Automatic Asymptotics in Isabelle/HOL

Manuel Eberl

Technische Universität München

8 June 2017

Agenda

1. What is Isabelle?
2. My work

What is Isabelle?

Isabelle

- One of the most popular ITPs (others: Coq, HOL Light, Mizar, PVS)

Isabelle

- One of the most popular ITPs (others: Coq, HOL Light, Mizar, PVS)
- Created in 1986 by Larry Paulson in Cambridge

Isabelle

- One of the most popular ITPs (others: Coq, HOL Light, Mizar, PVS)
- Created in 1986 by Larry Paulson in Cambridge
- A *generic* proof assistant that can handle different kinds of logics

Isabelle

- One of the most popular ITPs (others: Coq, HOL Light, Mizar, PVS)
- Created in 1986 by Larry Paulson in Cambridge
- A *generic* proof assistant that can handle different kinds of logics
- Nowadays mostly used with *Higher-Order Logic* (Isabelle/HOL)

Isabelle

- One of the most popular ITPs (others: Coq, HOL Light, Mizar, PVS)
- Created in 1986 by Larry Paulson in Cambridge
- A *generic* proof assistant that can handle different kinds of logics
- Nowadays mostly used with *Higher-Order Logic* (Isabelle/HOL)

HOL contains TND and choice, and many tools use them!

Isabelle

Isabelle developed out of the tradition of LCF theorem provers.

Idealised notion:

- Small kernel that implements basic logical inference

Isabelle

Isabelle developed out of the tradition of LCF theorem provers.

Idealised notion:

- Small kernel that implements basic logical inference
- Abstract type `thm` for proven facts

Isabelle

Isabelle developed out of the tradition of LCF theorem provers.

Idealised notion:

- Small kernel that implements basic logical inference
- Abstract type `thm` for proven facts
- Only the kernel can produce theorems

Isabelle

Isabelle developed out of the tradition of LCF theorem provers.

Idealised notion:

- Small kernel that implements basic logical inference
- Abstract type `thm` for proven facts
- Only the kernel can produce theorems
- Every proof procedure has to go through the kernel

Isabelle

Isabelle developed out of the tradition of LCF theorem provers.

Idealised notion:

- Small kernel that implements basic logical inference
- Abstract type `thm` for proven facts
- Only the kernel can produce theorems
- Every proof procedure has to go through the kernel

This ensures high trustworthiness of proofs.

Isabelle

Isabelle developed out of the tradition of LCF theorem provers.

Idealised notion:

- Small kernel that implements basic logical inference
- Abstract type `thm` for proven facts
- Only the kernel can produce theorems
- Every proof procedure has to go through the kernel

This ensures high trustworthiness of proofs.
(Reality is a bit more complicated)

Isabelle

Important features:

- Proof IDE based on jEdit

Isabelle

Important features:

- Proof IDE based on jEdit
- Structured proof language *Isar*

Isabelle

Important features:

- Proof IDE based on jEdit
- Structured proof language *Isar*
- Good automation

Isabelle

Important features:

- Proof IDE based on jEdit
- Structured proof language *Isar*
- Good automation
- Automatic counterexample search (QuickCheck, Nitpick)

Isabelle

Important features:

- Proof IDE based on jEdit
- Structured proof language *Isar*
- Good automation
- Automatic counterexample search (QuickCheck, Nitpick)
- Big mathematical library

Isabelle

Important features:

- Proof IDE based on jEdit
- Structured proof language *Isar*
- Good automation
- Automatic counterexample search (QuickCheck, Nitpick)
- Big mathematical library
- Even bigger *Archive of Formal Proofs*

Isabelle

Important features:

- Proof IDE based on jEdit
- Structured proof language *Isar*
- Good automation
- Automatic counterexample search (QuickCheck, Nitpick)
- Big mathematical library
- Even bigger *Archive of Formal Proofs*
- Code export to SML/OCaml/Scala/Haskell

Isar

Isar is a structured proof language that supports *forward* reasoning.

Isar

Isar is a structured proof language that supports *forward* reasoning.

Example

```
lemma closed_imp_lims_in_set:
  fixes A :: ‹('a :: metric_space) set›
  assumes ‹closed A› and ‹range f ⊆ A› and ‹f ⟶ x›
  shows ‹x ∈ A›
proof
  assume ‹x ∉ A›
  from ‹closed A› have ‹open (-A)›
  with ‹x ∉ A› obtain ε where ‹ε > 0› and ‹∀x'. dist x' x < ε ⟶ x' ∉ A›
  moreover from ‹f ⟶ x› and ‹ε > 0› obtain n where ‹dist (f n) x < ε›
  ultimately have ‹f n ∉ A›
  with ‹range f ⊆ A› show False
qed
```

Isar

Example

```
lemma closed_imp_lims_in_set:
  fixes A :: ‹('a :: metric_space) set›
  assumes ‹closed A› and ‹range f ⊆ A› and ‹f ⟶ x›
  shows ‹x ∈ A›
proof (rule ccontr)
  assume ‹x ∉ A›
  from ‹closed A› have ‹open (-A)› by auto
  with ‹x ∉ A› obtain ε where ‹ε > 0› and ‹∀x'. dist x' x < ε ⟶ x' ∉ A›
    by (auto simp: dist_norm open_dist)
  moreover from ‹f ⟶ x› and ‹ε > 0› obtain n where ‹dist (f n) x < ε›
    by (auto simp: tendsto_iff eventually_at_top_linorder)
  ultimately have ‹f n ∉ A› by auto
  with ‹range f ⊆ A› show False by auto
qed
```

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers
- Lists, algebraic datatypes, recursive functions, quotients

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers
- Lists, algebraic datatypes, recursive functions, quotients
- Basic algebra and number theory

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers
- Lists, algebraic datatypes, recursive functions, quotients
- Basic algebra and number theory
- Topology, limits, derivatives, integrals

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers
- Lists, algebraic datatypes, recursive functions, quotients
- Basic algebra and number theory
- Topology, limits, derivatives, integrals
- Measure and Probability theory

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers
- Lists, algebraic datatypes, recursive functions, quotients
- Basic algebra and number theory
- Topology, limits, derivatives, integrals
- Measure and Probability theory

The library is continuously extended and all old material ported to new versions.

The Library

The Isabelle distribution contains a large library of definitions and facts in HOL:

- Natural numbers, integers, reals, complex numbers
- Lists, algebraic datatypes, recursive functions, quotients
- Basic algebra and number theory
- Topology, limits, derivatives, integrals
- Measure and Probability theory

The library is continuously extended and all old material ported to new versions.

The *Archive of Formal Proofs* contains even more material that is continuously kept up-to-date.

Big Applications

Isabelle/HOL has been used for some big projects:

- Gödel's Incompleteness Theorems (Paulson, Isabelle, 2013)

Big Applications

Isabelle/HOL has been used for some big projects:

- Gödel's Incompleteness Theorems (Paulson, Isabelle, 2013)
- Kepler conjecture (Hales *et al.*, HOL Light + Isabelle, 2014)

Big Applications

Isabelle/HOL has been used for some big projects:

- Gödel's Incompleteness Theorems (Paulson, Isabelle, 2013)
- Kepler conjecture (Hales *et al.*, HOL Light + Isabelle, 2014)
- seL4 microkernel (Klein *et al.*, Isabelle, 2010)

Big Applications

Isabelle/HOL has been used for some big projects:

- Gödel's Incompleteness Theorems (Paulson, Isabelle, 2013)
- Kepler conjecture (Hales *et al.*, HOL Light + Isabelle, 2014)
- seL4 microkernel (Klein *et al.*, Isabelle, 2010)
- LTL model checker (Esparza *et al.*, Isabelle, 2013)

My Work

My main interests:

- Bringing more mathematics to Isabelle

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My PhD thesis:

Formally Verified Real Asymptotics

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My PhD thesis:

Formally Verified Real Asymptotics

Completed so far:

- Landau symbols

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My PhD thesis:

Formally Verified Real Asymptotics

Completed so far:

- Landau symbols
- Proof of the *Akra–Bazzi theorem*

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My PhD thesis:

Formally Verified Real Asymptotics

Completed so far:

- Landau symbols
- Proof of the *Akra–Bazzi theorem*
- Solving linear recurrences

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My PhD thesis:

Formally Verified Real Asymptotics

Completed so far:

- Landau symbols
- Proof of the *Akra–Bazzi theorem*
- Solving linear recurrences
- Euler–MacLaurin formula

My main interests:

- Bringing more mathematics to Isabelle
- Making Isabelle easier to use for mathematics

My PhD thesis:

Formally Verified Real Asymptotics

Completed so far:

- Landau symbols
- Proof of the *Akra–Bazzi theorem*
- Solving linear recurrences
- Euler–MacLaurin formula
- Asymptotics of factorial, Γ, $\psi^{(n)}$, erf, H_n

Some nice things we can easily do with these tools:

- Analyse complexity of divide & conquer algorithms: Merge Sort, Karatsuba multiplication, selection

Some nice things we can easily do with these tools:

- Analyse complexity of divide & conquer algorithms: Merge Sort, Karatsuba multiplication, selection
- Complexity of QuickSort

Some nice things we can easily do with these tools:

- Analyse complexity of divide & conquer algorithms: Merge Sort, Karatsuba multiplication, selection
- Complexity of QuickSort
- $\Omega(n \log n)$ lower bound for comparison sorts

Some nice things we can easily do with these tools:

- Analyse complexity of divide & conquer algorithms: Merge Sort, Karatsuba multiplication, selection
- Complexity of QuickSort
- $\Omega(n \log n)$ lower bound for comparison sorts
- Average number of integer divisors/co-primes/square-free integers

Some nice things we can easily do with these tools:

- Analyse complexity of divide & conquer algorithms: Merge Sort, Karatsuba multiplication, selection
- Complexity of QuickSort
- $\Omega(n \log n)$ lower bound for comparison sorts
- Average number of integer divisors/co-primes/square-free integers

What could be a suitable ambitious new project?

Problem: Asymptotics in Isabelle are ugly to prove!

Example: Lemma required for Akra–Bazzi

$$\lim_{x\to\infty} \left(1 - \frac{1}{b\log^{1+\varepsilon} x}\right)^p \left(1 + \frac{1}{\log^{\varepsilon/2}\left(bx + \frac{x}{\log^{1+\varepsilon} x}\right)}\right) - \left(1 + \frac{1}{\log^{\varepsilon/2} x}\right) = 0^+$$

Problem: Asymptotics in Isabelle are ugly to prove!

Example: Lemma required for Akra–Bazzi

$$\lim_{x\to\infty} \left(1 - \frac{1}{b \log^{1+\varepsilon} x}\right)^p \left(1 + \frac{1}{\log^{\varepsilon/2}\left(bx + \frac{x}{\log^{1+\varepsilon} x}\right)}\right) - \left(1 + \frac{1}{\log^{\varepsilon/2} x}\right) \;=\; 0^+$$

Original author: ‘Trivial, just Taylor-expand it!’

Problem: Asymptotics in Isabelle are ugly to prove!

Example: Lemma required for Akra–Bazzi

$$\lim_{x\to\infty} \left(1 - \frac{1}{b\log^{1+\varepsilon} x}\right)^p \left(1 + \frac{1}{\log^{\varepsilon/2}\left(bx + \frac{x}{\log^{1+\varepsilon} x}\right)}\right) - \left(1 + \frac{1}{\log^{\varepsilon/2} x}\right) \;=\; 0^+$$

Original author: 'Trivial, just Taylor-expand it!'

In Isabelle: 700 lines of messy proofs

lemma akra_bazzi_aux:
filterlim
$(\lambda x.\ (1 - 1/(b * \ln x\,\hat{}\,(1+\varepsilon))\,\hat{}\,p) *$
$(1 + \ln (b * x + x/\ln x\,\hat{}\,(1+\varepsilon))\,\hat{}\,(-\varepsilon/2)) -$
$(1 + \ln x\,\hat{}\,(-\varepsilon/2)))$
(at_right 0) at_top

lemma akra_bazzi_aux:
filterlim
$(\lambda x.\ (1 - 1/(b * \ln x\,\hat{}\,(1+\varepsilon))\,\hat{}\,p) *$
$(1 + \ln\ (b * x + x/\ln x\,\hat{}\,(1+\varepsilon))\,\hat{}\,(-\varepsilon/2)) -$
$(1 + \ln x\,\hat{}\,(-\varepsilon/2)))$
(at_right 0) at_top
by magic

lemma akra_bazzi_aux:
filterlim
$(\lambda x.\ (1 - 1/(b * \ln x \,\hat{}\, (1+\varepsilon)) \,\hat{}\, p) *$
$(1 + \ln (b * x + x/\ln x \,\hat{}\, (1+\varepsilon)) \,\hat{}\, (-\varepsilon/2)) -$
$(1 + \ln x \,\hat{}\, (-\varepsilon/2)))$
(at_right 0) at_top
by magic

This is what we would like to have.

lemma akra_bazzi_aux:
filterlim
$(\lambda x.\ (1 - 1/(b * \ln x \,\hat{}\, (1+\varepsilon)) \,\hat{}\, p) *$
$(1 + \ln (b * x + x/\ln x \,\hat{}\, (1+\varepsilon)) \,\hat{}\, (-\varepsilon/2)) -$
$(1 + \ln x \,\hat{}\, (-\varepsilon/2)))$
(at_right 0) at_top
by magic

This is what we would like to have.

Computer Algebra Systems can do this (sort of)

So why can't we?

Asymptotic Expansions

Asymptotic Expansions

Disclaimer:

- None of this was invented by me.

Asymptotic Expansions

Disclaimer:

- None of this was invented by me.
- I will not show actual Isabelle code for better readability.

Related Work

- *Asymptotic Expansions of exp–log Functions* by Richardson, Salvy, Shackell, van der Hoeven
- *On Computing Limits in a Symbolic Manipulation System* by Gruntz

We're looking for a compositional approach:

We're looking for a compositional approach:

Given the limits of $f(x)$ and $g(x)$, what is the limit of $f(x) \square g(x)$? (for $\square \in \{+, -, \cdot, /\}$)

We're looking for a compositional approach:

Given the limits of $f(x)$ and $g(x)$, what is the limit of $f(x) \square g(x)$? (for $\square \in \{+, -, \cdot, /\}$)

If the limits are $\in \mathbb{R}$: Obvious.

We're looking for a compositional approach:

Given the limits of $f(x)$ and $g(x)$, what is the limit of $f(x) \square g(x)$? (for $\square \in \{+, -, \cdot, /\}$)

If the limits are $\in \mathbb{R}$: Obvious.

But: $\infty - \infty = ?$ $\quad 0 \cdot \infty = ?$

We need to track more information than just the limit!

We need the *full asymptotic information*

Asymptotic Expansions

For $x \to 0$, we have:

$$e^x \sim 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \dots$$

$$\frac{1}{1+x} \sim 1 - x + x^2 - x^3 + \dots$$

Asymptotic Expansions

For $x \to 0$, we have:

$$e^x \sim 1 + x + \tfrac{1}{2}x^2 + \tfrac{1}{6}x^3 + \dots$$

$$\frac{1}{1+x} \sim 1 - x + x^2 - x^3 + \dots$$

This means: Cutting off $f(x) \sim a_0(x) + a_1(x) + \dots$ at term a_n yields error $O(a_{n+1}(x))$.

Asymptotic Expansions

For $x \to 0$, we have:

$$e^x \sim 1 + x + \tfrac{1}{2}x^2 + \tfrac{1}{6}x^3 + \dots$$

$$\frac{1}{1+x} \sim 1 - x + x^2 - x^3 + \dots$$

This means: Cutting off $f(x) \sim a_0(x) + a_1(x) + \dots$ at term a_n yields error $O(a_{n+1}(x))$.

Expansions contain the *full* asymptotic information.

Asymptotic Expansions

For $x \to 0$, we have:

$$e^x \sim 1 + x + \tfrac{1}{2}x^2 + \tfrac{1}{6}x^3 + \ldots$$

$$\frac{1}{1+x} \sim 1 - x + x^2 - x^3 + \ldots$$

This means: Cutting off $f(x) \sim a_0(x) + a_1(x) + \ldots$ at term a_n yields error $O(a_{n+1}(x))$.

Expansions contain the *full* asymptotic information.

They can be added/subtracted/multiplied/divided.

Asymptotic Expansions

For $x \to 0$, we have:

$$e^x \sim 1 + x + \tfrac{1}{2}x^2 + \tfrac{1}{6}x^3 + \dots$$

$$\frac{1}{1+x} \sim 1 - x + x^2 - x^3 + \dots$$

This means: Cutting off $f(x) \sim a_0(x) + a_1(x) + \dots$ at term a_n yields error $O(a_{n+1}(x))$.

Expansions contain the *full* asymptotic information.

They can be added/subtracted/multiplied/divided.

Limits can simply be 'read off'

Asymptotic Expansions

Not all functions have such easy expansions!
e. g. exp (at $\pm\infty$) and ln (at ∞, 0)

Asymptotic Expansions

Not all functions have such easy expansions!
e. g. exp (at $\pm\infty$) and ln (at ∞, 0)

Solution: later

Asymptotic Expansions

Not all functions have such easy expansions!
e. g. exp (at $\pm\infty$) and ln (at ∞, 0)

Solution: later

For now, we only consider expansions of the form

$$f(x) \sim c_0 x^{e_0} + c_1 x^{e_1} + \ldots$$

where $e_0 > e_1 > \ldots$

Asymptotic Expansions

How can one do concrete operations on these expansions?

Asymptotic Expansions

How can one do concrete operations on these expansions?

$$\textbf{type}\ \mathsf{Exp}\ =\ (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

Asymptotic Expansions

How can one do concrete operations on these expansions?

$$\textbf{type}\ \mathsf{Exp}\ =\ (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\mathsf{negate}\ :\ \mathsf{Exp} \to \mathsf{Exp}$$
$$\mathsf{negate}\ xs\ =\ [(-c, e)\ |\ (c, e) \leftarrow xs]$$

Asymptotic Expansions

How can one do concrete operations on these expansions?

$$\textbf{type}\ \mathsf{Exp}\ =\ (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\begin{aligned}
&\mathsf{negate}\ :\ \mathsf{Exp} \to \mathsf{Exp} \\
&\mathsf{negate}\ xs\ =\ [(-c, e) \mid (c, e) \leftarrow xs]
\end{aligned}$$

$$\begin{aligned}
&(+)\ :\ \mathsf{Exp} \to \mathsf{Exp} \to \mathsf{Exp} \\
&[]\ +\ ys\ =\ ys \\
&xs\ +\ []\ =\ xs
\end{aligned}$$

Asymptotic Expansions

How can one do concrete operations on these expansions?

$$\textbf{type}\ \mathsf{Exp} \;=\; (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\begin{aligned}
&\mathsf{negate} \;:\; \mathsf{Exp} \to \mathsf{Exp} \\
&\mathsf{negate}\ xs \;=\; [(-c, e) \mid (c, e) \leftarrow xs]
\end{aligned}$$

$$\begin{aligned}
&(+) \;:\; \mathsf{Exp} \to \mathsf{Exp} \to \mathsf{Exp} \\
&[] \;+\; ys \;=\; ys \\
&xs \;+\; [] \;=\; xs \\
&((c_1, e_1) :: xs) \;+\; ((c_2, e_2) :: ys) \\
&\quad \mid e_1 == e_2 \;=\; (c_1 + c_2, e_1) \;::\; xs + ys \\
&\quad \mid e_1 < e_2 \;=\; (c_1, e_1) \;::\; xs + ((c_2, e_2) :: ys) \\
&\quad \mid e_1 > e_2 \;=\; (c_2, e_2) \;::\; ((c_1, e_1) :: xs) + ys
\end{aligned}$$

Asymptotic Expansions – Multiplication

Multiplication with 'atomic' factor $c'x^{e'}$:

$$\text{scale} : \mathbb{R} \to \mathbb{R} \to \text{Exp} \to \text{Exp}$$
$$\text{scale}\ c'\ e'\ xs \ = \ [(c * c', e + e') \mid (c, e) \leftarrow xs]$$

Asymptotic Expansions – Multiplication

Multiplication with ‘atomic’ factor $c' x^{e'}$:

$$\begin{aligned} &\text{scale} \,:\, \mathbb{R} \to \mathbb{R} \to \text{Exp} \to \text{Exp} \\ &\text{scale}\ c'\ e'\ xs \ = \ [(c * c', e + e') \mid (c, e) \leftarrow xs] \end{aligned}$$

Multiplication of two expansions:

$$\begin{aligned} &(*) \,:\, \text{Exp} \to \text{Exp} \to \text{Exp} \\ &xs \ * \ [] \ = \ [] \\ &[] \ * \ ys \ = \ [] \end{aligned}$$

Asymptotic Expansions – Multiplication

Multiplication with 'atomic' factor $c'x^{e'}$:

$$\begin{aligned}
&\mathsf{scale} : \mathbb{R} \to \mathbb{R} \to \mathsf{Exp} \to \mathsf{Exp}\\
&\mathsf{scale}\ c'\ e'\ xs \ = \ [(c * c', e + e') \mid (c, e) \leftarrow xs]
\end{aligned}$$

Multiplication of two expansions:

$$\begin{aligned}
&(*) : \mathsf{Exp} \to \mathsf{Exp} \to \mathsf{Exp}\\
&xs \ * \ [] \ = \ []\\
&[] \ * \ ys \ = \ []\\
&((c_1, e_1) :: xs) \ * \ ((c_2, e_2) :: ys) \ =\\
&\quad (c_1 * c_2, e_1 + e_2) \ :: \ \mathsf{scale}\ c_1\ e_1\ ys + xs * ((c_2, e_2) :: ys)
\end{aligned}$$

Asymptotic Expansions – Power Series

Many functions have local power series expansions:

$$g(x) = \sum_{n=0}^{\infty} c_n x^n \qquad (\text{for } |x| < R)$$

e. g. $c_n := 1/n!$ for the exponential function.

Asymptotic Expansions – Power Series

Many functions have local power series expansions:

$$g(x) = \sum_{n=0}^{\infty} c_n x^n \qquad (\text{for } |x| < R)$$

e. g. $c_n := 1/n!$ for the exponential function.

If $f(x) \to 0$, one can substitute f into g:

$$g(f(x)) = \sum_{n=0}^{\infty} c_n f(x)^n$$

Asymptotic Expansions – Power Series

Many functions have local power series expansions:

$$g(x) = \sum_{n=0}^{\infty} c_n x^n \qquad (\text{for } |x| < R)$$

e. g. $c_n := 1/n!$ for the exponential function.

If $f(x) \to 0$, one can substitute f into g:

$$g(f(x)) = \sum_{n=0}^{\infty} c_n f(x)^n = c_0 + f(x) \sum_{n=0}^{\infty} c_{n+1} f(x)^n$$

Asymptotic Expansions – Power Series

Many functions have local power series expansions:

$$g(x) = \sum_{n=0}^{\infty} c_n x^n \qquad (\text{for } |x| < R)$$

e. g. $c_n := 1/n!$ for the exponential function.

If $f(x) \to 0$, one can substitute f into g:

$$g(f(x)) = \sum_{n=0}^{\infty} c_n f(x)^n = c_0 + f(x) \sum_{n=0}^{\infty} c_{n+1} f(x)^n$$

$$\begin{aligned}
&\mathsf{powser} \;:\; \mathbb{R}\ \mathsf{llist} \to \mathsf{Exp} \to \mathsf{Exp} \\
&\mathsf{powser}\ (c :: cs)\ xs \;=\; (c, 0)\ ::\ xs * \mathsf{powser}\ cs\ xs
\end{aligned}$$

Asymptotic Expansions – Reciprocal

Consider $f(x)$ with expansion $(c, e) :: xs$ where $c \neq 0$, i. e.

$$f(x) = c\,x^e + g(x)$$

where $g(x)$ expands to xs.

Asymptotic Expansions – Reciprocal

Consider $f(x)$ with expansion $(c, e) :: xs$ where $c \neq 0$, i. e.

$$f(x) = c\,x^e + g(x) = c\,x^e(1 + c^{-1}x^{-e}g(x))$$

where $g(x)$ expands to xs.

Asymptotic Expansions – Reciprocal

Consider $f(x)$ with expansion $(c, e) :: xs$ where $c \neq 0$, i. e.

$$f(x) = c\,x^e + g(x) = c\,x^e(1 + c^{-1}x^{-e}g(x))$$

where $g(x)$ expands to xs. Then:

$$f^{-1}(x) \sim c^{-1}x^{-e}\,(1 + c^{-1}x^{-e}g(x))^{-1}$$

Asymptotic Expansions – Reciprocal

Consider $f(x)$ with expansion $(c, e) :: xs$ where $c \neq 0$, i. e.

$$f(x) = c\,x^e + g(x) = c\,x^e(1 + c^{-1}x^{-e}g(x))$$

where $g(x)$ expands to xs. Then:

$$f^{-1}(x) \sim c^{-1}x^{-e}\left(1 + c^{-1}x^{-e}g(x)\right)^{-1}$$

Note the geometric series:

$$(1 + x)^{-1} = \sum_{n=0}^{\infty}(-1)^n x^n$$

Asymptotic Expansions – Reciprocal

Consider $f(x)$ with expansion $(c, e) :: xs$ where $c \neq 0$, i. e.

$$f(x) = c\,x^e + g(x) = c\,x^e(1 + c^{-1}x^{-e}g(x))$$

where $g(x)$ expands to xs. Then:

$$f^{-1}(x) \sim c^{-1}x^{-e}\left(1 + c^{-1}x^{-e}g(x)\right)^{-1}$$

Note the geometric series:

$$(1+x)^{-1} = \sum_{n=0}^{\infty}(-1)^n x^n$$

Therefore:

$$\begin{aligned}&\text{inv } ((c, e) :: xs) = \text{scale } (1/c)\ (-e)\\&\qquad (\text{powser } (\text{cycle } [1, -1])\ (\text{scale } (1/c)\ (-e)\ xs))\end{aligned}$$

Asymptotic Expansions – Other operations

Problem: Remember: $\ln x$ and $\exp x$ have no power series expansion for $x \to \infty$!

Asymptotic Expansions – Other operations

Problem: Remember: $\ln x$ and $\exp x$ have no power series expansion for $x \to \infty$!

Solution: Allow not only powers of x, but *products* of powers of an *asymptotic basis*.

Asymptotic Expansions – Other operations

Problem: Remember: $\ln x$ and $\exp x$ have no power series expansion for $x \to \infty$!

Solution: Allow not only powers of x, but *products* of powers of an *asymptotic basis*.

Example: $(e^x, x, \ln x)$ is an asymptotic basis and generates monomials $e^{ax} x^b \ln^c x$

$$e^{4x} + 2x^3 \ln x \triangleq [1 \cdot (4, 0, 0), 2 \cdot (0, 3, 1)]$$

Asymptotic Expansions – Other operations

Problem: Remember: $\ln x$ and $\exp x$ have no power series expansion for $x \to \infty$!

Solution: Allow not only powers of x, but *products* of powers of an *asymptotic basis*.

Example: $(e^x, x, \ln x)$ is an asymptotic basis and generates monomials $e^{ax} x^b \ln^c x$

$$e^{4x} + 2x^3 \ln x \triangleq [1 \cdot (4, 0, 0), 2 \cdot (0, 3, 1)]$$

Alternative hierarchical view: Coefficients of an expansion w. r. t. basis $b :: bs$ are *functions*, each of which has an expansion w. r. t. bs.

Asymptotic Expansions – Other operations

Problem: Remember: $\ln x$ and $\exp x$ have no power series expansion for $x \to \infty$!

Solution: Allow not only powers of x, but *products* of powers of an *asymptotic basis*.

Example: $(e^x, x, \ln x)$ is an asymptotic basis and generates monomials $e^{ax} x^b \ln^c x$

$$e^{4x} + 2x^3 \ln x \triangleq [1 \cdot (4, 0, 0), 2 \cdot (0, 3, 1)]$$

Alternative hierarchical view: Coefficients of an expansion w. r. t. basis $b :: bs$ are *functions*, each of which has an expansion w. r. t. bs.

$$e^{4x} + 2x^3 \ln x \triangleq [(4, (0, (0, 1))), (0, (3, (1, 2)))$$

Asymptotic Expansions – Other operations

Reading off limits is still easy:

$$f(x) \sim c \cdot b_1(x)^{e_1} \dots b_n(x)^{e_n} \; + \; \dots$$

Asymptotic Expansions – Other operations

Reading off limits is still easy:

$$f(x) \sim c \cdot b_1(x)^{e_1} \dots b_n(x)^{e_n} \; + \; \dots$$

Just determine first non-zero e_i:

Asymptotic Expansions – Other operations

Reading off limits is still easy:

$$f(x) \sim c \cdot b_1(x)^{e_1} \dots b_n(x)^{e_n} \; + \; \dots$$

Just determine first non-zero e_i:

- Limit is 0 if $e_i < 0$

Asymptotic Expansions – Other operations

Reading off limits is still easy:

$$f(x) \sim c \cdot b_1(x)^{e_1} \dots b_n(x)^{e_n} \; + \; \dots$$

Just determine first non-zero e_i:

- Limit is 0 if $e_i < 0$
- Limit is $\mathrm{sgn}(c) \cdot \infty$ if $e_i > 0$

Asymptotic Expansions – Other operations

Reading off limits is still easy:

$$f(x) \sim c \cdot b_1(x)^{e_1} \dots b_n(x)^{e_n} \; + \; \dots$$

Just determine first non-zero e_i:

- Limit is 0 if $e_i < 0$
- Limit is $\mathrm{sgn}(c) \cdot \infty$ if $e_i > 0$
- Limit is c if all $e_i = 0$

Asymptotic Expansions – Other operations

Before:

$$\textbf{type}\ \mathsf{Exp} \ = \ (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\mathsf{negate} \ : \ \mathsf{Exp} \to \mathsf{Exp}$$
$$\mathsf{negate}\ xs \ = \ [(-c, e) \mid (c, e) \leftarrow xs]$$

Asymptotic Expansions – Other operations

Before:

$$\textbf{type}\ \mathsf{Exp}\ =\ (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\mathsf{negate}\ :\ \mathsf{Exp} \to \mathsf{Exp}$$
$$\mathsf{negate}\ xs\ =\ [(-c, e)\ |\ (c, e) \leftarrow xs]$$

Now:

$$\textbf{type}\ \mathsf{Basis}\ =\ (\mathbb{R} \to \mathbb{R})\ \mathsf{list}$$

Asymptotic Expansions – Other operations

Before:

$$\textbf{type}\ \mathsf{Exp}\ =\ (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\mathsf{negate}\ :\ \mathsf{Exp} \to \mathsf{Exp}$$
$$\mathsf{negate}\ xs\ =\ [(-c, e) \mid (c, e) \leftarrow xs]$$

Now:

$$\textbf{type}\ \mathsf{Basis}\ =\ (\mathbb{R} \to \mathbb{R})\ \mathsf{list}$$
$$\textbf{datatype}\ \mathsf{Exp}\ :\ \mathsf{Basis} \to \mathsf{Type}\ \ \textbf{where}$$
$$\quad \mathsf{Const}\ :\ \mathbb{R} \to \mathsf{Exp}\ []$$
$$\quad \mathsf{Exp}\ \ \ \ :\ (\mathsf{Exp}\ bs \times \mathbb{R})\ \mathsf{llist} \to \mathsf{Exp}\ (b :: bs)$$

Asymptotic Expansions – Other operations

Before:

$$\textbf{type}\ \mathsf{Exp}\ =\ (\mathbb{R}\times\mathbb{R})\ \mathsf{llist}$$

$$\begin{aligned}&\mathsf{negate}\ :\ \mathsf{Exp}\to\mathsf{Exp}\\&\mathsf{negate}\ xs\ =\ [(-c,e)\mid(c,e)\leftarrow xs]\end{aligned}$$

Now:

$$\begin{aligned}&\textbf{type}\ \mathsf{Basis}\ =\ (\mathbb{R}\to\mathbb{R})\ \mathsf{list}\\&\textbf{datatype}\ \mathsf{Exp}\ :\ \mathsf{Basis}\to\mathsf{Type}\ \ \textbf{where}\\&\quad\mathsf{Const}\ :\ \mathbb{R}\to\mathsf{Exp}\ []\\&\quad\mathsf{Exp}\quad\ \ :\ (\mathsf{Exp}\ bs\times\mathbb{R})\ \mathsf{llist}\to\mathsf{Exp}\ (b::bs)\end{aligned}$$

$$\begin{aligned}&\mathsf{negate}\ :\ \mathsf{Exp}\ bs\to\mathsf{Exp}\ bs\\&\mathsf{negate}\ (\mathsf{Const}\ c)\ =\ -c\end{aligned}$$

Asymptotic Expansions – Other operations

Before:

$$\textbf{type}\ \mathsf{Exp} \;=\; (\mathbb{R} \times \mathbb{R})\ \mathsf{llist}$$

$$\begin{aligned} &\mathsf{negate} : \mathsf{Exp} \to \mathsf{Exp} \\ &\mathsf{negate}\ xs \;=\; [(-c, e) \mid (c, e) \leftarrow xs] \end{aligned}$$

Now:

$$\begin{aligned} &\textbf{type}\ \mathsf{Basis} \;=\; (\mathbb{R} \to \mathbb{R})\ \mathsf{list} \\ &\textbf{datatype}\ \mathsf{Exp} : \mathsf{Basis} \to \mathsf{Type}\ \ \textbf{where} \\ &\quad \mathsf{Const} : \mathbb{R} \to \mathsf{Exp}\ [] \\ &\quad \mathsf{Exp} \quad\ : (\mathsf{Exp}\ bs \times \mathbb{R})\ \mathsf{llist} \to \mathsf{Exp}\ (b :: bs) \end{aligned}$$

$$\begin{aligned} &\mathsf{negate} : \mathsf{Exp}\ bs \to \mathsf{Exp}\ bs \\ &\mathsf{negate}\ (\mathsf{Const}\ c) \;=\; -c \\ &\mathsf{negate}\ (\mathsf{Exp}\ xs) \;\;\;=\; \mathsf{Exp}\ [(\mathsf{negate}\ c, e) \mid (c, e) \leftarrow xs] \end{aligned}$$

Asymptotic Expansions – Logarithm

Same trick as before: Given

$$f(x) = c(x)\, x^e + g(x) \quad \text{with } g(x) \sim xs$$

Asymptotic Expansions – Logarithm

Same trick as before: Given

$$f(x) = c(x)\,x^e + g(x) \quad \text{with } g(x) \sim xs$$

we rearrange

$$\ln(f(x)) = \ln(c(x)\,x^e(1 + c(x)^{-1}x^{-e}g(x)))$$

Asymptotic Expansions – Logarithm

Same trick as before: Given

$$f(x) = c(x)\,x^e + g(x) \quad \text{with } g(x) \sim xs$$

we rearrange

$$\begin{aligned}\ln(f(x)) &= \ln(c(x)\,x^e(1 + c(x)^{-1}x^{-e}g(x))) \\ &= \ln c(x) + e\ln x + \ln\left(1 + c(x)^{-1}x^{-e}g(x)\right)\end{aligned}$$

Asymptotic Expansions – Logarithm

Same trick as before: Given

$$f(x) = c(x)\, x^e + g(x) \quad \text{with } g(x) \sim xs$$

we rearrange

$$\begin{aligned} \ln(f(x)) &= \ln(c(x)\, x^e(1 + c(x)^{-1}x^{-e}g(x))) \\ &= \ln c(x) + e \ln x + \ln\left(1 + c(x)^{-1}x^{-e}g(x)\right) \end{aligned}$$

and $t \mapsto \ln(1 + t)$ has a power series expansion. :)

Asymptotic Expansions – Logarithm

Same trick as before: Given

$$f(x) = c(x)\,x^e + g(x) \quad \text{with } g(x) \sim xs$$

we rearrange

$$\begin{aligned}\ln(f(x)) &= \ln(c(x)\,x^e(1 + c(x)^{-1}x^{-e}g(x))) \\ &= \ln c(x) + e\ln x + \ln\left(1 + c(x)^{-1}x^{-e}g(x)\right)\end{aligned}$$

and $t \mapsto \ln(1+t)$ has a power series expansion. :)

We might have to add $\ln x$ to our basis!

Asymptotic Expansions – Exponential

Exponential is much more complicated – too complicated for these slides.

Asymptotic Expansions – Exponential

Exponential is much more complicated – too complicated for these slides.

- Lots of case distinctions

Asymptotic Expansions – Exponential

Exponential is much more complicated – too complicated for these slides.

- Lots of case distinctions
- Can introduce ugly new basis elements like $\exp(x + 1/x)$

Asymptotic Expansions – Exponential

Exponential is much more complicated – too complicated for these slides.

- Lots of case distinctions
- Can introduce ugly new basis elements like $\exp(x + 1/x)$
- Lots of opportunities for implementation bugs

Asymptotic Expansions – Exponential

Exponential is much more complicated – too complicated for these slides.

- Lots of case distinctions
- Can introduce ugly new basis elements like $\exp(x + 1/x)$
- Lots of opportunities for implementation bugs
- Luckily, the Isabelle kernel caught them, of course. :)

What this looks like in Isabelle

Type α ms for multiseries with coefficients of type α

$$\textbf{datatype}\ \alpha\ \mathsf{ms}\ =\ \mathsf{MS}\ \ \text{"}(\alpha \times \mathsf{real})\ \mathsf{llist}\text{"}\ \ \text{"}\mathsf{real} \Rightarrow \mathsf{real}\text{"}$$

E.g. expansion of order 3 would be 'real ms ms ms'.

What this looks like in Isabelle

Type α ms for multiseries with coefficients of type α

$$\textbf{datatype}\ \alpha\ \mathsf{ms}\ =\ \mathsf{MS}\ \ \text{"}(\alpha \times \mathsf{real})\ \mathsf{llist}\text{"}\ \ \text{"}\mathsf{real} \Rightarrow \mathsf{real}\text{"}$$

E.g. expansion of order 3 would be 'real ms ms ms'.

Operations on ms are defined with corecursion,
proven correct with coinduction

What this looks like in Isabelle

Type α ms for multiseries with coefficients of type α

$$\textbf{datatype}\ \alpha\ \text{ms}\ =\ \text{MS}\ \ \text{"}(\alpha \times \text{real})\ \text{llist"}\ \ \text{"real} \Rightarrow \text{real"}$$

E.g. expansion of order 3 would be 'real ms ms ms'.

Operations on ms are defined with corecursion,
proven correct with coinduction

How do we turn this into a proof method?

Computing Expansions in Isabelle

Evaluation

So far, we can write down expansions as HOL terms

Evaluation

So far, we can write down expansions as HOL terms
– but how do we evaluate them?

Evaluation

So far, we can write down expansions as HOL terms
– but how do we evaluate them?

Isabelle has tools to evaluate terms *strictly*, but we need lazy evaluation.

Evaluation

So far, we can write down expansions as HOL terms
– but how do we evaluate them?

Isabelle has tools to evaluate terms *strictly*, but we need lazy evaluation.

I had to implement lazy evaluation of HOL terms.

Evaluation

Lazy evaluation framework:

- set of supported (co-)datatypes and their constructors

Evaluation

Lazy evaluation framework:

- set of supported (co-)datatypes and their constructors
- set of supported functions

Evaluation

Lazy evaluation framework:

- set of supported (co-)datatypes and their constructors
- set of supported functions
- set of function equations of the form $f(r, s, t) = \ldots$

Evaluation

Lazy evaluation framework:

- set of supported (co-)datatypes and their constructors
- set of supported functions
- set of function equations of the form $f(r, s, t) = \ldots$

The framework can

- determine whether a pattern matches a term (modulo rewriting)

Evaluation

Lazy evaluation framework:

- set of supported (co-)datatypes and their constructors
- set of supported functions
- set of function equations of the form $f(r, s, t) = \ldots$

The framework can

- determine whether a pattern matches a term (modulo rewriting)
- bring a term into head-normal form

Evaluation

Lazy evaluation framework:

- set of supported (co-)datatypes and their constructors
- set of supported functions
- set of function equations of the form $f(r, s, t) = \ldots$

The framework can

- determine whether a pattern matches a term (modulo rewriting)
- bring a term into head-normal form
- produce an Isabelle theorem of the form
  original term = reduced term

Does not support sharing

Evaluation

Problem:

- Addition of expansions involves comparisons of real numbers

Evaluation

Problem:

- Addition of expansions involves comparisons of real numbers
- 'Trimming' expansions involves zeroness tests of real functions

Evaluation

Problem:

- Addition of expansions involves comparisons of real numbers
- 'Trimming' expansions involves zeroness tests of real functions
- Both of these are difficult or even undecidable

Evaluation

Solution: Heuristic approach using Isabelle's automation

Evaluation

Solution: Heuristic approach using Isabelle's automation

- Use automation to determine signs – might fail

Evaluation

Solution: Heuristic approach using Isabelle's automation

- Use automation to determine signs – might fail
- Use automation to determine if function is identically zero – might cause non-termination

Evaluation

Solution: Heuristic approach using Isabelle's automation

- Use automation to determine signs – might fail
- Use automation to determine if function is identically zero – might cause non-termination
- User may have to supply additional facts

Evaluation

Solution: Heuristic approach using Isabelle's automation

- Use automation to determine signs – might fail
- Use automation to determine if function is identically zero – might cause non-termination
- User may have to supply additional facts

This works surprisingly well

Evaluation

Solution: Heuristic approach using Isabelle's automation

- Use automation to determine signs – might fail
- Use automation to determine if function is identically zero – might cause non-termination
- User may have to supply additional facts

This works surprisingly well

Additional backends (user input/Mathematica/Maple) possible

Proof method

At this point, we have all the ingredients:

- Parse and pre-process input expression

Proof method

At this point, we have all the ingredients:

- Parse and pre-process input expression
- Compute expansions bottom-up

Proof method

At this point, we have all the ingredients:

- Parse and pre-process input expression
- Compute expansions bottom-up
- Use evaluation framework to trim expansions/determine signs whenever necessary

Proof method

At this point, we have all the ingredients:

- Parse and pre-process input expression
- Compute expansions bottom-up
- Use evaluation framework to trim expansions/determine signs whenever necessary
- In the end: Trim expansion to determine leading term

Proof method

With some pre-processing, we can automatically prove statements of the form

- $f(x) \longrightarrow c$
- $f(x) \sim g(x)$
- $f(x) \in L(g(x))$ for any Landau symbol L

as $x \to l$ for $l \in \mathbb{R} \cup \{\pm\infty\}$

Proof method

With some pre-processing, we can automatically prove statements of the form

- $f(x) \longrightarrow c$
- $f(x) \sim g(x)$
- $f(x) \in L(g(x))$ for any Landau symbol L

as $x \to l$ for $l \in \mathbb{R} \cup \{\pm\infty\}$

f and g can be built from $+\ -\ \cdot\ /$ ln exp min max ^ $|\cdot|$ $\sqrt[n]{\cdot}$ without restrictions

Proof method

With some pre-processing, we can automatically prove statements of the form

- $f(x) \longrightarrow c$
- $f(x) \sim g(x)$
- $f(x) \in L(g(x))$ for any Landau symbol L

as $x \to l$ for $l \in \mathbb{R} \cup \{\pm\infty\}$

f and g can be built from $+ \; - \; \cdot \; /$ ln exp min max ^ $|\cdot|$ $\sqrt[n]{\cdot}$ without restrictions

sin, cos, tan at finite points also possible.

Proof method

Example

lemma $(\lambda n.\ (1 + 1/n) \mathbin{\hat{}} n) \longrightarrow \exp\ 1$
   by exp_log_asymptotics

Proof method

Example

lemma $(\lambda n.\ (1 + 1/n) \hat{}\ n) \longrightarrow \exp\ 1$
  by exp_log_asymptotics

Example

lemma $((\lambda x.\ (1 + y/x) \hat{}\ x) \longrightarrow \exp\ y)$ at_top
proof (cases $y = 0$)
  case False
  thus ?thesis **by** exp_log_asymptotics
qed simp_all

Example

lemma
assumes $c > 1$ **and** $k > 0$
shows $(\lambda n.\ n \,\hat{}\, k) \in o(\lambda n.\ c \,\hat{}\, n)$
using assms **by** exp_log_asymptotics

Example

lemma
assumes $c > 1$ **and** $k > 0$
shows $(\lambda n.\ n\,\hat{}\,k) \in o(\lambda n.\ c\,\hat{}\,n)$
using assms **by** exp_log_asymptotics

Example

lemma akra_bazzi_aux:
assumes $b \in \{0 <..< 1\}$ **and** $\varepsilon > 0$
shows filterlim $(\lambda x.$
$(1 - H / (b * \ln x\,\hat{}(1 + \varepsilon)))\,\hat{}\,p *$
$(1 + \ln (b * x + H * x / \ln x\,\hat{}(1 + \varepsilon))\,\hat{}(-\varepsilon/2)) -$
$(1 + \ln x\,\hat{}(-\varepsilon/2)))$
(at_right 0) at_top

Example

lemma
  assumes $c > 1$ **and** $k > 0$
  shows $(\lambda n.\ n\,\hat{}\,k) \in o(\lambda n.\ c\,\hat{}\,n)$
using assms **by** exp_log_asymptotics

Example

lemma akra_bazzi_aux:
  assumes $b \in \{0{<}..{<}1\}$ **and** $\varepsilon > 0$
  shows filterlim $(\lambda x.$
$$(1 - H/(b * \ln x\,\hat{}(1+\varepsilon)))\,\hat{}\,p\, *$$
$$(1 + \ln (b * x + H * x/\ln x\,\hat{}(1+\varepsilon))\,\hat{}(-\varepsilon/2)) -$$
$$(1 + \ln x\,\hat{}(-\varepsilon/2)))$$
    (at_right 0) at_top
  by (exp_log_asymptotics simp: mult_neg_pos)

Discussion

What works well:

- Surprisingly, all examples I tried take no more than a few seconds

Discussion

What works well:

- Surprisingly, all examples I tried take no more than a few seconds
- Algorithm copes very well with free variables that don't affect result

Discussion

What works well:

- Surprisingly, all examples I tried take no more than a few seconds
- Algorithm copes very well with free variables that don't affect result

Problems:

- If several cancellations occur, performance gets very bad

Discussion

What works well:

- Surprisingly, all examples I tried take no more than a few seconds
- Algorithm copes very well with free variables that don't affect result

Problems:

- If several cancellations occur, performance gets very bad
- Getting zeroness/sign tests to work can be trial & error

Discussion

What works well:

- Surprisingly, all examples I tried take no more than a few seconds
- Algorithm copes very well with free variables that don't affect result

Problems:

- If several cancellations occur, performance gets very bad
- Getting zeroness/sign tests to work can be trial & error
- Case distinctions have to be done manually

Discussion

What works well:

- Surprisingly, all examples I tried take no more than a few seconds
- Algorithm copes very well with free variables that don't affect result

Problems:

- If several cancellations occur, performance gets very bad
- Getting zeroness/sign tests to work can be trial & error
- Case distinctions have to be done manually
- Somewhat 'ad-hoc' formalisation

Discussion

- 5000 lines of Isabelle theory

Discussion

- 5000 lines of Isabelle theory
- 3000 lines of (untrusted) ML code

Discussion

- 5000 lines of Isabelle theory
- 3000 lines of (untrusted) ML code
- About 5 months of work so far

Discussion

- 5000 lines of Isabelle theory
- 3000 lines of (untrusted) ML code
- About 5 months of work so far
- Implementation was tricky to get right

Comparison to CASs

Similar algorithm by Gruntz in Maple in 1996

Comparison to CASs

Similar algorithm by Gruntz in Maple in 1996
(is now part of Mathematica)

Comparison to CASs

Similar algorithm by Gruntz in Maple in 1996
(is now part of Mathematica)

Back then, all CASs gave wrong results
for many of his test cases!

Comparison to CASs

Similar algorithm by Gruntz in Maple in 1996
(is now part of Mathematica)

Back then, all CASs gave wrong results
for many of his test cases!

Nowadays, most of them work

Comparison to CASs

23 test cases lie in the fragment we support

Comparison to CASs

23 test cases lie in the fragment we support

All of them work automatically

Comparison to CASs

23 test cases lie in the fragment we support

All of them work automatically

Maximum time: 1.726 s; Median: 0.311 s

Comparison to CASs

23 test cases lie in the fragment we support

All of them work automatically

Maximum time: 1.726 s; Median: 0.311 s

Mathematica and Maple do all of them very quickly and correctly

Comparison to CASs

23 test cases lie in the fragment we support

All of them work automatically

Maximum time: 1.726 s; Median: 0.311 s

Mathematica and Maple do all of them very quickly and correctly

Maxima, Sage, and SymPy fail on some of them

Comparison to CASs

23 test cases lie in the fragment we support

All of them work automatically

Maximum time: 1.726 s; Median: 0.311 s

Mathematica and Maple do all of them very quickly and correctly

Maxima, Sage, and SymPy fail on some of them

Maxima and Sage take very long for some of them and give wrong result for this:

$$\exp\left(\frac{\log\log\left(x + e^{\log x \log\log x}\right)}{\log\log\log\left(e^x + x + \ln x\right)}\right) \longrightarrow e$$

Comparison to CASs

How well are we doing?

Surprisingly, we are not that much slower (sometimes even faster) than Maple/Mathematica on many examples

Comparison to CASs

How well are we doing?

Surprisingly, we are not that much slower (sometimes even faster) than Maple/Mathematica on many examples

Also: All CASs seem to fail on the Akra–Bazzi example as soon as variables are involved

Comparison to CASs

How well are we doing?

Surprisingly, we are not that much slower (sometimes even faster) than Maple/Mathematica on many examples

Also: All CASs seem to fail on the Akra–Bazzi example as soon as variables are involved

In general, of course, Mathematica/Maple are much better in both scope and speed

Comparison to CASs

How well are we doing?

Surprisingly, we are not that much slower (sometimes even faster) than Maple/Mathematica on many examples

Also: All CASs seem to fail on the Akra–Bazzi example as soon as variables are involved

In general, of course, Mathematica/Maple are much better in both scope and speed

But: you have to trust the implementations.

Comparison to CASs

How well are we doing?

Surprisingly, we are not that much slower (sometimes even faster) than Maple/Mathematica on many examples

Also: All CASs seem to fail on the Akra–Bazzi example as soon as variables are involved

In general, of course, Mathematica/Maple are much better in both scope and speed

But: you have to trust the implementations.

Isabelle still isn't a CAS – but we're getting there.

Future Work

- Incomplete support for Γ, $\psi^{(n)}$, arctan

Future Work

- Incomplete support for Γ, $\psi^{(n)}$, arctan
- Cannot handle oscillating functions or complex-valued asymptotics

Future Work

- Incomplete support for Γ, $\psi^{(n)}$, arctan
- Cannot handle oscillating functions or complex-valued asymptotics
- User interaction for zeroness tests could be improved

Questions? Demo?