Reasoning about big enough numbers in Coq

Cyril Cohen

INRIA Saclay – Île-de-France
LIX École Polytechnique
INRIA Microsoft Research Joint Centre
cohen@crans.org

August 12, 2012
Games of epsilons

Standard presentation of analysis:
\[ \lim_{n \to \infty} x_n = a \] means that

\[ \forall \varepsilon > 0, \exists N, \forall n \geq N, |x_n - a| < \varepsilon \]
To show that

$$\lim_{\infty} x_n = a \quad \land \quad \lim_{\infty} y_n = b \quad \Rightarrow \quad \lim_{\infty} x_n y_n = ab$$

We can detail the proof like this (on the paper):

Let $\varepsilon > 0$ and let us find $N$ such that

$$\forall n \geq N, \ |x_n y_n - ab| < \varepsilon.$$
For all $n$, to show $|x_n y_n - ab| < \varepsilon$, it suffices to show that $|x_n| |y_n - b| < \frac{\varepsilon}{2}$ and $|x_n - a| |b| < \frac{\varepsilon}{2}$.

Since $(x_n)_n$ converges, it means $|x_n|$ is upper bounded by a number $m > 0$, so it suffices to show $|y_n - b| < \frac{\varepsilon}{2m}$ and $|x_n - a| < \frac{\varepsilon}{2(1 + |b|)}$. 
Finally, there exists $N_x$ and $N_y$ such that

$$\forall n > N_x, |x_n - a| < \frac{\varepsilon}{2(1 + |b|)}$$

$$\forall n > N_y, |y_n - b| < \frac{\varepsilon}{2m}$$

So taking $N = \max(N_x, N_y)$ works.
Demo 1

- The backward reasoning stops when we reach the `exists` in the goal.
- Finding the good $N$ is already painful with such a simple example.
Our contribution

- Allows to continue backward reasoning
- Hides the use of evar, using 3 tactics
- Simple to use, very close to approximate paper presentation.
How it works

Two steps:

- When you encounter an \texttt{exists}, you pose a “big enough variable”.
- When you must satisfy a condition like

\[ |x_n - a| < \frac{\varepsilon}{2(1 + |b|)} \]

you invoke the tactic \texttt{big}. If there is no dependency of the right hand side in the “big enough variable”, it solves it.

Demo 2
Implementation

Internally:

- The “big enough value” is the maximum of a list which tail is an evar, that gets instantiated progressively.
- Uses the tactic evar to create an evar.
- Uses apply to resolve an evar (or instantiate when there can be no ambiguity).
- Uses SSReflect pattern selection mechanism (cf ITP 2012 talk *A Language of Patterns for Subterm selection*).

The code is really really short.
Real life use

- My ITP 2012 talk **Construction of Real Algebraic Numbers in Coq**.
- Please try it.