Verifying software, verifying mathematics

Julio Rubio

Universidad de La Rioja
Departamento de Matemáticas y Computación

MAP 2011

Lorentz Center (Leiden), November 18th-December 2th, 2011

Partially supported by Ministerio de Educación y Ciencia, project MTM2009-13842-C02-01, and by European Commission FP7, STREP project ForMath, n. 243847.

Summary

- Software Engineering and Mathematics.
- Computing homotopy groups.
- Changing the view.
- Data structures are important.
- Abstraction barrier.
- Formalization as an experimental science.
- Where are we now?
- Conclusions.

Software Engineering and Mathematics

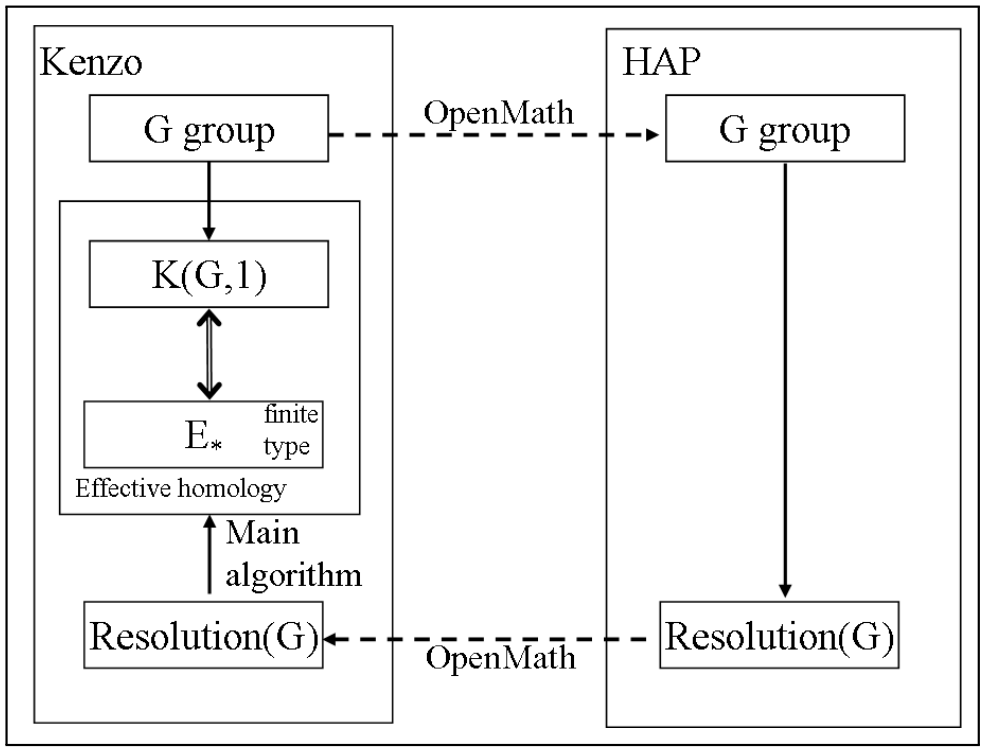
- Formal Methods in Software Engineering:
  Application of Mathematics *to* Computer Science.
- Objective of the talk:
  Applications of Software Engineering *to* Mathematics.
- (My) Background:
  Software verification of Computer Algebra systems.
- More concretely:
  Theorem proving (Isabelle, ACL2, Coq) for Kenzo specification and verification.

Computing homotopy groups (1/4)

- *Kenzo*: Sergeraert's program to compute in Algebraic Topology.
- Based on Sergeraert's notion of *effective homology*.
- *fKenzo*: Heras's friendly user interface for *Kenzo*.
- Computing homology and homotopy groups: demo.
- Romero's module: *Kenzo* extension to compute homology of groups.

Computing homotopy groups (2/4)

- In particular, Romero implemented an algorithm to compute the effective homology of $K(G,1)$, an Eilenberg-MacLane space.
- Idea: compute a resolution of the group G by means of Ellis's HAP/GAP, import it in Kenzo and construct a reduction from $C(K(G,1))$.



Computing homotopy groups (3/4)

- $K(G,1)$ is not simply connected ($\pi_1(K(G,1)) = G$)
- But its suspension is: $\Sigma K(G,1)$
- *On homotopy groups of the suspended classifying spaces*
  Roman Mikhailov and Jie Wu
  Algebraic and Geometric Topology 10(2010), 565 – 625
- $\pi_n(\Sigma K(G,1))$?
- Remarkable paper: computable aim (without computers!)
- *Theorem 5.4*: *Let A_4 be the 4-th alternating group.*
  Then $\pi_4(\Sigma K(A_4,1)) = \mathbb{Z}_4$

Computing homotopy groups (4/4)

- The effective homology of the suspension functor Σ was already implemented in *Kenzo*.
- Ana Romero makes *Kenzo* compute: $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_{12}$
- Let's repeat:

  Mikhailov & Hu: $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_4$

  Kenzo: $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_{12}$
- Then?

Changing the view (1/2)

- Standard protocol:
  1. Look for a bug (local)
  2. Look for a bug (in the whole new module)
  3. Look for a bug (in the Kenzo kernel)
  4. Look, look, look, ...
- Alternative protocol:
  1. Look at the journal paper
  2. A step in the proof seems missing
  3. Ask specialists (Sergeraert)
  4. Ask the authors: yes, *Kenzo* found the correct answer.

Changing the view (2/2)

- Standard view:
  1. Mathematics are correct "by principle"
  2. Software programs are unsafe "by construction"
  3. Software programs must be verified (using mathematical tools) to ensure its correctness.
- New view:
  1. Software verification ...
  2. ... *for* mathematics verification.

Data structures are important

- . . . in Mathematics.
- First lesson I learned: the formalized analysis of software is important in "standard" mathematics.
- Second lesson I learned: mathematics should be "designed".
- A case study: elementary (finite dimensional) linear algebra with Isabelle/HOL. (Aransay–Divasón)
- Reasoning with *sets* of vectors (Halmos, *Finite-Dimensional Vector Spaces*) is a very error prone approach.
- Solution: indexed sets (lists without duplicates).
- Moral: data structuring is important in Mathematics.
- George Gonthier's view: matrix is the (only one) data structure for finite-dimensional linear algebra.

Abstraction barrier (1/2)

- Is formalization the silver bullet against buggy mathematics?
- No: formalization should be verified.
- Independent proofs with different tools.
- Are the proving tools proven?

Abstraction barrier (2/2)

- Abstraction barrier: What are the basis for a formalization?
- Artificial example: to prove the Planar Jordan Curve Theorem, represent a Jordan curve as a triple (C, E, I) where $C \cup E \cup I = \mathbb{R}^2$ and $C \cap E = C \cap I = E \cap I = \emptyset$.
- Petitio principii!!!
- Another example: Gonthier's view doesn't allow one to ask about the possibility of an infinite dimensional basis.

Formalization as an experimental science

- To improve the reliability of formalizations:
  - Hypothesis (abstraction barrier) clearly stated.
  - Separation among axioms/definitions/statements.
  - Underlying logic infrastructure.
  - Enough documentation (beyond from both "lecture notes" papers and pure script files).
- . . . to allow others to reproduce the formalizations (perhaps with other tools).
- Interoperability (MKM) could be an issue here.

Where are we now? (1/3)

- How far are we from formalizing all the mathematics needed to prove $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_{12}$?
- "To prove" = "To verify the correctness of a program computing it".
- Quite far, but ...
- ... many things have already been formalized.

Where are we now? (2/3)

- Formalizing mathematics: the European Project ForMath
  European Commission FP7, STREP project ForMath: 2010-2013
  1. Homology (finite case, Smith Normal Form)
  2. Effective homology of bicomplexes (Coq)
  3. Normalization Theorem (ACL2)
- Previous formalization efforts:
  1. Correctness of simplicial sets in *Kenzo* (ACL2)
  2. Basic Perturbation Lemma (Isabelle/HOL)

Where are we now? (3/3)

- Still missing:
  1. Programs to compute homotopy groups (needs the infinite case).
  2. Romero's program dealing with the homology of groups (needs the infinite case, and also the XML connection with HAP/GAP).
- Long term research project (already 6 years...)
- Interesting?

Conclusions

- Theorem Provers are mature enough to tackle *real* mathematical problems.
- Specially interesting in conjunction with Computer Algebra systems (increasing reliability, *both* of software *and* mathematics).
- In particular: it demonstrates the usefulness of formalization for the "standard" mathematician.
- Mathematics as a kind of experimental engineering.