What is in Store for Coq.Interval

maybe...

Guillaume Melquiond

Current situation

Coq.Interval provides:

1. *Exact* operations on type $\mathbb{R} \cup \{\bot_R\}$.
2. Rounded arithmetic operations on *floating-point* numbers $\mathbb{F} = \{m \cdot \beta^e \mid m, e \in \mathbb{Z}\} \cup \{\bot_F\}$.
   Note: m and e are unbounded.
3. Arithmetic operations on *inf-sup intervals* $\mathbb{I} = \mathbb{F}^2 \cup \{\bot_I\}$.
   Note: this extends the arithmetic operations on $\mathbb{R} \cup \{\bot_R\}$
4. Interval enclosures of some *elementary functions*: $\mathbb{F} \to \mathbb{I}$.
5. Interval elementary functions: $\mathbb{I} \to \mathbb{I}$.
6. *Tactic* based on bisection and automatic differentiation for solving user goals.

Exact arithmetic on $\mathbb{R} \cup \{\bot_R\}$

Partial functions are a pain; I should never have introduced $\bot_R$.
(Original rationale: ability to extend functions outside their domain.)

In *Coquelicot*'s world, all the functions are *total* and infinitely differentiable.
(Though the derivatives have meaningful values only if the functions are actually differentiable. Similarly for integrals, power series, and so on.)

Goal: switch Coq.Interval from Reals to Coquelicot.

Floating-point and interval arithmetic

$\mathbb{F} = \{m \cdot \beta^e \mid m, e \in \mathbb{Z}\} \cup \{\bot_F\}$ is fine for implementing a multi-precision floating-point arithmetic inside Coq.
Unfortunately, the interval arithmetic is tightly tied to this format.

As a consequence, one cannot plug a different arithmetic on bounds, e.g. MPFR or native floating-point numbers or computational real numbers.

Goal: sever the link between floating-point numbers and interval arithmetic.

Implement new elementary functions

Currently, all the elementary functions perform an argument reduction and evaluate an *alternated power series*.
So interval elementary functions only use interval arithmetic operations.

Unfortunately, everything goes wrong for other evaluation schemes.
For instance, *Newton's interval iteration* for logarithm depends on the interval evaluation of exponential, which is not yet available at that time.

Goal: find a saner layout for the library.

Improve the tactic

At order 1, differentiability is overrated; *slopes* would work better.

- $f(X) = f(x_0) + (X - x_0) \cdot f'(X)$,
- $f(X) = f(x_0) + (X - x_0) \cdot \{ \frac{f(x)-f(x_0)}{x-x_0} \mid x \in X \}$.

The tactic lacks some order-n approach.
Goal: integrate the tools of CoqApprox into the tactic.