# Formalization of Mathematics: why Algebraic Topology?

*Julio Rubio*

Universidad de La Rioja
Departamento de Matemáticas y Computación

MAP Spring School 2012

Sophia Antipolis, March 12th-16th, 2012

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

# Summary

- Reasons to formalize mathematics.
- The *Kenzo* program.
- Homological processing of biomedical images.
- A multitool approach.
- Formalizing with Isabelle/HOL.
- Formalizing with ACL2.
- Formalizing with Coq/SSReflect.
- Conclusions and future work.

# Reasons to formalize mathematics

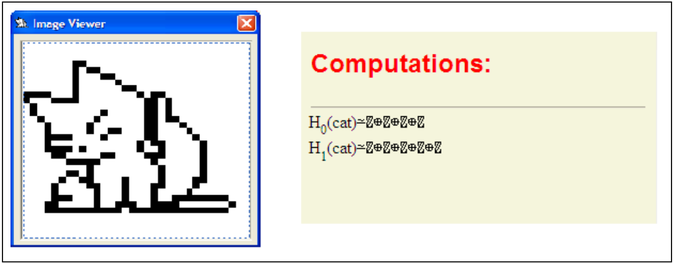
- Internal to the proving tools:
  - Checking expressiveness, testing, and so on.
- Internal to mathematics:
  - Foundations: Voevodsky's univalent foundations
  - Challenge: Gonthier on the classification of groups
  - Checking the correctness of a (computer) proof
    - Hales on the Kepler conjecture
    - Gonthier on the Four Color theorem
- Applications:
  - Verification of (mathematical) software
    - (Verification of hardware and software)
    - Reliable numerics: Spitters on computational analysis
    - Programs difficult to test
    - Programs for real-life problems

# Reasons to formalize mathematics

- Internal to the proving tools:
  - Checking expressiveness, testing, and so on.
- Internal to mathematics:
  - Foundations: Voevodsky's univalent foundations
  - Challenge: Gonthier on the classification of groups
  - Checking the correctness of a (computer) proof
    - Hales on the Kepler conjecture
    - Gonthier on the Four Color theorem
- Applications:
  - Verification of (mathematical) software
    - (Verification of hardware and software)
    - Reliable numerics: Spitters on computational analysis
    - Programs difficult to test
    - Programs for real-life problems

# The *Kenzo* program (1/3)

Algebraic Topology: the science of associating algebraic invariants with geometrical objects (topological spaces)




*Kenzo* is a computer algebra system (created by F. Sergeraert) devoted to Algebraic Topology.

# The *Kenzo* program (2/3)

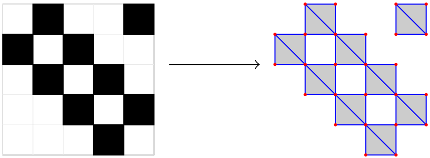
*Kenzo* can compute results difficult to reach by any other means.

- In particular, A. Romero enhanced *Kenzo* with an algorithm to compute the homotopy groups of suspended Eilenberg-MacLane spaces.
- *On homotopy groups of the suspended classifying spaces*
  Roman Mikhailov and Jie Wu
  Algebraic and Geometric Topology 10(2010), 565 – 625
- *Theorem 5.4*: *Let* $A_4$ *be the* 4*-th alternating group.*
  *Then* $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_4$
- Ana Romero makes *Kenzo* compute: $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_{12}$
- Let's repeat:
  Mikhailov & Wu: $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_4$
  *Kenzo*: $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_{12}$
- Then?

# The *Kenzo* program (3/3)

- In this particular case, *Kenzo* was right (i. e. the “theorem” in the paper wasn’t one).
- In addition, Romero’s program can compute more homotopy groups out of reaching by Mikhailov & Wu’s techniques.
- Therefore:
  1. Mathematics formalization ...
  2. ... *for* software verification ...
  3. ... *for* mathematics verification.
- Formalizing to prove $\pi_4(\Sigma K(A_4, 1)) = \mathbb{Z}_{12}$.
- “To prove” $=$ “To verify the correctness of a program computing it”.

# Homological processing of digital images (1/2)



- An image is represented by means of a list of lists of bits.
- Then we construct an associated *simplicial complex* (list of triangles).
- Homology groups are obtained by diagonalizing the *incidence matrices*.

# Homological processing of digital images (2/2)

$$\begin{tikzcd}
\text{Digital Image} \arrow[d] \\
\text{Simplicial Complex} \arrow[d] \\
\text{Incidence matrices} \arrow[d] \\
\text{Homology} \arrow[uuu, bend left]
\end{tikzcd}$$

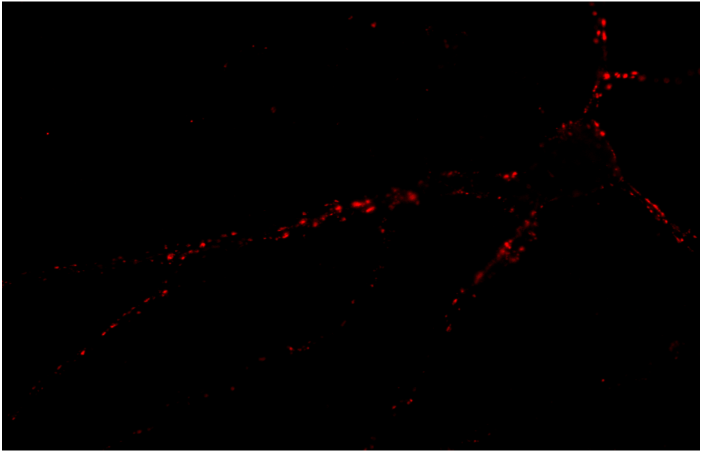
**Objective**

Certified computation of homology groups for digital images

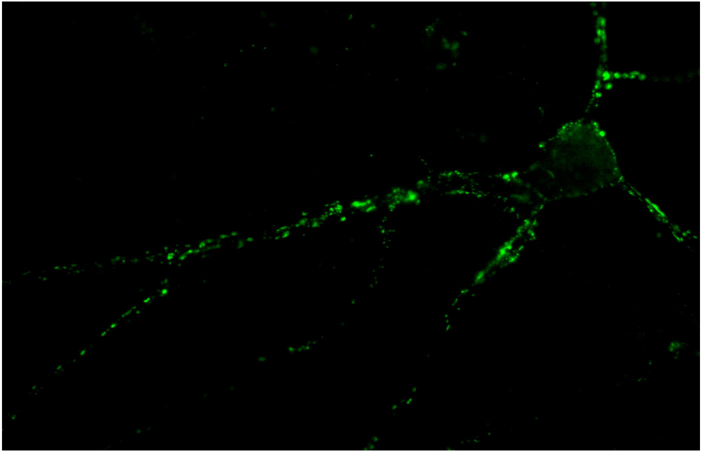
# Application to biomedicine: Counting synapses

- Synapses are the points of connection between neurons.
- Relevance: Computational capabilities of the brain.
- Procedures to modify the synaptic density may be an important asset in the treatment of neurological diseases (like Alzheimer).
- An automated and reliable method is necessary.

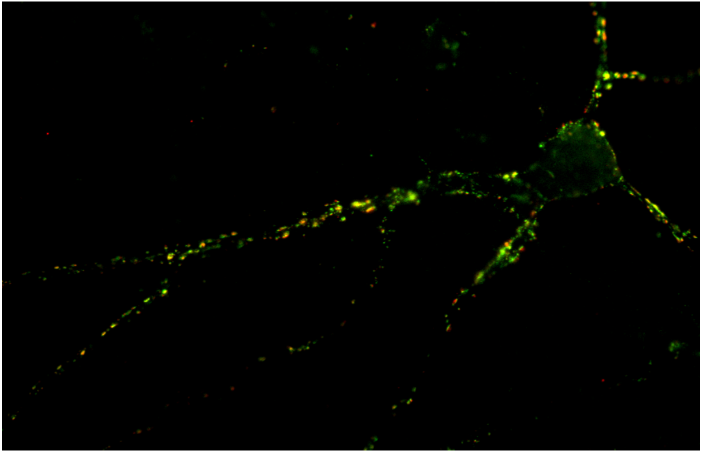
# Application to biomedicine: Counting synapses



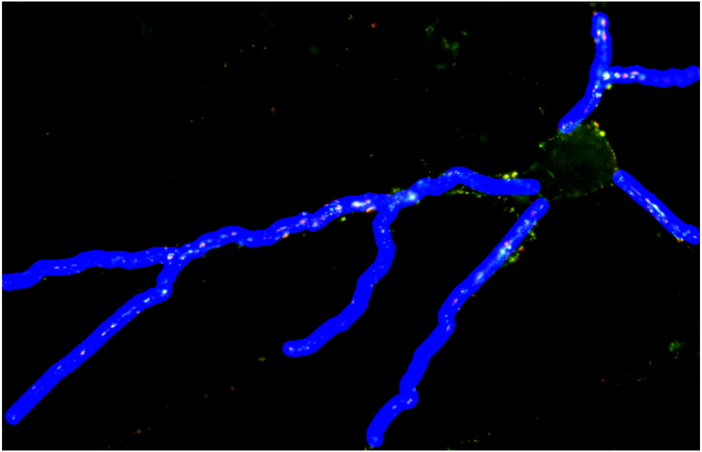
# Application to biomedicine: Counting synapses



# Application to biomedicine: Counting synapses



# Application to biomedicine: Counting synapses



# Application to biomedicine: Counting synapses

# Application to biomedicine: Counting synapses

- Counting synapses:
  - Measure the number of connected components of the last image.
  - Good benchmark to test our framework: computation of $H_0$.
  - SynapCountJ: software to measure synaptic density evolution.
- Therefore:
  1. Mathematics formalization ...
  2. ... *for* software verification ...
  3. ... *for* real-life applications.
- Formalizing Algebraic Topology for *Kenzo* verification: How?

# A multitool approach (1/2)

Many proving tools are available for the formalization engineer.
They differ regarding different aspects:

- Automated / Interactive
- Checkers / Provers
- First order / Higher order
- Classical / Constructive

Our problem also poses different issues:

- Formalization of basic algebraic structures and algorithms.
- Verification of concrete *Kenzo* code.
- Certified execution of homological programs (different from *Kenzo*).

# A multitool approach (2/2)

Our idea is to take the best from each tool:

- Isabelle/HOL to formalize algorithms in a *classical* setting.
- ACL2 to verify *first order* fragments of *Kenzo* code.
- Coq to provide executability where first order is not enough and the constructiveness is ensured.

# Mathematics: Homological Algebra (1/3)

- A chain complex is $\{(C_n, d_n)\}_{n\in\mathbb{Z}}$, where each $C_n$ is an abelian group, and each $d_n : C_n \to C_{n-1}$ is a homomorphism satisfying $d_n \circ d_{n+1} = 0, \forall n \in \mathbb{Z}$.
- *Homology groups:* $H_n(C, d) := Ker(d_n)/Im(d_{n+1})$.
- Given two chain complexes $\{(C_n, d_n)\}_{n\in\mathbb{Z}}$ and $\{(C'_n, d'_n)\}_{n\in\mathbb{Z}}$, a *chain morphism* between them is a family $f$ of group homomorphisms $f_n : C_n \to C'_n, \forall n \in \mathbb{Z}$ satisfying $d'_n \circ f_n = f_{n-1} \circ d_n, \forall n \in \mathbb{Z}$.

# Mathematics: Homological Algebra (2/3)

- Given two chain complexes $C := \{(C_n, d_n)\}_{n\in\mathbb{Z}}$ and $C' := \{(C'_n, d'_n)\}_{n\in\mathbb{Z}}$ a *reduction* between them is $(f, g, h)$ where
  - $f : C \to C'$ and $g : C' \to C$ are chain morphisms
  - and $h$ is a family of homomorphisms (called *homotopy operator*) $h_n : C_n \to C_{n+1}$.

  satisfying

  1. $f \circ g = 1$
  2. $d \circ h + h \circ d + g \circ f = 1$
  3. $f \circ h = 0$
  4. $h \circ g = 0$
  5. $h \circ h = 0$

- If $(f, g, h) : C \to C'$ is a reduction, then $H(C) \cong H(C')$.

# Mathematics: Homological Algebra (3/3)

- Given a chain complex $(C, d)$, a *perturbation* for it is a family $\rho$ of group homomorphisms $\rho_n : C_n \to C_{n-1}$ such that $(C, d + \rho)$ is again a chain complex (that is to say: $(d + \rho) \circ (d + \rho) = 0$).
- A reduction $(f, g, h) : (C, d) \to (C', d')$ and a perturbation $\rho$ for $(C, d)$ are *locally nilpotent* if
  $\forall x \in C_n, \exists m \in \mathbb{N}$ such that $(h \circ \rho)^m(x) = 0$.

### Basic Perturbation Lemma

Let $(f, g, h) : (C, d) \to (C', d')$ be a reduction and be $\rho$ a perturbation for $(C, d)$ which are locally nilpotent. Then there exists a reduction $(f_\infty, g_\infty, h_\infty) : (C, d + \rho) \to (C', d'_\infty)$.

### Basic Perturbation Lemma Algorithm

Given a chain complex $(C, d)$ with effective homology and $\rho$ a perturbation for it satisfying the local nilpotence condition, then $(C, d + \rho)$ is a chain complex with effective homology.

# Formalizing with Isabelle/HOL

- Isabelle/HOL is an interactive theorem proving environment.
- Higher Order Logic (HOL) allows the modeller to translate the "by hand" proofs to the computer, in a "quite" direct way.
- First milestone: Jesús Aransay's proof of the Basic Perturbation Lemma in Isabelle/HOL.
- Isabelle statement:

  **theorem** **(in** *BPL)* *BPL:* **shows** *reduction D'*
  $(\!|$ *carrier = carrier C, mult = mult C, one = one C, diff =*
  $(\lambda x.$ *if* $x \in$ *carrier C then* $(\mathit{differ}_C)\ x \otimes_C (f \circ \delta \circ \Psi \circ g)\ x$
  *else* $\mathbf{1}_C)$ $|\!)$ $(f \circ \Phi)$ $(\Psi \circ g)$ $(h \circ \Phi)$

- Further challenge: program extraction.

# Formalizing with ACL2 (1/3)

- ACL2 = A Computational Logic for Applicative Common Lisp ($ACL^2$).
- ACL2 is:
  - A programming language (an *applicative* subset of Common Lisp).
  - A logic (a restricted first-order one, with few quantifiers).
  - A theorem prover for that logic (on programs properties).
- Could *Kenzo* be verified in ACL2?
- ACL2 is first order...
- ...but *Kenzo* intensively uses higher-order functional programming (functional coding of infinite sets).
- Isabelle/HOL is a higher order tool (Coq too).
- Pragmatic approach: ACL2 verification of *first order* fragments of *Kenzo*.

# Formalizing with ACL2 (2/3)

- *Kenzo* way of working:
  1. Construction of constant spaces (spheres, Moore spaces, ...): $\sim 20\%$
  2. Construction of new spaces from other ones (cartesian products, loop spaces,...): $\sim 60\%$
  3. Perform some computations (homology groups): $\sim 10\%$

**Concrete Goal**

*Verify the correctness of Kenzo constructors of constant spaces*

- *Kenzo* first order logic fragments
- *Kenzo* code $\rightarrow$ ACL2

**Case Study**

*Each Kenzo Simplicial Set is really a simplicial set*

# Formalizing with ACL2 (3/3)

## Definition

A *simplicial set* $K$, is a union $K = \bigcup_{q \geq 0} K^q$, where the $K^q$ are disjoints sets, together with functions:

$$\begin{array}{lll} \partial_i^q : K^q \to K^{q-1}, & q > 0, & i = 0, \ldots, q, \\ \eta_i^q : K^q \to K^{q+1}, & q \geq 0, & i = 0, \ldots, q, \end{array}$$

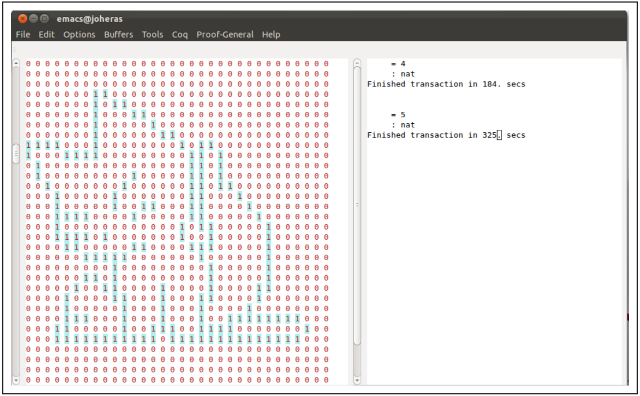
subject to the relations:

$$\begin{array}{llllll} (1) & \partial_i^{q-1}\partial_j^q & = & \partial_{j-1}^{q-1}\partial_i^q & \text{if} & i < j, \\ (2) & \eta_i^{q+1}\eta_j^q & = & \eta_{j+1}^{q+1}\eta_i^q & \text{if} & i \leq j, \\ (3) & \partial_i^{q+1}\eta_j^q & = & \eta_{j-1}^{q-1}\partial_i^q & \text{if} & i < j, \\ (4) & \partial_i^{q+1}\eta_i^q & = & \mathit{identity} & = & \partial_{i+1}^{q+1}\eta_i^q, \\ (5) & \partial_i^{q+1}\eta_j^q & = & \eta_j^{q-1}\partial_{i-1}^q & \text{if} & i > j+1, \end{array}$$

- Generic Simplicial Set Theory (J. Heras, on previous work by F. J. Martín-Mateos)
- From 4 definitions and 4 theorems
- Instantiates 3 definitions and 7 theorems
- The proof of the 7 theorems involves: 92 definitions and 969 theorems
- The proof effort is considerably reduced

# Formalizing with Coq/SSReflect

From digital images to homology in Coq:




- First order structures: possible also in ACL2.
- But . . . finite structures: all the power of SSReflect.
- Joint work inside the ForMath project: J. Heras, M. Dénès, G. Mata, A. Mörtberg, M. Poza, V. Siles, . . .

# Conclusions and future work

- Conclusions
  - Algebraic Topology is a good place to formalization
    - ⋆ The subject is rich enough (challenging tools and mathematics).
    - ⋆ We have a program difficult to test (*Kenzo*).
    - ⋆ We have a program with real-life applications (biomedical images).
  - In our context (software verification), executability is important.
  - Automation is necessary.
  - Big endeavor, team work is mandatory.
- Future work
  - From execution to *efficient* execution.
    - ⋆ Better algorithms (more math, more difficult to prove).
    - ⋆ Improving running environments in the proving tools.
  - Interoperability among the proving tools.