Math. Struct. in Comp. Science (2011), *vol.* 21, *pp.* 671–677.
doi:10.1017/S0960129511000235

Special Issue: Interactive Theorem Proving and the Formalisation of Mathematics

Guest Editors: Andrea Asperti and Jeremy Avigad

Preface

GÉRARD HUET

INRIA Paris-Rocquencourt
Email: Gerard.Huet@inria.fr

Received 29 November 2010; revised 15 December 2010

This special issue of *Mathematical Structures in Computer Science* is devoted to the theme of 'Interactive theorem proving and the formalisation of mathematics'.

The formalisation of mathematics started at the turn of the 20th century when mathematical logic emerged from the work of Frege and his contemporaries with the invention of the formal notation for mathematical statements called predicate calculus. This notation allowed the formulation of abstract general statements over possibly infinite domains in a uniform way, and thus went well beyond propositional calculus, which goes back to Aristotle and only allowed tautologies over unquantified statements.

At the *International Congress of Mathematicians* in 1900, Hilbert presented a list of 23 unsolved problems, the first of which being the design of an axiomatic system for mathematics capable of internally proving its own consistency. The search for a solution to this problem had profound implications for mathematical practice since it led Cantor and others to investigate set theory and its foundations in axiomatic systems expressed in variants of predicate calculus. Some kind of consensus was reached with the first-order axiomatisation of Zermelo and Fraenkel, possibly complemented with the axiom of choice (ZFC), but mysteries remained, such as the derivability of the continuum hypothesis, or even it soundness. It was unclear whether set theory was too strong (and possibly inconsistent) or too weak (and unable to prove its own consistency).

The matter was settled by Gödel, who destroyed all hopes of finding a consistent axiomatic system able to prove its own consistency. The best we could do is relative consistency proofs, such as Gödel's result that the continuum hypothesis is indeed consistent with ZFC. This is hardly satisfactory from a foundational point of view, but this line of research led to a consolidation of mathematical logic around two complementary points of view: proof theory along with the arithmetisation of syntax on the one hand and model theory within set theory on the other. Furthermore, the precise formulation of

mechanistic accounts of using arithmetics in the manipulation of formal expressions led to the investigation of the important area of recursive function theory in the 1940's. This resulted in the development of frameworks for computability, such as Kleene's recursive equations, Church's λ-calculus and Turing's abstract machines, and in a timely way to serve as abstractions and for endowing the emerging technology of electronic computers with informatics as a science of computing.

Meanwhile, mainstream mathematicians had become weary of such formalities. The attempt by Whitehead and Russell to design a variant of predicate calculus for actual mathematical practice culminated in their *Principia Mathematica*, a thick three-volume book giving a painstaking account of mathematical trivialities using an inscrutable notation. More elegant proof systems, such as Gentzen's systems of natural deduction and sequent calculus, were deemed to be still too low-level for fruitful use in mathematical practice, and mathematical logic was relegated to the status of an obscure specialty not really relevant to actual mathematical practice. Mathematicians were not too concerned about foundations any more, and the discovery of the independence of the continuum hypothesis by Paul Cohen in 1963 was taken lightly by most professional mathematicians, in spite of the methodological problems it raised for the sound development of a unified body of modular mathematics based on set theory.

A renewed interest in these matters started in the 1960's, when electronic computers became powerful enough to implement formal axiomatic systems and automate the search for formal proofs. Early computer scientists started to implement fragments of first-order logic, and to search for formal proofs using variants of Herbrand's theorem. In 1965, the discovery of the resolution method by Allen Robinson, using a notion of principal ideal in first-order instantiation (the so-called 'most general unifier' of two quantifier-free formulae), led to a flurry of investigations in a new research area called automated theorem proving, or, better, computational logic. Alas, concrete results were very disappointing, since only utter trivialities could be established by fully automated methods, and the treatment of essential methods such as reasoning about equality and induction was wholly unsatisfactory. Furthermore, the packaging of mathematical problems into obscure normal forms, with the replacement of existential quantifications by Skolem constants, and the distributivity blowups induced by conjunctive normal form prevented a human mathematician from helping in guiding the proof process in sensible ways, all intuition about the original problem having been thrown away like the proverbial baby with the bath water.

Despite these disappointing early attempts at the mechanisation of mathematics, which seemed to confirm the growing general doubts about the seriousness of artificial intelligence research, a few visionary endeavours in the 1970's sowed the seeds of future breakthroughs.

One line of research was to reject first-order predicate calculus as being too general, and to look for the mechanical investigation of simpler reasoning processes, where full automation could be obtained in special cases. One such line concerned inductive reasoning. In the early 1970's, Robert Boyer and J Moore started a 30-year long cooperation on the development of an automation of primitive recursive arithmetic known first as the Boyer–Moore Theorem Prover, then as ACL (an acronym for 'A Computational Logic'). This effort, building on the concurrent development of the programming language

LISP, was successful in building an automated prover robust enough to sustain the synthesis of the enormous formal proofs needed to establish the consistency of non-trivial software and hardware with respect to formal specifications. One could thus attain the level of confidence in their correctness necessary for their certification for safety-critical applications such as human transportation or for banking transactions. This system is still used to this day for hardware verification by Warren Hunt at the University of Texas, and was the main inspiration for the PVS program verifier developed by Nataraja Shankar at SRI and used for significant avionics applications.

Another specialisation was to look at the equational reasoning typical of algebraic proofs, and to focus it using systems of rewrite rules. A breakthrough in this direction occurred in 1970 with the pioneer work of Knuth and Bendix, which led to the development of sophisticated methods of equational reasoning in the 1970's and 1980's. The ensuing rewriting technology is now an unavoidable component of modern proof systems.

A completely orthogonal line of research was to look for a more general notation than first-order predicate calculus in order to design a powerful language suitable for the development of formal mathematics in computer-assisted systems. The computer would do the trivial administration of logical arguments, while the mathematician would steer the global proof development in high-level terms. Such an effort was pursued in the team of N. G. de Bruijn at Eindhoven University in the 1970's with the development of the Automath systems. The Automath language was a variant of typed λ-calculus with types structured as logical propositions, which were themselves represented in the calculus. The ensuing high-level language, or mathematical vernacular, was supposed to directly map into a mathematical argument as practised by human mathematicians, up to a linear blow-up factor. This visionary effort was ill-understood and under-estimated at the time, so Automath did not spread beyond its academic birthplace, but it had a very strong influence on further developments of the field.

Another effort in this attempt to use interactive environments for proof development, rather than fully automated mechanical theorem proving, was started in the 1970's by Robin Milner and his team at the University of Edinburgh in the framework of denotational semantics. This led to the LCF system, where proof search is guided by the execution of high-level tactics written in a safe programming language in the form of the meta-language ML. The LCF system was the inspiration for the architecture of many contemporary proof assistants, such as HOL, Isabelle, and Coq. Furthermore, the ML language emancipated itself from the guts of proof assistants to become a major line of development of functional programming languages, such as Standard ML, OCaml and Haskell.

Putting together the two strands of development, modern proof assistants mix powerful inductive and equational reasoning in reasoning systems where the user may guide a high-level proof development. Such systems develop by interactive means a data-structure reflecting a formal proof object, which may be guaranteed to respect the proof obligations of sophisticated inference rules. Furthermore, the inference rules themselves may be justified internally to a certain extent using a stratification of proofs allowing the system to prove its own consistency at one level in the next level, which is the best we may hope for in view of Gödel's incompleteness theorems.

A salient feature of modern proof assistants, which usually support some version of higher-order type theory allowing constructive proofs to be compiled into effective terminating procedures, allows the implementation of powerful reflection principles, where computation is used to prove theorems at a meta-theoretical level without incurring the actual production of an explicit proof object, which could be of enormous size.

During the last ten years this research into interactive theorem proving has had significant practical impact for the certification of computer systems at the software, hardware and middleware levels. But as a side-effect of the high-level nature of their specification languages, they have also begun to have a significant impact on the formalisation of mathematics. In 2005, Georges Gonthier managed to steer the Coq assistant, using as a plug-in a powerful system of tactics known as Ssreflect that allows reflection, into the development of a fully verified proof of the four-colour theorem, thereby stirring the interest of professional mathematicians.

The organisation of powerful libraries of formal mathematical developments is now a research area in its own right, and we can expect it to have a major impact on mainstream mathematical activity in the years to come, and can foresee a gradual shift of mathematical publication towards the development of standardised databases of mechanically refereed inter-linked mathematical libraries. This is not such a far-fetched scenario, since the Mizar endeavour has already started such a coordinated publication effort of formal mathematics with a certain amount of success.

This short introduction to early research in this field has explained how the formalisation of mathematics has evolved from speculations about the foundations of set theory into a major international effort in the development of computer-aided proof assistants. This special issue of *Mathematical Structures in Computer Science* presents a fairly representative selection of specific recent research strands in this new interdisciplinary field.

Three main themes are covered by the selection of papers in this special issue, namely:

— the formalisation of mathematics;
— methodology and proof infrastructure;
— foundations and meta-theoretic issues.

The first theme, the formalisation of mathematics, is covered by the papers by: Ioana Paşca; John Harrison; Yves Bertot, Frédérique Guilhot and Assia Mahboubi; and Claudio Sacerdoti Coen and Enrico Tassi. They cover significant developments of formal mathematics in real analysis, discrete geometry, algebra and topology, respectively, that are fully verified using the proof platforms Coq/Ssreflect, HOL Light and Matita.

Ioana Paşca's paper shows a significant development in real analysis, namely, Newton's method. This development is established in the Coq proof assistant, using Georges Gonthier's Ssreflect tactics library. This is a very impressive achievement since although Newton's method is well known, the precise statement of its validity under general conditions of the function and the initial point is far from trivial, and its formal proof demands quite an elaborate methodology. This paper provides a good explanation of the necessary mathematical background (some of it, such as Kantorovitch's Theorem, fairly recent), the formalisation of the main notions from real analysis in a proof assistant, the tools used to keep the notation compact enough for a human mathematician to grasp, the

subtleties of implementing the implicit notational confusions typical of a proof on paper, and, finally, the benefit of obtaining abstract formulations that are sometimes more general than the standard theorems as stated in textbooks. The author also shows that it is possible to go further than the standard formulation, which assumes infinite precision, and to refine conditions in order to prove convergence on sequences of rounded approximations, paving the way for the actual certification of numerical algorithms implemented on floating point representations of reals. This is an absolute *tour de force*, and may actually be a piece of new mathematics. The challenge here is to bridge the gap between numerical methods in applied mathematics and fully verified mathematics of the purest kind.

John Harrison's paper describes a formal proof of Pick's theorem developed in the HOL Light theorem prover. Pick's result is a theorem of discrete geometry relating the area of a simple polygon with vertices at integer lattice points to the number of lattice points in its interior and boundary. The proof of this result involves a variety of results from geometry, topology and algebra, such as variants of the Jordan Curve Theorem, results on the triangulation of polygons, a fair portion of Euclidean geometry and some linear algebra. The full and successful development of this piece of mathematics in the HOL Light proof assistant is thus an important landmark in formal mathematics. It shows that mathematical notions, such as the concept of 'inside', that are intuitively clear for a human, hide complex reasoning. It may be hoped that such case studies teach us how to design computational processes capturing such everyday intuition.

Yves Bertot, Frédérique Guilhot and Assia Mahboubi's paper discusses the problem of locating the roots of polynomials using Bernstein coefficients. They first explain the theory of special parametric polynomial curves known as Bézier curves, which were invented for computed aided design, and later used to define the spline curves used in computer graphics. These curves have been investigated from the computational point of view by de Casteljau. The related Bernstein polynomials allow an approximation of polynomials within a bounded interval using a finite set of rational numbers known as their Bernstein parameters. Finding the roots of a polynomial within a given interval may be achieved constructively by an iterative computational process on appropriate Bernstein parameters using de Casteljau's algorithm. This construction demands a modification of the intermediate value theorem, which can be used computationally in the case of polynomials – the authors call it the constructive intermediate value theorem. Putting all this together, the authors give a complete development within the Coq proof assistant, providing the essential infrastructure for the validation of efficient algorithms to isolate the real roots of an arbitrary polynomial up to a given approximation. This development is abstract enough to be modular in the sense of not depending on the particular representations used for reals, but just on their axiomatisation as a real closed field, and without assuming the intermediate value theorem. This piece of work is thus an important landmark on the road towards the certification of numerical methods.

Claudio Sacerdoti Coen and Enrico Tassi's paper is the last one directed to the first theme, the formalisation of mathematics, in this special issue. They present an investigation into the relationship between traditional topology and the more constructive version of 'pointless topology' as investigated, notably, by Giovanni Sambin in his 'Basic Picture' work. To this end, they define *overlap algebras*, which are algebraic structures designed to

reason constructively about subsets, and are a new and essential tool for formal topology. Their main result proves the existence of a functor between the categories of generalised topological spaces (called basic topologies) with and without points, respectively. They give a completely formal proof of this result in predicative intuitionistic logic using the Matita proof assistant. This shows that automated proof systems can be useful for validating complex contemporary mathematics. It is notable that this development was commissioned by the mathematician who originated pointless topology, and that it allowed the discovery of explicit complexity bounds for the constructions. The authors discuss how their investigation uncovered several inadequacies in the proof platform, which will be remedied in its next version, and, furthermore, helped to generalise the original paper mathematics by suggesting more abstract formulations.

The second theme of this special issue is methodology and proof infrastructure, and is represented by the papers by: Bas Spitters and Eelis van der Weegen; Frédéric Blanqui and Adam Koprowki; and Russell O'Connor. These papers discuss modular development in algebraic hierarchies using type classes, the certification of well-founded ordering descriptions and the use of classical mathematics for constructive logic, respectively.

Bas Spitters and Eelis van der Weegen propose a mechanism for structuring mathematics, specifically algebraic hierarchies with a notion of type classes recently introduced in the Coq proof assistant by Mathieu Sozeau. They successfully argue that the automatic coercion mechanisms solve several problems associated with a too explicit 'bundling' of algebraic components. They show how to build succesfully the hierarchy of monoids, rings and other standard algebraic structures. In the second part of the paper, they show how to develop categorical notions up to adjunction, then use this categorical background to develop categorical universal algebra. They arrive at the notion of the variety of an equational algebra, with an elegant formalisation that is fully consistent with modern presentations of universal algebra. This allows them to define N and Z as the initial algebras of the respective varieties. Finally, they offer an abstract characterisation of Q. This very elegant development systematically uses the constraint programming implicit in the unification solving mechanism of the prover in preference to explicit proof construction using tactics. The authors conclude with some suggestions for improving the underlying logical framework and core facilities. This paper is characteristic of the virtuous circle of improving proof assistants (here type classes in Coq) followed by the development of significant mathematical libraries (for universal algebra in this case) suggesting the next round of improvements of the logical framework and its proof infrastructure.

Frédéric Blanqui and Adam Koprowki's paper presents a comprehensive survey of the CoLoR library for well-founded relations within the Coq proof assistant. Such rewrite relations occur both as an implementation of canonical tactics for equality reasoning in algebraic theories where a canonical presentation exists and as functional programs over algebraic datatypes. The termination of such systems of rewrite rules is essential for the meta-theory of equational rewriting, and, in particular, its decidability. The authors give a survey of termination techniques. They describe the notion of a *termination certificate* allowing the verification of a termination argument given in this format. They discuss a compiler of such certificates into a Coq proof, called Rainbow, and compare it with the competing CiME3/Coccinelle and CeTA/IsaFoR systems.

Russell O'Connor's paper discusses the use of classical mathematics for constructive logic, a somewhat paradoxical proposal. There are well-known methods for embedding classical theorems in a constructive setting using a variety of translation techniques, such as Gödel's double negation interpretation, or by adding axioms such as excluded middle. In his paper, O'Connor goes further by showing that classical results can be applied to constructive mathematics without additional axioms. More precisely, he shows that dependent types allow the translation of classical definitions into constructive logic using monadic methods. He shows several applications of this general technique in the framework of the Coq proof assistant.

The third theme of this special issue, foundations and metatheoretic issues, is covered by the papers by: Mihnea Iancu and Florian Rabe; and Robin Adams and Zhaohui Luo. The former discusses the formalisation of mathematical foundations, and the latter the inter-operability of logical frameworks through mechanical translation of their proof scripts.

Mihnea Iancu and Florian Rabe discuss the formalisation of mathematical foundations. More precisely, they investigate how to represent three important foundational systems, namely, Isabelle/HOL, Mizar and ZFC set theory, within a common framework derived from Edinburgh's logical framework, as implemented in the Twelf platform. This allows them to investigate the embeddings of one foundation into another in a constructive setting that allows the mechanical extraction of the translation functions between the various foundational systems. This opens the way to reusing formal developments originating in one system in another system, as well as for experiments in cooperative efforts to develop a large body of modular mathematics using multiple platforms based on distinct but equally consistent foundations.

Robin Adams and Zhaohui Luo's paper, *A pluralistic approach to the formalisation of mathematics*, shows how proof scripts developed in one logical framework can be reused in another framework using Gambino and Aczel's notion of logic enriched type theories (LTT). They show how sound translations between LTTs allow a direct transfer of a proof development from one foundational framework to another using a translation library. They provide several case studies of such translations, including Gödel's double negation translation and Friedman's A-translation.

The papers included in this special issue are fairly representative of the major strands of research in the fast-developing fields of interactive theorem proving and the formalisation of mathematics. They show complementary methods and techniques, but also address interoperability concerns between the various proof platforms, which will be needed for the cooperative development of certified mathematics. The field is now mature enough to contribute to the certification of applied mathematics algorithms used in engineering. All this work points to the possibility of using such formal developments for the mechanical extraction of computational systems fit for safely critical applications, which is a crucial concern of modern engineering in many fields of human activity. Given this, we can foresee a future role for computational logic in bridging the gap between pure mathematics and the technologies essential to industry and services.