# Some Mathematical Case Studies in ProofPower-HOL

R.D. Arthan

Lemma 1 Ltd.
2nd Floor, 31A Chain Street,
Reading UK RG1 2HX
rda@lemma-one.com

**Abstract.** This paper gives an overview of three case studies in developing pure mathematical theory using ProofPower-HOL. The case studies, which currently cover a selection of basic material from the theories of real analysis, group theory and topology, expose some interesting issues for formalising mathematics.

## 1 Introduction

Apart from basic mathematical structures such as sets, functions, lists and numbers, applying an automated theorem-proving system to hardware and software engineering problems tends to involve mathematical theories of a rather different nature from the traditional subject matter of pure mathematics. Many researchers feel that engineering applications are the most important for automated theorem-proving. However, it is natural to try to formalise pure mathematical theories. Research into this goes back to the earliest days of electronic computation.

In a recent survey, Carlos Simpson [12] has identified numerous reasons why computer-assisted formalised mathematics should be of benefit to the mathematical community. Simpson gives many references to earlier work in this area as does John Harrison in his thesis [7] and his paper [6]. The Flyspeck project [5] is applying computer-assisted theorem proving to increase confidence in Thomas Hales' proof of the Kepler sphere-packing conjecture, a difficult proof involving a considerable element of computation which has caused problems for the traditional peer review process.

In 2001, the opportunity arose to develop, for use in program verification, a theory of real arithmetic for the ProofPower specification and proof system. It was a natural experiment to use this as the basis of a theory of real analysis and I spent some time late in 2001 working on that. In 2003, since the Jordan curve theorem[1] was felt to be a good challenge problem in some automated theorem-proving circles, I used ProofPower-HOL to prove what Henle [8] calls the

[1] Apparently, much progress has been made on the Jordan curve theorem using the Mizar system, but I not able to assess whether the proof of the general case in two dimensions is complete (see http://mizar.uwb.edu.pl/).

fundamental lemma in one of the classical proofs of this results. In retrospect, I view this as a highly instructive mistake: Henle's book is a very accessible account of elementary algebraic topology for beginning students. His fundamental lemma is essentially a calculation of the mod 2 homology groups of the plane. I formulated it as a combinatorial result about discrete gratings and proved it, the proof being fairly easy.

Unfortunately, connecting the fundamental lemma expressed as a combinatorial fact with the geometry involves several topological results, most notably what Henle calls Alexander's lemma. I quickly realised that I was going to have to cover quite a bit of geometry and topology to prove them. Now Henle's proofs are very carefully designed for the beginner; they appeal to geometric intuitions as much as to formal reasoning. Henle sets up topology as the topology of subsets of the plane and to follow his proofs as they stand would involve doing special cases of general results whose proofs are no harder formally than the special cases.

**Moral 1:** if you ask someone "have you proved the XYZ theorem?" and receive the reply that they have proved the "fundamental lemma" or the "main result" or similar, it is wise to scrutinise their formal account closely to find out what they have actually proved[2].

**Moral 2:** theorem-provers don't need spoon-feeding; it makes sense to prove things at the "right" level of generality and that will often be more general than in an account intended for beginners.

**Moral 3:** while there is no royal road to proving theorems, there are shortcuts; however, you have to choose your shortcuts very carefully to make sure you don't get lost.

I subsequently began some case studies in pure mathematics, trying to cover the material along the lines that it might be covered in a typical undergraduate or beginning graduate course. Carried far enough, this programme would have the Jordan curve theorem drop out as the two dimensional case of the Jordan-Brouwer separation theorem proved via the calculation of the homology groups of spheres, but that is a long way off. To date this work has covered the following topics.

- A more complete treatment of real analysis including the definitions and basic properties of the exponential and trigonometric functions and of $\pi$.
- Some group theory including the definitions and elementary properties up to the three isomorphism theorems and the Cayley representation theorem
- Enough abstract and metric space topology to define the notions of homotopy and the fundamental groupoid (and to prove that it is a groupoid).

[2] As a simpler example, I have still not seen a proof of the mutilated chess-board theorem as a theorem about dominoes and chess-board as geometrical objects, which is what they surely are. Just as in my problem with the Jordan curve theorem, the combinatorics is fairly easy, but the geometric realisation requires more work.

My objectives in this enterprise were somewhat vague: essentially, I just wanted to see how this material turns out and to compare notes[3] with other systems (Mizar, PVS, HOL Light, etc.). I was also specifically interested in developing the theory to the points where the main mathematical subjects of algebra, analysis, geometry and topology begin to interact and inform one another, e.g., in algebraic topology and differential geometry. During the course of the work, some definite themes have emerged:

– I have tried to provide natural and readable specifications of the mathematical concepts formalised. For example, I use differential equations rather than power series as the definitions of the trigonometric functions, since I consider that approach to have a more intuitive, geometrical appeal.
– I have tried to follow the development of pure mathematics both in fitting abstract notions to more concrete ones after the event and in using abstract notions that have not yet been formalised to inform more concrete work. For example, you don't need to develop abstract group theory to define the real numbers and show that they are a group under addition. You can even use group-theoretical thinking while you're developing the theory. However, once you have some abstract group theory, you should be able to apply that to the real numbers and other specific constructions you may make with them (e.g., real vector spaces[4]).
– I have tried to develop each theory at the "right" level of generality or abstraction: this often involves a compromise between making the task at hand feasible and making the results general enough to be useful. On the other hand, being more abstract is sometimes both more powerful in applications and easier! E.g., the fundamental groupoid of a topological space is technically often easier to work with than the fundamental group.

There is no new mathematics of any significance in any of this: just as there is no significant new mathematics in an undergraduate textbook. However, interesting details arise *en passant* and you do learn something as you go (for example, that integration is not needed to develop enough of the the theory of power series to introduce the exponential and transcendental functions).

This paper gives an overview of what has been done at the time of writing (May–July 2004) and discusses some of the issues for formalising mathematics that have been highlighted. Full details are given in the papers [1–3]. The structure of the sequel is as follows: section 2 introduces the ProofPower-HOL logic and system by means of a simple example which also illustrates, in microcosm, some of the formalisation issues encountered (a theory listing for this example is given as an appendix); section 3 gives an outline on what has been done in the three case studies; section 4 discusses how the approach of the case studies

[3] The present paper is not intended as a detailed presentation of comparisons between different systems, but I will air my personal viewss on some points of principle.

[4] Roger Jones and I have an embryonic theory of normed real vector spaces based on the group theory case study and including a coordinate-free definition of the Frechet derivative.

might scale to more complex problem domains; section 5 gives some concluding remarks.

## 2 An Example

ProofPower is a system supporting specification and proof in HOL and Z. It is founded on an LCF-style implementation in Standard ML[5] of the same polymorphic simple type theory as the other systems in the HOL family. ProofPower-HOL supports a syntax for specification adapted from the Z notation [13] intended to encourage well-documented formal specifications using familiar logical and mathematical notations. This document is written in that syntax, Its source form is a mixture of LaTeX and input for the ProofPower-HOL parser. The input for the parser is displayed using a special font and a (mostly) single-character mark-up for the mathematical symbols, so that, for example, when you see '∀' in this paper, what I typed and then saw on my screen was an upside-down 'A' too.

To illustrate the style of specification adopted in the case studies, let us develop a simple algebraic theory. If $G$ is a group, a *$G$-action* on a set $X$ is a correspondence between elements of $G$ and mappings of $X$ to itself such that multiplication in the group corresponds to composition of the mappings and the unit of the group corresponds to the identity mapping. A set $X$ equipped with a $G$-action is called a $G$-set. $G$-sets arise, for example, by considering groups of symmetries of geometrical objects. To give a concrete example of a group action before defining the concept of a group, let us consider the particular case when $G$ is $\mathbb{Z}$, the group of integers under addition.

So a $\mathbb{Z}$-set will comprise a pair comprising a set (called the *carrier set* of the $\mathbb{Z}$-set) together with an assignment to each integer of a function from the set to itself. To represent this abstract concept in HOL, let us consider the polymorphic class of all pairs comprising a set of elements of some type $'a$ together with a function mapping integers to total functions from $'a$ to itself. We can capture this in the following type abbreviation[6]

SML

```
declare_type_abbrev("ℤ_SET", ["'a"], ⌜:'a SET × (ℤ → 'a → 'a)⌝);
```

The type $'a$ here is a polymorphic type parameter. It can be instantiated to any type we please, for example, an element of the type[7] $\mathbb{R}\ \mathbb{Z}\_SET$ is the type that includes all $\mathbb{Z}$-actions on sets of real numbers. We will think of the above

[5] ML stands for "metalanguage". Standard ML is a functional programming language which serves as both the implementation language and the interactive command language for ProofPower.

[6] Here the "specification" comprises an ML command to achieve the desired effect, since the ProofPower-HOL parser does not provide a concrete syntax for this form of definition.

[7] HOL type constructors are generally postfix operators, for example '$\mathbb{Z}$ LIST' denotes the type of lists of integers.

type as providing a *signature* for a class of *structures* which are candidates to be $\mathbb{Z}$-sets. If $X$ is such a structure (i.e., a member of an instance of the above type), we will write $Car\ X$ for the carrier set and $(x\ **\ i)X$ for the action of an integer $i$ on an element $x$. Note that the action operation is ternary not binary: in informal mathematics, it is normal to let the reader infer from the context which mathematical structures are being deployed, but formally we must be explicit about this.

To achieve the above syntax, we first declare the string '$**$' to act as an infix symbol with the same numerical precedence (310) as arithmetic exponentiation.

SML

```
declare_infix(310, "**");
```

We now give a constant specification to introduce the new constants '$Car$' and '$**$'. A constant specification in ProofPower-HOL comprises two parts: the part above the line gives type ascriptions for the new constant or constants and the part below the line gives a predicate which is to be their defining property. In this case the defining property comprises two universally quantified equations defining the values of applications of the functions '$Car$' and '$**$' Parsing the constant specification maps onto a call of the primitive definitional principle *const_spec*. This principle requires an existence proof for the constants being introduced. The ProofPower-HOL infrastructure includes a range of procedures for discharging the existence proofs and these will automatically discharge the proof obligations for all of the definitions in this example.

HOL Constant

$$\boldsymbol{Car} : \ 'a\ \mathbb{Z}\_SET \to 'a\ SET;$$
$$\mathbf{\$**} : \ 'a \to \mathbb{Z} \to 'a\ \mathbb{Z}\_SET \to 'a$$

---

$$\forall\ (set,\ action)\bullet$$
$$\quad Car\ (set,\ action) = set$$
$$\wedge\ (\forall x\ i\bullet\ (x\ **\ i)\ (set,\ action) = action\ i\ x)$$

The above definition serves to provide a convenient syntax for the operations on the structures of interest. We can see this in the following definition which captures the laws that a candidate $\mathbb{Z}$-set must satisfy to be worthy of the name. The laws specify that: *(i)*, the carrier set is closed under the $\mathbb{Z}$-action; *(ii)*, addition of integers corresponds to composition of the corresponding actions; and, *(iii)*, 0 corresponds to the identity function.

HOL Constant

$$\mathbb{Z}\_\boldsymbol{Set} : 'a\ \mathbb{Z}\_SET\ SET$$

---

$$\forall\ X\bullet$$
$$\quad X \in\ \mathbb{Z}\_Set$$
$$\Leftrightarrow (\forall x\ i\bullet\ x \in\ Car\ X \Rightarrow\ (x\ **\ i)\ X \in\ Car\ X)$$

$\mid \wedge\ (\forall x\ i\ j\bullet\ x \in Car\ X \Rightarrow (x ** (i + j))\ X = ((x ** i)\ X ** j)\ X)$
$\mid \wedge\ (\forall x\bullet\ x \in Car\ X \Rightarrow\ (x ** \mathbb{NZ}\ 0)\ X = x)$

In addition to the specifications, the source of this document also contains the statements and proofs of a small selection of theorems about $\mathbb{Z}$-sets. ML proof scripts are not particularly informative even to the expert eye, except when they are brought alive by replaying them interactively, so they have been suppressed from the printed form of this document. There is a listing of the theory in the appendix. The reader is invited to refer to the appendix for the statements of the following two theorems which are both elementary consequences of the above definition. The first theorem says that acting by $i$ and then by $-i$ results in the identity function on the carrier set and the second gives a cancellation law.

*ℤ_set_minus_thm* *ℤ_set_cancel_thm*

We complete the example by defining the *orbit* of an element $x$ of a $\mathbb{Z}$-set $X$. The orbit comprises the set of all elements $y$ that can be reached from $x$ under the $\mathbb{Z}$-action.

HOL Constant

$\mid$ ***Orbit*** $: 'a\ \mathbb{Z}\_SET \to 'a \to 'a\ SET$

$\mid \forall\ X\ x\bullet\ Orbit\ X\ x =\ \{y \mid \exists i\bullet\ y = (x ** i)\ X\}$

The reader may again consult the appendix for the statements of the following two theorems. The first says that any element of a $\mathbb{Z}$-set belongs to its own orbit and the second says that any two orbits are either equal or disjoint. In other words, the orbits are the equivalence classes of an equivalence relation: "co-orbital".

*orbit_refl_thm* *orbit_disjoint_thm*

In this example, we have formalised a very elementary mathematical theory and developed some very elementary theorems about it. The proofs would serve almost as they stand to prove the same facts about $G$-sets for arbitrary groups $G$ given the theory of groups developed in [2]. This could then provide the basis of some much more interesting mathematics. For present purposes, the example serves to illustrate ProofPower in action and to introduce the style of presentation of analysis, topology and group theory in [1–3].

## 3 The Case Studies

### 3.1 Basic Analysis

The case study on analysis is presented in [1]. It builds on the ProofPower-HOL theory that introduces the real numbers as a complete ordered field and covers the following ground.

– polynomial functions on the real numbers
– limits of sequences of real numbers
– continuity of functions
– differentiation
– limits of function values
– uniform convergence of limits of functions
– series and power series
– special functions: exponential function, natural logarithm, sine and cosine.

Broadly similar subject matter has been formalised before in HOL Light by John Harrison [7] and by Hanne Gottliebsen [4] in PVS. There are also developments of analysis in Mizar and Coq and several other systems. In addition to proofs of theorems, the ProofPower treatment includes automated proof procedures for continuity-checking and calculating derivatives (as do the treatments of Harrison and Gottliebsen).

While I make no claim for novelty in the material covered, I would claim that the specifications are readable and natural and that the material that is covered is done comprehensively. For example, here are the definitions of the sin and cos functions and of Archimedes' constant, $\pi$, defined here as the positive generator of the additive group of roots of the sin function.

$$\boldsymbol{Sin\ Cos} : \mathbb{R} \to \mathbb{R}$$

$$\begin{aligned}
& \mathit{Sin}(\mathbb{NR}\ \mathit{0}) = \mathbb{NR}\ \mathit{0} \land \mathit{Cos}(\mathbb{NR}\ \mathit{0}) = \mathbb{NR}\ \mathit{1} \\
\land\ & (\forall x \bullet\ (\mathit{Sin}\ \mathit{Deriv}\ \mathit{Cos}\ x)\ x) \land (\forall x \bullet\ (\mathit{Cos}\ \mathit{Deriv}\ \sim(\mathit{Sin}\ x))\ x)
\end{aligned}$$

$$\boldsymbol{ArchimedesConstant} : \mathbb{R}$$

$$\begin{aligned}
& \mathbb{NR}\ \mathit{0} < \mathit{ArchimedesConstant} \\
\land\ & \mathit{Sin}(\mathit{ArchimedesConstant}) = \mathbb{NR}\ \mathit{0} \\
\land\ & (\forall x \bullet\ \mathit{Sin}\ x = \mathbb{NR}\ \mathit{0} \\
& \quad \Rightarrow (\exists m \bullet\ x = \mathbb{NR}\ m * \mathit{ArchimedesConstant}) \\
& \quad \lor\ (\exists m \bullet\ x = \sim(\mathbb{NR}\ m * \mathit{ArchimedesConstant})))
\end{aligned}$$

*declare_alias*("$\pi$", ⌜*ArchimedesConstant*⌝);

Here the notation $(f\ \mathit{Deriv}\ c)\ x$ means that function $f$ has derivative $c$ at $x$ and the function $\mathbb{NR}$ is the injection of the natural numbers into the reals. The alias declaration introduces the traditional name "$\pi$" as an alternative to "*ArchimedesConstant*".

The specifications of the trigonometric functions and of $\pi$ clearly require non-trivial consistency proofs. This involves a development of the theory of power series, including the general result on differentiating power series term-by-term (which avoids the need for introducing integration at this stage). The elementary properties of the exponential, logarithmic and trigonometric functions and $\pi$ are then developed "axiomatically" from the differential equations.

As observed in [7], several notions of limit arise and it is desirable to have common ways of dealing with them. Harrison's approach is via the general notion of convergence nets. I use the more homely device of reducing the notions in question to sequential convergence. For example, it is an easy consequence of the standard definition of continuity that a function $f$ is continuous at $x$ iff. $f$ maps any sequence converging to $x$ to a sequence converging to $f(x)$. Using this fact, statements about continuity reduce to statements about sequential convergence, and, by and large, this turns the $\forall\exists\forall$ quantifier structure of the usual $\epsilon$-$\delta$ arguments into simple universally quantified statements about sequential convergence. Proponents of non-standard analysis both in education and in theorem-proving sometimes advocate the simple quantifier structure of the definition of continuity in non-standard analysis as an advantage. Using sequential convergence achieves much the same effect in standard analysis. The text books tend not to stress this method of working if they mention it at all, probably because it fails to generalise to arbitrary topological spaces.

**Moral 4:** when you are using a theorem-prover, you do not need to adopt methods for their pedagogical value: unlike a student, the prover cannot develop bad habits, so you can freely use any method that works.

### 3.2 Group Theory

The case study on analysis deals with a single specific HOL type: the type $\mathbb{R}$ of real numbers. The case study [2] on group theory puts the polymorphism in HOL to work along much the same lines as the $\mathbb{Z}$-set example presented in section 2 above.

The case study begins with a treatment of equivalence relations, equivalence classes and the construction of quotient sets along the lines proposed by Larry Paulson [10]. This material comprises a lemma library which provides templates for working with equivalence relations, in particular, for defining functions on quotient sets. This supports the proof of the first isomorphism theorem in group theory, which is all about defining homomorphisms on quotient groups. It would serve a similar purpose in any of the common algebraic concrete categories (rings, modules over a ring, vector spaces over a field etc.) and in dealing with quotient spaces in topology.

The group theory itself begins with a definition of the signature of a group along similar lines to the signature for $\mathbb{Z}$-sets in the example above. The polymorphic notion of a group is then defined to be the set of all structures with this signature that satisfy the group laws.

Substructures and quotient structures in algebra are very important, so it is vital to deal smoothly with subgroups and quotient groups. Taken verbatim, the traditional explication of these concepts in set theory leads to significant notational and semantic difficulties. The problem is this: in doing the general theory, an expression like $x.y$ denoting the product of two elements of a group

$G$ actually contains three variables: the group elements '$x$', '$y$', and the multiplication operator '.'. Syntactic tricks allow one to preserve something like the traditional infix notation for such expressions. But there is a semantic problem when one needs to deal with subgroups: according to the traditional account, the '.', in $x.y$ will denote a different set-theoretic function in a subgroup $H$ from what it does in the containing group $G$. Coercing operations from subgroup to containing group or from one subgroup to another becomes an excessive burden.

My solution to this problem is to formulate all definitions relative to some carrier set of interest in such a way that the behaviour of operators or properties outside the carrier set is irrelevant. I advocate this approach in general for dealing with algebraic structures. The apparent extra complication actually achieves an economy, because when one is working with substructures, the operators and properties can all be those of the containing structure: you have no need to restrict them to the substructures or to worry about coercing the operations of one substructure into the operations of another.

As an example, I take the operations on a group $G$ to be total functions on the universe of the type of its elements whose behaviour outside the carrier set of $G$ is immaterial. The operations on a subgroup $H$ of $G$ must be represented by the same total functions. This involves no loss of generality and removes a much complexity in specifications and proofs. It may be objected that this approach gives the wrong notion of equality for groups (since the same group can be represented using two different ways of totalising the operations). However, in normal algebraic practice, one almost never needs to assert equality (as opposed to isomorphism) between two groups that are not known to be subgroups of some other group, and in that case equality has the usual meaning.

Using this approach, the three isomorphism theorems and the Cayley representation theorem are very easy to prove once one has derived the usual laws of equational reasoning in a group from the defining properties (and developed proof procedures to automate the application of these laws). Once the formalisation details were settled, it was routine and quick to prove these results.

In fact, I feel that the treatment in this case study demonstrates that polymorphic simple type theory is actually more natural than set theory for carrying out much of mathematics. For example, one can give the following very convenient definition of the symmetric group on a set $X$ (i.e., the group comprising all permutations of the set).

$\boldsymbol{SymGroup} : {}'a\ SET \to ({}'a \to {}'a)\ GROUP$

---

$\forall\ X \bullet\ SymGroup\ X = ($
$\quad \{f \mid OneOne\ f \land Onto\ f \land \forall y \bullet\ \neg y \in X \Rightarrow f\ y = y\}$, $(*\ Carrier\ \ set\ *)$
$\quad (\lambda f\ \ g \bullet \lambda x \bullet f(g\ \ x))$, $(*\ multiplication\ *)$
$\quad (\lambda x \bullet\ x)$, $(*\ unit\ \ \ element\ *)$
$\quad Inverse\ (*\ inverse\ *)$
$)$

Here the quadruple giving the structure has components as indicated by the comments and *Inverse* is the function that maps a 1-1 onto function to its inverse function. This definition has numerous advantages over the untyped set-theoretic version. In particular, if $X$ is a subset of $Y$, then the symmetric group on $X$ is a subgroup of the symmetric group of $Y$ as it stands, whereas this is only true "up to an isomorphism" in the standard set-theoretic account. Moreover, we can think of *SymGroup* $\{x \mid x = x\}$ as denoting the group of all permutations of the universe, sitting naturally inside the monoid of all self-mappings of the universe. This works very pleasantly: as the Cayley representation theorem states, any group is isomorphic to a group of permutations and so composition of 1-1 onto functions provides a universal prototype for the multiplication in a group, a fact which cannot even be stated properly in first-order set theory.

**Moral 5:** *Pace* Quine [11, article on "Mathematosis"], in a typed theory it is counter-productive to define the concept of a group so that the carrier set can be recovered from the set that represents the multiplication.

### 3.3 Topology

The case study in topology is perhaps the most advanced of the three in educational terms, but it still really only prvoides the beginnings of the subjects it deals with. The subjects covered are:

- abstract topology: topologies; construction of new topologies from old as (binary) product spaces or subspaces; continuity, Hausdorff spaces; connectedness; compactness.
- metric spaces: the definitions of metrics and product metrics and the result that product metrics induce product topologies; existence of Lebesgue numbers for open coverings of compact metric spaces.
- topology of the line and the plane: characterisation of connected subspaces of the line; continuity of addition and multiplication as functions on the plane.
- elementary homotopy theory: definitions of path-connectedness, the homotopy relation and the fundamental groupoid; proof that the homotopy relation is an equivalence relation and that the fundamental groupoid is a groupoid[8]

The definition of a topology is the usual one: a topology is a family of sets (referred to as *open sets*) that is closed under arbitrary unions and binary intersections.

[8] In fact, at the time of writing, all the theorems needed to justify the construction of the fundamental groupoid as a quotient of the path space have been proved, but these have not yet been brought in line with the theory of equivalence relations in [2].

$$
\begin{array}{|l}
\textbf{\textit{Topology}} : {}'a\ SET\ SET\ SET \\
\hline
\quad Topology = \\
\{\tau \mid (\forall V \bullet\ V \subseteq \tau \Rightarrow \bigcup V \in \tau) \land (\forall A\ B \bullet A \in \tau \land B \in \tau \Rightarrow A \cap B \in \tau)\}
\end{array}
$$

Since the carrier set of a topological space can readily be recovered as the union of all its open sets, the complications with signatures that arise in algebra do not arise. The signature has but one component and the thinking we decried for algebraic structures in Moral 3 now turns out to be very convenient.

The central notion of continuity takes the following form (defining an operator *Continuous* which is written postfix). Here $\sigma$ and $\tau$ are intended to be topologies (and will be in the statements of all theorems that use this definition). As in the group theory case study, we work throughout with ordinary HOL total functions, taking care to make the definitions of concepts such as continuity ignore the behaviour of the functions outside some carrier set of interest, in this case the *Space* of the topologies, defined as the union of their open sets as discussed above.

$$
\begin{array}{|l}
\mathbf{\$}\textbf{\textit{Continuous}} : ({}'a\ SET\ SET \times {}'b\ SET\ SET) \to ({}'a \to {}'b)\ SET \\
\hline
\forall \sigma\ \tau \bullet \quad (\sigma, \tau)\ Continuous = \\
\quad \{f \\
\quad \mid \quad (\forall x \bullet\ x \in Space\ \sigma \Rightarrow f\ x \in Space\ \tau) \\
\quad \land \quad (\forall A \bullet\ A \in \tau \Rightarrow \{x \mid x \in Space\ \sigma \land f\ x \in A\} \in \sigma)\}
\end{array}
$$

Again, as in the group theory, this approach has the merit of localising complexity in the definitions which would otherwise spread to other definitions and to the statements and proofs of theorems. If you try to mimic the representation of functions in set theory, functions have constantly to be restricted to subspaces, whereas this is unnecessary with the total function approach.

Space does not allow an extended discussion of the methods of proof in this case study. However, there is one open problem that is worth mentioning. There is a constant need in topological reasoning to prove that functions are continuous. In algebraic topology, functions are often constructed by patching together functions defined on subspaces of the domain. For example, in proving that addition of paths in the fundamental groupoid is associative, the following result is needed, where $Open_R$ denotes the usual topology on the real line.

$$
\begin{array}{|lll}
\forall k \bullet & (\forall t \bullet & k\ t = \\
& & if\ t \leq 1/4\ then\ \mathbb{NR}\ 2{*}t \\
& & else\ if\ t \leq 1/2\ then\ t + 1/4 \\
& & else\ (1/2){*}t + 1/2) \\
& \Rightarrow & k \in (Open_R,\ Open_R)\ Continuous
\end{array}
$$

The proofs of such facts are very mechanical and are reminiscent of what the automated proof procedures for continuity of algebraic combinations of continuous functions in the analysis case study do. However, there are two slight complications: *(i)*, you need to apply a simple "patching" lemma to justify the continuity of a function defined by cases and *(ii)*, in the general case some small amount of intelligence is needed to pick the right topologies on intermediate sets. For example, to show that a composite $f \circ g$ is continuous with respect to a topology $\sigma$ on the domain of $g$ and a topology $\tau$ on the range of $f$, you need to pick some topology on the range of $g$ that makes both $f$ and $g$ continuous. An algorithm to automate these proofs would be a great boon, but I do not yet have one. Joe Hurd's work on predicate subtyping [9] looks like a promising source of ideas.

**Moral 6:** There are a lot of new and challenging problems for proof automation in pure mathematics.

## 4 Will it scale?

An important question to ask of any case study in applying formal methods and theorem-proving in engineering applications is "will the proposed technique scale to real-life applications?". I believe the same applies to mathematical applications as well. Simpson [12] identifies what is probably one of the most important problems for more advanced pure mathematics: much use is made of structure which share a combination of algebraic, topological or geometrical properties. For example, the rich and important theory of Lie groups is an abstraction of the algebraic and geometric theory of groups of real or complex matrices. A Lie group is simultaneously a group and a smooth manifold, a smooth manifold being something that has a particular topological structure combined with a differential structure allowing analytic methods to be used. The issue then is, how to deal formally with the kind of reasoning that is endemic in mathematics where one just says something like "let G be a Lie group" and then freely appeals to the notations and theory of whichever of the underlying structures provides the facts one needs.

I believe the approach to modelling mathematical structures exemplified by the case study on groups and also by the $\mathbb{Z}$-set example in section 2 above will scale, subject to some slight modifications to the details, ideally supported by some extensions to the syntax offered by the parser (see [2] for more details on the latter).

The main change to the approach addresses the issue highlighted by Simpson in his example of Lie groups. To get things to scale, I would propose using labelled products rather than unlabelled products for the signatures of algebraic structures. To see how this would work, consider the notion of a field: Given our treatment of groups, a field can conveniently be thought of as two group structures on elements of the same type obeying certain laws[9].

Using labelled products, the signature for groups would be given by the following ProofPower-HOL labelled product type definition which defines a new polymorphic labelled product type *'a GROUP* with four components with the indicated labels and types. The component labels become the names of the functions that project the product type onto its component types.

HOL Labelled Product

$GROUP$

| | |
|---|---|
| $Car_G$ | $: 'a\ SET;$ |
| $Times_G$ | $: 'a \to 'a \to 'a;$ |
| $Unit_G$ | $: 'a;$ |
| $Inverse_G$ | $: 'a \to 'a$ |

Now we can define the signature for a field as a labelled product. Note that in both these labelled product definitions, in the interests of scalability to complex situations, we are decorating the component labels with subscripts to avoid clashes with other algebraic structures, e.g., rings would also have an additive group.

HOL Labelled Product

$FIELD$

| | |
|---|---|
| $AdditiveGroup_F$ | $: 'a\ GROUP;$ |
| $MultiplicativeGroup_F$ | $: 'a\ GROUP$ |

This captures the desired semantics, but creates some syntactic problems. For example, the expression 1 + x.y in a field $K$ would have to be written.

$$\begin{array}{l} Times_G\ (AdditiveGroup_F\ K) \\ \quad (Unit_G(MultiplicativeGroup_F\ K)\ ) \\ \quad (Times_G(MultiplicativeGroup_F\ K)\ x\ y) \end{array}$$

This syntactic problem can be overcome either by explicitly defining accessor functions as we did for $\mathbb{Z}$-sets and groups, or by extending the parser and type-checker to allow aliases for non-constant expressions, or perhaps, specifically for composite functions, so one could define + and . to be aliases allowing something like $1 + Kx.Ky$ to be written for the above term.

Simpson proposes a solution in dependent type theory to this problem in which mathematical structures are represented by functions from strings to component structures. This is not available to us in HOL, but I can think of no examples in mathematics where the statically typed approach sketched above would be semantically insufficient.

[9] This does not work in the traditional set-theoretic account, since the multiplicative structure of a field does not comprise a group unless it is restricted to the non-zero elements.

## 5 Summary

I have given an overview of three case studies in the use of the ProofPower-HOL theorem-prover on pure mathematical problem domains. This has highlighted some problems in giving a smooth formalisation. Solutions or partial solutions to these problems have been proposed. In particular, I have outlined a method for scaling the approach to the compound mathematical structures that predominate in modern century mathematics.

I have extracted some "morals" from the work done to date, and there is much more that could be said about good ways to go about capturing a useful and evolving body of pure mathematics in an automated theorem-proving system. However, agonising about the technical approach will be less productive than actually trying to do some mathematics and to learn from the attempt.

## Acknowledgments

My thanks are due to Matthew Franks, Hanne Gottliebsen, John Harrison, Roger Jones and Larry Paulson for their very helpful correspondence during the development of these case studies.

## References

1. R.D. Arthan. Mathematical case studies: Basic analysis. `http://www.lemma-one.com/ProofPower/examples/examples.html`, 2004.
2. R.D. Arthan. Mathematical case studies: Some group theory. `http://www.lemma-one.com/ProofPower/examples/examples.html`, 2004.
3. R.D. Arthan. Mathematical case studies: Some topology. `http://www.lemma-one.com/ProofPower/examples/examples.html`, 2004.
4. Hanne Gottliebsen. *Automated Theorem Proving for Mathematics: Real Analysis in PVS.* PhD thesis, University of St. Andrews, 2001.
5. T. Hales. The Flyspeck Project Fact Sheet. Technical report, `http://www.math.pitt.edu/~thales/flyspeck/index.html`, 2003.
6. John Harrison. Formalized Mathematics. Technical report, `http://www.cl.cam.ac.uk/users/jrh/papers/`, 1996.
7. John Harrison. Theorem Proving with the Real Numbers. Technical report, University of Cambridge Computer Laboratory, 1996.
8. Michael Henle. *A Combinatorial Introduction to Topology.* Dover Publications, Inc., 1979.
9. Joe Hurd. Predicate Subtyping with Predicate Sets. In Richard J.Boulton and Paul B. Jackson, editors, *Proceedings of TPHOLs 2001, LNCS 2152.* Springer-Verlag, 2001.
10. L. Paulson. Defining functions on equivalence classes. *Preprint: available at* `http://www.cl.cam.ac.uk/users/lcp/papers/Reports/equivclasses.pdf`, 2004.
11. W.V. Quine. *Quiddities.* Harvard University Press, 1987.
12. Carlos Simpson. Computer Theorem Proving in Math. *arXiv:math.HO/0311260 v2*, 20 February 2004.
13. J.M. Spivey. *The Z Notation: A Reference Manual, Second Edition.* Prentice-Hall, 1992.

# A THE THEORY $\mathbb{Z}$_set

## A.1 Parents

$\mathbb{Z}$

## A.2 Constants

| | |
|---|---|
| \$** | $'a \to \mathbb{Z} \to 'a\ SET \times (\mathbb{Z} \to 'a \to 'a) \to 'a$ |
| ***Car*** | $'a\ SET \times (\mathbb{Z} \to 'a \to 'a) \to 'a\ SET$ |
| $\mathbb{Z}$***_Set*** | $('a\ SET \times (\mathbb{Z} \to 'a \to 'a))\ SET$ |
| ***Orbit*** | $'a\ SET \times (\mathbb{Z} \to 'a \to 'a) \to 'a \to 'a\ SET$ |

## A.3 Type Abbreviations

| | |
|---|---|
| $'a$ $\mathbb{Z}$***_SET*** | $'a\ SET \times (\mathbb{Z} \to 'a \to 'a)$ |

## A.4 Fixity

| | |
|---|---|
| *Infix 310*: | ** |

## A.5 Definitions

***Car***

** $\quad \vdash \forall\ (set,\ action)$
$\quad\quad \bullet\ Car\ (set,\ action) = set$
$\quad\quad\quad \land\ (\forall\ x\ i \bullet\ (x\ **\ i)\ (set,\ action) = action\ i\ x)$

$\mathbb{Z}$***_Set*** $\quad \vdash \forall\ X$
$\quad\quad \bullet\ X \in \mathbb{Z}\_Set$
$\quad\quad\quad \Leftrightarrow (\forall\ x\ i \bullet\ x \in Car\ X \Rightarrow (x\ **\ i)\ X \in Car\ X)$
$\quad\quad\quad\quad \land\ (\forall\ x\ i\ j$
$\quad\quad\quad\quad \bullet\ x \in Car\ X$
$\quad\quad\quad\quad\quad \Rightarrow (x\ **\ (i + j))\ X = ((x\ **\ i)\ X\ **\ j)\ X)$
$\quad\quad\quad\quad \land\ (\forall\ x \bullet\ x \in Car\ X \Rightarrow (x\ **\ \mathbb{N}\mathbb{Z}\ 0)\ X = x)$

***Orbit*** $\quad \vdash \forall\ X\ x \bullet\ Orbit\ X\ x = \{y | \exists\ i \bullet\ y = (x\ **\ i)\ X\}$

### A.6 Theorems

$\boldsymbol{\mathbb{Z}\_set\_minus\_thm}$

$$\begin{array}{l} \vdash \forall\ X \\ \quad \bullet\ X \in \mathbb{Z}\_Set \\ \qquad \Rightarrow (\forall\ x\ i \bullet\ x \in Car\ X \Rightarrow ((x ** i)\ X ** \sim i)\ X = x) \end{array}$$

$\boldsymbol{\mathbb{Z}\_set\_cancel\_thm}$

$$\begin{array}{l} \vdash \forall\ X\ x\ y\ i \\ \quad \bullet\ X \in \mathbb{Z}\_Set \land x \in Car\ X \land y \in Car\ X \\ \qquad \Rightarrow ((x ** i)\ X = (y ** i)\ X \Leftrightarrow x = y) \end{array}$$

$\boldsymbol{orbit\_refl\_thm}$

$$\vdash \forall\ X \bullet\ X \in \mathbb{Z}\_Set \Rightarrow (\forall\ x \bullet\ x \in Car\ X \Rightarrow x \in Orbit\ X\ x)$$

$\boldsymbol{orbit\_disjoint\_thm}$

$$\begin{array}{l} \vdash \forall\ X \\ \quad \bullet\ X \in \mathbb{Z}\_Set \\ \qquad \Rightarrow (\forall\ x\ y \\ \qquad \bullet\ x \in Car\ X \land y \in Car\ X \\ \qquad\quad \Rightarrow Orbit\ X\ x \cap Orbit\ X\ y = \{\} \\ \qquad\qquad \lor Orbit\ X\ x = Orbit\ X\ y) \end{array}$$