Conclusions an further work

Conclusions

This work should be understood as a first step towards an integral assistant for research and teaching in Algebraic Topology. From our point of view, the main contributions accomplished in this work, gathered by chapters, are the following ones.

- Chapter 2:
  - We have showed how some *guidance* can be achieved in the field of Computational Algebraic Topology, without using standard Artificial Intelligence techniques. The idea is to build an infrastructure, giving a *mediated access* to an already-written symbolic computation system. Putting together both Kenzo itself and this infrastructure, we have produced the Kenzo framework which is able to be connected to different clients (desktop GUIs, web applications and so on). In general, this can imply a restriction of the full capabilities of the kernel system, but the interaction with it is easier and enriched, contributing to the objective of increasing the number of users of the system. This work has been presented in [HPRS08, HPR08b].
  - We have included the following intelligent enhancements in the Kenzo framework: (1) controlling the input specifications on constructors, (2) avoiding some operations on objects which will raise errors, (3) chaining methods in order to provide the user with new tools, and (4) using expert knowledge to obtain some results not available in the Kenzo system. This work has been published in [HP10b].
  - As a by-product, we have developed OpenMath Content Dictionaries devoted to Simplicial Algebraic Topology. Namely, for each one of the mathematical structures of the Kenzo system, an OpenMath Content Dictionary has been defined. The definitions given in these Content Dictionaries include the axiomatic parts and have been used to interoperate with the ACL2 deduction system. This work can be found in [HPR09c].
  - We have developed a first version of a system with the same aim that the Kenzo framework, but which can process queries concurrently and where persistent results are available. This work has been presented in [HPR09b].

- Chapter 3:
  - We have implemented a plug-in framework which allows us to add new functionality to the Kenzo framework without modifying the original source code. This plug-in framework has allowed us to increase the capabilities of the Kenzo framework by means of new Kenzo functionalities or the connection with other systems such as Computer Algebra systems or Theorem Prover tools.
  - Since the final users of the Kenzo framework are Algebraic Topology students, teachers or researchers that usually do not have a background in Common Lisp, we have developed a user-friendly front-end allowing a topologist (student, teacher or researcher) to use the Kenzo program guiding his interaction by means of the Kenzo framework, without being disconcerted by the Lisp technicalities which are unavoidable when using the Kenzo system. This work has been presented in [HPR08a, HPR09a].
  - Several profiles of interaction with our system can be considered; therefore, we have used the plug-in framework to be able to customize our graphical user interface. In this way, the user can configure the application depending on his needs. This work can be found in [HPR09d].
  - We have gathered the two frameworks (Kenzo framework and plug-in framework) and the graphical user interface in a new system called *fKenzo*. This work has been published in [HPRS11].
- Chapter 4:
  - We have achieved the integration, in the same system, of computation (thanks to both Kenzo and GAP) and deduction (by means of ACL2). Even if our proposal has a limited extend, both thematically and from the point of view of the core systems, we think it shows a solid line of research that could be exported to other areas of mathematical knowledge management. This work has been presented in [HPRR10].
  - We have not only achieved the integration of several tools in the same system, but also the interoperability among them (a problem much more difficult). On the one hand, we have automated the composability between Kenzo and GAP which was presented in [RER09] hiding the technical details of the integration of these systems to a final user. On the other hand, Kenzo, GAP and ACL2 work together in our framework to provide a powerful and reliable tool related to the construction of Eilenberg MacLane spaces of type $K(G, n)$ where G is a cyclic group; those spaces are instrumental in the computation of homotopy groups. This work has been published in [HPRR10].
- Chapter 5:
  - We have presented some programs that improve the functionality of Kenzo, generating simplicial complexes from its facets and building the simplicial set

canonically associated with a simplicial complex. Moreover, we have certified the correctness of the programs in the ACL2 Theorem Prover, increasing in this way the reliability of this new Kenzo module. This work has been presented in [HP10a].

- We have developed a new Kenzo module devoted to study 2D and 3D monochromatic images based on the simplicial complex module. In addition, we have formalized our algorithms in ACL2; in this way, if we use our programs in real life problems (for instance, in the study of medical images), we are completely sure that the results produced by our programs are always correct.
- We have presented an algorithm to build the effective homology of the pushout of simplicial sets with effective homology. As a by-product the wedge and join of simplicial sets can be built as particular cases of the pushout. The algorithms are implemented as a new Kenzo module which in turn is tested with some interesting case studies such as the projective space $P^2(\mathbb{C})$ or $SL_2(\mathbb{Z})$. This work has been presented in [Her10].
- We have integrated all the modules presented in this chapter in *fKenzo*.

- Chapter 6:
  - We have designed a framework to prove the correctness of simplicial sets as implemented in the Kenzo system. As examples of application we have given a complete correctness proof of the implementation in Kenzo of spheres, standard simplicial sets and simplicial sets coming from simplicial complexes (modulo a safe translation of Kenzo programs to ACL2 syntax). By means of the same generic theory the correctness of other Kenzo simplicial sets can be proved. This work has been published in [HPRR11].
  - We have provided a methodology to model mathematical structures in the ACL2 Theorem Prover. This modeling process has been used to partly implement an ACL2 algebraic hierarchy which is the foundation for verifying the correctness of the construction of spaces from other ones, and can be useful for works non related to our development.
  - We have presented a framework to prove the correctness of construction of spaces from other ones implemented in the Kenzo system. As examples of application, we have given, among others, a complete correctness proof of the implementation in Kenzo of the direct sum of chain complexes, the Easy Perturbation Lemma and the SES_1 theorem. By means of the same framework the correctness of other constructions can be proved.

To sum up, we can claim that we have developed an assistant for Algebraic Topology called *fKenzo*. This system not only provides a friendly front-end for using the Kenzo kernel, but also guides the user in his interaction with the system. In addition, this system can evolve at the same time as Kenzo. We have tested this feature with the development of several new Kenzo modules which have been integrated in *fKenzo* without

any special hindrance. Moreover, the functionality of the system can also be extended by means of the integration with other systems such as Computer Algebra systems or Theorem Prover tools which not only work individually, but also cooperate to produce new tools. Eventually, we can claim that we have developed a reliable tool since we have partly verified the kernel of our system with the ACL2 Theorem Prover.

Open problems and further work

Several research lines are still open related to the work presented in this memoir, we gather them by chapters:

- Chapter 2:
  - The Homotopy Expert System (Paragraph 2.2.3.2.3) could be endowed with a higher level language to describe rules that could be made available for an algebraic topologist without any special computer science knowledge.
  - The algorithm implemented in the Homotopy Algorithm Module (Paragraph 2.2.3.2.2) can be completed thanks to the development presented in [RER09] which allows us to compute the effective homology of Eilenberg MacLane spaces of type $K(G, n)$ where G is a cyclic group. In this way, all the homotopy groups of 1-reduced simplicial sets with effective homology could be computed.
  - One of the most important issues to be tackled in the next versions of the Kenzo framework is how organizing the decision on when (and how) a calculation should be derived to a remote server. Some ideas about this problem were presented in Section 2.4.
  - The architecture presented in Section 2.5 will allow us, to move to a distributed computing context. We are thinking of a federated architecture, where several rich clients (each one with its own tuple space) communicate with a central powerful computing server. The central server acts as a general repository and also provides computing power to deal with difficult calculations. The most complicated problem is to devise good heuristics to decide what is the meaning of the fuzzy predicate "to be a difficult computation".
- Chapter 3:
  - One of the feasible enhancements for *fKenzo* could be to find a suitable way (free from the Common Lisp syntax) of editing and handling elements of each constructed space. Thus we could approach the difficult question of introducing in *fKenzo* computations as the homology groups of $P^2(\mathbb{C})$, see Subsubsection 5.3.4.

- In the area of user interfaces, a feasible objective could be be the development of a Web User Interface (WUI). At this moment the structure of a possible WUI is provided by means of the XUL specifications, the job now would consist of connecting the WUI with the framework to obtain a new usable front-end.
- Also in the area of user interfaces, we could integrate a "drag and drop" equation editor, like DragMath [Bil], in the *fKenzo* user interface. The editor would let users build up mathematical expressions in a traditional two dimensional way, and then output the expression in OpenMath format. The OpenMath expression will be used to invoke the Kenzo framework.

- Chapter 4:
  - More computational and deduction capabilities could be integrated in our framework. From the Computer Algebra side, we can consider, for instance, the Cocoa [CoC] system, where some works related to Kenzo has been already done, see [dC08]. From the Theorem Proving side, we can consider Coq and Isabelle, which have been already used to formalize Kenzo algorithms, see [ABR08, DR11]. Moreover, as in the case of GAP and ACL2, the real interest does not lie in using them individually, but making them work in a coordinate and collaborative way to obtain new tools and results not reachable if we use individually each system.
  - It would be necessary to improve the interaction with the ACL2 system. At this moment the queries must be pre-processed (even if parametrized specification allows the system to cover complete families of structures: cyclic groups, spheres, simplicial complexes and so on); a comfortable way of introducing questions about the truth of properties of intermediary objects, dynamically generated during a computing session, should be provided.
  - We want to integrate more ACL2 certification capabilities in our system. The idea is to interact in a same friendly front-end with Kenzo and ACL2. For instance, the user could give information to construct a new simplicial set with Kenzo; then, he could provide minimal clues to ACL2 explaining why his construction is sensible; then ACL2 would produce a complete proof of the correctness of the construction. This kind of interaction between Computer Algebra systems and Theorem Provers would be very valuable, but severe difficulties related to finding common representation models are yet to be overcome.
  - A meta-language should be designed to specify how and when a new kernel system can be plugged in the framework. This capability and the necessity of orchestrating the different services suppose a real challenge, which could be explored by means of OpenMath technologies.

- Chapter 5:

- A quite natural research line consists of developing new algorithms and programs. For instance, a challenging task is the implementation of the dual notion of pushout, called pullback [Mat76].
- In Subsubsection 5.1.5.2, we have presented two different programs which construct the simplicial complex associated with a list of simplexes. One of those programs was only implemented in Common Lisp since it uses a memoization strategy which cannot be directly implemented in ACL2. However, the work presented in [BH06] provides the mechanism to memoize functions in ACL2, then an optimized version of our algorithm could be implemented in ACL2. Moreover, we could use the semantic attachment procedure presented in Subsection 6.1.3 to use the already implemented not optimized version for reasoning and the new optimized one for executing.
- In the same way that we have applied the algorithms related to simplicial complexes in the study of digital images, we could develop new programs to be applied in different contexts such as coding theory or robotics. Likewise that in the case of digital images, if we want to apply our programs to real life problems, we must be completely sure that the results produced by our programs are correct. Therefore, the formal verification of our programs with a Theorem Prover would be significant.

- Chapter 6:
  - With the acquired experience, the methodology presented in Section 6.2 could be extrapolated to other algebraic Kenzo data structures. So, the work presented in that section can be considered a solid step towards our objective of verifying in ACL2 first order fragments of the Kenzo Computer Algebra system.
  - The hierarchy presented in Subsection 6.3.4 should be enriched by means of the rest of mathematical data structures implemented in Kenzo, see Subsection 1.2.1.
  - The methodology presented in Section 6.4 could be applied to several cases. In particular, it would be interesting the formalization of the construction of the effective homology of the pushout which involves several of the constructions already presented in this memoir.
  - The formalization of the computation of homology groups of spaces remains as further work. In this case we could focus on the computation of homology groups of spaces of finite type thanks to the Effective Homology method. Then, computing each homology group can be translated to a problem of diagonalizing certain integer matrices. Therefore, we should formalize this diagonalization process.

All these problems are interesting enough to be undertaken, however, we cannot tackle all of them at the same time. Therefore, our main priorities are:

- *Libraries of formalized Homological Algebra and Algebraic Topology.* Throughout this memoir we have presented the formalization of several results related to Homological Algebra and Algebraic Topology in the ACL2 theorem prover. We are planning the formalization of more results in the future. To undertake this goal, we foresee the use of not only the ACL2 theorem prover but also the proof assistant Coq [BGBP08] as well as its SSReflect extension [GM09] and the libraries it provides. A first step in this line is the work [HPDR11], where the formalization in Coq of incidence matrices associated with simplicial complexes as well as the main theorem that gives meaning to the definition of homology groups are presented.

- *Integration of different Theorem Prover tools.* As explained in the previous item, we are planning to work simultaneously with two different proof assistants: ACL2 and Coq. This goal can be split in two different subgoals. On the one hand, we want to compare the different formalizations of the same problem in ACL2 and in Coq/SSReflect; a related work to this topic can be found in [AD10]. On the other hand, we are interesting in using ACL2 as oracle of inductive schemas for Coq/SSReflect; obtaining an interoperability which is, in some sense, similar to the one presented for the Kenzo and GAP Computer Algebra systems.

- *Applications to medical imagery.* The methods presented in this memoir related to digital images can be applied in the study of medical images. However, to this aim it is necessary to implement new tools in software systems. In addition, if we want to apply our methods in the study of medical images we must be completely sure that they are correct, therefore, a formalization process is required, too.

The main reason for choosing these goals, and no others, lies in the fact that they are three of the tasks assigned to La Rioja node of the Formath project [For]. This project has partially supported the work presented in this memoir.