Numbers in Coq

Yves Bertot

June 2023

The situation of numbers in the Coq system

Computation is a strong design attractor for Coq

- Incentive to reason about algorithms
- Being able to run these algorithms is nice
- Even more so with reflective tactics
  - Tactits that rely on internal computation

Many choices of data structures for representing integers and natural numbers

- Peano approach: 0 and successors
- base and position representations
  - Sequences of digits or bits
  - Preference for binary
  - Also possible to use binary tree representations

Ease of programming dependent on the choice of data structure

- Bare metal inductive type theory

Most advanced: computation with real numbers

WARNING: not available in JsCoq

- Challenge: a pocket calculator says that $e = 2.71828182\ldots$ and $\pi = 3.14159265\ldots$
- What is the sign of $3.14159265e - 2.71828182\pi$?

```
Require Import Reals Interval.Tactic Lra.

Open Scope R_scope.

Lemma real_exercise :
   0 < 3.14159265 * exp 1 -  2.71828182 * PI.
Proof.  interval. Qed.
```

The case of real numbers

Real numbers outside of the Coq-computable world

- Proof-based computation is still available
- Computing approximations
- *Fallible* computations

How does it work?

- Real numbers are in a type "assumed to exist", with 0, 1, $+$, ..., and complete archimedian field properties,
- e^x is defined as $\sum_{i=0}^{+\infty} \frac{x^i}{i!}$
- $\cos x$ is defined as $\sum_{i=0}^{+\infty} \frac{(-1)^i x^{2i}}{(2i)!}$
- PI is defined as twice the first positive root of cos.
- An extra theorem shows $PI = 4(4atan\frac{1}{5} - atan\frac{1}{239})$
- If the input is rational, each power series computation only uses rational numbers
- Intervals are computed at each step, and then combined
- The method has weaknesses, but works well in many cases

Weakness of the interval approach

```
Require Import Reals Interval.Tactic Lra.

Open Scope R_scope.

Lemma real_exercise2 x :
   1 / 2 ^ 10 <= x <= 1 -> 0 <= x - sin x.
Proof.
intros xint.
interval_intro (x - sin x) with
   (i_decimal, i_prec 120, i_bisect x, i_depth 8).
(* Long time of computation, result lower bound below 0 *)
interval_intro (x - sin x) with
   (i_decimal, i_taylor x, i_prec 120, i_bisect x, i_depth
lra.
Qed.
```

Didactic issues with `interval`

- Does not let the student practice skills
- Domain of practical application difficult to understand
  - This requires acquiring some skills, non-mathematical
- Useful to have for menial goals, but it feels too powerful

Rational numbers

- Use of the `Compute` command.
- Rational numbers are encoded as pairs of a signed integer and a positive number
- Exact operations are provided (no square root)
- Results are not normalized (for efficency reason)
- The normalization function must be called explicitely
- There is a specific equality for rational numbers, noted ==

Rational numbers are under-appreciated

- There are comparatively less theorems than for real numbers or integers
- Equality between rational numbers is only treated as an equivalence relation
- Naked eye comparison is uncomfortable

Binary integers

- A specific datatype to represent positive integers (no size limit)
- A wrapper to add signs: two types `positive` and `Z`
- Clumsy recursion: recursive calls are only available for half numbers
  - Good enough for usual operations: $+$, *, $/$, square root,
  - Clumsy for gcd, factorial
- Proof by induction requires more skill than for natural numbers
- The workhorse for many computation tools in Coq, including numeral notations in real numbers

Examples with integers

```
Require Import ZArith.
Open Scope Z_scope.

Check xO (xI xH).
  (*representation for 6, as positive number*)

Check Zpos (xO (xI xH)).
  (*representation for 6, as a signed integer *)

Definition Zfactorial (x : Z) :=
  snd (Z.iter x (fun '(x, f) => (x + 1, f * x)) (1, 1)).

Compute Zfactorial 6. (* returns 720 as expected *)

Compute Zfactorial 100.
  (* returns a huge number, no notable delay *)
```

Example proof with integers

- $\sqrt{2}$ is not rational
- Rephrased with integers :
  $\forall pq, 0 < q < p \rightarrow 2 * q^2 = p^2 \rightarrow \texttt{False}$
- **sketch of the proof**, if p is a minimal integer such that the equality holds, then p is even, and then the equality holds for q and $p/2$
- The key step is that if the square of p is even, then p is even, this take some time to express with existing theorems.
- Untold assumptions in this sketch are that p and q are positive (in particular, non-zero)

Demo time

DEMO