

Using reflection to solve some differential equations



Guillaume Allais

COQTAIL Junior Laboratory
ENS Lyon

August, 26th

Motivations

A simple toy example

All the features

COQTAIL defines new objects

- Power series

```
an  : Rseq
rho : infinite_cv_radius an
=====
sum an rho : R -> R
```

- N^{th} derivative

```
n    : nat
f    : R -> R
Dnf  : D n f
=====
nth_derive f Dnf : R -> R
```

With specific properties

- Trivial identities

- $\sum a_n r^{n \rho_1} == \sum a_n r^{n \rho_2}$
- $\sum (a_n + b_n) r^{n \rho} == \sum a_n r^{n \rho} + \sum b_n r^{n \rho}$

- Interactions

- A power series can be differentiated infinitely many times
- The shape of these derivatives is simple

With specific properties

- Trivial identities

- $\sum a_n r^{n \rho_1} == \sum a_n r^{n \rho_2}$
- $\sum (a_n + b_n) r^{n \rho} == \sum a_n r^{n \rho} + \sum b_n r^{n \rho}$

- Interactions

- A power series can be differentiated infinitely many times
- The shape of these derivatives is simple

Do we really want to deal with this by hand?

Reflection

- A datatype representing formulas
- A semantics connecting the datatype to the formulas

A simple toy example

```
Inductive side_equa : Set :=  
  | y      : forall (p : nat) (k : nat), side_equa  
  | plus   : forall (s1 s2 : side_equa), side_equa.
```

First semantics

- From ASTs to power series

$$\begin{aligned} \llbracket y(p, k) \rrbracket_{\mathbb{R}} \rho &= (\sum_n \rho(p) x^n)^{(k)} \\ \llbracket plus(s_1, s_2) \rrbracket_{\mathbb{R}} \rho &= \llbracket s_1 \rrbracket_{\mathbb{R}} \rho + \llbracket s_2 \rrbracket_{\mathbb{R}} \rho \end{aligned}$$

Second semantics

- From ASTs to coefficients' sequences

$$\begin{aligned} \llbracket y(p, k) \rrbracket_{\mathbb{N}} \rho &= \left(\frac{(n+k)!}{n!} \rho(p)_{n+k} \right) \\ \llbracket plus(s_1, s_2) \rrbracket_{\mathbb{N}} \rho &= \llbracket s_1 \rrbracket_{\mathbb{N}} \rho + \llbracket s_2 \rrbracket_{\mathbb{N}} \rho \end{aligned}$$

Main theorem

We can talk about coefficients' sequences to prove equalities on the corresponding power series.

$$\llbracket s_1 :=: s_2 \rrbracket_{\mathbb{N}}(\text{map } \pi_1 \rho)$$

$$\Downarrow$$

$$\llbracket s_1 :=: s_2 \rrbracket_{\mathbb{R}} \rho$$

Main theorem

We can talk about coefficients' sequences to prove equalities on the corresponding power series.

$$\llbracket s_1 :=: s_2 \rrbracket_{\mathbb{N}}(\text{map } \pi_1 \rho)$$

\Downarrow

$$\llbracket s_1 :=: s_2 \rrbracket_{\mathbb{R}} \rho$$

Ltac

- Quoting
- Normalizing
- Solving

Ltac

- Quoting

- `isconst s x : \mathbb{B}`

- Normalizing

- Solving

Ltac

- Quoting

- `isconst s x : \mathbb{B}`
- `add_var an rho env : $\mathbb{N} \star \mathcal{E}$`

- Normalizing

- Solving

Ltac

- Quoting

- `isconst s x : \mathbb{B}`
- `add_var an rho env : $\mathbb{N} \star \mathcal{E}$`
- `quote_side_equa env s x : $\mathcal{E} \star \text{side_equa}$`

- Normalizing

- Solving

Ltac

■ Quoting

- `isconst s x : \mathbb{B}`
- `add_var an rho env : $\mathbb{N} \star \mathcal{E}$`
- `quote_side_equa env s x : $\mathcal{E} \star \text{side_equa}$`

■ Normalizing

- `normalize_rec p s x : unit`

■ Solving

Ltac

■ Quoting

- `isconst s x : \mathbb{B}`
- `add_var an rho env : $\mathbb{N} \star \mathcal{E}$`
- `quote_side_equa env s x : $\mathcal{E} \star \text{side_equa}$`

■ Normalizing

- `normalize_rec p s x : unit`

■ Solving

- `solve_diff_equa : unit`

Examples

```
an : Rseq
ra : infinite_cv_radius an
rb : infinite_cv_radius an
=====
sum an ra == sum an rb
```

Examples

```
an : Rseq
ra : infinite_cv_radius an
rb : infinite_cv_radius an
=====
sum an ra == sum an rb
```

```
(([an, ra] , (y(0,0), y(0,0)))
```

Examples

```
an : Rseq
ra : infinite_cv_radius an
rb : infinite_cv_radius an
=====
sum an ra == sum an rb
```

```
(([an, ra] , (y(0,0), y(0,0)))
```

```
nth_derive (sum an ra) (D_infty_Rpser an ra 0) ==
nth_derive (sum an ra) (D_infty_Rpser an ra 0)
```

Examples

```
an : Rseq
ra : infinite_cv_radius an
rb : infinite_cv_radius an
=====
sum an ra == sum an rb
```

```
(([an, ra] , (y(0,0), y(0,0)))
```

```
nth_derive (sum an ra) (D_infty_Rpser an ra 0) ==
nth_derive (sum an ra) (D_infty_Rpser an ra 0)
```

```
an == an
```

```
an  : Rseq
bn  : Rseq
rab : infinite_cv_radius (an + bn + bn)
ra  : infinite_cv_radius an
rb  : infinite_cv_radius bn
rc  : infinite_cv_radius bn
=====
sum (an + bn + bn) rab ==
sum bn rb + sum an ra + sum bn rc
```

- Just a toy example however...
 - Fully automatic
 - Quite reflects the actual implementation

All the features

```
Inductive side_equa : Set :=
  | cst   : forall (r : R), side_equa
  | scal  : forall (r : R) (s : side_equa), side_equa
  | y     : forall (p : nat) (k : nat) (a : R), side_equa
  | opp   : forall (s1 : side_equa), side_equa
  | min   : forall (s1 s2 : side_equa), side_equa
  | plus  : forall (s1 s2 : side_equa), side_equa
  | mult  : forall (s1 s2 : side_equa), side_equa.
```


Any questions?

Sources: `http://coqtail.sf.net`