Pure Mathematics in a Mechanized Logic

John Harrison
Åbo Akademi University
Department of Computer Science
Lemminkäisenkatu 14 A
20520 Turku
`jharriso@abo.fi`

Abstract: It is widely believed that in principle it's possible to reduce most of present-day mathematics to reasoning in a formal logical system. The technical difficulty of actually doing so is quite formidable. However, the arrival of the computer is changing this situation, since computers are good at helping with such tedious symbolic manipulation. The computer formalization of mathematics is now a popular research topic. Here we report on our own development of mathematical analysis starting just from the axioms of simple type theory and reducing all reasoning (with the aid of the computer) to a formal deductive calculus of great simplicity.

1 Introduction

The idea of reducing reasoning to computation in some kind of mechanical calculus is quite an old one. For example, Hobbes [1] remarked that 'reasoning …is but reckoning', and even noted the difficulty of ensuring that logical reasoning is correct:

> For as Arithmeticians teach to adde and subtract in *numbers* [...] The Logicians teach the same in consequences of words [...] And as in Arithmetique, unpractised men must, and Professors themselves may often erre, and cast up false; so also in any other subject of Reasoning the ablest, most attentive, and most practised men, may deceive themselves, and inferre false conclusions.

Symbolic logic developed over the course of time in the hands of many people, Leibniz and Boole being prominent examples. The first logical systems which were adequate for the formalization of substantial parts of mathematics were developed by Frege and Peano. Frege concentrated more on logical and philosophical foundations; he wished to show that mathematics is analytic, *contra* Kant who claimed it is synthetic a priori. Peano was more interested in writing mathematics in a formal language. Together with his colleagues and assistants he published a substantial amount of formalized mathematics: his journal *Rivista di Matematica* was published from 1891 until 1906, and polished versions were collected in various editions of the *Formulaire* *de Mathématique*. The task of formalization stimulated several innovations, e.g. the identification of sequences with functions out of $\mathbb{N}$, which is now taken for granted.

These developments reached a synthesis in the work of Russell and Whitehead [2], who in *Principia Mathematica* gave a formal development of mathematics, starting from the barest logical and set-theoretic assumptions. In a way this was a triumph which has never been surpassed. But the resulting text demands extremely close attention from the reader, and there can be few who made the effort. Indeed, Russell's own intellect, he reports, never quite recovered from the strain of writing it. The idea of practically using these formal logics for real mathematics gradually fell into disrepute. On the other hand a number of authorities such as Bourbaki [3] seem to regard complete formalizability *in principle* as the ultimate arbiter of correctness. The arrival of the computer is changing this state of affairs, for several reasons.

- As one relies less on intuition and more on rules of formal manipulation, accuracy in those manipulations becomes more important. This is tedious for people, but checking conformance to formal rules is one of the things computers are very good at.
- The computer need not simply check the correctness of formal arguments, but can try to help in their construction; at one extreme, it is sometimes possible for computers to find proofs of theorems completely automatically.
- Among modern computer scientists, formal logic is quite widely known and substantially used; many regard formal logic as the 'applied mathematics' underlying computer science, just as real analysis etc. underlies physics and engineering.
- Computer correctness is itself a topic of concern, especially since computers are used in safety-critical systems like fly-by-wire aircraft. One means of verifying a design is to prove that a mathematical model of the system satisfies some formal properties, designed to correspond to real-world requirements. Such proofs are much more technically intricate, albeit perhaps shallower, than typical mathematics proofs, and of necessity involve very complicated formal manipulations.

There have already been a number of formalizations of significant parts of mathematics in compter theorem provers. One of the earliest examples is the work of Jutting, who formalized Landau's book on the construction of $\mathbb{R}$ in the AUTOMATH system [4]. The largest and most sustained development (as far as we know) is the Mizar project led by Trybulec [5]. Here, a large body of mathematics has been formalized, spanning pure set theory, algebra, analysis, topology, category theory and various unusual applications like mathematical puzzles and computer models. Our own work, which we describe here, is conducted in the HOL system, a descendant of Edinburgh LCF [6]. The system rigorously enforces a complete decomposition to simple primitive inferences, but at the same time, allows the user to write arbitrary programs to automate common patterns and generally render this task less tedious.

2 The HOL logic

Here we will quickly discuss the HOL types, logical syntax and deductive system.[1] HOL's logic is based on λ-calculus, a formalism invented by Alonzo Church. In HOL, as in λ-calculus, there are only four kinds of term:

- Constants, e.g. 1, 2, $\top$ (true) and $\perp$ (false). The set of constants is extensible by constant definition, which we discuss later.
- Variables, e.g. n, x, p. We assume some arbitrary denumerable set of variables; in HOL a variable may have any string as its name.
- Applications, e.g. $f(x)$. These represent the evaluation of a function f at an argument x. The brackets may be omitted, e.g. $f\ x$.
- Abstractions, e.g. $\lambda x.\ t[x]$. This example represents 'the function of x that yields $t[x]$'.

Abstractions are not often seen in informal mathematics, but they have at least two merits. First, they allow one to write anonymous function-valued expressions without naming them (occasionally one sees $x \mapsto t[x]$ used for this purpose), and since our logic is avowedly higher order, it's desirable to place functions on an equal footing with first-order objects in this way. Secondly, they make variable dependencies and binding explicit; by contrast in informal mathematics one often writes $f(x)$ in situations where one really means $\lambda x.\ f(x)$.

Logical and mathematical syntax is systematically implemented in terms of these primitive constructs of λ-calculus, using various new constant symbols. For example:

- Predication in logic, e.g. $P(x)$ ('predicate P holds of x') is just an application of P to x.
- The conjunction $p \wedge q$ ('p and q') is just $(\wedge p)q$ where $\wedge$ is the constant symbol representing conjunction. Note that although λ-abstractions can only have one argument, we can get the effect of many arguments by regarding a function as higher order. Here for example $\wedge$ takes one argument and yields another function. This device is known as 'currying' after the logician Haskell Curry.
- Universal quantification $\forall x.\ P[x]$ ('for all x, $P[x]$') is $\forall(\lambda x.\ P[x])$, i.e. a higher order function $\forall$ applied to a lambda-term. In general, constructs like summations $\Sigma_{i=0}^{n} t[i]$ and derivatives $\frac{d}{dx} f(x)$ which appear to bind variables are all implemented using a higher order function and λ-abstraction. This view is anyway natural in functional analysis, where differentiation is regarded as a (higher order) operator.

[1] Gordon and Melham [7] give a discussion that is both more leisurely and more precise, though the deductive system in the present version of HOL differs in detail from the one given there.

However, assuming basic properties like $(\lambda x. t[x])s = t[s]$, the system becomes inconsistent when augmented with basic logical operations. For example if $\neg$ is the logical negation operator ('not'), we can derive Russell's paradox of 'the set of all sets which are not members of themselves' (using predication $P\ x$ rather than set membership $x \in P$):

$$\vdash (\lambda x. \neg(x\ x))(\lambda x. \neg(x\ x)) = \neg((\lambda x. \neg(x\ x))(\lambda x. \neg(x\ x)))$$

Accordingly, Church [8] augmented λ-calculus with a theory of types, simplifying Russell's system from Principia Mathematica and giving what is often called 'simple type theory'. HOL follows this system quite closely, but allows type variables, giving a form of polymorphism. Every term has a unique *type* which is either one of the basic types or type variables or the result of applying a type constructor to other types. The only basic types in HOL are initially the type of booleans, *bool* and the infinite type of individuals ind; the only type operator is the function space constructor $\rightarrow$. We write $t : \gamma$ to indicate that a term t has type γ. Readers familiar with set theory may like to think of types as sets within which the objects denoted by the terms live, so $t : \gamma$ can be read as $t \in \gamma$. Note that the use of the colon is already standard notation when used for function spaces, i.e. one typically writes $f : A \rightarrow B$ rather than $f \in A \rightarrow B$.

Just as with the type systems of modern programming languages, functions may only be applied to arguments of the right type; only a function of type $f : \gamma \rightarrow \ldots$ may be applied to an argument of type γ. This restriction is decidable and applied at the level of the term syntax. In fact, when a term is written down, even with no type annotations at all, HOL is capable not only of deciding whether it is well-typed, but inferring a most general type for it and each of its subterms if it is. Thus the user can normally omit types and does not need to be bothered by them.

HOL takes as primitives just three constants, $=: \alpha \rightarrow (\alpha \rightarrow \mathit{bool})$ (equality), $\Rightarrow: \mathit{bool} \rightarrow (\mathit{bool} \rightarrow \mathit{bool})$ (implication) and $\varepsilon : (\alpha \rightarrow \mathit{bool}) \rightarrow \alpha$ (selection). The last of these is a device due to Hilbert: εP denotes 'some x such that $P(x)$'. Many systems of formal logic are deficient in that they do not allow one to give such descriptions, necessitating the introduction of additional symbols or complicated locutions to avoid them. HOL's deductive system is based on a system of 13 rules and 3 axioms. All these allow the derivation of sequents $\Gamma \vdash p$ ('if each term in Γ is true, so is p'), where all the terms involved have type *bool* (there is no distinguished syntactic class of formulas as in first order logic). Examples of rules include the transitivity of equality:

$$\frac{\Gamma \vdash x = y \quad \Delta \vdash y = z}{\Gamma \cup \Delta \vdash x = z}\ \texttt{TRANS}$$

so-called β-conversion, which expresses the fact that abstraction and application are converse operations:

$$\overline{\vdash (\lambda x.\, t[x])s = t[s]} \ \texttt{BETA\_CONV}$$

and the instantiation rule which allows the free variables in p to be assigned to arbitrary terms (subject to type restrictions), provided none of these variables appear free in Γ.

$$\frac{\Gamma \vdash p[x_1, \ldots, x_n]}{\Gamma \vdash p[t_1, \ldots, t_n]} \ \texttt{INST}$$

As well as these rules, there are principles of constant definition and type definition. Constant definition allows one, given a term t with no free variables and an unused constant name c, to add an additional equational axiom $\vdash c = t$. Often in logic one regards definitions as some kind of metalogical abbreviation. Here we actually use axioms in the object logic, for reasons of efficiency and simplicity: one can show that the addition of such axioms does not make the logic inconsistent; effectively one *is* just introducing shorthands. Similarly, the principle of type definition allows one to introduce a new type or type operator which is 'really' just an abbreviation for a set constraint. If for a predicate $P : \gamma \to bool$ one has a theorem $\vdash P\ t$ (asserting that the set with characteristic predicate P is nonempty) then one can introduce a new type (or type operator if γ contains type variables) in bijection with that set. We cannot go into more detail here. Some examples of definitions are the other logical operations $\top$ (true), $\forall$ (for all), $\exists$ (there exists), $\wedge$ (and), $\vee$ (or), $\perp$ (false) and $\neg$ (not).

$$\begin{array}{rcl}
\top & = & (\lambda x.\, x) = (\lambda x.\, x) \\
\forall & = & \lambda P.\, P = \lambda x.\, \top \\
\exists & = & \lambda P.\, \forall Q.\, (\forall x.\, P(x) \Rightarrow Q) \Rightarrow Q \\
\wedge & = & \lambda t_1.\, \lambda t_2.\, \forall t.\, (t1 \Rightarrow t2 \Rightarrow t) \Rightarrow t \\
\vee & = & \lambda t_1.\, \lambda t_2.\, \forall t.\, (t1 \Rightarrow t) \Rightarrow (t2 \Rightarrow t) \Rightarrow t \\
\perp & = & \forall P.\, P \\
\neg & = & \lambda t.\, t \Rightarrow \perp
\end{array}$$

These definitions look a bit obscure at first sight, but the first two express the intended meaning quite directly, while the next three are roughly the definitions of meets and joins in a lattice (think of implication as $\leq$). That concludes the logic proper, and in fact quite a bit of interesting mathematics, e.g. infinitary inductive definitions [9] can be developed just from that basis, without, moreover, using the ε operator. But for general use we adopt three more axioms. First there is an axiom of extensionality, which we encode as an η-conversion theorem: $\vdash (\lambda x.\ t\ x) = t$. Then there is the axiom giving the basic property of the ε operator, that it picks out something satisfying P whenever there is something to pick:

$$\vdash \forall x.\, P(x) \Rightarrow P(\varepsilon x.\, P[x])$$

Rather surprisingly, this also makes the logic classical, i.e. allows us to prove the theorem $\forall p.\, p \vee \neg p$; see [10] for the proof we use. Finally there is an axiom asserting that the type ind is infinite; the Dedekind/Peirce definition of 'infinite' is used:

$$\begin{array}{ll} \vdash \exists f : ind \to ind. & (\forall x_1, x_2.\, (f(x_1) = f(x_2)) \Rightarrow (x_1 = x_2)) \wedge \\ & \neg(\forall y.\, \exists x.\, y = f(x)) \end{array}$$

From this foundation, all the existing HOL theories, and the development of real analysis we describe in this paper, are derived by definitional extension. In a sense, the HOL mathematical development, of which this work represents the culmination, realizes at last the dreams of the logical pioneers.

3 Constructing the real numbers

As we have said, we do not introduce any additional axioms (other than definitional ones), so the task is to construct the real numbers in the traditional sense of finding a model for them in the mathematical world we do have available. This world includes the *natural* numbers $\mathbb{N}$, which we can define quite easily via type definition using a subset of ind as follows. If $f : ind \to ind$ is injective but not surjective (using ε and the axiom of infinity we can write down such an f), we can write down an element $z \notin ran(f)$. Now carve out a set inductively by:

$$\frac{}{z \in S}$$

$$\frac{x \in S}{f(x) \in S}$$

and define the type of natural numbers to be in bijection with S. It is clear that the principle of mathematical induction holds by definition, and the constructors are injective and distinct because f is injective and $z \notin ran(f)$.

There are many ways available of constructing the reals starting from something simpler. Perhaps the two best-known are Dedekind's and Cantor's constructions. Both of these require the construction of the rational numbers $\mathbb{Q}$ first. Dedekind's method identifies a real number x with the set of all rational numbers $r < x$, while Cantor's method identifies it with the set of all rational sequences which converge to x. For use as definitions, these can easily be rephrased so as not to mention the real numbers explicitly. For example, Dedekind's method defines reals as 'cuts' of the rationals, which are nontrivial downward-closed sets with no greatest element, while Cantor's method uses equivalence classes of rational Cauchy sequences.

These techniques are classical and quite elegant, but they require a certain amount of groundwork to construct the rationals first. Actually, if Dedekind's method is used, it is preferable to proceed from $\mathbb{N}$ to the positive rationals $\mathbb{Q}^+$ first, then to the positive reals $\mathbb{R}^+$ and finally to the reals. The reason is that one wants to define multiplication of cuts elementwise:

$$XY = \{xy \mid x \in X \wedge y \in Y\}$$

but if the cuts X and Y extend to $-\infty$, the result is not a cut. This route via $\mathbb{Q}^+$ and $\mathbb{R}^+$ was followed by Landau [11] in his classic monograph devoted to the reals construction.

Perhaps the most intuitive construction, which moreover does not require us to construct the rational first, is as positional (say, decimal or binary) expansions. This allows one to identify a (positive) real with an appropriately restricted function $\mathbb{N} \to \mathbb{N}$. However it turns out to be difficult to define all the operations, and prove that they have the required properties; multiplication is particularly awkward. However as shown by Behrend [12], it is not necessary to define multiplication or the multiplicative inverse explicitly. A structure $(X, +, <)$ is called a *system* (of magnitudes) if the following conditions are satisfied:

- $\forall x\ y.\ x + y = y + x$
- $\forall x\ y\ z.\ x + (y + z) = (x + y) + z$
- $\forall x.\ x \not< x$
- $\forall x\ y.\ x = y \vee x < y \vee y < x$
- $\forall x\ y.\ x < y \Leftrightarrow \exists d.\ y = x + d$

Examples are the positive reals, the positive rationals and the positive natural numbers. We say a system is *dense* if $\forall x, y.\ x < y \Rightarrow \exists z.\ x < z \wedge z < y$, and *complete* if every nonempty subset which is bounded above has a supremum (least upper bound). Note that the naturals are not dense while the positive rationals are not complete. We have the following theorem:

Theorem 1 *If $(X, +_X, <_X)$ is a system, and $(Y, +_Y, <_Y)$ a complete system, then for any $x \in X$, $y \in Y$, there is a unique homomorphism $h : X \to Y$ with $h(x) = y$.*
Proof: *See Behrend's paper.*

This can be applied to endomorphisms of $(\mathbb{R}^+, <, +)$. Pick an arbitrary element of $\mathbb{R}^+$ and call it 1. We know that for each $x \in \mathbb{R}^+$ there is a unique endomorphism $x^\sharp : \mathbb{R}^+ \to \mathbb{R}^+$ that maps $1 \mapsto x$. So *define* multiplication as follows:

$$xy = x^\sharp(y)$$

It is not hard to show that this has all the required properties. In order, one can see that it is left distributive, has 1 as a right identity, is associative, has 1 as a left identity, is right distributive, is commutative and finally has an inverse. See Behrend's paper for details.

We adopted yet another alternative, which attempts to equal the elegance and directness of Cauchy's method, but without requiring us to construct the rationals first. We say that a function $a : \mathbb{N} \to \mathbb{N}$ is 'nearly multiplicative' if:[2]

$$\exists B.\, \forall m, n \in \mathbb{N}.\ |na_m - ma_n| \leq B(m+n)$$

The idea is that (a_n) encodes a Cauchy sequence (x_n) with $O(1/n)$ convergence rate, via $x_n = a_n/n$. One can show [13] that this is equivalent to 'nearly additivity'.[3]

$$\exists B.\, \forall m, n \in \mathbb{N}.\ |a_{m+n} - (a_m + a_n)| \leq B$$

We define the positive reals as equivalence classes of such functions under the equivalence:

$$a \sim b \Leftrightarrow \exists B.\, \forall n \in \mathbb{N}.\ |a_n - b_n| \leq B$$

defining the ordering similarly:

$$a \leq b \Leftrightarrow \exists B.\, \forall n \in \mathbb{N}.\ a_n \leq b_n + B$$

and addition pointwise:

$$(a+b)_n = a_n + b_n$$

Apparently it is necessary to rescale and define multiplication by $(ab)_n = a_n b_n / n$. However a few easy manipulations with the nearly-multiplicative property shows that *function composition* is equivalent. The equivalence is used to establish commutativity, and all the other properties are immediate (for example, function composition is always associative). Completeness is fairly straightforward too; if n^* denotes the injection of $n \in \mathbb{N}$ into $\mathbb{R}$, then if S is a set which is bounded above, the following defines its supremum:

$$r_n = max\{r \mid \exists a \in S.\ r^* \leq n^* a\}$$

The multiplicative inverse can either be derived abstractly from denseness and completeness or explicitly defined by $a_n^{-1} = n^2/a_n$. For details, see [13].

[2] Even though we deal with the naturals, we define $|x - y|$ as absolute difference.

[3] Nearly-additive functions were suggested as a representation of the reals by Schanuel, though he did not regard it as a useful construction because of the apparent difficulty of proving that multiplication is commutative.

4 Formalizing analysis

One of the triumphs of mathematics in the nineteenth century was the rigorization of analysis. People like Cauchy, Bolzano and Weierstrass gave precise 'ϵ-δ' definitions of notions such as limits, continuity, differentiation and integration. For example a function $f : \mathbb{R} \to \mathbb{R}$ is said to be continuous (on $\mathbb{R}$) precisely when:

$$\forall x \in \mathbb{R}.\, \forall \epsilon > 0.\, \exists \delta > 0.\, \forall x'.\, |x - x'| < \delta \Rightarrow |f(x) - f(x')| < \epsilon$$

Now in a rigorous treatment, this is actually a *definition*. However it is important, if our theories of continuous functions are to have their psychological or practical significance, that this correspond to our intuitive notion of what a continuous function is. That is, even if unobvious at first sight, it must be seen in retrospect to be the 'right' definition. Is this the case here? Perhaps the most intuitive feature of continuous functions is that they attain intermediate values:

$$\begin{aligned} \forall x, x', z \in \mathbb{R}.\;\; & x < x' \wedge (f(x) < z < f(x') \vee f(x') < z < f(x)) \\ & \Rightarrow \exists w.\, x < w < x' \wedge f(w) = z \end{aligned}$$

This property is called 'Darboux continuity'. Why not take that as the definition of continuity instead? In general, where there are several apparently equally attractive choices to be made, which one should be selected?

Such worries can be allayed when the various plausible-looking definitions are all provably equivalent. For example, the alternative definitions of computability in terms of Turing machines, Markov algorithms, general recursive functions, untyped λ-calculus, production systems and so forth all turned out so. But here this is not the case: continuity implies Darboux continuity but not conversely (we shall see an example later). The usual definition of continuity has probably won out because it is intuitively the most satisfying, leads to the tidiest theory or admits the most attractive generalization to other metric and topological structures. But note that it also led to the counterintuitive pathologies of real analysis such as Bolzano's and Weierstrass's examples of everywhere continuous nowhere differentiable functions. This shows how the rigorous version of a concept can take on an unexpected life of its own.

Our HOL formalization of analysis follows the techniques that have now become standard in mathematics. It does not require such difficult analyses of informal concepts, since modern analysis is already quite rigorous. The most difficult decision we had to take was the precise theory of integration to develop (more on this later). But sometimes the demands of complete formalization and the exigencies of computer implementation can throw new light on informal, even though rigorous, concepts. In what follows, we will emphasize some of the unusual features of our development.

4.1 The degree of abstraction

Modern mathematics has attained a high level of abstraction. This is for a complicated mixture of reasons. It deepens one's understanding of mathematics if one can see very clearly which assumptions and particularities are actually necessary, and if so what role they play. Moreover, the abstract versions of concepts are often aesthetically beautiful, or so general that they admit surprising applications and bring out new connections. There are also more pragmatic reasons for abstraction, though. First the proofs can actually be easier in a more abstract framework. Arguably, topological proofs of results in real analysis are often more elegant and straightforward than those rooted in particular properties of the real line. (For example, the proof that the image of a connected set under a continuous function is connected is perhaps more direct than the usual proofs of the intermediate value theorem.) Moreover, they allow an economy of effort, since it is often possible to apply the same theorem in many specific situations, e.g. properties of continuous functions to $\mathbb{R}$, $\mathbb{R}^n$ and $\mathbb{C}$.

Our initial approach to analysis was quite utilitarian, so we only pursued abstraction when it seemed likely to achieve economy of effort. Since the effort involved in mechanizing proofs is greater than usual, this is especially important, though on the other hand a programmable theorem prover like HOL may allow one to apply 'similar' proofs in different situations. We developed theories of topology, but in general found that since so far we have only used the concrete case of $\mathbb{R}$, this did not achieve any real saving. However another general theory, of convergence nets, was genuinely useful, since we avoided proving many similar theorems about different kinds of limits (pointwise, sequential, at $\pm\infty$, ...).

4.2 Clarifying variable binding

One attractive feature of analysis into λ-calculus is that the status of free and bound variables becomes more explicit. For example the everyday Leibniz derivative notation as in $\frac{d}{dx}(x^2) = 2x$ conceals a rather subtle point. The variable x obviously occurs free in the right-hand side of this equation and bound in the left. But it isn't *just* a bound variable on the left, because changing the variable name changes the right hand side too! If we write the above out formally using our definitions, we get $deriv\ (\lambda x. x^2)\ x = 2x$. Now the situation on the left hand side becomes clear. There are really two separate instances of x, one bound, one free, which the Leibniz notation conflates. A precisely similar situation can occur in integration, but here the standard notation separates the two instances:

$$\int_0^x 2x\ dx = x^2$$

Indeed, careful authors usually abandon the Leibniz derivative notation in more advanced work, or indicate with a subscript the point at which the resulting derivative is to be calculated.

Another derivative notation that seems clear is the use of the prime, $f'(x)$. However the author has witnessed a heated debate on `sci.math` among good mathematicians over whether $f'(g(x))$ denotes the derivative of f evaluated at $g(x)$ (this view seems most popular) or the derivative of $f \circ g$ evaluated at x ('the prime notation (f') is a shorthand notation for derivative of a univariate function with respect to the free variable.')

4.3 The problem of partial functions

There is a respect, however, in which the HOL formalization does not work out so well. Consider the following two theorems:

1. If $x_n \to a$ and $y_n \to b$ as $n \to \infty$, then $x_n + y_n \to a + b$ as $n \to \infty$.

2. If $lim_{n\to\infty} x_n = a$ and $lim_{n\to\infty} y_n = b$ then $lim_{n\to\infty} x_n + y_n = a + b$.

Intuitively, these look pretty much identical. However there is a subtle difference between them. The statement $lim_{n\to\infty} x_n = a$ analyzes down to LIM $(\lambda n.\ x_n) = a$. Now LIM has type $(\mathbb{N} \to \mathbb{R}) \to \mathbb{R}$, and like all the functions in HOL, it is *total*. This means that even when applied to a nonconvergent sequence it yields some value. So the statement $\vdash lim_{n\to\infty} x_n = a$ does not allow us to deduce that x_n is convergent at all, still less any more advanced theorems about composition of limits. Therefore in the HOL development we rely on the relational forms. Another effect of total functions is that because we define $x/y = xy^{-1}$, we have $0/0 = 0$, because whatever 0^{-1} might be, it is a real number and multiplying it by zero gives zero.

If one looks at analysis texts they usually say something like 'if $s_n \to l$ as $n \to \infty$, then we write $lim_{n\to\infty} s_n = l$'. So taken literally, reading 'if' as 'iff' as one conventionally does for definitions, the whole construct is some kind of *contextual definition* which is really to be regarded as a shorthand for the relational statement. Contextual definitions are those that do not necessarily refer directly to well-defined entities in the object logic, but are to be regarded as shorthands for possibly quite structurally different statements there. The use of proper classes in ZF set theory is a good example, and one can even look at cardinal arithmetic this way, that is, $|A| = |B|$ is really an abbreviation for 'there exists a bijection between the sets A and B', and so on.

From one point of view then, our sticking to a purely relational formalization is defensible, in that it could be said to constitute an *analysis* of what the informal statement means. One could even say that it is through the metalanguage that the functional versions should be interpreted, not in the object logic. However, the fact remains that the relational form can be very inconvenient to work with. The inconvenience comes to the fore when such constructs are nested, the prime example being the nested differentiation operations in differential equations. Here, differential equations as written in textbooks need to be accompanied by a long string of differentiability assumptions that in informal usage are understood implicitly. (Even though some equality expressions are contextually defined, equality is still always treated as substitutive.) Therefore one might ask: is there a better formalism in which such statements can be formalized in a way that keeps them close to informal convention?

One simple possibility is to use an untyped system like standard set theory. Here, the lack of types gives more freedom to adopt special values in the case of undefinedness. For example, many functions which we want to be undefined in certain places have a natural 'type' $X \to Y$. (For example, the inverse function $\mathbb{R} \to \mathbb{R}$, and the limit operation $(\mathbb{N} \to \mathbb{R}) \to \mathbb{R}$ are just two we have discussed here.) One might adopt the convention of extending their ranges to $Y \cup \{Y\}$ and giving the function the value Y in the case of undefinedness. This would seem to give many of the benefits of a first-class notion of definedness without complicating the foundational system.

A more radical approach is to adopt a logic in which certain terms can be 'undefined'. Then, on the understanding that $s = t$ means 's and t are defined and equal', the statement $lim\ s = l$ would, assuming that l is defined, incorporate the information that s does actually converge. Such a logic is implemented by the IMPS system [14], which was expressly designed for the formalization of mathematics, and otherwise has a type theory not dissimilar to HOL's. However some dislike the idea of complicating the underlying logic, either because of a desire for implementation simplicity, or because of concern that proofs are likely to become cluttered with definability assumptions. The original LCF systems [6, 15] implemented a similar logic (a special 'bottom' element denoted undefinedness), as did the first version of the LAMBDA hardware verification tool, and in both cases, the clutter was considerable.

An alternative, long used by the Nuprl system and more recently by PVS, is to make all functions total, but exploit a facility for transparent subtypes. In such a scheme, lim would have not type $(\mathbb{N} \to \mathbb{R}) \to \mathbb{R}$ but rather $\{s \mid \exists l.\, s \to l\} \to \mathbb{R}$. Now the logic is still almost as simple (subtypes are desirable for other reasons too, anyway), and if $lim\ s = l$ is well-typed, then s must converge. However there is still an additional load of proof obligations involved in typechecking, which is no longer decidable as in HOL. Moreover, it means that the underlying types can become extremely complicated and counterintuitive. Finally, it is less flexible than the IMPS approach, which, as it actually features a definedness operator, can employ its own bespoke notions of equality. For example in many informal contexts it seems that the equality $s = t$ is interpreted as what IMPS calls 'quasi-equality', i.e. 'either s and t are both undefined, or they are both defined and equal'. On such a reading, for example, the equation $\frac{d}{dx}(1/x) = -1/x^2$ is strictly valid. Actually [16] uses a special asymmetric 'Venturi tube' equality meaning 'if the left hand side is defined then so is the right and they are equal'. For other contrasts between the approaches to undefinedness, consider the following (derived from a multiplicative inverse operation that is undefined at 0):

$$\forall x \in \mathbb{R}.\, tan(x) = 0 \Rightarrow \exists n \in \mathbb{Z}.\, x = n\pi$$

or the following:

$$\forall x \in \mathbb{R}.\ (x^{-1})^{-1} = x$$

and

$$\forall x \in \mathbb{R}.\ x^{-1} > 0 \Leftrightarrow x > 0$$

4.4 The Carathéodory derivative

For practical use, we want to be able to prove theorems about the derivatives of specific functions. The various combining theorems (derivatives of sum and product etc.) are mostly straightforward. The easiest way to prove the rules for inverses and quotients is to prove first that $\lambda x.x^{-1}$ has derivative $-1/x^2$ at all points x other than 0, and then use the chain rule. First however we need to prove the chain rule, and that does turn out to be trickier than expected. In Leibnizian notation the theorem is very suggestive:

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx}$$

It would seem that to prove it we need simply observe that the above is true for finite differences Δx, and consider the limit. However this does not work easily, because we have to consider the possibility that Δu may be zero even when Δx is not; crudely speaking, the problem is that limits are not compositional. If $f(x) \to y_0$ as $x \to x_0$ and $g(y) \to z_0$ as $y \to y_0$, it may not be the case that $(g \circ f) \to z_0$ as $x \to x_0$. The reason is that the definition of limit:

$$\forall \epsilon > 0.\ \exists \delta > 0.\ \forall y.\ 0 < |y - y_0| < \delta \Rightarrow |g(y) - z_0| < \epsilon$$

includes the extra property that $0 < |y - y_0|$, i.e. $y \neq y_0$. This is of course necessary since in many situations (e.g. the derivative) the function whose limit is being considered might be undefined or nonsensical at $y = y_0$.

The usual proofs of the chain rule therefore split into two separate cases, which makes the result rather messy. There are rumours that a large proportion of American calculus texts get the proof wrong. Certainly, the author has seen one which explicitly noted that the chain rule's proof is too complicated to be given. There is a way out, however, Continuity *is* compositional, and the chain rule follows quite easily from the following alternative characterization of differentiability, due to Carathéodory. A function f is differentiable at x with derivative $f'(x)$ iff there is a function g_x, continuous at x and with value $f'(x)$ there, such that for all x':

$$f(x') - f(x) = g_x(x)(x' - x)$$

The equivalence with the usual definition is easy to establish. The theorem about the differentiation of inverse functions is also eased by using the Carathéodory characterization [17].

4.5 The transcendental functions

The transcendental functions are defined by their power series. After a fairly hard proof showing that it is admissible to differentiate these series term-by-term inside their circle of convergence, we have theorems like $\frac{d}{dx}sin(x) = cos(x)$ as well as some theorems about degenerate cases, like $sin(0) = 0$ and $cos(0) = 1$. Interestingly, these suffice for the proof of most (all?) of the basic algebraic properties. The trick used to prove identities $f(x) = 0$ is to prove that $f'(x) = 0$, indicating that f is constant, and that $f(a) = 0$ for some particularly convenient value a. For example to prove $sin^2(x) + cos^2(x) = 1$ we differentiate the left hand side w.r.t. x and get $2sin(x)cos(x) - 2cos(x)sin(x) = 0$, and the result follows. Slight variants of this work for other situations too, e.g. to prove $exp(x+y) = exp(x)exp(y)$, differentiate $exp(x+y)exp(-x)$ w.r.t. x, and to show the addition formulas for sin and cos, prove them both together by showing that the following has zero derivative w.r.t. x and is zero at $x = 0$.

$$\begin{aligned}\lambda x.\ \ &(sin(x+y) - (sin(x)cos(y) + cos(x)sin(y)))^2 + \\ &(cos(x+y) - (cos(x)cos(y) - sin(x)sin(y)))^2\end{aligned}$$

This technique is quite simple and powerful, but does not seem to be widely exploited in analysis books. The only example we have found is an isolated use in [18], proving the addition formulas via differentiating $sin(a+x)cos(b-x) + cos(a+x)sin(b-x)$.

4.6 Gauge integration

A consequence of the definitional approach is that we must be particularly careful about the way we define mathematical notions. In some cases, the appropriate definitions are uncontroversial. However many areas of mathematics offer a range of subtly different approaches. Integration is a particularly difficult case [19, 20]. For a long time it was considered as the problem of quadrature (finding areas). However the discovery by Newton and Leibniz that it is broadly speaking a converse operation to differentiation led to many people thinking of it that way instead. Undergraduate mathematics courses usually present the Riemann integral. At a more advanced level, Lebesgue theory or some more abstract descendant seems dominant.

The relationship among these different integrals is quite complicated. Any function with a step discontinuity fails to be Newton integrable, and there are functions which are Lebesgue but not Riemann integrable, a classic case being the characteristic (indicator) function of the rationals.

$$f(x) = \begin{cases} 1 & \text{if } x \in \mathbb{Q} \\ 0 & \text{if } x \notin \mathbb{Q} \end{cases}$$

On the other hand, the derivative of the following function has, by definition, a Newton integral, but it is not Lebesgue integrable over any interval containing 0.

$$f(x) = \begin{cases} 0 & \text{if } x = 0 \\ x^2 sin(1/x^2) & \text{if } x \neq 0 \end{cases}$$

We elected to develop the gauge integral [21, 22] in HOL. This integral is a simple generalization of the Riemann integral which integrates all Lebesgue integrable functions and all derivatives, maintains the attractive convergence theorems (Dominated and Monotone Convergence) of the Lebesgue integral, and also obeys the Fundamental Theorem of Calculus without any additional conditions beyond f's differentiability.

$$\int_a^b f'(x)dx = f(b) - f(a)$$

A particularly attractive feature of this is that we can check the correctness of many integrals by differentiating. This has been exploited by Théry and the present author [23]: we use the Maple computer algebra system to propose antiderivatives and then attempt to verify them in HOL by proof. This is a small step in the important research goal of combining the convenience of these systems with the rigour of theorem proving.

References

[1] T. Hobbes. *Leviathan*. Andrew Crooke, 1651.

[2] A. N. Whitehead and B. Russell. *Principia Mathematica (3 vols)*. Cambridge University Press, 1910.

[3] N. Bourbaki. *Theory of sets*. Elements of mathematics. Addison-Wesley, 1968. Translated from French 'Théorie des ensembles' in the series 'Eléments de mathématique', originally published by Hermann in 1968.

[4] R. P. Nederpelt, J. H. Geuvers, and R. C. de Vrijer, editors. volume 133 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, 1994.

[5] A. Trybulec. The Mizar-QC/6000 logic information language. *ALLC Bulletin (Association for Literary and Linguistic Computing)*, 6:136–140, 1978.

[6] M. J. C. Gordon, R. Milner, and C. P. Wadsworth. *Edinburgh LCF: A Mechanised Logic of Computation*, volume 78 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.

[7] M. J. C. Gordon and T. F. Melham. *Introduction to HOL: a theorem proving environment for higher order logic*. Cambridge University Press, 1993.

[8] A. Church. A formulation of the Simple Theory of Types. *Journal of Symbolic Logic*, 5:56–68, 1940.

[9] J. Harrison. Inductive definitions: automation and application. In Phillip J. Windley, Thomas Schubert, and Jim Alves-Foss, editors, *Higher Order Logic Theorem Proving and Its Applications: Proceedings of the 8th International Workshop*, volume 971 of *Lecture Notes in Computer Science*, pages 200–213, Aspen Grove, Utah, 1995. Springer-Verlag.

[10] M. J. Beeson. *Foundations of constructive mathematics: metamathematical studies*, volume 3 of *Ergebnisse der Mathematik und ihrer Grenzgebiete*. Springer-Verlag, 1984.

[11] E. Landau. *Grundlagen der Analysis*. Leipzig, 1930. English translation by F. Steinhardt: 'Foundations of analysis: the arithmetic of whole, rational, irrational, and complex numbers. A supplement to textbooks on the differential and integral calculus', published by Chelsea; 3rd edition 1966.

[12] F. A. Behrend. A contribution to the theory of magnitudes and the foundations of analysis. *Mathematische Zeitschrift*, 63:345–362, 1956.

[13] J. R. Harrison. Theorem proving with the real numbers. Technical report, University of Cambridge Computer Laboratory, New Museums Site, Pembroke Street, Cambridge, CB2 3QG, UK, 1996. Author's PhD thesis.

[14] W. Farmer, J. Guttman, and J. Thayer. IMPS: an interactive mathematical proof system. In M. E. Stickel, editor, *10th International Conference on Automated Deduction*, volume 449 of *Lecture Notes in Computer Science*, pages 653–654, Kaiserslautern, Federal Republic of Germany, 1990. Springer-Verlag.

[15] L. C. Paulson. *Logic and computation: interactive proof with Cambridge LCF*. Number 2 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1987.

[16] P. J. Freyd and A. Scedrov. *Categories, allegories*. North-Holland, 1990.

[17] S. Kuhn. The derivative à la Carathéodory. *The American Mathematical Monthly*, 98:40–44, 1991.

[18] R. Haggarty. *Fundamentals of Mathematical Analysis*. Addison-Wesley, 1989.

[19] D. van Dalen and A. F. Monna. *Sets and Integration: an Outline of the Development*. Wolters-Noordhoff, 1972.

[20] I. N. Pesin. *Classical and modern integration theories*. Academic Press, 1970.

[21] J. Kurzweil. Generalized ordinary differential equations and continuous dependence on a parameter. *Czechoslovak Mathematics Journal*, 82:418–446, 1958.

[22] R. Henstock. A Riemann-type integral of Lebesgue power. *Canadian Journal of Mathematics*, 20:79–87, 1968.

[23] J. Harrison and L. Théry. Extending the HOL theorem prover with a computer algebra system to reason about the reals. In Jeffrey J. Joyce and Carl Seger, editors, *Proceedings of the 1993 International Workshop on the HOL theorem proving system and its applications*, volume 780 of *Lecture Notes in Computer Science*, pages 174–184, UBC, Vancouver, Canada, 1993. Springer-Verlag.