The “Initiation to formal proofs” course

Pierre Rousselin (with Marie Kerjean and Micaela Mayero)

Université Paris 13, Villetaneuse

Le Val d’Ajol
June, 22 2023

Context

Organization of the course

Content

Impressions, feedbacks

Context

Organization of the course

Content

Impressions, feedbacks

Public

- Université Paris 13, Villetaneuse (northern suburbs of Paris)
- First year undergraduate students in maths+CS double major
- two groups of 25 students each

History of the course

- first edition in fall 2021
- new specific course for these students
- at the interface of maths and CS
- focused on activity and rigour
- (replaces a 18h methodology course)

Goals of the course

- ??

Goals of the course

- ??
- The aim is actually to make the students write formal proofs of mathematical statements...

Goals of the course

- ??
- The aim is actually to make the students write formal proofs of mathematical statements...
- ... in the hope that it will help them in their maths+CS studies.

Goals of the course

- ??
- The aim is actually to make the students write formal proofs of mathematical statements...
- ... in the hope that it will help them in their maths+CS studies.
- Work on rigour and problem-solving.
- Side goal: create a group dynamic in "double-licence" by giving the students a challenging specific course

Context

Organization of the course

Content

Impressions, feedbacks

Choices

- We had to make some choices under many constraints (duration of the course, lack of time to prepare new content, etc) and external influences (people to work with, *software foundations* by Pierce and al., ...)
- These choices are certainly not the only possible ones and very likely not the best ones.
- Our goal in this talk is not to say "one should choose this", but rather "we chose this" and explain, when possible, why we did so.

Choices and preparation

- only hands-on practical sessions (no lectures) with homeworks
- As in *Software Foundations*, the course is a set of `.v` source files with examples, exercises and comments.

Choices (2)

- Do not hide stuff: it's ok to talk about intuitionnist logic, right-associativity of `->`, ...
- Passionate students should be able to write their own theorems and prove them (autonomy).
- However, writing your own functions or types is not an objective.
- The maths+CS side is embraced: it's ok to write ascii bytes in a file using a text editor.
- Restrictions:
  - no booleans (two logics would be too much)
  - no inductive propositions (too much to digest in such a small course)

Starting point

- limits of sequences as final goal
- prerequisites: logics, natural numbers and real numbers.

Plan of the course

- Propositional (intuitionnist) logic (+ additional exercises)
- Natural numbers and induction (+ additional exercises)
- Predicate calculus (“Set theory” à la Coq) (+ additional exercises)
- First homework
- Real numbers as a field (algebra)
- Second homework
- Real numbers as an ordered field
- Absolute value and distance on real numbers
- Convergence of real-valued sequences
- Final test

Context

Organization of the course

Content

Impressions, feedbacks

`LogiquePropositionnelle.v`

- The aim is to introduce the usual connectors of propositional (intuitionnist) logic.
- We always start with a commented example...
- ... followed by exercises
- Always two sides: “how to prove it?” and “how to use it?”
- The order is `->`, `False` and `not`, `and`, `or`.
- In the end and in the additional exercises, the excluded middle is discussed and used (but we don’t really need it explicitly in the rest of the course...)

Tactics for propositional logic

- `->` `apply` to use, `intros` to prove
- `/\` `split` to prove, `destruct` to use
- `\/` `left` or `right` to prove, `destruct` to use (proof by exhaustion)
- `<->` `split` to prove, `destruct` to transform into two `->`
- `False` `destruct` to use and prove anything, `exfalso` to change any goal to `False`

Conclude with `exact` or `assumption`.

Only backward reasoning at this point.

Contract

- Strong implicit contract between students and teachers:
- every exercise should be feasible only with what has previously been shown.
- Always start with an example.
- Reduced number of tactics.

Naturels.v

- Peano's natural numbers
- Coq can compute (`Fixpoint`, `simpl`, `Compute`)
- induction
- `rewrite` and unification
- Natural number game (associativity and commutativity of multiplication from scratch)

Tactics introduced with $\mathbb{N}$

- `rewrite`
- `induction`
- `simpl` (debatable)
- `discriminate`
- `injection`
- `f_equal`

`CalculPredicat.v`

- Existential quantifier: using (`destruct`) and proving (`exists`).
- "Subsets of a type `A`", actually `A -> Prop`.
- Injections, surjections, bijections
- Negation of `forall` and `exists` with the excluded middle is discussed.

CorpsOrdonné.v

We start working with Coq's `Reals`.

- "Axioms" of an ordered field
- No more computation, only `rewrite`
- "Real numbers game": from "axioms" to $0 < 1$
- We progressively introduce forward reasoning (`apply ... in`, `rewrite ... in`, `assert`, `replace`).

RInégalités.v and Rabs_R_dist.v

- New in 2022
- Students struggle with inequalities and absolute values
- They needed more exercises before studying sequences.

Suites.v

- Before studying sequences, automation is shown (file `Auto.v`).
- Actually some inequalities on $\mathbb{N}$ could **not** be proved manually by students.
- First analysis lemma:

```
Lemma small_zero: forall x,
  (forall eps, eps > 0 -> (Rabs x) < eps) -> x = 0.
```

- The given example is:

```
Theorem UL_sequence (Un : nat -> R) (l1 l2 : R) :
  Un_cv Un l1 → Un_cv Un l2 → l1 = l2.
Proof.
  unfold Un_cv.
  intros Hl1 Hl2.
  (* On va montrer que la distance entre l1 et l2
     est aussi petite qu'on veut. *)
  apply small_dist_equal.
  (* Soit eps > 0. *)
  intros eps Heps.
  (* Soit n1 tel que pour tout n >= n1, |Un - l1| < eps / 2. *)
  destruct (Hl1 (eps / 2)) as [n1 Hn1]. lra.
  (* Soit n2 tel que pour tout n >= n2, |Un - l2| < eps / 2. *)
  destruct (Hl2 (eps / 2)) as [n2 Hn2]. lra.
  (* Soit n3 = max(n1, n2). *)
  remember (max n1 n2) as n3 eqn:n3_max.
  (* ... *)
Qed.
```

Context

Organization of the course

Content

Impressions, feedbacks

In the end

- Propositional logic with natural deduction is well understood
- Almost all students can prove simple equalities in $\mathbb{N}$ by induction
- Some difficulties with predicate calculus, but knowing that it be harder helped this year
- Working with real numbers is mostly ok with equations, inequalities are harder (but this gets better with practice).
- In 2021, only one student managed to prove a non-trivial analysis theorem. In 2022, about 6 of them proved a significant part of the `Suites.v` file. Can this be increased without more hours?

In the end (2)

- It is not really possible at this point to quantify the impact of this course on the students.
- It is clear though that it helped create a solid “double-licence” group.
- During the second semester, the avarage grade in double-licence this year was about 14/20, usually 5 more points than computer science students (and even better when compared to maths students).
- A group of students was very willing to continue with formal proofs (unfortunately, I didn’t manage to find time to write more exercises...)

Food for thoughts

- What is the real value of this course for students?
- When and how should we introduce forward reasoning?
- When and how should we introduce automation?
- Is it possible to go from coq proofs to pen and paper proofs?
- What's next?