Applying ACL2 to the Formalization of Algebraic Topology: Simplicial Polynomials

L. Lambán[*], *F.J. Martín-Mateos*[**], *J. Rubio*[*] *and* *J.-L. Ruiz-Reina*[**]

[*] Dpto. de Matemáticas y Computación (Universidad de La Rioja, Spain)
[**] Dpto. de Ciencias de la Comp. e Inteligencia Artificial (Universidad de Sevilla, Spain)

Introduction

- Kenzo symbolic computation system: a Common Lisp program devoted to Algebraic (Simplicial) Topology.
    - A research tool: used to obtain relevant results in the field, neither confirmed nor refuted by any other means.
- The following question makes sense: Is it Kenzo correct?
- Our goal: we want to ***formally prove correcteness properties*** of the algorithms implemented in Kenzo
- Since Kenzo is coded in Common Lisp, ACL2 seems a natural candidate for this task
    - Is it first-order enough to reason about algebraic topology?

Introduction

- Formal proofs of Kenzo properties imply the following:
  1. Formal correctness proofs of the implemented algorithms
  2. Formalizing the underlying theory: algebraic and simplicial topology
- Regarding the first issue, some formal verification of functions implemented in Kenzo has already been carried out (*Calculemus 2009*)
- This talk is about the second issue: formalization in ACL2 of some aspects of the theory of Simplicial Topology
  - Our first step: formal proof of the *Normalization Theorem* of Simplicial Topology

Simplicial sets

- Simplicial Topology is a subarea of Topology studying topological properties of spaces by means of combinatorial models.
- A *simplicial set* is a graded set $\{K_n\}_{n \in \mathbb{N}}$ (*n-simplexes*) together with operators $\partial_i^{(n)} : K_n \to K_{n-1}$ and $\eta_i^{(n)} : K_n \to K_{n+1}$ (*faces* and *degeneracies*, resp.), satisfying the following *simplicial identities*:

$$
\begin{array}{lllll}
(1) & \partial_i^{n-1}\partial_j^n & = & \partial_j^{n-1}\partial_{i+1}^n & \text{if} \quad i \geq j, \\
(2) & \eta_i^{n+1}\eta_j^n & = & \eta_{j+1}^{n+1}\eta_i^n & \text{if} \quad i \leq j, \\
(3) & \partial_i^{n+1}\eta_j^n & = & \eta_{j-1}^{n-1}\partial_i^n & \text{if} \quad i < j, \\
(4) & \partial_i^{n+1}\eta_j^n & = & \eta_j^{n-1}\partial_{i-1}^n & \text{if} \quad i > j+1, \\
(5) & \partial_i^{n+1}\eta_i^n & = & \partial_{i+1}^{n+1}\eta_i^n & = \quad id^n,
\end{array}
$$

Simplicial sets: some intuition

- Simplicial sets are an abstraction, but we can give some geometrical and combinatorial intuition.
- Geometrical: spaces resulting from triangulation of topological spaces:
  - n-simplexes in K_n can be seen as n dimensional "triangles"
  - The operators $\partial_i^{(n)}$ gives us the "sides" of the triangle (or "faces" of a tetrahedron).
- A particular simplicial set can also give us some combinatorial intuition:
  - n-simplexes: non-decreasing integer lists $[a_0, a_1, \ldots, a_n]$ (vertices of the "triangle")
  - $\partial_i^{(n)}$: delete the i-th element
  - $\eta_i^{(n)}$: duplicate the i-th element
  - This gives some intuition about the meaning of the simplicial identities

Simplicial sets: some intuition

- Simplicial sets are an abstraction, but we can give some geometrical and combinatorial intuition.
- Geometrical: spaces resulting from triangulation of topological spaces:
  - n-simplexes in K_n can be seen as n dimensional “triangles”
  - The operators $\partial_i^{(n)}$ gives us the “sides” of the triangle (or “faces” of a tetrahedron).
- A particular simplicial set can also give us some combinatorial intuition:
  - n-simplexes: non-decreasing integer lists $[a_0, a_1, \ldots, a_n]$ (vertices of the “triangle”)
  - $\partial_i^{(n)}$: delete the i-th element
  - $\eta_i^{(n)}$: duplicate the i-th element
  - This gives some intuition about the meaning of the simplicial identities

Simplicial sets: some intuition

- Simplicial sets are an abstraction, but we can give some geometrical and combinatorial intuition.
- Geometrical: spaces resulting from triangulation of topological spaces:
  - n-simplexes in K_n can be seen as n dimensional “triangles”
  - The operators $\partial_i^{(n)}$ gives us the “sides” of the triangle (or “faces” of a tetrahedron).
- A particular simplicial set can also give us some combinatorial intuition:
  - n-simplexes: non-decreasing integer lists $[a_0, a_1, \ldots, a_n]$ (vertices of the “triangle”)
  - $\partial_i^{(n)}$: delete the i-th element
  - $\eta_i^{(n)}$: duplicate the i-th element
  - This gives some intuition about the meaning of the simplicial identities

Simplicial sets

- A *simplicial set* is a graded set $\{K_n\}_{n\in\mathbb{N}}$ (*n-simplices*) together with operators $\partial_i^{(n)} : K_n \to K_{n-1}$ and $\eta_i^{(n)} : K_n \to K_{n+1}$ (*faces* and *degeneracies*, resp.), satisfying the following *simplicial identities*:

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

Defining simplicial sets in ACL2

A generic simplicial set using `encapsulate`

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


  ;;; Inside this encapsulate, we assume analogously
  ;;; all the simplicial identities.

  .....)
```

- `(K n x)` represents $x \in K_n$,
- `(d m i x)` and `(n m i x)` represent $\eta_i^{(m)}(x)$ and $\partial_i^{(m)}(x)$, resp.

Chain complexes

- The set of *n-chains* (denoted as $C_n(K)$) is the abelian group freely generated by K_n.
  - That is, linear combinations of elements of K_n with integer coefficients
  - In ACL2, ordered lists of pairs of the form `(i . x)`, where `i` is a non-null integer and `x` is a n-simplex
- The *differential* is defined on $x \in K_n$ as $d_n(x) = \sum_{i=0}^{n}(-1)^i \partial_i^{(n)}(x)$
  - Extended by linearity to chains, defining $d_n : C_n(K) \rightarrow C_{n-1}(K)$
- It can be proved that $d_n \circ d_{n+1} = 0$ (*differential property*)
- In Algebra, we say that $\{(C_n(K), d_n)\}_{n \in \mathbb{N}}$ is a *chain complex*
- Algebraic properties of the chain complex associated to a simplicial set give us topological information

Proving simplicial topology theorems in ACL2

- An example: an (informal) proof of $d_n \circ d_{n+1} = 0$.
  - $d_n = \sum_{i=0}^{n} (-1)^i \partial_i^{(n)}$ and $d_{n+1} = \sum_{i=0}^{n+1} (-1)^i \partial_i^{n+1}$
  - If we omit the superindexes, we can recursively define: $d_{n+1} = (-1)^{n+1}\partial_{n+1} + d_n$.
  - Therefore, $d_n \circ d_{n+1} = [(-1)^n \partial_n + d_{n-1}][(-1)^{n+1}\partial_{n+1} + d_n] =$ $= -\partial_n \partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1} \partial_{n+1} + d_{n-1} d_n$.
  - By induction, $d_{n-1} d_n = 0$, so: $d_n \circ d_{n+1} = -\partial_n \partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1} \partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n \partial_n \partial_{n+1} + d_{n-1} \partial_{n+1}$.
  - Applying the lemma, $d_n \circ d_{n+1} = 0$. QED.

Proving simplicial topology theorems in ACL2

- An example: an (informal) proof of $d_n \circ d_{n+1} = 0$.
  - $d_n = \sum_{i=0}^{n} (-1)^i \partial_i^{(n)}$ and $d_{n+1} = \sum_{i=0}^{n+1} (-1)^i \partial_i^{n+1}$
  - If we omit the superindexes, we can recursively define: $d_{n+1} = (-1)^{n+1} \partial_{n+1} + d_n$.
  - Therefore, $d_n \circ d_{n+1} = [(-1)^n \partial_n + d_{n-1}][(-1)^{n+1} \partial_{n+1} + d_n] = = -\partial_n \partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1} \partial_{n+1} + d_{n-1} d_n$.
  - By induction, $d_{n-1} d_n = 0$, so: $d_n \circ d_{n+1} = -\partial_n \partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1} \partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n \partial_n \partial_{n+1} + d_{n-1} \partial_{n+1}$.
  - Applying the lemma, $d_n \circ d_{n+1} = 0$. QED.

Proving simplicial topology theorems in ACL2

- An example: an (informal) proof of $d_n \circ d_{n+1} = 0$.
  - $d_n = \sum_{i=0}^{n} (-1)^i \partial_i^{(n)}$ and $d_{n+1} = \sum_{i=0}^{n+1} (-1)^i \partial_i^{n+1}$
  - If we omit the superindexes, we can recursively define: $d_{n+1} = (-1)^{n+1}\partial_{n+1} + d_n$.
  - Therefore, $d_n \circ d_{n+1} = [(-1)^n \partial_n + d_{n-1}][(-1)^{n+1}\partial_{n+1} + d_n] =$ $= -\partial_n\partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1} + d_{n-1}d_n$.
  - By induction, $d_{n-1}d_n = 0$, so: $d_n \circ d_{n+1} = -\partial_n\partial_{n+1} + (-1)^n\partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n \partial_n \partial_{n+1} + d_{n-1}\partial_{n+1}$.
  - Applying the lemma, $d_n \circ d_{n+1} = 0$. QED.

Proving simplicial topology theorems in ACL2

- An example: an (informal) proof of $d_n \circ d_{n+1} = 0$.
  - $d_n = \sum_{i=0}^{n}(-1)^i \partial_i^{(n)}$ and $d_{n+1} = \sum_{i=0}^{n+1}(-1)^i \partial_i^{n+1}$
  - If we omit the superindexes, we can recursively define: $d_{n+1} = (-1)^{n+1}\partial_{n+1} + d_n$.
  - Therefore, $d_n \circ d_{n+1} = [(-1)^n \partial_n + d_{n-1}][(-1)^{n+1}\partial_{n+1} + d_n] =$ $= -\partial_n \partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1} + d_{n-1}d_n$.
  - By induction, $d_{n-1}d_n = 0$, so: $d_n \circ d_{n+1} = -\partial_n \partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n \partial_n \partial_{n+1} + d_{n-1}\partial_{n+1}$.
  - Applying the lemma, $d_n \circ d_{n+1} = 0$. QED.

Proving simplicial topology theorems in ACL2

- An example: an (informal) proof of $d_n \circ d_{n+1} = 0$.
  - $d_n = \sum_{i=0}^{n}(-1)^i \partial_i^{(n)}$ and $d_{n+1} = \sum_{i=0}^{n+1}(-1)^i \partial_i^{n+1}$
  - If we omit the superindexes, we can recursively define: $d_{n+1} = (-1)^{n+1}\partial_{n+1} + d_n$.
  - Therefore, $d_n \circ d_{n+1} = [(-1)^n \partial_n + d_{n-1}][(-1)^{n+1}\partial_{n+1} + d_n] =$ $= -\partial_n\partial_{n+1} + (-1)^n\partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1} + d_{n-1}d_n$.
  - By induction, $d_{n-1}d_n = 0$, so: $d_n \circ d_{n+1} = -\partial_n\partial_{n+1} + (-1)^n\partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n \partial_n\partial_{n+1} + d_{n-1}\partial_{n+1}$.
  - Applying the lemma, $d_n \circ d_{n+1} = 0$. QED.

Proving simplicial topology theorems in ACL2

- An example: an (informal) proof of $d_n \circ d_{n+1} = 0$.
  - $d_n = \sum_{i=0}^{n}(-1)^i \partial_i^{(n)}$ and $d_{n+1} = \sum_{i=0}^{n+1}(-1)^i \partial_i^{n+1}$
  - If we omit the superindexes, we can recursively define: $d_{n+1} = (-1)^{n+1}\partial_{n+1} + d_n$.
  - Therefore, $d_n \circ d_{n+1} = [(-1)^n \partial_n + d_{n-1}][(-1)^{n+1}\partial_{n+1} + d_n] =$ $= -\partial_n\partial_{n+1} + (-1)^n \partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1} + d_{n-1}d_n$.
  - By induction, $d_{n-1}d_n = 0$, so: $d_n \circ d_{n+1} = -\partial_n\partial_{n+1} + (-1)^n\partial_n d_n + (-1)^{n+1} d_{n-1}\partial_{n+1}$
  - Lemma: $\partial_n d_n = (-1)^n \partial_n \partial_{n+1} + d_{n-1}\partial_{n+1}$.
  - Applying the lemma, $d_n \circ d_{n+1} = 0$. QED.

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Proving simplicial topology theorems in ACL2

- Although more complicated than the previous one, most of the proofs we have to deal with have the same features:
  - The superindexes can be omited (later safely recovered)
  - We calculate with symbolic expressions involving linear combinations of composition of face and degeneracy maps.
  - Definitions by recursion, proofs by induction
  - We apply equational properties about linearity, compositions of functions and the simplicial indentities.
  - The simplexes (and chains) on which the expressions are applied play no role in the proof
- To reflect this in our formal proofs, we introduce the framework of *simplicial polynomials:*
  - First-order ACL2 objects representing linear combinations of compositions of simplicial operators

Simplicial terms in ACL2

- Simplical terms represent composition of simplicial operators
- Note: the simplicial identities define a canonical form
  - Any composition of simplicial operators is equal to a unique composition of simplicial operators of the form

$$\eta_{i_k} \cdots \eta_{i_1} \partial_{j_1} \cdots \partial_{j_l}$$

  with $i_k > \cdots > i_1$ and $j_1 < \cdots < j_l$
- Example:
  - The composition $\partial_5^5 \eta_3^4 \partial_1^5 \partial_2^6 \eta_4^5$ can be put as $\eta_3 \eta_2 \partial_1 \partial_2 \partial_5$ and this can be represented by the two-element list `((3 2) (1 2 5))`.
- A *simplicial term* is a pair of lists of natural numbers in such a canonical form, representing a composition of simplicial operators

Simplicial polynomials

- A *simplicial polynomial* is a symbolic expression representing linear combinations of simplicial terms
  - Example: $3 \cdot \eta_5\eta_4\eta_2\partial_1\partial_3 - 2 \cdot \eta_3\eta_2\partial_1$
- In ACL2, simplicial polynomials are represented as lists of pairs of integers and simplicial terms.
  - Only in normal form: the list is ordered w.r.t. a total order on terms and we only allow non-null coefficients
  - Example: `((3 . ((5 4 2) (1 3))) (-2 . ((3 2) (1))))`
- That is, simplicial polynomials are first-order canonical representations of functions from $C_n(K)$ to $C_m(K)$

The ring of simplicial polynomials

- Sum and product of simplicial polynomials can also be defined, reflecting addition and composition of the functions represented (and returning its results also in normal form).
- For example:
  - $\boldsymbol{p}_1 = 3 \cdot \eta_4\eta_1\partial_3\partial_6\partial_7 - 2 \cdot \eta_1\partial_3\partial_4$
  - $\boldsymbol{p}_2 = \eta_3\partial_4\partial_6 + 2 \cdot \eta_1\partial_3\partial_4$
  - $\boldsymbol{p}_1 + \boldsymbol{p}_2 = \eta_3\partial_4\partial_6 + 3 \cdot \eta_4\eta_1\partial_3\partial_6\partial_7$
  - $\boldsymbol{p}_1 \cdot \boldsymbol{p}_2 = -2\cdot\eta_1\partial_3\partial_4\partial_6 - 4\cdot\eta_2\eta_1\partial_2\partial_3\partial_4\partial_5 + 3\cdot\eta_4\eta_1\partial_4\partial_6\partial_7\partial_8 + 6\cdot\eta_4\eta_2\eta_1\partial_2\partial_3\partial_4\partial_7\partial_8$
- We proved in ACL2 that the set of simplicial polynomials together with the addition and composition operations form a *ring with identity*
  - The ring of simplicial polynomials was obtained as an (automatic) instantiation of a generic ring of linear combinations of elements of a monoid
- We extensively apply ring properties in our proofs

Simplicial polynomials: a tool

- Note: our final goal is to do formalizations based on the functions `(K ...)`, `(d ...)` and `(n ...)` introduced by the previous `encapsulate`
  - Since that is a faithful and precise formalization of the notion of simplical set (what we call the *standard* framework)
- Simplicial polynomials are only a tool for doing that, trying to reflect our informal calculations by hand
- Once a property is proved in the polynomial framework, we must “lift” the property to the standard framework.

Lifting properties

- To "lift" properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \boldsymbol{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To “lift” properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \boldsymbol{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To “lift” properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \boldsymbol{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To "lift" properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \boldsymbol{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To "lift" properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \mathbf{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To "lift" properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \mathbf{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To "lift" properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \mathbf{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

Lifting properties

- To “lift” properties we define an evaluation function:
  - `eval-sp`($\boldsymbol{p}$,n,c) evaluates a polynomial $\boldsymbol{p}$ on a chain $c \in C_n(K)$
  - Key property: `eval-sp` is an homomorphism from the ring of polynomials to the ring of functions on chains
  - Note: `eval-sp` reintroduces the dimension (and this only makes sense when $\boldsymbol{p}$ is *valid* for dimension n)
- Example: proof of $d_n \circ d_{n+1}(c) = 0$, for all $c \in C_{n+1}(K)$
  - We define the function d_n (in the standard framework)
  - We also define the polynomial $\boldsymbol{d}_n$, representing d_n
  - We prove in the simplicial polynomial ring the formula $\boldsymbol{d}_n \cdot \boldsymbol{d}_{n+1} = \boldsymbol{0}$ (as sketched by the previous hand proof)
  - We prove that $\boldsymbol{d}_n$ is valid for dimension n
  - We prove that `eval-sp`($\boldsymbol{d}_n$,n,c)$= d_n(c)$
  - Finally, we apply `eval-sp` to both sides of the polynomial formula and we obtain the desired property in the standard framework

A non-trivial example: the Normalization Theorem

- The *homology groups* of a simplical set K are the quotient groups $H_n(C(K)) = Ker(d_n)/Im(d_{n+1})$
  - Homology groups provide topological information and are the main objects to be computed by Kenzo
- In fact, Kenzo builds a simpler chain complex with the same homology groups:
  - We say that a n-simplex x is *degenerate* if exists $y \in K_{n-1}$ such that $x = \eta_i^{(n)}(y)$ for some $0 \leq i \leq n$. Otherwise, it is *non-degenerate*
  - Let $C_n^N(K)$ denote the free abelian group generated by non-degenerate simplexes
  - Let $f_n : C_n(K) \to C_n^N(K)$ be the function that eliminates the degenerate addends of a chain (*normalization function*)
  - Let $d_n^N = f_n \circ d_n$
  - Then $\{(C_n^N(K), d_n^N)\}_{n \in \mathbb{N}}$ is a chain complex
- **Normalization Theorem**: $H_n(C(K)) \cong H_n(C^N(K)), \forall n \in \mathbb{N}$

A non-trivial example: the Normalization Theorem

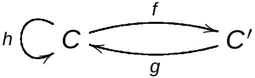
- The *homology groups* of a simplical set K are the quotient groups $H_n(C(K)) = Ker(d_n)/Im(d_{n+1})$
  - Homology groups provide topological information and are the main objects to be computed by Kenzo
- In fact, Kenzo builds a simpler chain complex with the same homology groups:
  - We say that a n-simplex x is *degenerate* if exists $y \in K_{n-1}$ such that $x = \eta_i^{(n)}(y)$ for some $0 \leq i \leq n$. Otherwise, it is *non-degenerate*
  - Let $C_n^N(K)$ denote the free abelian group generated by non-degenerate simplexes
  - Let $f_n : C_n(K) \to C_n^N(K)$ be the function that eliminates the degenerate addends of a chain (*normalization function*)
  - Let $d_n^N = f_n \circ d_n$
  - Then $\{(C_n^N(K), d_n^N)\}_{n \in \mathbb{N}}$ is a chain complex
- **Normalization Theorem**: $H_n(C(K)) \cong H_n(C^N(K)), \forall n \in \mathbb{N}$

A non-trivial example: the Normalization Theorem

- The *homology groups* of a simplical set K are the quotient groups $H_n(C(K)) = Ker(d_n)/Im(d_{n+1})$
  - Homology groups provide topological information and are the main objects to be computed by Kenzo
- In fact, Kenzo builds a simpler chain complex with the same homology groups:
  - We say that a n-simplex x is *degenerate* if exists $y \in K_{n-1}$ such that $x = \eta_i^{(n)}(y)$ for some $0 \leq i \leq n$. Otherwise, it is *non-degenerate*
  - Let $C_n^N(K)$ denote the free abelian group generated by non-degenerate simplexes
  - Let $f_n : C_n(K) \rightarrow C_n^N(K)$ be the function that eliminates the degenerate addends of a chain (*normalization function*)
  - Let $d_n^N = f_n \circ d_n$
  - Then $\{(C_n^N(K), d_n^N)\}_{n\in\mathbb{N}}$ is a chain complex
- **Normalization Theorem**: $H_n(C(K)) \cong H_n(C^N(K)), \forall n \in \mathbb{N}$

The Normalization Theorem: a stronger version

- A *strong homotopy equivalence* is a 5-tuple (C, C', f, g, h)



where $C = (M, d)$ and $C' = (M', d')$ are chain complexes, $f : C \to C'$ and $g : C' \to C$ are chain morphisms, $h = (h_i : M_i \to M_{i+1})_{i \in \mathbb{N}}$ is a family of homomorphisms (called *homotopy operator*), which satisfy the following three properties for all $i \in \mathbb{N}$:

(1) $f_i \circ g_i = id_{M'_i}$
(2) $d_{i+2} \circ h_{i+1} + h_i \circ d_{i+1} + g_{i+1} \circ f_{i+1} = id_{M_{i+1}}$
(3) $f_{i+1} \circ h_i = 0$

If, in addition the 5-tuple satisfies the following two properties:

(4) $h_i \circ g_i = 0$
(5) $h_{i+1} \circ h_i = 0$

then we say that it is a *reduction*.

The Normalization Theorem: a stronger version

- A reduction between chain complexes describes a situation where homological information is preserved
- That is, if (C, C', f, g, h) is a reduction, then $H_n(C) \cong H_n(C'), \forall n \in \mathbb{N}$
- We have proved a reduction version of the Normalization Theorem
- That is, we have defined appropriate f, g and h and proved that $(C(K), C^N(K), f, g, h)$ is a reduction.

A conjecture

- In J. Rubio, F. Sergeraert, *“Supports Acycliques and Algorithmique”*, Astérisque **192** (1990), it was experimentally found the following formula for $(C(K), C^N(K), f, g, h)$
  - f_n is the normalization function.
  - $g_n = \sum (-1)^{\sum_{i=1}^{p} a_i+b_i} \eta_{a_p} \dots \eta_{a_1} \partial_{b_1} \dots \partial_{b_p}$ where the indexes range over $0 \leq a_1 < b_1 < \dots < a_p < b_p \leq n$, with $0 \leq p \leq (n+1)/2$.
  - $h_n = \sum (-1)^{a_{p+1}+\sum_{i=1}^{p} a_i+b_i} \eta_{a_{p+1}} \eta_{a_p} \dots \eta_{a_1} \partial_{b_1} \dots \partial_{b_p}$ where the indexes range over $0 \leq a_1 < b_1 < \dots < a_p < a_{p+1} \leq b_p \leq n$, with $0 \leq p \leq (n+1)/2$.

  and claimed there, without proof, that they define a strong homotopy equivalence
- Our contribution:
  - We did a hand proof of the conjecture
  - We formalized it in ACL2, thus proving the reduction version of the Normalization Theorem

A conjecture

- In J. Rubio, F. Sergeraert, *“Supports Acycliques and Algorithmique”*, Astérisque **192** (1990), it was experimentally found the following formula for $(C(K), C^N(K), f, g, h)$
  - f_n is the normalization function.
  - $g_n = \sum (-1)^{\sum_{i=1}^{p} a_i + b_i} \eta_{a_p} \dots \eta_{a_1} \partial_{b_1} \dots \partial_{b_p}$ where the indexes range over $0 \leq a_1 < b_1 < \dots < a_p < b_p \leq n$, with $0 \leq p \leq (n+1)/2$.
  - $h_n = \sum (-1)^{a_{p+1} + \sum_{i=1}^{p} a_i + b_i} \eta_{a_{p+1}} \eta_{a_p} \dots \eta_{a_1} \partial_{b_1} \dots \partial_{b_p}$ where the indexes range over $0 \leq a_1 < b_1 < \dots < a_p < a_{p+1} \leq b_p \leq n$, with $0 \leq p \leq (n+1)/2$.

  and claimed there, without proof, that they define a strong homotopy equivalence
- Our contribution:
  - We did a hand proof of the conjecture
  - We formalized it in ACL2, thus proving the reduction version of the Normalization Theorem

The main theorems proved

- THEOREM: `F-chain-morphism`
  $(m \in \mathbb{N}^+ \wedge c \in C_m(K)) \to d^N_m(f_m(c)) = f_{m-1}(d_m(c))$
- THEOREM: `G-chain-morphism`
  $(m \in \mathbb{N}^+ \wedge c \in C^N_m(K)) \to g_{m-1}(d^N_m(c)) = d_m(g_m(c))$
- THEOREM: `F-G-H-property-1`
  $(m \in \mathbb{N} \wedge c \in C^N_m(K)) \to f_m(g_m(c)) = c$
- THEOREM: `F-G-H-property-2`
  $(m \in \mathbb{N}^+ \wedge c \in C_m(K)) \to d_{m+1}(h_m(c)) + h_{m-1}(d_m(c)) = c - g_m(f_m(c))$
- THEOREM: `F-G-H-property-3`
  $(m \in \mathbb{N} \wedge c \in C_m(K)) \to f_{m+1}(h_m(c)) = 0$
- THEOREM: `F-G-H-property-4`
  $(m \in \mathbb{N} \wedge c \in C^N_m(K)) \to h_m(g_m(c)) = 0$
- THEOREM: `F-G-H-property-5`
  $(m \in \mathbb{N} \wedge c \in C_m(K)) \to h_{m+1}(h_m(c)) = 0$

Some comments on the proof of the Normalization Theorem

- The core of the proof is carried out in the polynomial framework, guided by our hand proof
- The expressions involved are highly combinatorial. For example, this is the polynomial for h_4:

  $\eta_0 - \eta_1 + \eta_1\eta_0\partial_1 - \eta_1\eta_0\partial_2 + \eta_1\eta_0\partial_3 - \eta_1\eta_0\partial_4 + \eta_2 + \eta_2\eta_0\partial_2 - \eta_2\eta_0\partial_3 + \eta_2\eta_0\partial_4 - \eta_2\eta_1\partial_2 + \eta_2\eta_1\partial_3 - \eta_2\eta_1\partial_4 - \eta_3 + \eta_3\eta_0\partial_3 - \eta_3\eta_0\partial_4 - \eta_3\eta_1\partial_3 + \eta_3\eta_1\partial_4 + \eta_3\eta_2\partial_3 - \eta_3\eta_2\partial_4 - \eta_3\eta_2\eta_0\partial_1\partial_3 + \eta_3\eta_2\eta_0\partial_1\partial_4 + \eta_4 + \eta_4\eta_0\partial_4 - \eta_4\eta_1\partial_4 + \eta_4\eta_2\partial_4 - \eta_4\eta_2\eta_0\partial_1\partial_4 - \eta_4\eta_3\partial_4 + \eta_4\eta_3\eta_0\partial_1\partial_4 - \eta_4\eta_3\eta_0\partial_2\partial_4 + \eta_4\eta_3\eta_1\partial_2\partial_4$
- But the style of the proofs is similar to the simple example presented previously.
- Properties are lifted from the polynomial framework to the standard framework.

Some comments on the proof of the Normalization Theorem

- Note: the polynomial framework is not expressive enough to state the theorem. For example:
  - The normalization function cannot be expressed as a polynomial
  - Some transformations have to be applied to obtain a reduction from a strong homotopy equivalence, not expressed as polynomials.
- Therefore, some additional proofs in the standard framework are needed.

Conclusions and further work

- We have presented an approach to proving Algebraic Topology theorems in a first-order setting
  - We use the ACL2 theorem prover, because our long term goal is to verify properties of a Common Lisp system
- Proof effort: 99 definitions, 565 lemmas, 158 hints
  - Part of the formalization is automatically generated as instances of other generic theories
- Our next step: Eilenberg-Zilber theorem, an important theorem in algebraic topology, about the homology of product spaces.
- Thank you!

Conclusions and further work

- We have presented an approach to proving Algebraic Topology theorems in a first-order setting
  - We use the ACL2 theorem prover, because our long term goal is to verify properties of a Common Lisp system
- Proof effort: 99 definitions, 565 lemmas, 158 hints
  - Part of the formalization is automatically generated as instances of other generic theories
- Our next step: Eilenberg-Zilber theorem, an important theorem in algebraic topology, about the homology of product spaces.
- Thank you!