Mathematics, Algorithms and Proofs 2006

Symbolic Computation of A_∞-structures

Ainhoa Berciano Alcaraz (UPV–EHU)*

In this poster, we show some problems related with the computation of A_∞-structures in Algebraic Topology.

1. Basic Notions of A_∞-structures

Introduction

- The framework of our work is the field of Effective Algebraic Topology.
- An important problem is to know perfectly the explicit morphisms of an A_∞-(co)algebra in general.
- And, of course, to obtain explicit formulas is better in order to codify a program because of the complicated expresions involved.
- To simplify the notation, we will speak about A_∞-structures, referring to A_∞-algebras or A_∞-coalgebras according to the context.
- In our examples, we will only describe the case of A_∞-coalgebras.
- For concrete computer work, we use the **Kenzo program**, a symbolic system specific for Effective Algebraic Topology.
- To implement these structures we have to add a module of 2500 lines to kenzo (**ARAIA** and **CRAIC**).

A_∞-structures

- An A_∞-*coalgebra* is a dg-module M, with a family of morphisms $\Delta_i : M \to M^{\otimes i}$ $(i \geq 1)$, of degree $i-2$, such that for all $i \geq 1$:

$$\sum_{n=1}^{i} \sum_{k=0}^{i-n} (-1)^{n+k+nk} (1^{\otimes i-n-k} \otimes \Delta_n \otimes 1^k) \Delta_{i-n+1} = 0.$$

- Using the tensor trick and the perturbation lemma, it is possible to deduce such a structure from a reduction from a given coalgebra over our dg-module M.

Symbolic Representation

- In the computational framework, we code an A_∞-coalgebra as a morphism of degree zero from M to the tensor algebra of M, TM.
- We "lost" the real degree of the morphisms, but it is an small abuse of notation, with the purpose of coding the program as simple as possible.

2. Problems

"Infinite" loops

- The first problem to solve is the translation of mathematical categories into computational classes.
  - It is well known that given two coalgebras, C and D, the tensor product of $C \otimes D$ is a coalgebra, where the coproduct, $\Delta : C \otimes D \to (C \otimes D)^{\otimes 2}$, is given by the formula $(1 \otimes T \otimes 1)(\Delta_C \otimes \Delta_D)$.
  - In our "computational world", there exists a class **coalgebra** and of course, we are interested in the above property. Because of the rule that the tensor product of two coalgebras is a coalgebra, when the program defines the morphism Δ, it applies this rule, and it realizes that $(C \otimes D)^{\otimes 2}$ has to be a coalgebra too. So, it tries to define first the last object as a coalgebra and in a recursive way it falls down in an infinite loop.

Coherence

- **Mathematically**, using an A_∞-structure is more complex than to do it with a (co)algebra.
  - In fact an A_∞-algebra is an algebra wich is non-necessarily associative, but where the associativity property yet satisfied *up to homotopy*.
- **Computationally**, we have to reverse this point of view and to consider the class of A_∞-algebras as a *superclass* of the class of algebras, that is as a class of *more general* ("weaker") objects.

3. Solutions: Inheritance and classes

Inheritance

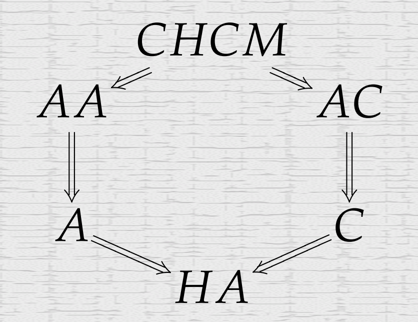
- About the "infinite" loops, we decided to use a *lazy* programming style: the **slot-unbound** Lisp generic function allows to implement a redundant slot *dynamically only when required*, so avoiding the mentioned infinite loop.

Extension of the class family

- It is clear that a dg-algebra induces a chain complex, so in the next diagrams, we use the notation:
  - CHCM=chain complex; A=dg-algebra; C=dg-coalgebra; HA=dg-Hopf algebra.
  - AA=A_∞-algebra; AC=A_∞-coalgebra.

Schedule

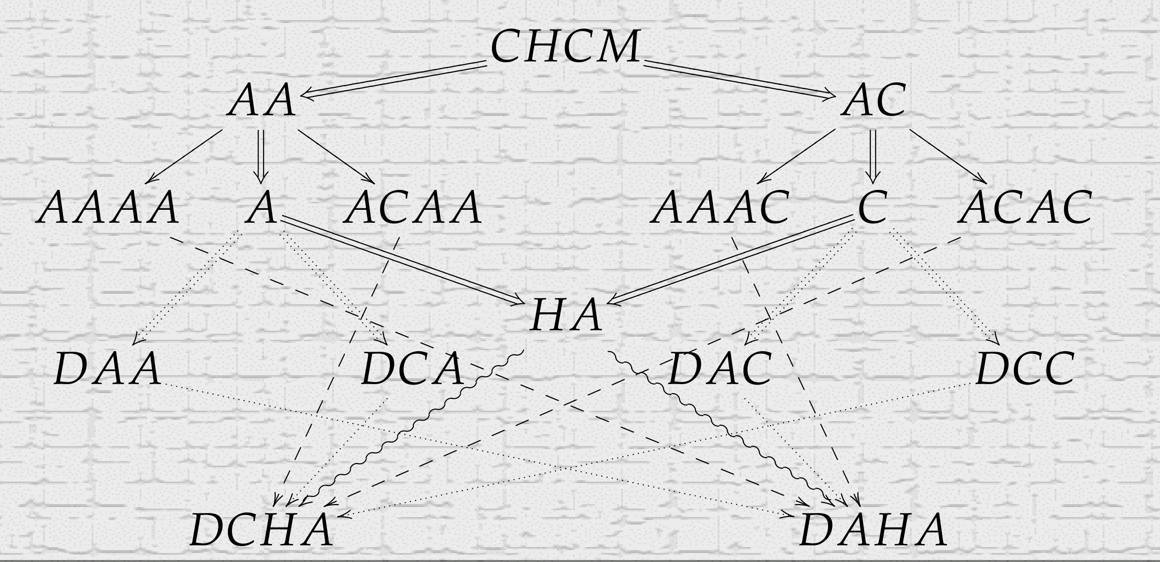
- Graphically, the structure is



4. Examples of computation

Computations of A_∞-structures via reductions

- If we want to obtain the A_∞-structure induced by a reduction, we need as an **input** a reduction, where the top chain complex is in fact a dg-(co)algebra. Applying **ARAIA** (resp. **CRAIC**), the output is the "same", but the bottom chain complex has now an explicit A_∞-structure. Because we would like to compare the A_∞-structure induced by a reduction with the trivial one in the case that the bottom chain complex would be an "structure", we have to modify the classes and to add new ones.
- Here, the notation added is:
  - AAAA=an objet with a doble A_∞-algebra; dually AAAC, ACAC and ACAA.
  - DAA=an algebra plus an A_∞-algebra; dually DCA, DAC, DCC, DCHA, DAHA.



References

[1] A. Berciano, P. Real, *The A_∞-coalgebra structure of the Zp-homology of Eilenberg-Mac Lane spaces*, Proceedings EACA 2004.

[2] P. Graham, Ansi Common Lisp, Prentice Hall, 1996.

[3] F. Sergeraert, *The computability problem in algebraic topology*. Adv. Math., **104**, n. 1, 1–29, 1994.

[4] R. Schön, *Effective Algebraic Topology*, Mem. A.M.S. **451**, 1991.

*Partially supported by project UPV-EHU00127.310-E-15916 and by the PAICYT research project FQM–296.