Toward the use of a proof assistant to teach mathematics.

Julien Narboux
under the supervision of
Hugo Herbelin

LIX, École Polytechnique

ICTMT7, 26 July 2005, Bristol, UK

Outline

- The impact of the use of software on the proving activity is a well addressed issue in the litterature.
- CAS, DGS, .
- There are software whose sole purpose is to produce proofs : the proof assistants.

- The impact of the use of software on the proving activity is a well addressed issue in the litterature.
- CAS, DGS, .
- There are software whose sole purpose is to produce proofs : the proof assistants.

- The impact of the use of software on the proving activity is a well addressed issue in the litterature.
- CAS, DGS, PA.
- There are software whose sole purpose is to produce proofs : the proof assistants.

CAS (Maple, MuPAD, Mathematica . . .)

Definitions, Questions → Results

Computation

Theorem Prover (Otter, Vampire . . .)

Definitions, Axioms, Statement → True, False, I don't know, Nothing.

Automatic proof

Proof assistant (Coq (84), Isabelle, HOL, PVS (90's). . .)

Axioms, Statement, Interractive Proof → Correct or not

Interractive proof

Proof oriented software

- For example :
  - logic oriented (Hyperproof . . .)
  - geometry oriented (Geometrix, Baghera . . .)
  - algebra oriented (MathXpert . . .)
- They are :
  - User friendly
  - Give hints to the student

Proof assistants

- Not specialized
- A very large span of applications
- A very high level of confidence
- Real mathematics

What can we prove ?

- Programs (Line 14 of Paris' subway . . .)
- Mathematical statements (The fondamental theorem of algebra (Henk Barendregt's group), The four colours theorem (Gonthier, Werner). . .)

What can we prove ?

- Programs (Line 14 of Paris' subway . . .)
- Mathematical statements (The fondamental theorem of algebra (Henk Barendregt's group), The four colours theorem (Gonthier, Werner). . .)

Why ?

- To understand what a proof is.
- To ensure correctness of the proof (The four colours theorem again).
- To generate proofs that could not be done by hand, either
  - Proof of programs (often long but straightforward proofs
    with too many cases for an exhaustive search).
- For teaching.

Why ?

- To understand what a proof is.
- To ensure correctness of the proof (The four colours theorem again).
- To generate proofs that could not be done by hand, either
  - Proof of programs (often long but straightforward proofs with too many cases for an exhaustive search).
  - Proof of mathematical statements (The four colours theorem).
- For teaching.

Why ?

- To understand what a proof is.
- To ensure correctness of the proof (The four colours theorem again).
- To generate proofs that could not be done by hand, either
  - Proof of programs (often long but straightforward proofs with too many cases for an exhaustive search).
  - Proof of mathematical statements (The four colours theorem).
- For teaching.

Why ?

- To understand what a proof is.
- To ensure correctness of the proof (The four colours theorem again).
- To generate proofs that could not be done by hand, either
  - Proof of programs (often long but straightforward proofs with too many cases for an exhaustive search).
  - Proof of mathematical statements (The four colours theorem).
- For teaching.

Why ?

- To understand what a proof is.
- To ensure correctness of the proof (The four colours theorem again).
- To generate proofs that could not be done by hand, either
  - Proof of programs (often long but straightforward proofs with too many cases for an exhaustive search).
  - Proof of mathematical statements (The four colours theorem).
- For teaching.

Why ?

- To understand what a proof is.
- To ensure correctness of the proof (The four colours theorem again).
- To generate proofs that could not be done by hand, either
  - Proof of programs (often long but straightforward proofs with too many cases for an exhaustive search).
  - Proof of mathematical statements (The four colours theorem).
- For teaching.

Coq

Coq (http://coq.inria.fr/)

- a free software (GPL2),
- based on the Calculus of Inductive Constructions,
- developped at INRIA in the LogiCal team,
- since 1984.

Coq

Coq (http://coq.inria.fr/)

- a free software (GPL2),
- based on the Calculus of Inductive Constructions,
- developped at INRIA in the LogiCal team,
- since 1984.

Coq

Coq (http://coq.inria.fr/)

- a free software (GPL2),
- based on the Calculus of Inductive Constructions,
- developped at INRIA in the LogiCal team,
- since 1984.

Coq

Coq (http://coq.inria.fr/)

- a free software (GPL2),
- based on the Calculus of Inductive Constructions,
- developped at INRIA in the LogiCal team,
- since 1984.

What you need to trust :

- The theory behind Coq.
- The Coq kernel implementation match the theory.
  Coq : $>$ 130000 lines of code
  The kernel : $<$ 11000 lines of code
- Your hardware, operating system and Ocaml compiler.
- Yours axioms.

What you need to trust :

- The theory behind Coq.
- The Coq kernel implementation match the theory.
  Coq : $>$ 130000 lines of code
  The kernel : $<$ 11000 lines of code
- Your hardware, operating system and Ocaml compiler.
- Yours axioms.

What you need to trust :

- The theory behind Coq.
- The Coq kernel implementation match the theory.
  Coq : $>$ 130000 lines of code
  The kernel : $<$ 11000 lines of code
- Your hardware, operating system and Ocaml compiler.
- Yours axioms.

What you need to trust :

- The theory behind Coq.
- The Coq kernel implementation match the theory.
  Coq : $>$ 130000 lines of code
  The kernel : $<$ 11000 lines of code
- Your hardware, operating system and Ocaml compiler.
- Yours axioms.

1.
$$\forall xyz, x = y \wedge y = z \rightarrow x = z$$

2.
$$\sum_{k=0}^{k=n} 2k + 1 = (n+1)^2$$

A few difficulties :

- $f(x, y)$ is noted $(f\ x\ y)$.
- $A \rightarrow B \rightarrow C$ is used to express $A \wedge B \rightarrow C$.
- $\neg A$ is defined by $A \rightarrow \mathit{False}$.

Let's start :

```
1 subgoal
x : nat
y : nat
z : nat
H : x = y /\ y = z
______________________________(1/1)
x = z
```

Examples:

We know that :

- x, y and z are natural numbers and
- $x = y \wedge y = z$.

We need to show that :

- $x = z$.

How can I prove something ?

The proof can be described step-by-step using :

- case distinction
- absurd
- induction
- application of a theorem
- computation
- rewriting
- and sometimes automation
- ...

Some examples now.

- It clarifies what we know, what we want to prove, what are the theorems, lemmas, axioms, definitions...
- It is rigourous.
- It helps to understand the logic.
- It clarifies what the logical rules are.
- It is fair: the proof is correct iff it is accepted by the system.

- It clarifies what we know, what we want to prove, what are the theorems, lemmas, axioms, definitions…
- It is rigourous.
- It helps to understand the logic.
- It clarifies what the logical rules are.
- It is fair: the proof is correct iff it is accepted by the system.

- It clarifies what we know, what we want to prove, what are the theorems, lemmas, axioms, definitions...
- It is rigourous.
- It helps to understand the logic.
- It clarifies what the logical rules are.
- It is fair: the proof is correct iff it is accepted by the system.

- It clarifies what we know, what we want to prove, what are the theorems, lemmas, axioms, definitions…
- It is rigourous.
- It helps to understand the logic.
- It clarifies what the logical rules are.
- It is fair: the proof is correct iff it is accepted by the system.

- It clarifies what we know, what we want to prove, what are the theorems, lemmas, axioms, definitions...
- It is rigourous.
- It helps to understand the logic.
- It clarifies what the logical rules are.
- It is fair: the proof is correct iff it is accepted by the system.

Limitations

- Notations
- Error messages
- Interface
- Associativity-Commutativity
- Not enough automation
- Too much automation

Limitations

- Notations
- Error messages
- Interface
- Associativity-Commutativity
- Not enough automation
- Too much automation

Limitations

- Notations
- Error messages
- Interface
- Associativity-Commutativity
- Not enough automation
- Too much automation

Limitations

- Notations
- Error messages
- Interface
- Associativity-Commutativity
- Not enough automation
- Too much automation

Limitations

- Notations
- Error messages
- Interface
- Associativity-Commutativity
- Not enough automation
- Too much automation

Limitations

- Notations
- Error messages
- Interface
- Associativity-Commutativity
- Not enough automation
- Too much automation

Work done or in progress

PCoq	A gui to ease the usage of Coq.
F. Guilhot's work	A formalization of high school geometry in Coq.
CoqWeb	An interface for solving exercises online using Coq (Work in progress).
DrGeoCaml	A gui for interactive proof in geometry using Coq (Work in progress).

- DGS to *conjecture*
- CAS to *compute*
- PA to *prove*

Thank you !

- DGS to *conjecture*
- CAS to *compute*
- PA to *prove*

Thank you !