Short introduction by example to Coq and formalising ZF $\subseteq$ ZF$_\varepsilon$ in Coq

Jaime Gaspar*

2 July 2014

1 Short introduction by example to Coq[1]

Proof assistants are computer programs that help mathematicians to prove theorems and to formally verify the correctness of proofs. Proof assistants are nowadays one of the more exciting areas in the intersection of mathematical logic and computer science. For example, one particularly exciting achievement is the formal verification of the proof of the four colour theorem using the proof assistant Coq.

In this talk we give a very elementary introduction to Coq by means of a very simple example, namely the proofs of the following theorems.

- If $\leq$ is a non-strict partial order, then $<$ defined by $x < y \Leftrightarrow x \leq y \wedge x \neq y$ is a strict partial order.
- If $<$ is a strict partial order, then $\leq$ defined by $x \leq y \Leftrightarrow x < y \vee x = y$ is a non-strict partial order.

We divide the talk into the following four parts.

*Universitat Rovira i Virgili, Department of Computer Engineering and Mathematics, Av. Països Catalans 26, E-43007 Tarragona, Catalonia, jaime.gaspar@urv.cat [affiliation disputable]. Centro de Matemática e Aplicações (CMA), FCT, UNL. Financially supported by the Martí Franquès Research Fellowship Programme grant number 2013PMF-PIPF-24 of the Universitat Rovira i Virgili.

[1]Keywords: Coq; proof assistant; formal verification; partial order.

Introduction to Coq We explain, by means of the theorems mentioned, that to formalise a proof in Coq we need to tell Coq the following four things.

Language For example, to introduce a binary relation S on $\mathbb{N}$ by the code `Variable S : nat -> nat -> Prop` and to introduce another binary relation N on $\mathbb{N}$ defined as $N(x, y) \Leftrightarrow S(x, y) \vee x = y$ by the code `Definition N x y := S x y \/ x = y`.

Axioms For example, to introduce the irreflexivity axiom $\forall x \in \mathbb{N}\, \neg S(x, x)$ by the code `Axiom irreflexivity : forall x : nat, ~S x x`.

Theorem For example, to introduce the reflexivity theorem $\forall x \in \mathbb{N}\, N(x, x)$ by the code `Theorem reflexivity : forall x : nat, N x x`.

Proof For example, to introduce the proof "take x, unfold $N(x, x)$ into $S(x, x) \vee x = x$, prove the right part $x = x$ by the reflexivity of $=$" of the reflexivity theorem by the codes "`intro x`, `unfold N`, `right`, `reflexivity`".

Achievements of Coq We discuss what is achieved with this kind of formal verification.

Applications of Coq to education We address the application of Coq to education by mentioning how we can use Coq to learn the following topics.

Logic For example, propositional calculus.

Arithmetic For example, Peano arithmetic.

Algebra For example, group theory.

Geometry For example, Euclidean geometry.

Set theory For example, Zermelo-Fraenkel set theory.

Practical aspects We address the following practical aspects of Coq.

Using Coq How to use Coq online (no installation needed) and offline (installation needed).

Tutorials Where to find tutorials and manuals for Coq to learn more.

We keep this talk short, simple and sweet.

2 Formalising ZF $\subseteq$ ZF$_\varepsilon$ in Coq[2]

Jean-Louis Krivine [1] introduced a variant ZF$_\varepsilon$ of ZF (Zermelo-Fraenkel set theory without the axiom of choice) with

- two set memberships:
  - the old extensional set membership $\in$;
  - a new nonextensional set membership ε;
- axioms saying that $\in$ is the "extensional collapse" of ε.

Then he proved ZF $\subseteq$ ZF$_\varepsilon$ (that is, every theorem of ZF is also a theorem of ZF$_\varepsilon$).

In this talk we present a little formalisation in Coq (a proof assistant) of Krivine's proof of ZF $\subseteq$ ZF$_\varepsilon$. Admittedly, we are hoping for comments from Coq experts to help us to write a short article about the formalisation. The talk is divided into the following four parts.

Introduction to ZF$_\varepsilon$ We briefly introduce ZF$_\varepsilon$ along the lines above.

Introduction to Coq We briefly introduce Coq by means of a very simple example of a theory.

Language The theory has a language with propositional variables P and Q.

Axioms The theory has the axioms P and $P \Rightarrow Q$.

Theorem The theory proves the theorem Q.

Proof The theory proves Q by the proof "by the axiom $P \Rightarrow Q$, to prove Q suffices to prove P, which is an axiom".

Formalisation of ZF $\subseteq$ ZF$_\varepsilon$ in Coq We present our formalisation by showing key bits of the Coq code.

Language For example, the set membership ε is introduced by the code `Parameter epsilon : Set -> Set -> Prop`.

Axioms For example, the axiom of pairing of ZF$_\varepsilon$ is introduced by the code `Axiom Pair : forall a b : Set, exists c : Set, a ε c /\ b ε c`.

[2]Keywords: Coq; proof assistant; formal verification; ZF$_\varepsilon$; ZF; set theory.

Theorem For example, the theorem of paring of ZF is introduced by the code `Theorem Pair : forall a b : Set, exists c : Set, a ∈ c /\ b ∈ c`.

Proof For example, the theorem of pairing of ZF is proved by some code `Proof ... Qed` that is a bit complicated so we omit it here.

Contributions and problems We briefly discuss contributions to education and technical problems of our formalisation.

Contributions For example, the extremely detailed level of the formalisation provides a good exercise in applying the axiom schema of comprehension.

Problems For example, we added new axioms to Coq, and this raises the question of whether Coq with those new axioms is consistent.

We keep this talk short, simple, and sweet.

References

[1] Jean-Louis Krivine. Realizability algebras II: new models of ZF + DC. *Logical Methods in Computer Science*, 8(1:10):1–28, 2012. http://arxiv.org/abs/1007.0825.