Reasoning about big enough numbers in Coq

Cyril Cohen

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

Abstract. This proposal shows a way to deal with the mathematical idiom "big enough" in the Coq proof assistant. It is indeed difficult to use this idiom concisely in formalized constructive analysis. We describe a tactic and a proof style to keep the reasoning as close as possible to the paper mathematics style. The methodology was built for and successfully applied to a construction of real algebraic numbers.

Introduction

In analysis, we use the idiom "for big enough values of n ...". This presentation does not exhibit the values that this "big" n could take. Of course, the user can manually enter himself a value. The success in stating a value vouches for the non circularity of the definition, and the success in completing the proof with it demonstrates that it was big enough indeed. However, the proof is usually independent from the actual value of n and depends only on its existence.

Statements of intermediate lemmas can be presented in such a way that the constraints on n appear explicitly in the proof and could be detected automatically. In order to implement this, we rely on Coq existential variables: an existential variable can be created at the beginning of the proof and set *a posteriori*. We build a Coq tactic that updates the value of this existential variable. We successfully applied this methodology to the construction of real algebraic numbers [1], which full development is available at http://perso.crans.org/cohen/work/realalg.

1 A traditional and a Coq presentation

Let us consider the proof of the following statement: Given two Cauchy sequence $(x_n)_n$ and $(y_n)_n$, if $\lim_{n\to\infty} x_n y_n \neq 0$ then $\lim_{n\to\infty} y_n \neq 0$.

The standard proof would be: since $\lim_{n\to\infty} x_n y_n \neq 0$ there exists N_1 and δ_1 such that $x_n y_n \geq \delta_1$ for all $n \geq N_1$. And since x_n is a Cauchy sequence, there exists N_2 and δ_2, such that $|x_n| \leq \delta_2$ for all $n \geq N_2$. So for all $n \geq \max(N_1, N_2)$, $(y_n)_n$ stays greater than $\frac{\delta_1}{\delta_2}$ which completes the proof.

The Coq presentation is almost the same except that the existence of δ_1 and δ_2 are respectively rephrased this way:

```
Lemma lboundP (x : creal) (x_neq0 : x ≠ 0) i :
  cauchymod x (lbound x_neq0) ≤ i →lbound x_neq0 ≤ |x i|.
Lemma uboundP (x : creal) i : |x i| ≤ ubound x.
```

In order to prove that $\lim_{n\to\infty} y_n \neq 0$, we pose a big enough n and show that for all $i \geq n$, `(lbound xy_neq0 / ubound x ≤ |y i|)`

At some point the application of the lemma `lboundP` generates the subgoal `(cauchymod (x * y) (lbound xy_neq0) ≤ i)` which happens to be solvable by updating `n` to be at least the value on the left hand side of the inequality.

2 Methodology and implementation

A number falls into the "big enough" category if any bigger number would fit too. Thus, we create in the context a variable `i` defined as `(max_seq ?s)`, the maximum of an existential sequence `?s`. Then, the update consists in the instantiation of `?s` using `(n :: ?s')` where `n` is the new value. We build a tactic `big` that finds occurrences of terms of the form `(n ≤ i)` in the goal and replaces them by true and adds `n` to `?s`.

In order to use this tactic we must ensure that any lemma we will use on a "big enough" number i will defer all the constraints on i as side conditions of the form $n \leq i$, as for example in `lboundP`. The reader may also look at definitions of `cauchymodP`, `diffP`, `le_crealP`, ... which are stated the same way.

We implemented the tactic in `Ltac`. It relies on SSReflect pattern selection mechanism [2] to ensure the robustness of our tactic: we select sequentially all the subterms of the form $n < i$, and try to apply our tactic to each of them.

We also had to deal with an issue in the version 8.3 of Coq where one cannot instantiate evars when no goal remains. Thus, we artificially created a logical cut when creating the evar, so that the user is prompted with a trivial goal in the end, during the resolution of which he could set the tail of the existential sequence to be the empty list.

3 Conclusion

Our tactic helps the user to reason like in paper mathematics by letting him pose an arbitrary big value and explain latter why it can be big enough. We successfully applied this method to a construction of real algebraic numbers.

This tactic uses integers, but it might be interesting to generalize it to "small enough" element, even though in an Archimedian domain, it could be simulated by $\frac{1}{n}$ for a big enough n. We could also generalize this for any lattice.

References

1. Cohen, C.: Construction of real algebraic numbers in Coq. In: Proceedings of ITP 2012 (2012), to appear
2. Gonthier, G., Tassi, E.: A language of patterns for subterm selection. In: Proceedings of ITP 2012 (2012), to appear