# Introduction

The interest in Mechanized Mathematics appeared at the beginning of Computer Science, or even before: see the work of Alan Turing [Tur36], one of the fathers of Computer Science. Leaving scientific applications of numerical analysis (which gave rise to the first computers) far behind, from the fifties Symbolic Computation started to be developed with the aim of reproducing the mathematicians way of working, with the help of computers. Symbolic Computation has two branches: Computer Algebra and Mechanized Reasoning based on computational logic. For the early history of Computer Algebra, see the reference [DST93]. Related to both Mechanized Reasoning and proof assistant tools we can quote the influential GPS (General Problem Solver) of Newell and Simon [NSS59], whose first version appeared around 1955, linked to the birth of Artificial Intelligence.

These two lines of symbolic manipulation (Computer Algebra and Mechanized Reasoning) underwent a parallel development in the next 40 years, without too much relation between them. However, the scene radically changed 15 years ago, when the Calculemus initiative [Cal], funded by European projects, set out the challenge of integrating both Computer Algebra (computation) and Mechanized Reasoning (deduction) systems. In the same line, the MAP (Mathematics, Algorithms and Proofs) network [MAP], founded in 2003, works in the foundations of that integration mainly based on *constructive mathematics*.

Moreover, at the same time, with the success of Internet technologies, several options appeared to make feasible (symbolic) mathematics on the Net. Perhaps, the most solid contribution was the one of the initiative IAMC (Internet Accessible Mathematical Computation) [IAM]. From this point on, several architectures (such as "buses" [CH97], [MATd], "brokers" [Mata], [Matb], generic servers [Matc], or based on web services [DSW05], [SSW04])) and design patterns [Dus06] have been proposed, culminating in both the MoNET (Mathematics on the Net) paradigm [Mon] and the mathematics on semantic web [CDDP04]. The specification of XML standards, namely MathML [A$^+$08] and OpenMath [Con04], has been instrumental in all the quoted contributions, since they provide communication protocols to transfer mathematical knowledge. A successful initiative, but independent of these standards, is SAGE [Ste]; an alternative to currently dominant paradigms.

From the confluence of these lines (the one related to the integration of computation and deduction, and the one focused on the mathematics on the Net), a community de-

voted to Mathematical Knowledge Management (MKM) [MKM] emerged in 2003. Its final goal is the development of integral assistants for mathematics including computation, deduction, internet access and powerful user interfaces able to make the daily work of mathematical researchers easier, as well as influencing the strategies in mathematical teaching. Of course, this is a quite ambitious goal which retakes some of the classic Artificial Intelligence topics.

After this general introduction about the scope of this thesis, let us present now our application context: Algebraic Topology. This mathematical subject studies topological spaces by means of algebraic means, in particular through algebraic invariants (groups or rings, usually). This allows one to study interesting properties about topological spaces by means of statements about groups which are often easier to prove.

However, in spite of being an abstract mathematical subject, Algebraic Topology methods can be implemented in software systems and then applied to different contexts such as coding theory [Woo89], robotics [Mac03] or digital image analysis [GDMRSP05, GDR05] (in this last case, in particular in the study of medical images [SGF03]).

We can say that the work presented in this memoir tries to particularize the MKM work to the Algebraic Topology scope.

In this work, the Kenzo program [DRSS98], a Computer Algebra system devoted to research in Algebraic Topology, will be instrumental. This system has been mainly developed by Francis Sergeraert and is implemented in Common Lisp [Gra96]. The Kenzo system implements the Effective Homology method appeared in the eighties trying to make available real algorithms for the computation of homology and homotopy groups (two of the most important invariants in Algebraic Topology). Introduced by Francis Sergeraert in [Ser87] and [Ser94], the present state of this technique is described in [RS97] and [RS06].

From the year 2000 the work of the Programming and Symbolic Computation Team of University of La Rioja, supervised by Julio Rubio, has been, in some way, parallel to the one outlined for the general discipline. After a first stage where efforts were devoted both to increase and improve the algorithms and programs of Kenzo, a new research line was launched, without leaving the previous one, dedicated to apply Formal Methods of Software Engineering to the Kenzo system. Related to this research, relevant results were achieved about algebraic specification of Kenzo and, in general, of object oriented software systems (see, for instance, [LPR03] and [DLR07]). Deepening in this line, the Isabelle/HOL proof assistant was used to verify in [ABR08] a very important algorithm in Homological Algebra: the Basic Perturbation Lemma. In the same line, we can find the work of [DR11] where the Effective Homology of the Bicomplexes (another important result in Homological Algebra) was formalized in Coq, or the proof of the Normalization Theorem [LMMRRR10] in the ACL2 Theorem Prover. Regarding Algebraic Topology on the Net, a first approach was published in [APRR05], where a (partial) remote access to the Kenzo system was achieved using CORBA [Gro].

In this memoir the three previous research lines (*development of algorithms*, *formal-*

*ization with proof assistant tools* and *implementation of user interfaces* for Algebraic Topology) converge. To be more concrete, the goal of this work has consisted in developing an integral assistant for research in Algebraic Topology. The "integral" adjective means that the assistant not only provides a graphical interface for using the Kenzo kernel, but also guides the user in his interaction with the system, and, as far as possible, produces certificates about the correctness of the computations performed. As a by-product, several subgoals have been achieved. First of all, we have improved the usability and the accessibility of Kenzo, trying to increase in this way the number of users who can take profit of Kenzo. Moreover, we have also enhanced the computation capabilities of the Kenzo system providing, on the one hand, new Kenzo modules and, on the other hand, a connection with the GAP Computer Algebra system [GAP]. In addition, the goal of integrating computation (Kenzo) and deduction (ACL2 [KM]) is reached in this work by means of the formalization of Kenzo programs in the ACL2 Theorem Prover.

The rest of this memoir is organized as follows. The first chapter includes some preliminary notions and results that will be used in the rest of the work. Basic notions about Homological Algebra, Simplicial Topology and Effective Homology are presented in the first section of this chapter. The second part introduces the Kenzo Computer Algebra system, as well as some of its more important features. Finally, the ACL2 Theorem Prover is briefly presented.

After this first chapter, the memoir is split into three different parts. Chapters 2 and 3 are devoted to improve the usability and the accessibility of the Kenzo system. The computational capability of Kenzo is increased in Chapters 4 and 5 by means of the connection with other systems and the development of new Kenzo modules verified in ACL2. Chapter 6 is fully devoted to prove the correctness of some Kenzo programs by means of ACL2.

Chapter 2 presents an environment, called Kenzo framework, which provides a mediated access to the Kenzo system, constraining the Kenzo functionality, but providing guidance to the user in his navigation on the system. The architecture, the components and the execution flow of the Kenzo framework are described in this chapter. Moreover, at the end of the chapter, two ongoing works related to increase the computation capabilities of the Kenzo framework by means of ideas about remote and distributed computations are presented.

Once we have presented the Kenzo framework, we want to be able to increase the capabilities of the system by means of new Kenzo functionalities or the connection with other systems such as Computer Algebra systems or Theorem Prover tools but without modifying the kernel source code. Moreover, we realize that the interface of the Kenzo framework is not desirable for a human user, since it is based on XML [B$^+$08]; then, a more suitable way of interacting with the Kenzo framework should be provided. Chapter 3 is devoted to present how these problems have been handled. The first part of this chapter is focused on introducing a plug-in framework which allows us to extend the Kenzo framework without modifying the source code. A customizable graphical user

interface which makes the interaction with the Kenzo framework easier is presented in the second part of this chapter. This graphical user interface improves the usability and the accessibility of the Kenzo system. The whole system, that is to say, the combination of the two frameworks and the front-end, is called *fKenzo*, an acronym for friendly Kenzo. The next two chapters focus on extending *fKenzo* by means of the connection with other systems and new Kenzo functionalities respectively.

Chapter 4 describes the integration of several systems in *fKenzo*. First of all, the Computer Algebra system GAP [GAP] is linked to *fKenzo*, allowing the computation of group homology. Moreover, the GAP system is not only used individually but also some of its programs are composed with the Kenzo system in order to obtain new tools, which increase the computational capabilities of *fKenzo*; this part of the memoir is an enhancement to a previous work [RER09]. The second part of this chapter is devoted to integrate the ACL2 Theorem Prover in *fKenzo*, achieving in this way the integration of computation and deduction in the same system. Likewise that in the GAP case, the ACL2 system is not only used individually but it is also combined with Kenzo to increase the reliability of the new programs of our system. From this chapter on, ACL2 will play a key role in our work.

Chapter 5 carries out a contribution to the algorithms and programs of the Kenzo system by means of new Kenzo modules which in turn extend the *fKenzo* system. First of all, a new Kenzo module in charge of working with simplicial complexes is presented. This module about simplicial complexes is the foundation to develop a setting to analyze monochromatic digital images. Namely, a new module which can be used to study digital images is explained in the second part of this chapter. Finally, the necessary algorithms and programs to build the effective homology of the pushout of simplicial sets are presented in the last section of the chapter. As an application of the pushout, we compute with Kenzo the first homotopy groups of the suspension of the classifying space of $SL_2(\mathbb{Z})$, the group of $2 \times 2$ matrices with determinant 1 over $\mathbb{Z}$, with the group operations of ordinary matrix multiplication and matrix inversion. Moreover, this chapter is not only devoted to present the new Kenzo modules but also to increase the reliability of the programs implemented in that modules by means of the ACL2 Theorem Prover.

Chapter 6 is devoted to prove the correctness of some programs implemented in the Kenzo system by means of ACL2. In the first part of this chapter, a set of ACL2 tools which will be fundamental onwards are presented. The second part is focussed on proving the correctness of the constant Kenzo simplicial set constructors. Subsequently, an infrastructure for modeling mathematical structures in ACL2 is introduced. Eventually, a methodology to verify the correctness of the construction of Kenzo spaces from other ones applying topological constructors is presented in the last section of this chapter.

The memoir ends with a chapter which includes conclusions and further work, a glossary and the bibliography.