Mathematics, Linguistics and Types

Gérard Huet

Théories modernes des types

IHPST, Paris, Mars 2006

Prelude

Thanks for IHPST ("Institut d'Histoire et de Philosophie des Sciences et des Techniques") for inviting us to a debate on "Théories modernes des types" – "Contemporary Type Theory".

I was invited to present "Calcul des Constructions" – "Calculus of Constructions" but I declined. Not that "Calcul des Constructions" is uninteresting, but first of all it is a rather technical topic, its precise description and statement of its main properties would take more than one hour, and this would be a rather futile repetition of stuff which is already 20 years old. Moreover its design methodology is more interesting, as a formal framework for expressing Mathematics, and more generally rigorous argumentation.

Frameworks

Such a framework may be implemented on computers, and thus hopefully lead to useful rigorous reasoning tools. This is basically the Automath program, now more than 30 years old, but still the leading paradigm in this endeavour. The Calculus of Constructions is one point in a space of type theories, one corner in Barendregt's cube, a bit stronger than usual Automath languages, maybe unnecessary stronger, and actually not expressive enough to express two fundamendal constructions, namely inductive types and co-inductive types. From the Calculus of Constructions evolved a Calculus of inductive and co-inductive constructions, which is the core formalism manipulated by a reasoning tool called Coq. Coq is a proof assistant, which helps a mathematician formally verify, and to a certain extent discover proofs of mathematical facts, such as the 4 colour theorem.

Social necessity

The interesting philosophical discussion concerns the soundness of this program, regarding its epistemic value, and notably its usefulness in establishing rigorous standards of reasoning. We first remark that reasoning tools are absolutely essential social requisites, at a time when most industrial, commercial, financial, legal, transportation, health, food, information, etc processes use sophisticated information systems, whose correct and safe operation can only be established through appropriately formal verifications.

Mathematics as winner in the Knowledge Economy

Verifying formally mathematical statements such as the 4-colour theorem appears as a luxury compared to these vital economic concerns. But the main lesson that we learned over many years of development of informatics is that not only arbitrarily deep mathematics are needed in order to do proofs of computer system correctness, but furthermore that mathematical proofs are the very fabric of these virtual artefacts that are computer processes. The endeavour of formal mathematics is a booming business. I am personally in the process of opening a Joint Research Laboratory between INRIA and Microsoft, of which one of the foremost projects concerns Automation of Mathematics. Companies like "Trusted Logic" pay proof engineers to craft mathematical developments insuring the safe operation of smartcard transactions - bank consortia are paying dear money for the production of formal mathematics.

Mathematics and Linguistics

I chose as title of my talk "Mathematics, Linguistics and Types" since I would like to suggest that Type Theory is a better linguistic medium for developing Mathematics than the usual mixture of natural language and formulas that is traditionally used for communicating mathematical results. This point about a proper "mathematical vernacular", clearly articulated by Pr de Bruijn a long time ago, is not well taken by most professional mathematicians, who undervalue linguistic aspects underlying their activity. They see language as a benign facility, and fail to appreciate the importance of notation, seen as a secondary issue. We were taught that mathematics is "l'art de raisonner juste avec des figures fausses". Linguistics is taken as granted, like prose for Mr Jourdain. Who will dare comparing as intellectual disciplines Mathematics and Linguistics, a kind of semi-scientific branch of Humanities ?

Questioning the rigour of Mathematics

I am actually painfully conscious of this social handicap, since I attempt to teach Computational Linguistics to snotty students of Ecole Normale Supérieure majoring in Maths. The various course instructors attempt to get their attention by presenting the contents of their material in a heavy mathematical fashion. I chose in the contrary to get their attention by questioning the rigour of Mathematics by pointing out linguistic problems.

A few paradoxes

It is very easy to run into paradoxes. For instance, let us prove that $1 = 2$. 1 is the numerator of $1/2$. Similarly, 2 is the numerator of $2/4$. But we know that $1/2 = 2/4$, by easy simplification. Substituting equal for equal, we get $1 = 2$. If you carry out this piece of reasoning in front of a mathematician, he will usually get angry and shout some 'obvious' objection. However, such objections vary a lot according to the person, leading to doubts as to whether similar problematic implicit quotienting, or confusions of 'signifiant' and 'signifié', or mixture of extensional notions with intentional ones, could crop up in a real mathematical text. For instance, it is easy to prove categoricity of arithmetic using induction over the characteristic predicate of the standard part of an arbitrary model. This proof is illegitimate since this predicate is not first-order definable, but you need to be a professional logician to have in mind such subtelties.

Even simpler linguistic paradoxes

Such paradoxes are not deep foundational problems. They arise from a mixture of natural language and informal mathematical notation. Naming, scoping, and quantifiying mathematical notions is a delicate matter. Even a seemingly innocuous piece of reasoning such as $P \Rightarrow P$ may lead to catastrophy when P is instanciated. Consider English, when P is taken as 'any number is odd'. We get 'if any number is odd, then any number is odd', and then from the fact that 1 is odd we may deduce that any number, such as 2, is odd.

More notational paradoxes

We know that we should not write $1/0$. But when one does algebraic reasoning, we shall write down formulas such as a/b, where a and b may be complex expressions depending on some parameters. How do we know whether b may be null for certain values of these parameters ? How do we keep track of such side conditions ? In linguistic terms, the fact that $b \neq 0$ is a *presupposition* of the sentence where a/b occurs. Thus the logical argumentation underlying the verification of a piece of mathematics ought to take care of *présupposé* as well as *posé*, and there is no systematic convention to take care of the former – such contextual conditions are somehow implicit from the use of certain notations. A notation like $\sqrt{e}$ will have the additional problem of being ambiguous whenever it is not meaningless, hiding a logical disjunction.

Worse notational paradoxes

Actually, set notation itself suffers from complex conditions of usage - what is the precise condition under which one may write an expression such as $\{x \mid P(x)\}$? Do we need to understand the precise comprehension axioms of the underlying set theory in order to understand whether such notation makes sense, and if so, which precise object it yields ?

What is the meaning of $\aleph_1$ in view of the independence of the continuum hypothesis ?

Logic comes in

In Mathematics, Linguistics and Types, we miss two important topics: Logic and Informatics. Somehow Type Theory stands for an intimate mixture of the two. Logic has a long history - from Philosophy, to Mathematics, to Informatics. Of course there remains a current of philosophical logic, and an active field of Mathematical Logic, but somehow the logic mainstream, specially Proof Theory, cannot be dissociated from Informatics. Logical tools such as lambda-calculus and linear logic are the main objects of design and study of Informatics, together with a host of algebraico-logical notions which percolated from the study of computer processes, such as finite automata and transducers, streams, continuations, processes, games, etc.

From Language to Logic and back

Such tools are now finding appropriate use in the modelling of linguistics, at all levels – morphology, syntax, semantics, pragmatics. This interplay between disciplines is indeed very important, and it goes both way – logic and language use the same cognitive apparatus and are co-referential as ways of creation and communication of valid knowledge.

Prehistorical digression

In the old days, when computers appeared, a new discipline arose, called "Computer Science" in the USA. There was a tension between Mathematics and Electrical Engineering on who would teach the new notions, and the new discipline organized itself in more or less ad hoc ways. Mathematicians chose to isolate themselves from what they saw as technological contamination, and consequently there was a deficit in proper mathematical training. Students going through the curriculum of undergraduate computer science studies were hopelessly poor in basic reasoning skills like quantification manipulation or induction. Even if they had a reasonable training in Calculus, they lacked proper acquaintance with Algebra and Combinatorics.

My first encounter with Logic

My own experience. I was trained as an aeronautics engineer, and thus was taught a fair amount of complex analysis and linear algebra. In parallel I followed the first Master curriculum established in Informatics at the Faculté des Sciences of the University of Paris when it still existed, back in 1967. The mathematical apparatus was completely different, and incredibly ill-adapted to what it pertained to bring, namely rigour in reasoning. There was a course in mathematical logic, let me be mercyful and not say anything about it.

My first encounter with Recursion

There was also a course in recursion theory, thanks to Bernard Jaulin, which had the merit of being rigorous, and had at least some vague plausibility as foundations for recursive computations over integers.

Algebra wins

Then there was Schützenberger's fascinating course on advanced combinatorics (Ramsey numbers, monoid equations).

Tale of an embarrassing mismatch between Algebra and Logic.

Also Schutz's association with Roger Lyndon gave me the hint to look up the "Notes on Logic" by this author, which opened my eyes about logic by providing a very clear algebraic presentation of the important material in 90 pages - in my opinion an unsurpassed introduction to logic to this date.

Weird logical tautologies

In a standard mathematical logic textbook, first order models are taken as non empty sets. This important convention is told "en passant", much like excluded middle is wired in at the start of propositional calculus. This is a convenience for the standard inference rules of first order quantification. I like to tease mathematician friends with the remark that $\forall x\ P(x) \Rightarrow \exists x\ P(x)$. If they give a moment's thought to this remark, they immediately come with the empty set as a counter example. When I insist that this is a valid statement in the standard predicate calculus, they usually get annoyed and nasty.

Mathematicians despise Logic

Actually, most mathematicians don't care about logic. Mathematical logic grew out of consistency analysis, following Hilbert's program, and other meta-mathematical considerations. Gödel's incompleteness results were a blow to the whole mathematical edifice, but this crisis was overcome like the previous ones, such as irrationality and transcendance. Mathematical logic became a highly specialised subfield of mathematics, far from being as prestigious as algebraic geometry or number theory. It even got ridiculed by Paul Cohen, who considered the independence of the continuum hypothesis as a rather simple exercise - obtaining forcing as a kind of reverse engineering construction.

Drinking problems

Gilles Kahn made a point of debunking standard logic in Coq's tutorial with the "Drinker's theorem". Statement. In any bar, there is a person who, if he drinks, every one in the bar drinks. Proof. If everyone drinks, take any one. If someone does not drink, he is a witness to the truth of the statement. However, we have 2 difficulties. First, how do we know there are no other situations ? Answer: by excluded middle. Second, what happens when the bar closes and is therefore empty ? So there are many hidden hypotheses which are implicit in the linguistic formulation of the problem and its solution.

From bars to rings

This "theorem" is a pearl, since it mixes in a minimal way three difficulties in the standard apparatus - models are non empty, excluded middle is wired in, and material implication is used instead of entailment. So you get a formal proof which is easily defeated by common sense. With an ironical phrasing in terms of lowly concepts such as drinking in a bar, whereas mathematics allows itself the use of concrete terminology like rings, fields and matrices, but always with noble connotations !

But perhaps one might question the use of real-world terminology at all. What helps intuition is also a danger of introducing implicit assumptions. See the work by Lakatos for an interesting discussion.

Type Theory commits less than Set theory

Here we see a very clear instance of the superiority of type theory as a linguistic medium for reasoning over the standard abstraction of first-order logic. The bar has to be postulated non-empty, since otherwise there is no way to get the witness. Similarly if you do not postulate excluded middle you will not get anywhere. Type theory is not fundamentally different from predicate calculus, it is just less commital in terms of basic means of knowledge.

Type Theory commits more than Linear Logic

Type Theory is however not minimal in its use of basic principles, since it allows non-linearity, both for contraction (axioms are not consumed, they are reusable) and for thinning (irrelevant assumptions are discharged). This is because it uses lambda-calculus, and not linear logic as its foundation.

The epistemic justification for this choice lies in two basic postulates about knowledge: it can be memorized and reused, while irrelevant knowledge may be ignored. Further properties such as insensibility to temporal ordering of independent pieces of knowledge are derived metaproperties (combinators).

Lambda calculus

There is no good understanding of type theory if you do not know lambda-calculus. Type theory constructions are typed lambda-terms, the types being some logical formulas themselves formulated in lambda-calculus. Lambda-typed lambda terms, that was from the start the motto of Automath. The rest is details about formation rules for those types. But the main outcome of this endeavour is to get a full language for mathematics, including abstraction, axioms, definitions and proofs. Furthermore those proofs are not only concrete objets, but executable algorithms as well, paving the way to sophisticated mixtures of computation and deduction, and ultimately powerful uses of reflexion principles.

Lambda calculus was slow in percolating

When one looks back into the work of Gentzen in the 30's, one can find lambda-calculus in properly presented natural deduction, of course, and sequent calculus may be seen as the search space for such proofs. But this story took a long time to unravel through the work of Kleene, Curry, Howard, de Bruijn, Martin-Löf, Girard, etc. Even though Church's notation was well-known by professional logicians, Prawitz' book on natural deduction does not even allude to lambda calculus. As late as 1980, lambda calculus specialists were a sect of less than 20 members worldwide. Even today, I would bet there are not 50 people who are able to state Böhm's separation theorem correctly, let alone prove it.

Denotational semantics of programming languages

Lambda-calculus actually came to informatics through the development, in the 80's, of denotational semantics by Dana Scott and his followers. This fueled a lot of not-so-relevant research in continuous lattices, and ended up cluttering constructive reasoning with an unsavoury dose of abstract nonsense - category theory imposed by intimidation rather than necessity for proper abstraction.

Hyper specialization hinders progress

Actually, the influence of lambda-calculus grew in at least two areas: design of programming languages (Peter Landin for Algol 60, then ISWIM, the precursor of ML), and semantics of natural language (Richard Montague, who applied Church's simple theory of types to the propositional semantics of language). But it took a long time to recognize that the categorial grammatical framework of Lambek was a non-commutative precursor of linear logic. This is a clear case of hyper-specialization of disciplines, missing important common conceptual apparatus.

In search of a language for formal maths

The search for a perfect reasoning language by Leibniz got only grotesque modern avatars by Frege's graphical notation and Russell and Whitehead absurdly opaque notation for Principia Mathematica. De Bruijn's Automath notation, in the 60's, was both too early and too late. Too early to be properly understood before Type Theory became popular in proof assistants research circles. Too late to serve as language for Bourbaki's endeavour. Bourbaki knew about lambda-notation, and even contemplated using it as mathematical notation, but they failed to capture the lambda abstraction with two-dimensional diagrams and gave up, indulging in a set-theoretic semi-rigorous dialect.

Type Theory is better than Axiomatic Logic

We are not so much interested in axiomatisation as we are in actual definition of concepts (constructions of sorts). In Goldblatt's words, in his preface to his book on Topoi: "[We are interested in] understanding the house that we mentally build for ourselves to live in".

A most important mathematical concern is the art of writing definitions in the right way, as well as the craft of knowing when to allude to a definition - we are not interested in cut-free proofs, heaven, this is a logician device for consistency proofs, not a mathematically sensible operation.

We do not want cut free, we want clutter free.

Methods rather than facts

The less we assume, the more opportunities we get of discovering interesting methods. A case in point is the Continuum Hypothesis. What is interesting is not so much that it can be taken or discarded, but it is the forcing method in itself, a very powerful construction.

Actually, as Paul Cohen says "A point of view that the author feels may eventually come to be accepted is that the CH is obviously false." This sounds like an interesting research program: find the right induction principle to *refute* CH.

Free variables, arbitrary choices of points, quantification

Perlis' law. Someone's free variable is someone else's bound variable.

Logical interpretation of this law : These are concerns of lambda calculus, which deals correctly with binding, abstraction, functional notation, and hypothesis management. Furthermore, lambda reduction gives a general solution to solving definitional equality replacement by higher-order substitution.

Of course lambda reduction is only one kind of definitional equality. What is equally important is *recursion*, which comes with inductive types.

The equality quagmire

What on earth does $x = y$ mean ? The number of interpretations boggles the mind. Leibniz equality like in Church's simple theory of types, with substitutivity for free by lambda reduction, extensional equality for sets, abstract equality over a structure, leading to setoids, etc.

If one uses the right constructions, category theory can be accommodated conservatively, as Amokrane Saibi showed.

Category theory tried to treat concepts modulo isomorphism. Then you wind up with statements like: "it is not just equal, it is equal on the nose". Why should the nose of a category theorist matter for mathematical truth?

Right concerns about modularity

We do not want vague overloading, we want mathematical modules, which compose in well understood ways, in the lambda-calculus tradition.

Wouldn't it be nice if we could *define* what mathematicians mean by Algebra, Geometry, Arithmetic, Topology, Measure theory, Analysis, Probability theory. What is Combinatorics the natural composition of ? What part of Analysis is needed to extend Arithmetic into Number theory ? Is Category theory is some sense at the root of mainstream mathematics, or is it Set theory or some well-understood combination of both ?

This would put Logic in its right corner, as some contemplation of Galois connections between syntactic combinatorial structures and set theoretic models.

Logic is too simplistic

Logic is to mathematics what logical semantics is to linguistics, a poor approximation of meaning. Furthermore we want words to be associated not just with meanings, but with references, and this is where definitions matter.

If you use notation like $\{x|P(x)\}$ or even a/b or $\sqrt{e}$, you do not want to express just the stated fact, but the presupposed assumption as well. We need an Automath context to take care correctly of side conditions.

Modern Type Theory as a vehicle for serious Maths

The recent completion of the Four Colour Theorem as a formally verified piece of Type Theory sets the state of the art in formal mathematics. It uses a slick axiomatisation of hypermaps and represents a significant improvement over the proof by Robertson et al., itself an improvement over a messy mixture of maths and assembly programming by Appel and Haken.

As Georges Gonthier states : "We believe that our success was largely due to the fact that we approached the Four Colour Theorem mainly as a *programming* problem, rather than as a *formalization* problem".

From programs-for-proofs to programs-as-proofs

Towards a true Mathematician assistant

Mixing the why and the how. A high-level tactic language.

Note on Proof assistants versus Logical frameworks.

Putting it all together

Now we know how to model correctly the semantics of natural language through *continuations*, a notion imported from denotational semantics of programming language, and which generalizes Richard Montague's seminal semantic work in linguistics. Since continuations are basically typed lambda terms, one may dream of formal mathematical developments actually developed through a natural language discourse, using the naturally developed cognitive apparatus of the human mathematician.

Then we shall get a true mathematical consistent discourse and the proper tools to search for interesting mathematical stories.