Tutorial
Formalization of Algebraic Topology
Talk 4

ACL2: Going down to first order. The case of Simplicial Topology

Julio Rubio

Universidad de La Rioja
Departamento de Matemáticas y Computación

Mathematics, Algorithms, Proofs, MAP 2009

Monastir (Tunisia), December 14th-18th, 2009

Summary

- Introduction.
- Rewriting systems and Simplicial Topology.
- Quantifier elimination.
- Infrastructure to prove $C(K) \Longrightarrow C^{ND}(K)$.
- Conclusions and further work.
- Conclusions of the tutorial.

Introduction

- 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).
- M. Andrés, L. Lambán, J. R., J. L. Ruiz-Reina.
  Formalizing Simplicial Topology in ACL2.
  Workshop ACL2 2007, Austin University, pp. 34-39.
- L. Lambán, F. J. Martín-Mateos, J. R., J. L. Ruiz Reina.
  When first order is enough: the case of Simplicial Topology.

The category Δ^* and the simplicial set Δ

- Recall: category Δ^*
  - Objects: $\mathbf{n} = \{0, 1, \ldots, n\}, \forall n \in \mathbb{N}$.
  - Morphisms: $\mu : \mathbf{n} \to \mathbf{m}$, increasing.
- Each morphism $\mu : \mathbf{n} \to \mathbf{m}$ can be identified with a list (its image) $(\mu_0, \ldots, \mu_n)$ where $0 \leq \mu_i \leq m, \forall 0 \leq i \leq n$.
- A canonical (universal) simplicial set Δ can be defined as the simplicial *complex* with
  $\Delta(\mathbf{n}) = \{(a_0, a_1, \ldots, a_n); a_0 \leq a_1 \leq \ldots \leq a_n \text{ and } a_i \in \mathbb{N}\}$.
- Roughly speaking:
  Δ encodes the same information as the category Δ^*.
- Rough consequence:
  all the properties of Δ which can be proved by using only the *simplicial identities* can be extended to *any* simplicial set.

Standard encoding of degenerate simplexes

- Recall:
  Given a simplicial set K and a simplex $x \in K_n$, there exists a unique expression $x = \eta_{i_1} \dots \eta_{i_t} \overline{x}$, with $\overline{x}$ *non-degenerate* (i.e. $\overline{x} \notin Im(\eta_j), \forall j$), and $0 \leq i_t < \dots < i_1$ (t could be equal to 0).
- Rewording it in terms of the simplicial set Δ:
- Any simplex l of Δ can be expressed in a unique way as a pair $(dl, l0)$ such that: $l = degenerate(dl, l0)$ with $l0$ a non-degenerate simplex and dl a strictly increasing list.
- Or more generally, expressed in terms of ACL2 elements:
  Any ACL2 list l can be expressed in a unique way as a pair $(dl, l0)$ such that: $l = degenerate(dl, l0)$ with $l0$ without two consecutive elements equal and dl a strictly increasing degeneracy list.

Simplicial identities as rewriting rules

- You can prove the previous theorem in ACL2, inside the simplicial set Δ (it is a problem of list manipulation) or...
- ...you can give a more abstract proof based only in the simplicial identities seen as *rewriting rules*.
- Only two identities are needed for this concrete result:
  - $\eta_i \eta_j = \eta_{j+1} \eta_i$ if $i \leq j$
  - $\partial_i \eta_i = id$.
- That gives two kind of rewriting rules:
  - $\eta_i \eta_j \longrightarrow_o \eta_{j+1} \eta_i$ if $i \leq j$ (*o-rules*, ordering rules)
  - $\partial_i \eta_i \longrightarrow_r id$ (*r-rules*, reduction rules).
- This allows defining, in ACL2, an *abstract* reduction system (framework previously developed by J. L. Ruiz-Reina and F. J. Martín-Mateos).

Properties of the simplicial rewriting system

- It is necessary only to prove two properties on this formal system:
  - It is noetherian.
  - It is locally confluent.
- Then by using the formalization of Newman's Lemma in J. L. Ruiz-Reina, J. A. Alonso, M. J. Hidalgo, F. J. Martín-Mateos, *Formal Proofs About Rewriting Using ACL2*. Annals of Mathematics and Artificial Intelligence **36** (2002) 239–262.
- we can prove in ACL2 that the simplicial rewriting systems is *convergent*
- and then the canonical decomposition $x = \eta_{i_1} \ldots \eta_{i_t} \overline{x}$ follows.

From Simplicial to Algebraic Topology

- Can this "theoretical computer science" (= rewriting systems) approach be generalized?
- Many results in Algebraic Topology take the form:

$$\forall K \; \forall n \; \forall x \in K_n, T(x) = T'(x)$$

  where T and T' are linear combinations of *simplicial operators* (i. e. sequences of face and degeneracy operations).
- For instance: $\forall K \; \forall n \; \forall x \in K_n, d_n d_{n+1}(x) = 0$.
- In principle, this kind of statements requires:
  - Higher order logic.
  - Dependent types.

Quantifier elimination

- $\forall K \ \forall n \ \forall x \in K_n, T(x) = T'(x)$
- If we work in the universal simplicial set Δ:
  $\forall n \ \forall x \in \Delta_n, T(x) = T'(x)$.
- But $x \in \Delta_n$ implies x can be interpreted as any list of length $n + 1$.
- Thus: $\forall n \ \ T^{(n)} = T'^{(n)}$.
- Can we even eliminate this last quantifier to obtain as statement:

$$T = T'?$$

Example

- A (faulty) proof of $d_n \circ d_{n+1} = 0$.
  - $d_{n+1} = (-1)^{n+1}\partial_{n+1}^{(n+1)} + (-1)^n\partial_n^{(n+1)} + \ldots - \partial_1^{(n+1)} + \partial_0^{(n+1)}$ and
    $d_n = (-1)^n\partial_n^{(n)} + (-1)^{n-1}\partial_{n-1}^{(n)} + \ldots - \partial_1^{(n)} + \partial_0^{(n)}$
  - Let us skip the superindices:
    $d_{n+1} = (-1)^{n+1}\partial_{n+1} + (-1)^n\partial_n + \ldots - \partial_1 + \partial_0$ and
    $d_n = (-1)^n\partial_n + (-1)^{n-1}\partial_{n-1} + \ldots - \partial_1 + \partial_0$
  - Thus: $d_{n+1} = (-1)^{n+1}\partial_{n+1} + d_n$.
  - $d_n \circ d_{n+1} = [(-1)^n\partial_n + d_{n-1}][(-1)^{n+1}\partial_{n+1} + d_n] =$
    $= -\partial_n\partial_{n+1} + (-1)^n\partial_n d_n + (-1)^{n+1}d_{n-1}\partial_{n+1} + d_{n-1}d_n$.
  - By induction: $d_n \circ d_{n+1} = -\partial_n\partial_{n+1} + (-1)^n\partial_n d_n + (-1)^{n+1}d_{n-1}\partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n\partial_n\partial_{n+1} + d_{n-1}\partial_{n+1}$.
  - QED.
- Only using induction+simplification (ACL2!).
- Can this kind of heuristic reasoning be formalized?

Three models

- Idea: when working over Δ, if a simplicial equation is true in a dimension n, it is also true $\forall m \geq n \ldots$
- ... because for any simplicial *complex* faces and degeneracies are defined in a *generic* way (i.e. a way independent from the concrete complex and the concrete dimensions).
- Three layers:
  - *Model 1:* Simplicial sets expressed as graded sets, and functions defining faces and degeneracies (and chain complexes over them).
  - *Model 2:* Simplicial rewriting rules (symbolic, without evaluation on simplices), but with dimension annotations.
  - *Model 3:* Simplicial terms and polynomials without dimension annotations.
- *From Model 2 to Model 1:* trough the universal property of Δ.
- *From Model 3 to Model 2:* for each proof carried out over *Model 3*, a dimension n can be computed such that the proof can be translated to *Model 2* for all $m \geq n$.
- The three layers can be formalized in ACL2.

The first model in ACL2

The higher-order aspect of the first model can be simulated in ACL2 by means of an *encapsulate*.

```
(encapsulate
 (((K * *) => *)
  ((d * * *) => *)
  ((n * * *) => *))
  ...
 (defthm simplicial-id1
   (implies (and (K n x)
                 (natp n)
                 (natp i)
                 (natp j)
                 (<= j i)
                 (< i n))
    (equal (d (- n 1) i (d n j x))
           (d (- n 1) j (d n (+ 1 i) x)))))
   ...
```

Third model: simplicial terms

- A *simplicial operator* is any sequence of faces and degeneracies.
- Example: $\partial_3\eta_0\partial_3\partial_2\eta_5$.
- A *simplicial term* is a simplicial operator in canonical form.
- In the example: $\eta_3\eta_0\partial_2\partial_3\partial_4$.
- In ACL2 a simplicial term is represented as a pair of two lists of natural numbers, the first one strictly decreasing, and the second one strictly increasing.
- In the example: `((3 0) (2 3 4))`
- Simplicial terms can be composed, following the simplicial rules.
- We have proved in ACL2 that the set of simplicial terms together with this binary operation form a monoid (the unity being the list with two empty lists).

Third model: simplicial polynomials

- Given a monoid $(\mathcal{T}, \circ, \mathbf{1})$, we can construct the set $\mathcal{P}$ of linear combinations (with integer coefficients) over $\mathcal{T}$.
- By extending the product in $\mathcal{T}$, we can endow $\mathcal{P}$ with a ring structure.
- This construction can be formalized in ACL2 as a *generic theory* (a tool previously developed in ACL2 by J. L. Ruiz-Reina and F. J. Martín-Mateos).
- Example of theorem inferred:

```
(defthm cmp-pol-pol-add-pol-pol-distributive-l
  (implies (and (pol-p p1)
                (pol-p p2)
                (pol-p p3))
     (equal (cmp-pol-pol (add-pol-pol p1 p2) p3)
            (add-pol-pol (cmp-pol-pol p1 p3)
                         (cmp-pol-pol p2 p3)))))
```

The differential example revisited

The “heuristic” proof of $d_n \circ d_{n+1} = 0$ can be now formalized in ACL2

```
(defthm cmp-d-d=0
  (implies (and (natp n)
                (< 0 n))
    (equal (cmp-sp-sp (d n) (d (1+ n)))
           (add-pol-pol-id))))
```

Not only it can be formalized, but it can be highly automated.

Furthermore, it can be “lifted” to *Model 1* (through *Model 2*) in ACL2 and expressed in the standard textbook way.

A reduction from $C(K)$ to $C^{ND}(K)$

- Recall:
  - Let K be a simplicial set.
    - ⋆ Define: $C_n(K) := \mathbb{Z}[K_n]$, free $\mathbb{Z}$-module generated by n-simplexes.
    - ⋆ Define: $d_n(x) := \sum_{i=0}^{n}(-1)^i \partial_i x$ over generators, and extend linearly.
  - Define $C^{ND}(K) := C(K)/D(K)$, where $D_n(K) := \mathbb{Z}[K_n^D]$, with K_n^D the set of degenerate n-simplexes of K.
  - Theorem: there exists a reduction $(f, g, h) : C(K) \Rightarrow C^{ND}(K)$.
- We are going to use the previous infrastructure on the ring of simplicial polynomials to give an ACL2 proof of this result.

An experimental result

- In
  J. R., F. Sergeraert.
  Supports Acycliques and Algorithmique.
  Astérisque **192** (1990) pp. 35-55.
- we have found experimentally the following formula for $(f, g, h) : C(K) \Rightarrow C^{ND}(K)$.
  - f is simply the canonical projection.
  - $g_n = \sum (-1)^{\sum_{i=1}^{p} a_i + b_i} \eta_{a_p} \ldots \eta_{a_1} \partial_{b_1} \ldots \partial_{b_p}$
    where the indexes range over $0 \leq a_1 < b_1 < \ldots < a_p < b_p \leq n$, with $0 \leq p \leq (n+1)/2$.
  - $h_n = \sum (-1)^{a_{p+1} + \sum_{i=1}^{p} a_i + b_i} \eta_{a_{p+1}} \eta_{a_p} \ldots \eta_{a_1} \partial_{b_1} \ldots \partial_{b_p}$
    where the indexes range over $0 \leq a_1 < b_1 < \ldots < a_p < a_{p+1} \leq b_p \leq n$, with $0 \leq p \leq (n+1)/2$.
- and we claimed there, without proof, that they define a homotopy equivalence.

Obtaining a reduction

- Other proofs were known, but no one (up to our knowledge) is given by means of explicit programmable formula.
- In fact (f, g, h) does not define a reduction, but only a homotopy equivalence.
- Our definitions satisfy:
  1. $fg = id$
  2. $dh + hd + fg = id$
  3. $fh = 0$, but
  4. $hg \neq 0$
  5. $hh \neq 0$
- Nevertheless, there is a generic procedure to obtain an *actual* reduction from (f, g, h) satisfying (1) and (2).
- This can be encoded in *Model 1*, since it does not require complex rewriting.

Devising an ACL2 proof

- The simplicial ring technique can be applied over *one* space/chain complex, but in the statement there are now *two* chain complexes.
- Solution: do not pass too early to the quotient.
- We model everything on $C(K)$, the "big" chain complex.
  - The morphism f is replaced by the simplicial polynomial $F = id$.
  - The morphism g is replaced by a simplicial polynomial G (thus it is interpreted as a morphism $C(K) \rightarrow C(K)$).
  - The homotopy operator h is replaced by a simplicial polynomial H.
- By applying induction and simplification over the simplicial ring, we prove in ACL2
  - $dG = Gd$
  - $dH + Hd + G = id$
- and several properties proving that G and H are well behaved with respect to degeneracies.
- Then *Model 1* can be used to express the theorem in the usual terms.

Conclusions and further work

- Conclusions:
  - ACL2 can be used to formalize (part of) Simplicial and Algebraic Topology.
  - Going down to first order, through the *simplicial ring*, a higher degree of automation is reached.
  - In ACL2, we are always verifying Common Lisp programs, close relatives of Kenzo ones.
- Further work:
  - To continue exploring and extending the first order simplicial ring technique.
  - Up to now, we have been guided by Kenzo requirements:
    - ⋆ The Kenzo representation of degenerate simplexes (proved correct by means of ACL2, Calculemus 2009).
    - ⋆ Justifying why in Kenzo we can work with the smaller chain complex $C^{ND}(K)$.
  - *Next step:* Eilenberg-Zilber theorem (the bridge between Geometry and Algebra).

Conclusions of the tutorial

- Algebraic Topology seems a good area to experiment with the formalization and mechanization of Mathematics:
  - Infinite dimensional spaces occur there in a natural (and unavoidable) way.
  - It is needed to deal with complicated algebraic structures hierarchies.
  - There are difficult combinatorial proofs.
- In summary: logic is complicated in Algebraic Topology, and combinatorics too.
- Challenge guiding our approach: the verification of the Kenzo system. (Formal mathematics *for* program verification.)
- Our multi-tool approach seems to be suitable:
  - Isabelle/HOL to get proofs as close as possible to those of books and papers.
  - Coq when the constructiveness of proofs needs to be ensured.
  - ACL2 when first order is enough, and we need to be very near the Kenzo Common Lisp code.