul k Log2h)) (mul k Log2l) in
  let t2 := mul t t in
  let p := add p0 (mul t2 (add p1 (mul t2 p2))) in
  let q := add q0 (mul t2 (add q1 (mul t2 q2))) in
  pow2 (Zfloor k + 1) *
    (add (div (mul t p) (sub q (mul t p))) (1/2)).

Theorem exp_correct :
  forall x : R,
  generic_format radix2 (FLT_exp (-1074) 53) x ->
  -746 <= x <= 710 ->
  Rabs ((cw_exp x - exp x) / exp x) <= 1 * pow2 (-51).
```

Outline

Method Error

Relative error between $2f(t)$ and $\exp t$ for $|t| \leq 355 \cdot 2^{-10}$.

All the operations are assumed to be infinitely precise.



Bounding Method Error using CoqInterval

```
Lemma method_error :
  forall t : R,
  let t2 := t * t in
  let p := p0 + t2 * (p1 + t2 * p2) in
  let q := q0 + t2 * (q1 + t2 * q2) in
  let f := (t * p) / (q - t * p) + 1/2 in
  Rabs t <= 355 / 1024 ->
  Rabs ((2*f - exp t) / exp t) <= 23 * pow2 (-62).
Proof.
intros t t2 p q f Ht.
unfold f, q, p, t2, p0, p1, p2, q0, q1, q2.
interval with (i_bisect_taylor t 9, i_prec 70).
Qed.
```

Outline

Searching for Proofs using Gappa

How to prove . . . ?

$$x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n, P_1, \ldots, P_m \vdash \bot.$$

Searching for Proofs using Gappa

How to prove . . . ?

$$x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n, P_1, \ldots, P_m \vdash \bot.$$

Gappa

1. symbolical instantiation of theorems (backward reasoning)

Searching for Proofs using Gappa

How to prove . . . ?

$$x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n, P_1, \ldots, P_m \vdash \bot.$$

Gappa

1. symbolical instantiation of theorems (backward reasoning)
2. numerical instantiation of theorems (forward reasoning)

Searching for Proofs using Gappa

How to prove . . . ?

$$x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n, P_1, \ldots, P_m \vdash \bot.$$

Gappa

1. symbolical instantiation of theorems (backward reasoning)
2. numerical instantiation of theorems (forward reasoning)
3. proof simplification

Searching for Proofs using Gappa

How to prove . . . ?

$$x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n, P_1, \ldots, P_m \vdash \bot.$$

Gappa

1. symbolical instantiation of theorems (backward reasoning)
2. numerical instantiation of theorems (forward reasoning)
3. proof simplification
4. Coq proof generation

Searching for Proofs using Gappa

Interval arithmetic

$$u + v \in \mathbf{w} \quad \Leftarrow \quad u \in \mathbf{u} \wedge v \in \mathbf{v} \wedge \mathbf{u} + \mathbf{v} \subseteq \mathbf{w}.$$

Searching for Proofs using Gappa

Interval arithmetic

$$u + v \in \mathbf{w} \quad \Leftarrow \quad u \in \mathbf{u} \wedge v \in \mathbf{v} \wedge \mathbf{u} + \mathbf{v} \subseteq \mathbf{w}.$$

Refinement

$$x \in \mathbf{x} \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge x \in \mathbf{x_2} \wedge \mathbf{x_1} \cap \mathbf{x_2} \subseteq \mathbf{x},$$

$$x \in \mathbf{x} \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge x \notin \mathbf{x_2} \wedge \mathbf{x_1} \setminus \mathbf{x_2} \subseteq \mathbf{x}.$$

Searching for Proofs using Gappa

Interval arithmetic

$$u + v \in \mathbf{w} \quad \Leftarrow \quad u \in \mathbf{u} \wedge v \in \mathbf{v} \wedge \mathbf{u} + \mathbf{v} \subseteq \mathbf{w}.$$

Refinement

$$x \in \mathbf{x} \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge x \in \mathbf{x_2} \wedge \mathbf{x_1} \cap \mathbf{x_2} \subseteq \mathbf{x},$$
$$x \in \mathbf{x} \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge x \notin \mathbf{x_2} \wedge \mathbf{x_1} \setminus \mathbf{x_2} \subseteq \mathbf{x}.$$

Arbitrary formulas

$$P[\top] \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge P[x \in \mathbf{x_2}] \wedge \mathbf{x_1} \subseteq \mathbf{x_2},$$
$$P[\bot] \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge P[x \in \mathbf{x_2}] \wedge \mathbf{x_1} \cap \mathbf{x_2} = \emptyset,$$
$$P[\bot] \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge P[x \notin \mathbf{x_2}] \wedge \mathbf{x_1} \subseteq \mathbf{x_2},$$
$$P[\top] \quad \Leftarrow \quad x \in \mathbf{x_1} \wedge P[x \notin \mathbf{x_2}] \wedge \mathbf{x_1} \cap \mathbf{x_2} = \emptyset.$$

Searching for Proofs using Gappa

Equality

$$x \in \mathbf{z} \quad \Leftarrow \quad y \in \mathbf{z} \wedge x = y.$$

Searching for Proofs using Gappa

Equality

$$x \in \mathbf{z} \quad \Leftarrow \quad y \in \mathbf{z} \wedge x = y.$$

Backward propagation

$$u = (u \cdot v)/v \quad \Leftarrow \quad u \neq 0 \wedge \mathcal{U}[u \cdot v],$$
$$v = (u \cdot v)/u \quad \Leftarrow \quad u \neq 0 \wedge \mathcal{U}[u \cdot v].$$

Searching for Proofs using Gappa

Equality

$$x \in \mathbf{z} \quad \Leftarrow \quad y \in \mathbf{z} \wedge x = y.$$

Backward propagation

$$u = (u \cdot v)/v \quad \Leftarrow \quad u \neq 0 \wedge \mathcal{U}[u \cdot v],$$
$$v = (u \cdot v)/u \quad \Leftarrow \quad u \neq 0 \wedge \mathcal{U}[u \cdot v].$$

Nonzero

$$u \neq 0 \quad \Leftarrow \quad |u| \in \mathbf{u} \wedge 0 \notin \mathbf{u},$$
$$u \cdot v \neq 0 \quad \Leftarrow \quad u \neq 0 \wedge v \neq 0.$$

Outline

Round-Off Errors and Dependency Effect

Bounding errors through rewriting

- $(\tilde{u} - \tilde{v}) - (u - v)$
- $\dfrac{\tilde{u} \cdot \tilde{v} - u \cdot v}{u \cdot v}$

Round-Off Errors and Dependency Effect

Bounding errors through rewriting

- $(\tilde{u} - \tilde{v}) - (u - v) = (\tilde{u} - u) - (\tilde{v} - v)$.
- $\dfrac{\tilde{u} \cdot \tilde{v} - u \cdot v}{u \cdot v}$

Round-Off Errors and Dependency Effect

Bounding errors through rewriting

- $(\tilde{u} - \tilde{v}) - (u - v) = (\tilde{u} - u) - (\tilde{v} - v)$.
- $\dfrac{\tilde{u} \cdot \tilde{v} - u \cdot v}{u \cdot v} = \dfrac{\tilde{u} - u}{u} + \dfrac{\tilde{v} - v}{v} + \dfrac{\tilde{u} - u}{u} \cdot \dfrac{\tilde{v} - v}{v}$.

Round-Off Errors and Dependency Effect

Bounding errors through rewriting

- $(\tilde{u} - \tilde{v}) - (u - v) = (\tilde{u} - u) - (\tilde{v} - v)$.
- $\dfrac{\tilde{u} \cdot \tilde{v} - u \cdot v}{u \cdot v} = \dfrac{\tilde{u} - u}{u} + \dfrac{\tilde{v} - v}{v} + \dfrac{\tilde{u} - u}{u} \cdot \dfrac{\tilde{v} - v}{v}$.

Work only on structurally similar subexpressions.

Dissimilar Subexpressions

How to bound . . . ?

$$x - \circ(\lfloor x \cdot \texttt{InvLog2} \rceil \cdot \texttt{Log2h}).$$

Dissimilar Subexpressions

How to bound . . . ?

$$x - \circ(\lfloor x \cdot \texttt{InvLog2} \rceil \cdot \texttt{Log2h}).$$

It would be simpler for . . .

$$(x \cdot \texttt{InvLog2}) \cdot \texttt{InvLog2}^{-1} - \circ(\lfloor x \cdot \texttt{InvLog2} \rceil \cdot \texttt{Log2h}).$$

Dissimilar Subexpressions

How to bound . . . ?

$$x - \circ(\lfloor x \cdot \mathtt{InvLog2} \rceil \cdot \mathtt{Log2h}).$$

It would be simpler for . . .

$$(x \cdot \mathtt{InvLog2}) \cdot \mathtt{InvLog2}^{-1} - \circ(\lfloor x \cdot \mathtt{InvLog2} \rceil \cdot \mathtt{Log2h}).$$

Tell Gappa that . . .

$$x = x \cdot \mathtt{InvLog2} \cdot \mathtt{InvLog2}^{-1}.$$

Outline

Proof Summary

Argument reduction

- Case $|x| \leq \frac{5}{16}$ using a manual proof and gappa.
- $x = x \cdot \mathtt{InvLog2} \cdot \mathtt{InvLog2}^{-1}$ using `field`.
- $x - k \cdot \log 2 = x - k \cdot \mathtt{Log2h} - k \cdot (\log 2 - \mathtt{Log2h})$ using `ring`.
- $\mathtt{Log2l} - (\log 2 - \mathtt{Log2h}) \in [-2^{-102}; 0]$ using `interval`.
- Range and absolute error using gappa.

Proof Summary

Argument reduction

- Case $|x| \leq \frac{5}{16}$ using a manual proof and gappa.
- $x = x \cdot \texttt{InvLog2} \cdot \texttt{InvLog2}^{-1}$ using `field`.
- $x - k \cdot \log 2 = x - k \cdot \texttt{Log2h} - k \cdot (\log 2 - \texttt{Log2h})$ using `ring`.
- $\texttt{Log2l} - (\log 2 - \texttt{Log2h}) \in [-2^{-102}; 0]$ using `interval`.
- Range and absolute error using gappa.

Entire algorithm

- Relative method error using `interval`.
- Error between $\exp t$ and $\exp(x - k \log 2)$ using `interval`.
- Relative error using gappa.

Proof Summary

Argument reduction

- Case $|x| \leq \frac{5}{16}$ using a manual proof and gappa.
- $x = x \cdot \texttt{InvLog2} \cdot \texttt{InvLog2}^{-1}$ using `field`.
- $x - k \cdot \log 2 = x - k \cdot \texttt{Log2h} - k \cdot (\log 2 - \texttt{Log2h})$ using `ring`.
- $\texttt{Log2l} - (\log 2 - \texttt{Log2h}) \in [-2^{-102}; 0]$ using `interval`.
- Range and absolute error using gappa.

Entire algorithm

- Relative method error using `interval`.
- Error between $\exp t$ and $\exp(x - k \log 2)$ using `interval`.
- Relative error using gappa.

A few hours of work, 65 lines of Coq script. ✓