

Four Approaches to Automated Reasoning with Differential Algebraic Structures*

Jesús Aransay¹, Clemens Ballarin², and Julio Rubio¹

¹ Dpto. de Matemáticas y Computación. Univ. de La Rioja. 26004 Logroño (Spain).
{jesus-maria.aransay,julio.rubio}@dmc.unirioja.es

² Institut für Informatik. Technische Univ. München. D-85748 Garching (Germany).
ballarin@in.tum.de

Abstract. While implementing a proof for the Basic Perturbation Lemma (a central result in Homological Algebra) in the theorem prover Isabelle one faces problems such as the implementation of algebraic structures, partial functions in a logic of total functions, or the level of abstraction in formal proofs. Different approaches aiming at solving these problems will be evaluated and classified according to features such as the degree of mechanization obtained or the direct correspondence to the mathematical proofs. From this study, an environment for further developments in Homological Algebra will be proposed.

1 Introduction

EAT [15] and Kenzo [5] are software systems written under Sergeraert's direction for symbolic computation in algebraic topology and homological algebra. These systems have been used to compute remarkable results (for instance, some homology groups of iterated loop spaces) previously unknown. Both of them are based on the intensive use of functional programming techniques, which enable in particular to encode and handle at runtime the infinite data structures appearing in algebraic topology algorithms. As pointed out in [4], algebraic topology is a field where challenging problems remain open for computer algebra systems and theorem provers.

In order to increase the reliability of the systems, a project to formally analyze fragments of the programs was undertaken. In the last years, several results have been found related to the algebraic specification of data structures, some of them presented in [9]. Following these results, the algorithms dealing with these data structures have to be studied. The long term goal of our research project is to get certified versions of these algorithms, which would ensure the correctness of the computer algebra systems to a degree much greater than current hand-coded programs. To this end, a tool for extracting code from mechanized proofs could be used. As a first step towards this general goal, our concrete objective in this paper is to explore several possibilities to implement proofs of theorems in algebraic topology by using a theorem prover. As theorem prover, the tactical

* Partially supported by MCyT, project TIC2002-01626 and by CAR ACPI-2002/06

prover Isabelle [12] has been chosen, mainly, because it is the system the authors are most familiar with.

A first algorithm for which we intend to implement a proof is the Basic Perturbation Lemma (hereinafter, BPL), since its proof has an algorithm associated used in Kenzo as one of the central parts of the program. In Section 2, the statement of the BPL, as well as a collection of lemmas leading to its proof, will be given. In Section 3, a naïve attempt to implement the proofs of these lemmas is introduced. Section 4 describes an approach using the existing tools in Isabelle. In Section 5, we try to avoid the problems arising from partiality and provide a generic algebraic structure embedding most of the objects appearing in the problem. In Section 6, an approach based on a new Isabelle feature, instantiation of locales, will be commented on. The paper ends with a conclusions section.

2 Some Homological Algebra

In the following definitions, some notions of homological algebra are briefly introduced (for further details, see [10], for instance).

Definition 1. A graded group C_* is a family of abelian groups indexed by the integers, $C_* = \{C_n\}_{n \in \mathbb{Z}}$, with each C_n an abelian group. A graded group homomorphism $f: A_* \rightarrow B_*$ of degree k ($\in \mathbb{Z}$) between two graded groups A_* and B_* is a family of group homomorphisms, $f = \{f_n\}_{n \in \mathbb{Z}}$, with $f_n: A_n \rightarrow B_{n+k}$ a group homomorphism $\forall n \in \mathbb{Z}$. A chain complex is a pair (C_*, d_{C_*}) , where C_* is a graded group, and d_{C_*} (the differential map) is a graded group homomorphism $d_{C_*}: C_* \rightarrow C_*$ of degree -1 such that $d_{C_*} d_{C_*} = 0_{\text{hom } D_* D_*}$. A chain complex homomorphism between two chain complexes (A_*, d_{A_*}) and (B_*, d_{B_*}) is a graded group homomorphism $f: A_* \rightarrow B_*$ (degree 0) such that $f d_{A_*} = d_{B_*} f$.

Let us note that the same family of homomorphisms $f = \{f_n\}_{n \in \mathbb{Z}}$ can be considered as a graded group homomorphism or a chain complex homomorphism. If no confusion arises, C_* will represent both a graded group and a chain complex; in the case of a chain complex homomorphism, the differential associated to C_* will be denoted by d_{C_*} .

Definition 2. A reduction $D_* \Rightarrow C_*$ between two chain complexes is a triple (f, g, h) where: (a) the components f and g are chain complex homomorphisms $f: D_* \rightarrow C_*$ and $g: C_* \rightarrow D_*$; (b) the component h is a homotopy operator on D_* , that is, a graded group homomorphism $h: D_* \rightarrow D_*$ of degree 1; (c) the following relations are satisfied: (1) $fg = \text{id}_{C_*}$; (2) $gf + d_{D_*} h + h d_{D_*} = \text{id}_{D_*}$; (3) $fh = 0_{\text{hom } D_* C_*}$; (4) $hg = 0_{\text{hom } C_* D_*}$; (5) $hh = 0_{\text{hom } D_* D_*}$.

Reductions are relevant since the homology of chain complexes is preserved by them and they allow to pass from a chain complex where the homology is unknown to a new one where homology is computable.

Definition 3. Let D_* be a chain complex; a perturbation of the differential d_{D_*} is a homomorphism of graded groups $\delta_{D_*}: D_* \rightarrow D_*$ (degree -1) such that $d_{D_*} +$

δ_{D_*} is a differential for the underlying graded group of D_* . A perturbation δ_{D_*} of d_{D_*} satisfies the nilpotency condition with respect to a reduction $(f, g, h): D_* \Rightarrow C_*$ whenever the composition $\delta_{D_*}h$ is pointwise nilpotent, that is, $(\delta_{D_*}h)^n(x) = 0$ for an $n \in \mathbb{N}$ depending on each x in D_* .

Under certain conditions, reductions can be “perturbed” to obtain new reductions easier to work with. This is expressed in the BPL.

Theorem 1. Basic Perturbation Lemma — *Let $(f, g, h): D_* \Rightarrow C_*$ be a chain complex reduction and $\delta_{D_*}: D_* \rightarrow D_*$ a perturbation of the differential d_{D_*} satisfying the nilpotency condition with respect to the reduction (f, g, h) . Then, a new reduction $(f', g', h'): D'_* \Rightarrow C'_*$ can be obtained where the underlying graded groups of D_* and D'_* (resp. C_* and C'_*) are the same, but the differentials are perturbed: $d_{D'_*} = d_{D_*} + \delta_{D_*}$, $d_{C'_*} = d_{C_*} + \delta_{C_*}$, and $\delta_{C_*} = f\phi\delta_{D_*}g$; $f' = f\phi$; $g' = (1 - h\phi\delta_{D_*})g$; $h' = h\phi$, where $\phi = \sum_{i=0}^{\infty} (-1)^i (\delta_{D_*}h)^i$.*

The BPL is a key result in algorithmic homological algebra (in particular, it is crucial for EAT [15] and Kenzo [5]). It ensures that when a perturbation is discarded (usually in chain complexes of infinite nature), a new reduction between chain complexes can be algorithmically obtained and thus the process to obtain a chain complex with computable homology can be implemented in a symbolic computation system. The BPL first appeared in [16] and was rewritten in modern terms in [3]. Since then, plenty of proofs have been described in the literature (see, for instance, [7,14]). We are interested in a proof due to Sergeraert [14]. This proof is divided into two parts:

Part 1. Let ψ be $\sum_{i=0}^{\infty} (-1)^i (h\delta_{D_*})^i$. From the BPL hypothesis, the following equalities are proved: $\psi h = h\phi$; $\delta_{D_*}\psi = \phi\delta_{D_*}$; $\psi = 1 - h\delta_{D_*}\psi = 1 - \psi h\delta_{D_*} = 1 - h\phi\delta_{D_*}$; $\phi = 1 - \delta_{D_*}h\phi = 1 - \phi\delta_{D_*}h = 1 - \delta_{D_*}\psi h$.

Part 2. With these equalities, it is possible to give a collection of lemmas providing the new reduction between the chain complexes (and therefore producing the algorithm associated to the BPL).

In the rest of the paper, we focus on the second part. The collection of lemmas will be now presented, and later the sketch of a proof, which combines these lemmas, will be given. The sketch of the proof shows the constructive nature of the proof, which would permit us to obtain an algorithm from it. In the following sections we will explain the different attempts we have studied to implement the proofs of these lemmas.

Lemma 1. *Let $(f, g, h): D_* \Rightarrow C_*$ be a chain complex reduction. There exists a canonical and explicit chain complex isomorphism between D_* and the direct sum $\ker(gf) \oplus C_*$. In particular, $F: \text{im}(gf) \rightarrow C_*$ and $F^{-1}: C_* \rightarrow \text{im}(gf)$, defined by $F(x) = f(x)$ and $F^{-1}(x) = g(x)$, are inverse isomorphisms of chain complexes and $\text{im}(gf) = \ker(\text{id}_{D_*} - gf)$.*

Let us denote by $\text{inc}_{\ker(p)}$ the canonical inclusion homomorphism $\text{inc}_{\ker(p)}: \ker(p) \rightarrow D_*$ given by $x \mapsto x$, with $p = d_{D_*}h + hd_{D_*}$. It is well defined since $\ker(p)$ is a chain subcomplex of D_* .

Lemma 2. *Let D_* be a chain complex, $h: D_* \rightarrow D_*$ (degree 1) a homomorphism of graded groups, satisfying $hh = 0$ and $hd_{D_*}h = h$. Let $p = d_{D_*}h + hd_{D_*}$ (from the reduction properties in Definition 2 follows that $id_{D_*} - p = gf$). Then $(id_{D_*} - p, inc_{\ker(p)}, h)$ is a reduction from D_* to $\ker(p)$.*

Lemma 2 is used to give a (very easy) constructive proof of the following result.

Lemma 3. *Assuming the conditions of the BPL, and the equalities of Part 1, there exists a canonical and explicit reduction $D'_* \Rightarrow \ker(p')$, where $p' = d_{D'_*}h' + h'd_{D'_*}$.*

Lemma 4. *Assuming the conditions of the BPL, and the equalities of Part 1, there exists a canonical and explicit isomorphism of graded groups between $\ker(p)$ and $\ker(p')$, where $p = d_{D_*}h + hd_{D_*}$ and $p' = d_{D'_*}h' + h'd_{D'_*}$.*

Lemma 5. *Let A_* be a chain complex, B_* a graded group and $F: A_* \rightarrow B_*$, $F^{-1}: B_* \rightarrow A_*$ inverse isomorphisms between graded groups. Then, the graded group homomorphism (degree -1) $d_{B_*} := Fd_{A_*}F^{-1}$ is a differential on B_* such that F and F^{-1} become inverse isomorphisms between chain complexes.*

Lemma 6. *Let $(f, g, h): A_* \Rightarrow B_*$ be a reduction and $F: B_* \rightarrow C_*$ a chain complex isomorphism. Then (Ff, gF^{-1}, h) is a reduction from A_* to C_* .*

Sketch of the BPL proof — By applying Lemma 3, a reduction $D'_* \Rightarrow \ker(p')$ is obtained. Then, by Lemma 4, a graded group isomorphism between $\ker(p')$ and $\ker(p)$ is built. Now, from Lemma 1, one can conclude that $\ker(p) = \ker(id_{D_*} - gf) = \text{im}(gf) \cong C_*$, and an explicit isomorphism of graded groups between $\ker(p')$ and C_* is defined (by composition). The differential of $\ker(p')$ has to be transferred to C_* by applying Lemma 5, giving a new chain complex C'_* , with the property that $\ker(p') \cong C'_*$ as chain complexes. By applying Lemma 6 to $D'_* \Rightarrow \ker(p')$ and $\ker(p') \cong C'_*$, an explicit reduction from D'_* to C'_* is obtained. When the homomorphisms obtained in the different lemmas are consecutively composed, the equalities in the BPL statement are exactly produced.

3 A symbolic approach

Our first approach consists in considering the objects in the statements (the homomorphisms) as generic elements of an algebraic structure where equational reasoning can be carried out. The idea is to identify, for instance, the elements of a ring of homomorphisms with the elements of a generic ring. Then, calculations in this ring are identified with proof steps in the reasoning domain (homomorphisms in the example). We call this a *symbolic approach* since homomorphisms are represented simply by symbols (as elements of a generic ring) without reference to their nature as functions. In our case, one of the additional difficulties of the proofs we intend to implement is that most of them require the properties of the various domains (with also elements of different nature and type) involved in the proof; but when trying to implement mathematics in computers, an abstraction process is always needed to produce the translation of elements of the

computer system employed into the elements of mathematics. This process is of great importance, since it may clarify what has been done and to which extent we can rely on the results obtained in the computer, and will be briefly described at the end of this section; for this symbolic approach, the abstraction process can lead to identify the different structures embedding homomorphisms with a simple structure embedding all of them; the idea will be again used in Section 5 and its importance will be observed also in Section 6.

The proof of Lemma 2 illustrates most of the problems that have to be solved: the implementation of complex algebraic structures and the implementation of homomorphisms. In addition, one must work with the homomorphisms in two different levels simultaneously: equationally, like elements of an algebraic structure, and also like functions over a concrete domain and codomain. These reasons made us choose this lemma to seek the more appropriate framework. Along this section this framework is precisely defined in the symbolic approach, then the proved lemmas are explained and finally some properties of this framework are enumerated³.

The following abstractions can be carried out:

- The big chain complex (D_*, d_{D_*}) is by definition a graded group with a differential operator, and $(\ker p, d_{D_*})$ is a chain subcomplex of it. The endomorphisms of (D_*, d_{D_*}) are the elements of a generic ring R .
- Some special homomorphisms between (D_*, d_{D_*}) and $(\ker p, d_{D_*})$ and endomorphisms of $(\ker p, d_{D_*})$ are seen as elements of R (for instance, the identity, the null homomorphism or some contractions).

Some of the computations developed under this construction of a generic ring can be then identified with some parts of the proof of Lemma 2. On the other hand, some other properties can not be proved in this framework since some (relevant) information about the structures involved and their properties is lost. This framework, being too generic, permits to avoid the problems of the concrete implementation of homomorphisms.

We will now give an example of how some of the properties having to be proved in the lemma can be represented in this framework. According to Def. 2 we have to prove the 5 characteristic properties of the reduction given in the conclusion of the lemma. From the five equalities, two fit in this framework and can be derived equationally inside of it. The first one is property (5), i.e.

$$hh = 0_R$$

The proof is trivial since the statement follows directly from premises of Lemma 2 and its proof can be implemented as in the mathematical proof. The second example of a proof that can be given inside this framework is property (3), i.e.

$$\text{if } hdh = h \text{ and } hh = 0 \text{ and } p = dh + hd \text{ then } (1_R - p)h = 0_R,$$

whose implementation in Isabelle can be given as

³ A similar pattern will be followed in the other approaches.

lemma (*in ring*) *property.three*: assumes $d \in \text{carrier } R$ and $h \in \text{carrier } R$
and $h * h = 0$ and $h * d * h = h$ and $p = d * h + h * d$
shows $(1 - p) * h = 0$
proof - from prems show ?thesis by algebra **qed**

Some comments on this proof: firstly, regarding the accuracy of the statement and, again, the abstraction process mentioned before, it should be pointed out that from the comparison of the proposed lemma in Isabelle and the property stated in Lemma 2 one difference is observed; namely, Lemma 2 says $(\text{id}_{D_*} - p)h = 0_{\text{hom } D_* \text{ ker } p}$ whereas the Isabelle proof corresponds exactly to $(\text{id}_{D_*} - p)h = 0_{\text{hom } D_* D_*}$ since there is only one ring involved in the Isabelle context. This is not a problem in this concrete equality since the value of $0_{\text{hom } D_* \text{ ker } p}$ and $0_{\text{hom } D_* D_*}$ is equal at every point (because $\text{ker } p$ is a graded subgroup of D_*); so far, the proof implemented in Isabelle, although having a formal difference with the original proof, can be considered to be an exact implementation of the mathematical proof. Nevertheless, this kind of situation is the one that will lead us to seek a more appropriate environment where the different domains and algebraic structures can be represented.

It is also worth to emphasize the tool support for this style of reasoning:

1. use of *locales* is of advantage since it clarifies the syntax, shortens the proof and creates contexts with local assumptions; in our case just by adding (*in ring*) in our lemma a specific context is built where R is a generic ring and all the theorems proved in the existing *ring* locale appear like facts.
2. the *algebra* tactic valid for rings automates proofs looking for a normal form of the given expression (0, in this case).

This approach has some advantages. First of all, it is quite simple and intuitive, which has the consequence that proofs are quite close to the mathematical proofs obtained and can be easily understood. As will be seen in Section 4, when more elaborate approaches are discussed, it is also possible that the size of the proofs turns them into something unfeasible. Moreover, Isabelle has among its standard libraries enough theories to produce proofs in the context of the symbolic approach in a completely automatic way; these generic proofs can be used where only equational reasoning is required. In addition to this, the basic idea of this approach will be useful in Sections 5 and 6.

There are also some drawbacks of this method. Firstly, we cannot prove the other properties needed to complete the implementation the proof of the lemma. They can not be proved, since information about the domain of the homomorphisms or about the concrete definition of the homomorphisms in such domains is required. For instance, it is not possible to derive with the tools of this framework that $p|_{\text{ker } p} x = 0$. A second disadvantage observed is the high level of abstraction required to pass from the mathematical context to the given context in Isabelle. The type assigned to homomorphisms in this framework, where they are considered ring elements, is just a generic type α , whereas something more similar to the mathematical definition of homomorphism would be at least desirable (for instance, $\alpha \rightarrow \beta$). In particular, neither the differential group (D_*, d_{D_*}) nor the

elements of the domain D_* appear. Therefore, the conceptual distance between Isabelle code and the mathematical content is too large. From a more practical point of view, the previous discussion on types shows that the computational content of homomorphisms (that is to say, their interpretation as functions) is lost, which prevents code extraction.

4 A set theoretic approach

The previous approach offers a high level of abstraction but is insufficient to prove our theorems. In this section we present a framework where the algebraic structures are again represented by records, but homomorphisms are now represented as functions.

There are two main components in this framework. One is the algebraic structures involved and the other one is the homomorphisms between these structures. In Isabelle, declarations of constants (including algebraic structures or homomorphisms) consist of a type⁴ and a defining axiom. Algebraic structures are implemented over extensible record types (see [11]); this permits record subtyping and parametric polymorphism, and thus algebraic structures can be derived using inheritance. As an example, a differential group can be defined in Isabelle as

$$\begin{aligned} \mathbf{record} \ \alpha \ \mathit{diff_group} &= \alpha \ \mathit{monoid} + \\ &\ \mathit{diff} :: \alpha \Rightarrow \alpha \\ \mathit{diff_group} \ C &\equiv \mathit{ab_group} \ C \wedge \mathit{diff} \in \mathit{hom} \ C \ C \wedge \\ &\ \forall x \in \mathit{carrier} \ C. \ \mathit{diff} \ \mathit{diff} \ x = \mathit{one} \ C \end{aligned}$$

For homomorphisms, again a type and a definition must be declared. In the Isabelle type system all functions are total, and therefore homomorphisms (which are partial functions over generic types) are implemented through total functions; the homomorphisms between two given algebraic structures will be the set of all total functions between two types, for which the homomorphism axioms hold⁵

$$\begin{aligned} \mathit{hom} &:: [(\alpha, \theta) \ \mathit{monoid_scheme}, (\beta, \gamma) \ \mathit{monoid_scheme}] \Rightarrow (\alpha \Rightarrow \beta) \ \mathit{set} \\ \mathit{hom} \ A \ B &\equiv \{f. f \in \mathit{carrier} \ A \rightarrow \mathit{carrier} \ B \wedge f(x *_A y) = (fx) *_B (fy)\} \end{aligned}$$

Since in Isabelle equality ($=$) is total on types, problems arise when comparing homomorphisms (or partial functions, in general). A way of getting out of this situation is to use “arbitrary”, which denotes (for every type) an arbitrary but unknown value. Doing so is sound, because types are inhabited. Partial functions can be simulated by assigning them to “arbitrary” outside of their domain.

With these definitions we have implemented the complete proof of Lemma 1. This proof takes advantage of the above mentioned inheritance among algebraic structures, starting from sets, with the definition of a bijection between sets, and then introducing the inherited structures step by step. Following this scheme,

⁴ In Isabelle syntax, $f :: \alpha \Rightarrow \beta$ denotes a total function from type α to type β .

⁵ In Isabelle syntax, $\{x.f(x)\}$ denotes set comprehension.

800 lines of Isabelle code were needed to both specify the required structures and implement the proofs, and a readable proof was obtained.

We would like to point out the following about this approach:

1. A higher degree of accuracy (in comparison to the previous approach) has been obtained; now the relationship between the objects implemented in the proof assistant and the mathematical concepts present in the proof is much clearer.
2. The representation of algebraic structures in Isabelle and Kenzo is performed in the same way, using records where each field represents an operator. On the other hand, homomorphisms have a simpler representation in Isabelle since they are just elements of a functional type whereas in Kenzo they are also records, allowing that way to keep explicit information over their domain and codomain.
3. Converting homomorphisms into total functions by the use of “arbitrary” makes things a bit more complex. From the formal point of view, it is difficult to identify a homomorphism containing an arbitrary element, i.e., a conditional statement “if $x \in G$ then fx else arbitrary”, with any mathematical object; there is a gap in the abstraction process that cannot be directly filled. A solution to clarify this process by identifying the elements outside the domain with a special element of the homomorphism’s codomain will be proposed in Section 5; with this idea are avoided the non defined values of the total function representing a concrete homomorphism. This solution has been proposed before and its disadvantages are well known, but making a careful use of it can make mechanization easier.
4. Homomorphisms are represented by conditional statements, and working with n homomorphisms at the same time one has to consider 2^n cases. There are two solutions to this problem. The first one consists in enhancing the homomorphisms with an algebraic structure allowing to reason with them like elements of this structure in an equational way (for instance, an abelian group or a ring). With this implementation, proofs can sometimes avoid to get into the concrete details of the representation of homomorphisms in the theorem prover. To some extent this can be understood as the development of a new level of abstraction; some steps (or computations) are developed at the level of the elements of the homomorphisms domains whereas some are developed at the level of the structure where homomorphisms are embedded. This forces to implement proof steps more carefully and also to make a correct choice of the homomorphisms that can be provided with an algebraic structure, and will be discussed in Section 5. A second possible solution not implemented yet using equivalence classes will be mentioned as future work in Section 7.

5 A homomorphism approach

If we capitalize the advantages of both the symbolic approach and the set theoretic approach and we combine them carefully, it is possible to obtain a new

framework where the level of abstraction is as suitable as in the set theoretic approach and the degree of mechanization is as high as in the symbolic approach. The idea is to develop a structure where the domain of the homomorphisms and also the homomorphisms can be found at the same time, with homomorphisms represented by elements of functional type (or something similar), which would allow working with them at a concrete level, but with the same homomorphisms being part of an algebraic structure. Since various algebraic structures are involved in the problem (at least (D_*, d_{D_*}) and $(\ker p, d_{D_*})$), there is not a simple algebraic structure containing all the homomorphisms appearing in the problem. Clearly endomorphisms of (D_*, d_{D_*}) form a ring, and also the ones in $(\ker p, d_{D_*})$, as well as the homomorphisms from (D_*, d_{D_*}) into $(\ker p, d_{D_*})$ form an abelian group; the automation of proofs in such a complicated environment would be hard; some ideas will be proposed in Section 7.

Another possible solution is explained in this section. It involves considering just one simple algebraic structure where all the homomorphisms can be embedded, and then develop tools allowing to convert a homomorphism from this structure into one of the underlying ones. Taking into account that $\ker p$ is a substructure of D_* , we will consider as our only structure the ring of endomorphisms $R = \text{hom}(D_*, d_{D_*})(D_*, d_{D_*})$. Later we will introduce the tools allowing to convert homomorphisms from R into homomorphisms, for instance, of $\text{hom}(\ker p, d_{D_*})(\ker p, d_{D_*})$, but just to illustrate the benefits of this approach we give a brief example here:

Example 1. Proving the fact “assumes $d \in \text{hom}(D_*, d_{D_*})(D_*, d_{D_*})$ and $h \in \text{hom}(D_*, d_{D_*})(D_*, d_{D_*})$ shows $p = dh + hd \in \text{hom}(D_*, d_{D_*})(D_*, d_{D_*})$ ”, due to partiality matters, requires several reasoning steps. When providing homomorphisms with an algebraic structure this is a trivial proof, since rings are closed under their operations. \square

In order to embed homomorphisms into a ring, it is necessary to choose carefully the elements and also the operators. Firstly, there can be only one representant for each homomorphism, and here partiality appears again, since otherwise both $\lambda x.(\text{one } G)$ and $\lambda x.(\text{if } x \in G \text{ then one } G \text{ else } \textit{arbitrary})$ could be identities in this ring. Secondly, operators must be closed over the structure, and thus they rely strongly on the chosen representation for homomorphisms. Without going in further depth, we decided to consider the carrier of the ring R formed by the completions $\lambda x.(\text{if } x \in G \text{ then } fx \text{ else one } G)$, because then the generic composition operator \circ can be used in Isabelle (whereas this was not possible with the *extensional functions* based on “arbitrary” in Isabelle). At a second stage, we had to implement the tools allowing to convert an element of $\text{hom}(D_*, d_{D_*})(D_*, d_{D_*})$ into one of, for instance, $\text{hom}(\ker p, d_{D_*})(\ker p, d_{D_*})$ (under precise conditions). This would allow to implement proofs for facts such as $\langle \text{id } -p, D, \ker p \rangle \circ \langle h, D, D \rangle = \langle 0, D, \ker p \rangle$ (property (3) in Def. 2, needed for Lemma 2) in a human readable style such as⁶:

⁶ The \dots are just an abbreviation meaning “the previous expression” in sequential calculations.

Example 2. assumes $\langle p, D, D \rangle \circ \langle h, D, D \rangle = \langle h, D, D \rangle$

shows $\langle \text{id} - p, D, \ker p \rangle \circ \langle h, D, D \rangle = \langle 0, D, \ker p \rangle$

proof -

have $\langle \text{id} - p, D, D \rangle \circ \langle h, D, D \rangle = (\langle \text{id} D, D \rangle - \langle p, D, D \rangle) \circ \langle h, D, D \rangle$

(...)

also have $\dots = \langle 0, D, D \rangle$

finally have one: $\langle \text{id} - p, D, D \rangle \circ \langle h, D, D \rangle = \langle 0, D, D \rangle$

from prems have two: $\text{im}(\text{id} - p) \subseteq \ker p$

from one and two

have $(\langle \text{id} - p, D, D \rangle \circ \langle h, D, D \rangle = \langle 0, D, D \rangle) \equiv (\langle \text{id} - p, D, \ker p \rangle \circ \langle h, D, D \rangle = \langle 0, D, \ker p \rangle)$

*then show $\langle \text{id} - p, D, \ker p \rangle \circ \langle h, D, D \rangle = \langle 0, D, \ker p \rangle$ **qed*** □

In order to obtain this degree of expressiveness, two major modifications must be made. The first one is related to the implementation of homomorphisms. The information about their domain and codomain must be explicit, otherwise it is not possible to convert them from $\langle f, D, D \rangle$ to $\langle f, D, \ker p \rangle$. Secondly, some lemmas allowing to modify these triples must be introduced (and proved) in Isabelle. These lemmas permit to change the domain and codomain of homomorphism provided that certain conditions are satisfied; when the composition of two homomorphisms is equal to a third one, the domain and codomain of the three homomorphisms can be changed and the equality holds. This collection of lemmas can be summarized in the following one, a generic version of all of them where all the algebraic structures (domains and codomains of the homomorphisms) can be changed:

Lemma 7. Laureano's Lemma- *Let $\langle g, C, D \rangle$ and $\langle f, A, B \rangle$ be two homomorphisms between chain complexes satisfying $\langle g, C, D \rangle \circ \langle f, A, B \rangle = \langle h, A, D \rangle$ and let A' be a chain subcomplex from A , B' a chain subcomplex from C' , $\text{im } f$ contained on B' , and $\text{im } h$ contained on D' . Then $\langle g, C', D' \rangle \circ \langle f, A', B' \rangle = \langle h, A', D' \rangle$.*

With this lemma and the new proposed representation for homomorphisms a framework with the following advantages is built. From the point of view of capability, it is possible to implement all the properties needed for Lemma 2. It should be also emphasized that the size of the proofs is manageable and sometimes the proofs require some human guidance in order to finish. Moreover, the framework can be transferred to other problems dealing with homomorphisms and particular reasoning about them. Embedding homomorphisms (even between different algebraic structures) in only one algebraic structure can help to easily implement proofs of properties about these homomorphisms (and avoids the implementation of more elaborated algebraic structures); moreover, Lemma 7 can be easily generalized for compositions of n homomorphisms. From the formal point of view, all the computations are carried out in this algebraic structure and all the operations needed can be identified as simplifications inside the ring, or applications of Lemma 7; the abstraction process from the implemented proof to the mathematical proof can be accurately defined.

On the other hand, the amount of concepts that need to be mechanized is rather large. In addition to this, we have observed that the implementation of

homomorphisms as records sometimes slows down the interactive proof steps. A likely cause for this loss of performance are inefficiencies in the record package of Isabelle2003⁷. A record encoding a homomorphism is a rather large data structure and contains a lot of redundant information. For instance, in $f: D_* \rightarrow D_*$, the corresponding record contains four copies of D_* , and D_* itself is already a quite large record. It appears that when several homomorphism are involved in a proof, the redundancies cause a slowed response time of the interactive environment.

6 Instantiating locales

A final method based on the instantiation of locales (see [1]), a tool recently implemented in Isabelle that could be of great help for this kind of problems, is also considered. Locales, which are light-weight modules, define local scopes for Isabelle; some fixed assumptions are made and theorems can be proved from these assumptions in the context of the locale.

With the instantiation of locales it is not possible to define generic frameworks, but the method has a great expressiveness for concrete problems. The approach presented in this section is based to some extent on the ideas introduced in the first approach, considering homomorphisms embedded in an algebraic structure, but with the clear advantage that now several algebraic structures can be defined at the same time (this will allow us to introduce rings R and R' , abelian groups A and A' , and so on) and the elements of these structures can be instantiated with their real components. For instance, R could be instantiated with $\text{hom}(D_*, d_{D_*})(D_*, d_{D_*})$ as carrier set and the usual composition \circ as product, R' with carrier $\text{hom}(\ker p, d_{D_*})(\ker p, d_{D_*})$ and operation \circ as product, A' with carrier $\text{hom}(D_*, d_{D_*})(\ker p, d_{D_*})$ and so on. This permits to work with homomorphisms at two different levels. In a first one, they can be considered as elements of the algebraic structures and the computations between the elements of these algebraic structures are identified with the steps of the proofs. In the second level, the operations of the algebraic structures can be instantiated to their definition (for instance, “mult = \circ ” or “sum = $\lambda fg.\lambda x.fx *_D gx$ ”) as well as the structures (in our case, the differential groups or chain complexes), in order to complete the proof steps requiring this information (for instance, $p|_{\ker p} x = 0$). The structure of the locale needed to implement the proofs of Lemma 2 in Isabelle would now look as follows:

```
locale hom_completion_environment = comm_group G + comm_group K +
ring R + ring R0 + comm_group A + comm_group A0 + var p +
assumes R = (carrier = hom_completion G G, mult = op  $\circ$ , one = ...)
and R' = ...
defines K = ker p
```

⁷ These have been removed in the latest release, Isabelle2004, but we have not yet ported our proofs.

First the structures (the variables) needed for the local scope are fixed and then their components are defined. Once all these assumptions have been made, the statement of lemmas will have the following appearance⁸:

lemma (*in hom_completion_environment*) *reduction_property_one*:
assumes $p \in \text{carrier } R$
show $(\text{one } R \ominus_3 p) \circ (\lambda x. \text{if } x \in \text{carrier } \ker p \text{ then id } x \text{ else one } \ker p) = \text{one } R'$

By specifying (*in hom_completion_environment*) we introduce all the facts present in the locale *hom_completion_environment* and then they can be used as theorems in our lemma. The previous statement corresponds to property (2) of Def. 2 specialized for Lemma 2. Using these tools, we get a framework very close to the one in the mathematical lemmas. Moreover, proofs can be mechanized in a readable style quite similar to the mathematical proofs, and only some repetitive steps have to be added to explicitly obtain the value of the fields of the fixed algebraic structures when needed; proofs are easy to read and similar to the ones made “by hand”. All these features make this framework quite satisfactory. On the other hand, from the formal point of view there is also a drawback. The operation \circ appearing in the statement of the lemma can not be identified with any of the operations of the framework. It composes elements of different algebraic structures such as $\text{hom}(\ker p, d_{D_*}) (\ker p, d_{D_*})$, $\text{hom}(D_*, d_{D_*}) (\ker p, d_{D_*})$ but whose relation has not been made explicit at any point inside the locale. The composition \circ is valid in this case since all the elements of the carrier sets are implemented through functions, and therefore they can be composed. Even in other cases, the operation relating the different structures could be implemented, but there is no mathematical correspondence for this external \circ ; this produces a gap in the abstraction function that permits to identify mathematical objects with their representation in the theorem prover.

7 Conclusions and Further Work

A design “from bottom to top” has been used to implement the proof of the BPL in Isabelle. Instead of considering algebraic topology as a whole and formalizing generic concepts of this theory, our approach started from a very concrete (and relevant) lemma (the BPL) in this area that could allow to estimate the real capabilities from Isabelle to work with differential structures and their homomorphisms. We also started from a simple attempt (in Section 3) that gave us helpful ideas for the following approaches about what tools were directly available in Isabelle and those other tools that should be supplied to the system. The Isabelle tools introduced in Section 4 were not enough to produce readable proofs. Then, in Section 5, an original framework where several computations with homomorphisms can be carried out was suggested. The implementation of these tools led us to the problem of what changes must be made into partial

⁸ In Isabelle syntax, \ominus_3 is a reference to the “minus” operation in the third structure of our locale definition, ring R .

functions in order to provide them with an enveloping algebraic structure. Our choice was to introduce completions, identifying the elements outside the domain of the partial function with a distinguished element, the identity of the image structures. Similarly, the *extensional* functions already available in Isabelle could have been used, just with some modifications in the composition of functions. The result is that a generic algebraic structure can not be directly defined from the homomorphisms since there are multiple functions representing each of them. In particular, it was shown that inside this framework implementations could be given for all the properties in Lemma 2, and that from the formal point of view, there was a clear abstraction function between the objects implemented in Isabelle and the mathematical objects required. Finally, in Section 6, the possibilities of a recent Isabelle feature (instantiation of locales) were studied; this new tool fits with several problems and so far would allow to implement all the proofs of the lemmas in Section 2. We have implemented a proof of all the properties in Lemma 2. From this study, we conclude that the approach based on the instantiation of locales is the most promising for further developments in homological algebra.

Some problems remain open, and more work has yet to be done:

1. To complete the proofs of all the lemmas in Section 2 and then give a complete implementation of the proof of the BPL. The proof obtained should satisfy all the requirements needed to extract a program from it. At this point, we could explore the possibilities of extracting code in Isabelle (see, for instance, [2]). In addition to this, a comparison will be needed with other provers where code extraction has been used in non trivial cases (see, for instance, the work on Buchberger's algorithm in Coq [17]).
2. As it has been explained in Section 4, partial functions can be implemented by using equivalence classes. Actually, the solutions proposed here (using completion functions or the *extensional* functions implemented in Isabelle) are just different ways of giving elements to represent these (non-implemented) equivalence classes. The implementation would force to redefine all the concepts regarding homomorphisms (composition, identities, . . .) but would produce a definitive way to deal with partial functions in the Isabelle/HOL system. Following the instructions given in [13], the definition should be quite feasible.
3. A ringoid (see [10]) is an algebraic structure containing homomorphisms and endomorphisms over different algebraic structures. To give an implementation of it could be useful for several problems in group theory. An attempt to introduce category theory in Isabelle has been already made (see [6]), and some similarities can be found between the implementation of morphisms given there and the representation of homomorphisms that we proposed in Section 5, needed to work in a more effective way with them. Specification of ringoids is not a complicated task in Isabelle. Nevertheless, the benefits and drawbacks of this new approach with respect to our third and fourth approaches should be carefully studied.

4. In order to develop algebraic topology, one of the basic concepts needed (see definitions in Section 2) is infinite complexes (of modules, for instance). Some attempts have been made to produce an implementation of them (see [8]), but definitive results have not been obtained yet. For their implementation, lazy lists or extensible records might be used, but these ideas will be studied in future work.

Acknowledgements. We are grateful to Wolfgang Windsteiger and the anonymous referees for the useful comments on the previous version of this paper.

References

1. C. Ballarin, *Locales and Locale Expressions in Isabelle/Isar*. In Stefano Berardi et al., TYPES 2003, LNCS vol. 3085, pp. 34 - 50.
2. S. Berghofer, *Program Extraction in Simply-Typed Higher Order Logic*, TYPES 2002, LNCS vol. 2646, pp. 21-38
3. R. Brown, *The twisted Eilenberg-Zilber theorem*, *Celebrazioni Arch. Secolo XX*, *Simp. Top.* (1967), pp. 34-37.
4. J. Calmet, *Some Grand Mathematical Challenges in Mechanized Mathematics*, In T. Hardin and Renaud Rioboo, editors, *Calculemus 2003*, pp. 137 - 141.
5. X. Dousson, F. Sergeraert and Y. Siret, *The Kenzo program*, <http://www-fourier.ujf-grenoble.fr/~sergerar/Kenzo/>
6. J. Glimming, *Logic and Automation for Algebra of Programming*, Master Thesis, Maths Institute, University of Oxford, August 2001, available at <http://www.nada.kth.se/~glimming/publications.shtml>.
7. V. K. A. M. Gugenheim, *On the chain complex of a fibration*, *Illinois Journal of Mathematics* 16 (1972), pp. 398-414.
8. H. Kobayashi, H. Suzuki and H. Murao, *Rings and Modules in Isabelle/HOL*, In T. Hardin and Renaud Rioboo, editors, *Calculemus 2003*, pp. 124 - 129.
9. L. Lambán, V. Pascual and J. Rubio, *An object-oriented interpretation of the EAT system*, *Appl. Algebra Eng. Commun. Comput.* vol. 14(3) pp. 187-215.
10. S. Mac Lane, *Homology*, Springer, 1994.
11. W. Naraschewski and M. Wenzel, *Object-Oriented Verification based on Record Subtyping in Higher-Order Logic*, In J. Grundy and M. Newey, editors, *TPHOLs'98*, LNCS vol. 1479, pp. 349-366.
12. T. Nipkow, L. C. Paulson and M. Wenzel, *Isabelle/HOL: A proof assistant for higher order logic*, LNCS vol. 2283, 2002.
13. L. Paulson, *Defining Functions on Equivalence Classes*, Report available at <http://www.cl.cam.ac.uk/users/lcp/papers/Reports/equivclasses.pdf>
14. J. Rubio and F. Sergeraert, *Constructive Algebraic Topology*, Lecture Notes Summer School in Fundamental Algebraic Topology, Institut Fourier, 1997.
15. J. Rubio, F. Sergeraert and Y. Siret, *EAT: Symbolic Software for Effective Homology Computation*, Institut Fourier, Grenoble, 1997.
16. W. Shih, *Homologie des espaces fibrés*, *Publications Mathématiques de l'I.H.E.S.* 13, 1962.
17. L. Théry, *Proving and computing: A certified version of the Buchberger's algorithm*, in *Proceeding of the 15th International Conference on Automated Deduction*, LNAI vol. 1421, 1998.