HAL
archives-ouvertes.fr

A Type-Free Formalization of Mathematics where Proofs are Objects

Gilles Dowek

▶ **To cite this version:**

Gilles Dowek. A Type-Free Formalization of Mathematics where Proofs are Objects. [Research Report] RR-2915, INRIA. 1996. <inria-00073782>

HAL Id: inria-00073782
https://hal.inria.fr/inria-00073782

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE

A type-free formalization of mathematics where proofs are objects

Gilles Dowek

N° 2915

Juin 1996

PROGRAMME 2

Calcul symbolique,
programmation
et génie logiciel

ISSN 0249-6399

INRIA
ROCQUENCOURT

A type-free formalization of mathematics where proofs are objects

Gilles Dowek *

Programme 2 — Calcul symbolique, programmation et génie logiciel

Projet Coq

Rapport de recherche n° 2915 — Juin 1996 — 24 pages

Abstract: We present a first order untyped axiomatization of mathematics where proofs are objects in the sense of Heyting-Kolmogorov functional interpretation. The consistency of this theory is open.

Key-words: formalization of mathematics, set theory, functionnal interpretation of proofs, reflection, Gödel's theorem, Tarski's theorem

(Résumé : tsvp)

*Gilles.Dowek@inria.fr

Unité de recherche INRIA Rocquencourt

Une formalisation des mathématiques non typée dans laquelle les démonstrations sont des objets

Résumé : On présente une axiomatisation des mathématiques en logique du premier ordre non typée dans laquelles les démonstrations sont des objets au sens de l'interprétation fonctionnelle de Hetying et Kolmororov. La cohérence de cette théorie est ouverte.

Mots-clé : formalisation des mathématiques, théorie des ensembles, interprétation fonctionnelle des démonstrations, réflexion, théorème de Gödel, théorème de Tarski

Introduction

As mathematical truth is not decidable, mathematical statements need to be proved. In the usual formalizations of the language of mathematics, proofs are sequences of propositions produced by some deduction rules. Such proofs are not terms of the language i.e. not elements of the universe of the discourse. Without going back to ancient greeks for whom the universe of the mathematical discourse contained only natural numbers, this situation can be compared to that of functions that were used, but not considered as objects before the seventeenth century or sets that were not considered as objects before the nineteenth century.

In contrast, in natural languages, the truth of a proposition can be justified by a complement. For example, the truth of the proposition "there are natural numbers such that $x^2 + y^2 = z^2$" can be justified by adding a complement "there are natural numbers such that $x^2 + y^2 = z^2$, e.g. $3, 4, 5$". In contrast, the truth of this proposition must be decided by some algorithm to avoid endless justifications of justifications. It must be decided also by an algorithm that the first proposition is true as the second is.

To do the same thing in a formal way, we want to define a language with a symbol pr such that if P is a provable proposition, then there is a term t such that $t \in pr(P)$ is also a provable proposition and moreover the truth of this proposition can be established by a proof-checking algorithm. We do not need every provable proposition of the form $t \in pr(P)$ to be established by the algorithm, but we need that for each provable proposition P there is at least one term t such that the proposition $t \in pr(P)$ is established by the algorithm. Such a term t is called a *checkable proof-term*.

Having such proof-terms can be useful. For instance, if we skolemize the proposition

$$\forall x\ \forall y\ (y \neq 0) \Rightarrow \exists z\ x = y * z$$

we introduce a binary function symbol / and the proposition

$$\forall x\ \forall y\ (y \neq 0) \Rightarrow x = y * (x/y)$$

Although we cannot prove the proposition $1 = 0 * (1/0)$ we can form the term $1/0$. This term is usually considered as meaningless. But the meaningfulness of such a term may be difficult to establish in particular, as we cannot decide if a real number is 0 or not. If, in contrast, we skolemize the proposition

$$\forall x\ \forall y\ \forall p\ (p \in pr(y \neq 0)) \Rightarrow \exists z\ x = y * z$$

then we get a ternary function symbol and the proposition

$$\forall x\ \forall y\ \forall p\ (p \in pr(y \neq 0)) \Rightarrow x = y * /(x, y, p)$$

Again, we can form the meaningless term $/(1, 0, t)$ but now we can easily check that t is not a checkable proof of $0 \neq 0$ and thus that this term is meaningless [18].

Deciding if a term $/(x, y, p)$ is meaningful or not permits to decide if we get back x or not when we multiply it by y.

The same holds when we want to define a function on a quotient set. If f is a function from a set A to a set B and R is an equivalence relation on A, we can define a function $g = Quo(f, A, B, R)$ from A/R to B if the proposition

$$\forall x \in A\ \forall y \in A\ R(x, y) \Rightarrow f(x) = f(y)$$

is provable. Again, this term is meaningless when this proposition is not provable. Meaningfulness is decidable if we include the proof of this proposition in the term g, $g = Quo(f, A, B, R, p)$.

A last example is the choice operator (resp. descriptions operator). The term $C(A)$ is meaningful only when A is a non empty set (resp. a singleton). Providing a proof that A is nonempty (resp. a singleton), $C(A, p)$ permits to decide if such an expression is meaningful.

Giving such an existence proof permits also to define this choice operator. A proof of the existence of an object in A is a pair consisting in an object and a proof that this object is in A. Thus the chosen object

$C(A, < a, p >)$ can be a. This way, the choice operator can be defined as the function projecting a pair on its first component. Thus, there is no need to extend the language with a new operator because the choice operator can be defined in the language [22].

Having a choice operator taking a proof in argument is needed to program computers in the language of mathematics [21, 19, 24]. In this case, we want to have, as usual, the possibility to define a function using the choice operator. For instance, as we know that there exists a function f such that

$$(f\ 0\ m) = m$$

$$(f\ (S\ n)\ m) = (S\ (f\ n\ m))$$

we want to express it by the term

$$+ = C(\{f \in N \to N \to N \mid \forall m \in N\ (f\ 0\ m) = m \wedge \forall n \in N\ \forall m \in N\ (f\ (S\ n)\ m) = (S\ (f\ n\ m))\})$$

But, we want also to be able to execute this program, i.e. for instance to compute the term $(S\ (S\ (S\ 0)))$ from the term $(+\ (S\ (S\ 0))\ (S\ 0))$. This computation cannot be done when the choice operator takes only a set in argument, but it can when this operator takes also a (constructive) proof of the existence of an element in this set.

Such proof-terms are also useful in automated theorem proving. Trying to prove the proposition P can be reduced to searching a term t such that $t \in pr(P)$ is decided by the algorithm.

Having proofs as objects is also a natural way to construct truth predicates (see, for instance, [13]) taking $\exists y\ (y \in pr(x))$ for $\mathcal{T}(x)$ and thus to formalize indirect arguments through reflection. For instance we may want to state an axiom expressing that an undecided statement asserting the existence of some natural number verifying some decidable property is false.

At last, we could hope that some metamathematical results, like Gödel's incompleteness theorem, showing the existence of a true but unprovable proposition would be theorems in such a theory

$$\exists P\ (P \wedge \neg \exists x\ (x \in pr(P)))$$

(however, this is still not the case in the the simple approach suggested below, see the discussion in section 3.5.2).

1 State of the art

This idea of having proofs as genuine mathematical objects has already been developed in several ways.

1.1 Proof theory

An early occurrence of proofs considered as mathematical objects is the notion of proof as defined in Frege-Hilbert systems, and then in natural deduction and sequent calculus. Since Gentzen's work, such proofs have been extensively studied in *proof theory*. However, the goal of proof theory is to study the proofs of some theories in the usual language of mathematics, not to extend the language of mathematics by internalizing the proofs of the language itself. Thus, goals are differents, but tools can be shared.

1.2 Reflection

Another approach to proofs as objects comes from the proof of Gödel's incompleteness theorems. These proofs require the construction of a proposition $Proof$ in arithmetic such that $Proof(n, p)$ express that n is the Gödel number of a proposition A and p the Gödel number of a proof of A. Gödel numbers may be avoided if one considers a theory of trees instead as a theory of natural numbers [12]. But, in both cases, the proof terms (numbers or trees) are dependent of the way proofs are written. Two proofs differing by a renaming of bound variables are different objects, two proofs differing by the permutation of two steps are

different objects and two proofs differing by cut elimination are different objects. A proof t of a proposition P containing a free variable x is a closed term, thus to construct a proof of $P[x \leftarrow u]$ we cannot substitute x by u in t, but we must apply a function mimicking substitution at the level of the encoding. The same holds if the proof uses an hypothesis. Thus, although this encoding meets its goal and permits to prove incompleteness theorems, it does not respect proofs structure and it does not provide a simple and direct expression of proofs.

1.3 Proofs according to Heyting and Kolmogorov

Such a simple and direct expression can be provided by Heyting-Kolmogorov interpretation. As opposed to formal proof trees or Gödel numbers of proof trees, this interpretation of proofs as mathematical objects respects proof structure: two proofs differing by a renaming of bound variables are equal objects, two proofs differing by the permutation of two steps are equal objects and two proofs differing by cut elimination are equal objects. A proof of a proposition P containing a free variable x is expressed by a term t containing also this variable x and $t[x \leftarrow u]$ is a proof of $P[x \leftarrow u]$. A proof t using an hypothesis is expressed by a term containing a free variable x standing for a proof of this hypothesis and if u is a proof of this hypothesis, then $t[x \leftarrow u]$ is also a proof of P.

Heyting-Kolmogorov interpretation is only defined for intuitionistic proofs. This is not as much a restriction as it could seem because classical mathematics can be built within intuitionistic mathematics, for instance taking the excluded middle as an axiom or defining classical connectives by double negation.

Definition (Heyting-Kolmogorov interpretation)

- A proof of a proposition of the form $A \Rightarrow B$ is a function mapping any proof of A to a proof of B.
- A proof of a proposition of the form $\forall x\ A$ is a function mapping any object a to a proof of $A[x \leftarrow a]$.
- A proof of a proposition of the form $A \wedge B$ is a pair formed with a proof of A and a proof of B.
- A proof of a proposition of the form $\exists x\ A$ is a pair formed with an object a and a proof of $A[x \leftarrow a]$.
- A proof of a proposition of the form $A \vee B$ is either a proof of A or a proof of B.
- There is no proof of $\bot$.

The proposition $\neg A$ is an alternative notation for $A \Rightarrow \bot$, the proposition $A \Leftrightarrow B$ for $(A \Rightarrow B) \wedge (B \Rightarrow A)$ and the proposition $\top$ for $\bot \Rightarrow \bot$.

This interpretation has been used in proof theory to express proofs of various logical systems in typed lambda-calculi (Curry [9], Tait [26], Howard [17], Girard [14], Krivine and Parigot [19], etc.). It has also been used to formalize mathematics with proofs as objects (de Bruijn [4], Martin-Löf [22], Coquand and Huet [8], Paulin [23], etc.). In proof theory, proofs are encoded in a language of functions independent of the logical formalism (even if this formalism provides a language for functions) while in the formalization of mathematics proofs are encoded in the language of functions of the logical formalism itself.

1.4 Propositions as types v.s. propositions as sets

From Heyting-Kolmogorov interpretation, a proof of a proposition $A \Rightarrow B$ is a function mapping proofs of A to proofs of B. Thus, it is an element of the set of functions from the set of proofs of A to the set of proofs of B.

In proof theory where proofs are encoded in typed lambda-calculi or in typed combinatoric languages, but with no sets available, this has often been stated as the fact that proofs of $A \Rightarrow B$ have type $A' \rightarrow B'$ where A' (resp. B') is the type of proofs of A (resp. B). Thus propositions and types (i.e. propositions of the logical formalism and types of the lambda-calculus) have isomorphic structures. In formalization of mathematics with proofs as objects, propositions and types (i.e. propositions and types of the logical formalism) also have

isomorphic structure. This isomorphism is used to ensure decidability of proof-checking: the decidability of type-checking implies that of proof-checking.

In formalization of mathematics with proofs as objects, this propositions-as-types choice has however some drawbacks. In typed formalizations of mathematics (Whitehead and Russell [27], Church [5], etc.) types are syntactical devices used to restrict the formation of terms and propositions. Thus, because types play this double game, the relation between a proposition and one of its proofs cannot be expressed in the language. Indeed, if we had a symbol pr in such a language, the proposition $p \in pr(A)$ would either be well typed and true when p is a proof of A or ill-typed when it is not, thus we cannot express in the language a well-typed proposition expressing that some term is not a proof of a proposition. In such formalisms, the statement $p \in pr(P)$, usually written $p : P$, is expressed in language different of that of propositions. Moreover, using a typed formalization of mathematics introduces a distinction between the notions of type and set, that one may want to avoid [11].

To be able to express such a proposition, we propose in this paper an untyped formalization of mathematics where proofs are objects. Thus, the decidability of proof-checking will not be a property built in the language, but a fact proved *a posteriori*. Although, the language proposed here is untyped, and permits the expression of a symbol pr, it takes a lot from the typed languages proposed by de Bruijn [4], Martin-Löf [22], Coquand and Huet [8] and Paulin [23]. More precisely, this language can be seen as an untyped formulation of the Calculus of Constructions [8].

1.5 Beeson's theory C

Another source of inspiration of this language is Beeson's theory C [2] that also uses Heyting-Kolmogorov interpretation of proofs in an untyped setting. This theory introduces a modal operator $proof$ such that $proof(p, P)$ expresses that p is a proof of the proposition P.

A minor difference is that, in the language proposed here, propositions are objects, thus we do not need to introduce a modal operator, but a predicate symbol or a function symbol is enough.

Another minor difference is that the theory C uses a formalization of mathematics based on a partial combinatory language, i.e. functions have no explicit domain of definition and self application paradoxes are avoided by using a predicate $\downarrow$ for *denoting terms*, while we use a more common notion of function explicitly given with a domain of definition. The theory C also permits quantification over all the universe while our quantifiers are bounded.

A more important difference is that we provide a proof-checking algorithm recognizing the truth of some propositions of the form $t \in pr(P)$ and we show that whenever a proposition P is provable, there is a term t such that the proposition $t \in pr(P)$ is recognized by the algorithm (although not all true propositions of the form $t \in pr(P)$ are). As a consequence we can drop usual proofs rules and use proof-terms to justify, in practice, the truth of propositions.

At last, while in the theory C, a proof of the proposition $A \Rightarrow B$ is a function f from proofs of A to proofs of B *together with a proof that f is a function from proofs of A to proofs of B*, here, a proof of $A \Rightarrow B$ is merely a function f from proofs of A to proofs of B. For checkable proof-terms, the fact that t expresses a function from proofs of A to proofs of B must be recognized by the algorithm. If we want to justify the truth of a proposition P by a non checkable proof-term t justifying the proposition $t \in pr(P)$ by a checkable proof term u. Then we have to use the axiom $\forall x \in pr(P)\ P$. Call c the proof of this axiom, the term $(c\ t\ u)$ is a checkable proof-term of the proposition P.

Thus we can still use indirect arguments by proving that some object is a proof of some proposition, but the use of this axioms forces us to take into account that the justification of justifications must come to an end, where the soundness of the justification must be recognized by an algorithm.

2 A variant of set theory

We want to extend the language of mathematics to include a symbol pr such that $pr(A)$ is the set of proofs of A. The first candidate to such an extension is the most common formalization of mathematics, i.e. set

theory formalized with Zermelo's axioms or Zermelo-Fraenkel axioms. However extending set theory with a notion of explicit proofs requires a few minor modifications of set theory itself. This section is devoted to the presentation of such modifications. The language developed here is rather close to the untyped formulation of type theory developed in [11].

2.1 Set theory

Set theory is a first order theory in a language containing two binary predicate symbols $=$ and $\in$. Deduction rules can be any formulation of deduction, we use a formulation of natural deduction. We start with equality axioms: the identity axiom

$$x = x \tag{1}$$

and Leibniz scheme

$$a = b \Rightarrow P[z \leftarrow a] \Rightarrow P[z \leftarrow b] \tag{2}$$

Then come axioms expressing the existence of some sets. For instance, the power set axiom states that for each set x there is a set X such that the elements of X are the subsets of x

$$\forall x\ \exists X\ \forall y\ ((y \in X) \Leftrightarrow \forall z\ ((z \in y) \Rightarrow (z \in x)))$$

The union axiom and the pairing axiom are formulated in a similar way. The subset scheme (restricted comprehension scheme) expresses that for each set x we can build the subset of x containing the elements verifying the property P. Thus for each proposition P such that $z_1, ..., z_n$ are the free variables of P minus x we have the axiom

$$\forall z_1\ ...\ \forall z_n\ \forall a\ \exists y\ \forall x\ (x \in y) \Leftrightarrow ((x \in a) \wedge P)$$

Using this scheme we can, for instance, deduce the existence of the empty set

$$\exists y\ \forall x\ \neg(x \in y)$$

2.2 Existential axioms v.s. algebraic axioms

This scheme does not provide any notation for the empty set. This choice is not a very good one when we want to use set theory as a practical language to formalize mathematics. Indeed, we cannot express the proposition $\emptyset = \emptyset$ but only a proposition $\exists x\ ((x = x) \wedge (E\ x))$ where E is a characteristic property of the empty set. Moreover as we want to express proofs as objects, for instance a proof of the proposition $A \Rightarrow A$ as the identity function over proofs of A, we want a term id for this object to be able to express the proposition $id \in pr(A \Rightarrow A)$ and not $\exists y\ (y \in pr(A \Rightarrow A))$. Indeed, to ensure the decidability of proof-checking we want to use the information carried by the term id.

An explicit presentation of set theory is obtained by skolemizing the axioms. For instance, skolemizing the power set axiom introduces a function symbol $\mathcal{P}$ and an axiom

$$\forall x\ \forall y\ ((y \in \mathcal{P}(x)) \Leftrightarrow \forall z\ ((z \in y) \Rightarrow (z \in x)))$$

In this language, the power set of some set x is now written $\mathcal{P}(x)$.

Skolemizing the union axiom introduces a function symbol $\bigcup$ and a notation $\bigcup(a)$ for the union of the elements of a. Skolemizing the pairing axiom introduces a function symbol $\{,\}$ and a notation $\{a, b\}$ for the pair containing the object a and the object b.

Skolemizing the subset scheme introduces an infinite number of function symbols $h_{x,z_1,...,z_n,P}$ and the axioms

$$\forall z_1\ ...\ \forall z_n\ \forall a\ \forall x\ (x \in h_{x,z_1,...,z_n,P}(z_1, ..., z_n, a)) \Leftrightarrow ((x \in a) \wedge P)$$

We write $\{x \in a \mid P\}$ for the term $h_{x,z_1,...,z_n,P}(z_1, ..., z_n, a)$. Notice that the free variables of $\{x \in a \mid P\}$ are those of P minus x and those of a. In this language we can form the term $\{x \in a \mid P\}$ only when P contains no Skolem symbols. Nested abstractions can be built by applying the symbol $h_{x,z_1,...,z_n,P}$ to terms $b_1, ..., b_n$ containing Skolem symbols, but the variables free in these terms cannot be bound in P. This language is however equivalent to one with the full power of nested abstraction (see, for instance, [10]).

2.3 Functions as primitive objects

The next point concerns functions. In set theory, functions are defined as functional relations and relations as sets of ordered pairs. Thus to express a function, for instance the function square, we need to express first the set of pairs i.e. $G = \{< x, y > \in N \times N \mid x * x = y\}$ and then a proof of the proposition

$$\forall x \in N \ \exists_1 y \in N \ \ < x, y > \in G$$

Thus a function is expressed by a set and a proof. If, following Heyting-Kolmogorov interpretation, we want to express a proof of $A \Rightarrow A$ as a function mapping proofs of A to proofs of A, we express this proof as a function, i.e. as a set $G = \{< x, y > \in pr(A) \times pr(A) \mid x = y\}$ and a proof of the proposition

$$\forall x \in pr(A) \ \exists_1 y \in pr(A) \ \ < x, y > \in G$$

Thus we get a circular definition: a proof is expressed by a set and a proof. This circularity could be avoided if the proof-checking algorithm could establish by itself the functionality of G, but this seems to be rather difficult in the general case. In contrast, if we express the function above by the term $x \mapsto x$ or rather $x \in pr(A) \mapsto x$, then to establish that this term expresses a function of $pr(A) \to pr(A)$ we only need to prove the proposition

$$\forall x \in pr(A) \ x \in pr(A)$$

and this proposition can be established by the proof-checking algorithm (see section 4 below).

Here, we take a formalization of mathematics where functions are primitive objects, i.e. we axiomatize and not define what functions are. But, the important point its not that $\{< x, y > \in pr(A) \times pr(A) \mid x = y\}$ and $x \in pr(A) \mapsto x$ are different objects, it is that we need also the notation $x \in pr(A) \mapsto x$ for this object.

We first introduce a new function symbol α (for "apply"). We write $(f \ a)$ for $\alpha(f, a)$ and $(f \ a_1 \ ... \ a_n)$ for $(...(f \ a_1) \ ... \ a_n)$. There is a notational difference between the application of a function symbol f to a term a, written $f(a)$, and the "application" of a term a to a term b, written $(a \ b)$, that is an alternative notation for $\alpha(a, b)$.

Then we take a functional comprehension scheme

$$\exists f \ \forall x_1 \in a_1 \ ... \ \forall x_n \in a_n \ (f \ x_1 \ ... \ x_n) = t$$

Notice that to build a function, we need to give its domain $a_1, ..., a_n$, but we do not need to give its codomain. In set theory the codomain of such a function can always be constructed with the replacement scheme.

When we skolemize this axiom scheme we introduce function symbols $f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}$ where $z_1, ..., z_p$ are the free variables of t minus $x_1, ..., x_n$, and an axiom

$$\forall x_1 \in a_1 \ ... \ \forall x_n \in a_n \ (f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}(a_1, ..., a_n, z_1, ..., z_p) \ x_1 \ ... \ x_n) = t$$

We write $x_1 \in a_1 ..., x_n \in a_n \mapsto t$ for the term $f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}(a_1, ..., a_n, z_1, ..., z_p)$

Then we need to axiomatize the notion of function space. We introduce a new function symbol $\to$ to construct such function spaces. We write $a \to b$ for $\to (a, b)$ and $a_1 \to ... \to a_{n-1} \to a_n$ for $a_1 \to (... \to (a_{n-1} \to a_n)...)$. We take the axioms

$$(f \in A \to B) \Rightarrow (a \in A) \Rightarrow (\alpha(f, a) \in B)$$

$$((\forall x_1 \in a_1 \ ... \ \forall x_n \in a_n \ t \in b) \Rightarrow f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}(a_1, ..., a_n, z_1, ..., z_p) \in a_1 \to ... \to a_n \to b$$

Remark As shown in [10] we cannot iterate the definition of functions of one argument to build function of n arguments.

Remark Let B be the function from N to $\mathcal{P}(N)$ mapping a natural number x to the singleton $\{x\}$ and f be the function $x \in N \mapsto x$. We have

$$\forall x \in N \ (f \ x) \in N$$

From this proposition we can deduce $f \in N \to N$. But we also have

$$\forall x \in N\ (f\ x) \in (B\ x)$$

and we do not have any notation for the set of functions verifying this property. To express proofs, we want to give a name and a notation for this set. Thus, we extend the function symbol $\to$ to a function symbol Π and we take the axiom

$$f \in \Pi(A, B) \Rightarrow a \in A \Rightarrow (f\ a) \in (B\ a) \tag{3}$$

and the functional comprehension scheme

$$\exists f\ \forall x_1 \in a_1\ ...\ \forall x_n \in (a_n\ x_1\ ...\ x_{n-1})\ (f\ x_1\ ...\ x_n) = t$$

When we skolemize this scheme, we introduce function symbols $f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}$ and an axiom

$$\forall x_1 \in a_1\ ...\ \forall x_n \in (a_n\ x_1\ ...\ x_{n-1})\ (f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}(a_1, ..., a_n, z_1, ..., z_p)\ x_1\ ...\ x_n) = t \tag{4}$$

We also take the axiom

$$(\forall x_1 \in a_1\ ...\ \forall x_n \in (a_n\ x_1\ ...\ x_{n-1})\ t \in (b\ x_1\ ...\ x_n)) \Rightarrow f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}(a_1, ..., a_n, z_1, ..., z_p) \in \Pi(a_1, ..., \Pi(a_n, b)) \tag{5}$$

At last we take a functional extensionality axiom

$$\forall f \in \Pi(A, B)\ \forall g \in \Pi(A, B)\ (\forall x \in A\ (f\ x) = (g\ x)) \Rightarrow f = g \tag{6}$$

Now the symbol $\to$ is not needed anymore, the term $A \to B$ is an alternative notation for $\Pi(A, x \in A \mapsto B)$ where x is a variable not occurring in B.

Remark (Lambda-calculus as an abuse of notation) In this language we can form the term $x \in A \mapsto t$ only when t contains no Skolem symbols. Nested abstractions can be built by considering a term t applying the symbol $f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}$ to terms $u_1, ..., u_p$ containing Skolem symbols, but the variables free in these terms cannot be bound in t [10]. This language is however equivalent to one with the full power of nested abstraction. Each time we want to build the function $x \in A \mapsto t$, we first replace every abstractions $y_1 \in B_1, ..., y_n \in B_n \mapsto u$ in t by the term $((x \in A, y_1 \in B_1, ..., y_n \in B_n \mapsto u)\ x)$. This way x in not free in abstractions subterms of t' and the function $x \in A \mapsto t'$ can be constructed with the scheme above.

2.4 Propositions as objects

To form a term $pr(P)$ we need the statement P to be a term. Statements must be both terms and propositions of the language. Thus instead of taking $\in$ and $=$ to be predicate symbols, we take them to be functions symbols. This permits to have atomic propositions as objects. The case of propositions formed with connectors and quantifiers is studied below.

The expression $0 = 0$ is now a term and not a proposition. This term can be seen as the content (*lexis*) of the proposition.

We introduce also an individual symbol O for a set containing contents of propositions.

Then we need a unary predicate symbol ε to assert a proposition-term, i.e. to express that the proposition is indeed true. We take an extensionality axiom expressing that two equivalent propositions have the same content

$$((\varepsilon(x) \Leftrightarrow \varepsilon(y))) \Rightarrow \varepsilon(x = y) \tag{7}$$

Now, in the notation $\{z \in A \mid P\}$, P also is a term.

2.5 Sets as functions

When functions are primitive objects, sets can in turn be defined as their characteristic functions, i.e. sets need not be primitive objects anymore. Thus, the term $a \in B$ is now an alternative notation for $(B\ a)$ and $\{x \in A \mid P\}$ for $x \in A \mapsto P$.

Remark As functions are defined with a domain of definition, when we define a set A by its characteristic function $f \in B \to O$, we cannot say anything of $(f\ x)$ if x is outside of B. Thus we need a special axiom to state that elements outside of B are not in A (i.e. all the elements of A are in B).

$$(\varepsilon(a \in (B \to O)) \wedge \varepsilon(b \in a)) \Rightarrow \varepsilon(b \in B) \tag{8}$$

Remark Now the term $\mathcal{P}(A)$ is an alternative notation for $A \to O$. Notice that if A is an element of $B \to O$ and $A \neq B$ then $A \notin \mathcal{P}(A)$. Indeed, A cannot have both the domain A and B. But the set $|A| = x \in A \mapsto \top$ belongs to $\mathcal{P}(A)$ and with the axiom above we have $(x \in A) \Leftrightarrow (x \in |A|)$.

2.6 Connectors and Quantifiers

Taking $=$ and $\in$ as function symbol permits to have atomic propositions as objects. To have all propositions as objects we take new individual symbols $\wedge$, $\vee$, $\Rightarrow$ and $\bot$. To avoid confusion, from now on, we write $\bigwedge$, $\bigvee$, $\Rightarrow$, $\bot$, $\forall$ and $\exists$ for the true connectors and quantifiers of the language. Then we take the axioms

$$\varepsilon(\wedge \in (O \to O \to O)) \tag{9}$$

$$\varepsilon(\vee \in (O \to O \to O)) \tag{10}$$

$$\varepsilon(\Rightarrow \in (O \to O \to O)) \tag{11}$$

$$\varepsilon(\bot \in O) \tag{12}$$

$$\varepsilon(\wedge\ A\ B) \Leftrightarrow (\varepsilon(A) \bigwedge \varepsilon(B)) \tag{a}$$

$$\varepsilon(\vee\ A\ B) \Leftrightarrow (\varepsilon(A) \bigvee \varepsilon(B)) \tag{b}$$

$$\varepsilon(\Rightarrow\ A\ B) \Leftrightarrow (\varepsilon(A) \Rightarrow \varepsilon(B)) \tag{c}$$

$$\varepsilon(\bot) \Leftrightarrow \bot \tag{d}$$

Then, we want to take some symbols allowing to build contents of propositions built by quantification. In the languages of propositions quantifiers are unbounded, i.e. we can quantify over all the objects of the universe. But Heyting-Kolmogorov interpretation of proofs restricts the use of quantifiers to bounded ones. Indeed, a proof of the unboundedly quantified proposition $\forall x\ P$ would need to be a function mapping any object of the universe to a proof. The domain of this function would need to be the set of all objects and postulating the existence of such a set leads to known paradoxes. Thus, it seems that if we want to express proofs with Heyting-Kolmogorov interpretation, we have to banish the use of unbounded quantification.

Thus, we introduce a unary function symbol $\forall$ and axioms

$$\varepsilon(\forall(A) \in ((A \to O) \to O)) \tag{13}$$

$$\varepsilon(\forall(A)\ f) \Leftrightarrow \bigforall x\ (\varepsilon(x \in A) \Rightarrow \varepsilon(f\ x)) \tag{e}$$

Now the proposition $\forall x \in A\ P$ is an alternative notation for $(\forall(A)\ (x \in A \mapsto P))$.

In the same way, we introduce a unary function symbols $\exists$ and axioms

$$\varepsilon(\exists(A) \in ((A \to O) \to O)) \tag{14}$$

$$\varepsilon(\exists(A)\ f) \Leftrightarrow \bigexists x\ (\varepsilon(x \in A) \bigwedge \varepsilon(f\ x)) \tag{f}$$

and the proposition $\exists x \in A\ P$ is an alternative notation for $(\exists(A)\ (x \in A \mapsto P))$.

The proposition $\exists x \in A$ is an alternative notation for $\exists x \in A\ \top$.

Remark Not all propositions have contents, because the language of proposition uses unbounded quantifiers, while that of contents of propositions bounded ones.

Remark According to the Heyting-Kolmogorov interpretation, a proof of a proposition of the form $(\forall(A)\ B)$ would be a function f mapping any object a of A to a proof of $(B\ a)$. But if a is a term, to know if $(f\ a)$ is a proof of $(B\ a)$ we have to decide if a is in A or not. Thus a better choice is to take for proofs of $(\forall(A)\ B)$ a function mapping any objet a and any proof b of $a \in A$ to a proof of $(B\ a)$.

This choice is close to that of naïve Heyting-Kolmogorov interpretation. In such an interpretation, a proof of $(\forall(A)\ B)$ is a proof of $\forall x\ ((x \in A) \Rightarrow (B\ x))$ and thus a function mapping any object a and proof b of the proposition $a \in A$ to a proof of $(B\ a)$. The only difference is that we use the fact that quantification is bounded to give a domain to this function.

In the same way a proof of a proposition of the form $(\exists(A)\ B)$ is a triple $< a, b, c >$ where a is an element of A, b a proof of the proposition $a \in A$ and c a proof of the proposition $(B\ a)$.

Remark The connectors and quantifiers in the axioms (1) to (6) can be read as function symbols, thus we only need to add a ε predicate symbol in front of these propositions. Axiom (7) and (8) can be rewritten

$$\varepsilon((x \Leftrightarrow y) \Rightarrow (x = y)) \tag{7}$$

$$\varepsilon((a \in (B \to O)) \wedge (b \in a)) \Rightarrow (b \in B) \tag{8}$$

Remark The axioms above and below contain free variable. What is meant is the universal closure of these propositions with the universal unbounded quantifier (not the function symbol). Axioms seems to be the only place where unbounded quantification is required.

2.7 Pairing, ordered pairs, disjoint unions and numbers

In this theory, if we have a set A, we can build, its definable subsets, its power set $(A \to O)$, and if A is a set of sets (i.e. if it belongs to $(B \to O) \to O$ for some set B) we can define the union of its elements as a subset of B, i.e. as an element of $B \to O$.

If we compare this with Zermelo's set theory, only one construction is missing: the possibility to construct pairs $\{A, B\}$ with any two elements. Of course, if A and B already belong to a common set C then we can construct the subset of C of objects equal to A or B, but we cannot, for instance, construct the set $\{A, \{A\}\}$.

In the construction of mathematical objects, pairs are used at several places, first they are used to build ordered pairs $< a, b >= \{\{a\}, \{a, b\}\}$. Then, they are used to form unions of any to sets $A \cup B = \bigcup\{A, B\}$ and thus disjoint unions $A \oplus B = (\{0\} \times A) \cup (\{1\} \times B)$. At last they are used to form numbers $0 = \emptyset$, $1 = \{0\}$, $2 = \{0, 1\}$, etc.

Instead of taking this pairing construct, we rather take as primitive ordered pairing and disjoint unions, i.e. we take function symbols $\times$, $pair$, π, $\otimes$, i, j and δ and the axioms expressing the meaning of these symbols. In fact, like for dependent function space, we want dependent cartesian products. An element of $\Sigma(A, B)$ is a pair $pair(A, B, a, b)$ such that a is an element of A and b is an element of $(B\ a)$.

The term $< a, b >_{A,B}$ is an alternative notation for $pair(A, B, a, b)$. The term $A \times B$ is an alternative notation for $\Sigma(A, x \in A \mapsto B)$.

$$\varepsilon(\forall x \in A\ \forall y \in (B\ x)\ (< x, y >_{A,B} \in \Sigma(A, B))) \tag{15}$$

$$\varepsilon((f \in \Pi x \in A\ ((B\ x) \to C)) \Rightarrow \pi(A, B, C, < x, y >_{A,B}, f) = (f\ x\ y)) \tag{16}$$

$$\varepsilon((x \in A) \Rightarrow (i(A, B, x) \in (A \oplus B))) \tag{17}$$

$$\varepsilon((y \in B) \Rightarrow (j(A,B,y) \in (A \oplus B))) \tag{18}$$

$$\varepsilon(\delta(A,B,C,i(A,B,x),f,g) = (f\ x)) \tag{19}$$

$$\varepsilon(\delta(A,B,C,j(A,B,x),f,g) = (g\ x)) \tag{20}$$

Remark The condition $f \in \Pi x \in A\ ((B\ x) \to C)$ in axiom (16) above defines the so-called *weak dependent cartesian product*. Indeed, we can build the function mapping an element of $\Sigma(A,B)$ to its first component by defining $f = x \in A, y \in (B\ x) \mapsto x$, proving $f \in \Pi x \in A\ ((B\ x) \to A)$ and defining the function $\pi_1 = a \in \Sigma(A,B) \mapsto \pi(A, x \in A \mapsto B, A, a, f)$. But we cannot construct the function mapping an element of $\Sigma(A,B)$ to its second component because $g = x \in A, y \in (B\ a) \mapsto y$ fails to be in a set $\Pi x \in A\ ((B\ x) \to C)$. In contrast we can build the function mapping an element of $A \times B$ to its second component by defining $g = x \in A, y \in B \mapsto y$, proving $g \in A \to B \to B$ and defining the function $\pi_2 = a \in A \times B \mapsto \pi(A, x \in A \mapsto B, B, a, g)$.

Remark The term $\pi_1(A,B,a)$ is an alternative notation for $a \in \Sigma(A,B) \mapsto \pi(A, (x \in A \mapsto B), A, a, (x \in A, y \in (B\ x) \mapsto x))$.

Remark The term $< a, b, c >_{A,B,C}$ is an alternative notation for

$$< a, < b, c >_{(B\ a), y \in (B\ a) \mapsto (C\ a\ y)} >_{A, x \in A \mapsto \Sigma((B\ x), y \in (B\ x) \mapsto (C\ x\ y))}$$

and the term $\pi'(A,B,C,D,e,f)$ for

$$\pi(A, x \in A \mapsto \Sigma((B\ x), y \in (B\ x) \mapsto (C\ x\ y)), D, e,$$
$$(x \in A, z \in \Sigma((B\ x), y \mapsto (C\ x\ y)) \mapsto \pi((B\ x), y \in (B\ x) \mapsto (C\ x\ y), D, z, (f\ x))))$$

We have

$$f \in \Pi x \in A\ \Pi y \in (B\ x)\ ((C\ x\ y) \to D) \Rightarrow \pi'(A,B,C,D,< a,b,c >, f) = (f\ a\ b\ c)$$

Remark To construct natural numbers, we consider an infinite set I of atoms, i.e. a set having a non surjective injection.

$$\varepsilon(\exists z \in I\ \exists s \in I \to I\ (\forall x \in I\ (\neg((s\ x) = z))) \wedge \forall x \in I\ \forall y \in I\ ((s\ x) = (s\ y) \Rightarrow (x = y)))$$

When we skolemize this axiom we get

$$\varepsilon((\forall x \in I\ (\neg((s\ x) = z)))) \tag{21}$$

$$\varepsilon(\forall x \in I\ \forall y \in I\ ((s\ x) = (s\ y) \Rightarrow (x = y))) \tag{22}$$

We can then either construct natural numbers as finite cardinals in $(I \to O) \to O$ or take numbers as atoms, defining n as being the object $(s\ ...\ (s\ z)...)$ and thus the set of numbers as the smallest set containing z and closed by s.

2.8 A first order theory

The axioms (1)-(22) and (a)-(f) above define a first order theory in the language containing the only predicate symbol ε and the function symbols $=$, O, $\wedge$, $\vee$, $\Rightarrow$, $\bot$, $\forall$, $\exists$, Π, α, $f_{(a_1,...,a_n),(z_1,...,z_p),(x_1,...,x_n),t}$, Σ, $pair$, π, $\oplus$, i, j, δ, I, z, s.

As usual, when functions are primitive and not defined as sets of pairs, the comprehension scheme alone does not provide enough functions to formalize mathematics, the description axioms (or the axiom of choice) needs to be added (see section 3.4).

2.9 Consistency

The relative consistency of the axioms above with respect to Zermelo-Fraenkel set theory seems to be easy to establish. Interpreting the set O by $\{0,1\}$ and following the usual definitions of function spaces, cartesian products, disjoint unions and numbers in set theory.

The differences between this theory and Zermelo-Fraenkel set theory concern three points.

- the axioms are skolemized,
- the predicate symbols $=$ and $\in$ are decomposed into two function symbols $=$ and $\in$ and a predicate symbol ε and consequently a set O is introduced and connectors and quantifiers are replicated as function symbols,
- function spaces, cartesian products, disjoint unions and numbers are axiomatized and not defined thus we can drop the pairing axiom, the replacement scheme and have a slightly weaker union axiom.

The third point is in some sense optional. We could take the Zermelo-Fraenkel axioms instead and define these notions as usual in set theory, provided we extend the language to be allowed to write $x \in N \mapsto x * x$ and not only $\{< x, y >\in N \times N \mid y = x * x\}$ (it is well-known that there is little differences between defining some notions and axiomatizing them first and then proving consistency by constructing a model.) However this permits to formalize mathematics with weaker axioms than Zermelo-Fraenkel and as we shall see to avoid some paradoxes when we add proofs as objects.

3 Proofs as objects

3.1 The symbol *pr*

We add a function symbol pr that maps the contents of propositions to the sets of their proofs. We extend the comprehension scheme to instances containing the symbol pr. We take the axioms:

$$\varepsilon(pr(P \Rightarrow Q) = pr(P) \to pr(Q)) \tag{23}$$

$$\varepsilon(pr(P \wedge Q) = pr(P) \times pr(Q)) \tag{24}$$

$$\varepsilon(pr(P \vee Q) = pr(P) \oplus pr(Q)) \tag{25}$$

$$\varepsilon(pr(\bot) = \emptyset_I) \tag{26}$$

$$\varepsilon(pr(\forall(A)\ f) = (\Pi\ x \in A\ \Pi y \in pr(x \in A)\ pr(f\ x))) \tag{27}$$

$$\varepsilon(pr(\exists(A)\ f) = (\Sigma\ x \in A\ \Sigma y \in pr(x \in A)\ pr(f\ x))) \tag{28}$$

Example

$$(x \in O, y \in pr(x) \mapsto y) \in pr(\forall X \in O\ (X \Rightarrow X))$$

3.2 Leibniz scheme, equality of denotations and equality of meanings

In contrast with the comprehension scheme, we do not extend Leibniz scheme with instances containing the symbol pr. Indeed even if we can prove $a = b$ we do not want $p \in pr(P\ a)$ and $p \in pr(P\ b)$ to be equivalent propositions. We do not want every proof of $(P\ a)$ to be a proof of $(P\ b)$ because we want the proof of $(P\ b)$ to be built from the proof of $(P\ a)$ and the proof of $a = b$. This behavior of equality can be compared with the behavior of equality in some modal logics where from "The murder = Professor Moriarty" and "Sherlock Holmes knows (The murder = The murder)" we cannot deduce "Sherlock Holmes knows (The murder = Professor Moriarty)". This relation between modal logic and the provability operator has already been noticed in many ways [15, 2, 3]. A consequence is that equality cannot be interpreted as equality in a

model: even if a and b are two terms provably equal they are not always interpreted as the same object in the model. Equality is just a casual binary predicate symbol.

We may now introduce another symbol $\equiv$ for genuine equality, and axioms

$$\varepsilon(x \equiv x) \tag{29}$$

$$\varepsilon(a \equiv b \Rightarrow P[z \leftarrow a] \Rightarrow P[z \leftarrow b]) \tag{30}$$

Where P here can be any proposition, i.e. may contain the symbol pr.

The proposition $a = b$ expresses that the terms a and b have the same denotation while the proposition $a \equiv b$ expresses that they have the same meaning. Indeed if $a \equiv b$ then any proof of $(P\ a)$ is a proof of $(P\ b)$. In other words, two propositions A and B have the same meaning if what what is needed to be done to prove A is what is needed to be done to prove B and two terms a and b have the same meaning is $P[x \leftarrow a]$ and $P[x \leftarrow b]$ always have the same meaning.

If we only have the two axioms above, we can prove propositions of the form $a \equiv b$ only when a and b are the same term. But, if R is a decidable equivalence relation on terms such that if $a\ R\ b$ then $a = b$ is a provable proposition, then we can extend the equality of meanings by an axiom scheme $a \equiv b$ for each pair of R-equivalent terms keeping decidability of proof-checking (compare with the opposition internal equality/external equality in Martin-Löf type theory [22] or and in the Calculus of Constructions [8] and with Plotkin-Andrews program in automated theorem proving [25, 1]).

Consequently, the extensionality axioms do not jeopardize the intentional aspects of terms. The axiom

$$\varepsilon((x \Leftrightarrow y) \Rightarrow (x = y))$$

only states that if x and y are equivalent propositions, they have the same denotation, not that they have the same meaning.

3.3 Truth as provability

Now, we wish to be able to prove propositions by constructing an object that is a proof of this proposition. Also we wish to prove an implication $A \Rightarrow B$, proving B using not only the truth, but also the proof of A. Thus, we take the axiom "truth = proof" i.e.

$$\varepsilon(P \Leftrightarrow \exists x \in pr(P))$$

This axiom can de decomposed in two axioms: the *conecessitation* axiom

$$\varepsilon(\forall x \in pr(P)\ P) \tag{31}$$

and the *necessitation* axiom

$$\varepsilon(P \Rightarrow (\exists x \in pr(P)))$$

When we skolemize this axiom we introduce a new function symbol c and the axiom becomes

$$\varepsilon(P \Rightarrow (c(P) \in pr(P))) \tag{32}$$

Remark When proofs are encoded as Gödel numbers, if P is an unprovable proposition valid in the standard model of arithmetic, then the proposition

$$P \Rightarrow \exists x\ proof(x, \text{`}P\text{'})$$

is false in the standard model. Thus the standard model is not a model of the necessitation axiom. In other words the necessitation axiom implies the existence of non standard numbers for proofs of true but unprovable propositions (McGee [20] shows that even very weak conditions on a truth predicate imply ω-inconsitency and thus the loss of the standard model of arithmetic.) Here, it only implies the existence of non standard functions, i.e. more functions that can be proved to exist with the usual axioms.

Remark The use of dummy proofs $c(P)$ can jeopardize constructivity, i.e. the possibility to compute a value from any term. Thus, from a constructive point of view, we might want to drop this axiom.

3.4 The axiom of choice

The functional comprehension scheme permits to prove the existence of explicitly definable functions. Usually the descriptions axiom or the axiom of choice is added to be able to define, for instance, addition. When proofs are objects, the choice operator can take in argument, a proof of the existence of an element in A and this operator can be the first projection. But we still need an axiom to state that $\pi_1(p)$ has the right property

$$\forall p \in pr(\exists(A)\ P))\ ((\pi_1(A, x \in A \mapsto pr(x \in A) \times pr(P\ x), p)) \in A \\ \wedge(P\ (\pi_1(A, x \in A \mapsto pr(x \in A) \times pr(P\ x), p))))$$

(33)

To prove this proposition we would need the second projection [22] $(p \in pr(\exists(A)\ P) \mapsto < \pi_1\pi_2 p, \pi_2\pi_2 p >>)$.

3.5 Paradoxes

The consistency of this theory is open and can be doubted. We have two kinds of paradoxes that could jeopardize its consistency. First, the proofs as objects principle permits to construct too large sets or functions with a too large domains allowing to reproduce variants of Russell's or Burali-Forti's paradox. Then Gödel's incompleteness theorems and Tarski's undefinability theorem could be turned into paradoxes when we take the axiom "truth = proof".

3.5.1 Russell-like paradoxes

Remark The language presented in this paper is not polymorphic, because we cannot build, for instance the polymorphic identity, taking as argument a set A and an element x of A and giving back x

$$id = A \in ?, x \in A \mapsto x$$

We can build a function taking as argument a proposition A, a proof x of A and giving it back

$$id = A \in O, x \in pr(A) \mapsto x$$

but sets of the form $pr(A)$ are not all the sets, in particular, the set O in not one of them. This prevents a naïve encoding of the (inconsistent) polymorphic higher order logic [7].

Like in set theory, we can define $Refl(A, R)$ as an alternative notation for the proposition $\forall x \in A\ (R\ x\ x)$, and we can define the set of reflexive relations over a given set $\{R \in A \to A \to O \mid \forall x \in A\ (R\ x\ x)\}$ but there is no set of all reflexive relations.

Like in set theory, we can introduce an axiom stating the existence of a set C_0 containing I and O and closed by the usual operations, then a set C_1 containing I, O, C_0, etc. very closely to the predicative polymorphism universe hierarchy of [6].

Remark Above, we have given a presentation where function spaces, cartesian products, disjoint unions and numbers are axiomatized and not defined. We may wonder what could happen if we had kept Zermelo-Fraenkel axioms and defined such notions.

In that case, extending the replacement scheme with instances containing occurrences of the symbol pr permits to construct the set $\Omega = \{pr(x) \mid x \in O\}$.

Then the union axiom permits to form the set of all proofs $\Omega' = \bigcup(\Omega)$. If a is any object of the universe and p a proof of $a \in A$ and t a proof of $\top$, then the set Ω' contains an encapsulation $< a, p, t >$ of a as a proof of $\exists x \in A\ \top$ and this is enough to express Russell's paradox:

$$R = \{x \in \Omega' \mid x \notin \pi_1(x)\}$$

we have $R \in \mathcal{P}(\Omega')$, let p be a proof of this proposition. We have

$$< R, p, t > \in pr(\exists y \in \mathcal{P}(\Omega')\ \top) \in \Omega$$

thus

$$< R, p, t > \in \bigcup(\Omega)$$

$$< R, p, t > \in \Omega'$$

thus

$$< R, p, t > \in R \Leftrightarrow < R, p, t > \notin \pi_1(< R, p, t >)$$

$$< R, p, t > \in R \Leftrightarrow < R, p, t > \notin R$$

This justifies the need to weaken Zermelo-Fraenkel set theory, as we did above.

Remark In the language above, we have no way to construct the second projection mapping any ordered pair to its second element. If we could we would get an inconsistent system, because we could express Coquand's strong dependent cartesian product paradox [6].

Indeed consider the proposition

$$A_0 = \exists x \in O\ \exists R \in pr(x) \to pr(x) \to O\ \top$$

A proof of this proposition is a tuple formed with a proposition P an proof that P is a proposition, a binary relation R on proofs of P, a proof that R is a binary relation on proofs of P and a proof of the proposition $\top$. Any binary relation over the proof of any proposition can therefore be encapsulated into a proof of A_0 and with the second projection, from such a proof we can get back the proposition and the relation. This is enough to express Girard's paradox [14, 6].

Even if we weaken set theory as we did in section 2, the set of the proofs of the proposition $\exists x \in O\ \exists R \in pr(x) \to pr(x) \to O\ \top$ is large enough to express Coquand's strong dependent cartesian product paradox. There are two ways to avoid this paradox. Either, like in the Calculus of Constructions [8] and in this paper, we weaken the cartesian product to avoid the second projection, but this forces us to take the axiom of choice as an axiom, or, as it is done in Martin-Löf type theory [22], we avoid quantification over sets and relations. A less radical position might be to avoid quantifications over sets of proofs but not over all sets.

3.5.2 Gödel-like paradoxes

Gödel's first incompleteness theorem is often read "there is a true proposition that is not provable". Thus we may try to reproduce the proof of Gödel's theorem within this theory to prove the inconsistency of the axiom "truth = proof". We would construct a predicate G such that

$$(G\ P\ x) = \exists y \in pr(P\ x)$$

then we would construct H such that

$$(H\ x) = \neg(G\ x\ x)$$

and

$$A = (H\ H)$$

yielding

$$A = (H\ H) = \neg(G\ H\ H) = \neg\exists y \in pr(H\ H) = \neg\exists y \in pr(A)$$

hence with the axiom "truth = proof"

$$A \Leftrightarrow \neg A$$

which is contradictory.

This contraction can also be seen as a consequence of Tarski's theorem that there is no predicate $\mathcal{T}$ such that $P \Leftrightarrow \mathcal{T}(P)$. Again, we would construct a predicate G such that

$$(G\ P\ x) = \mathcal{T}(P\ x)$$

then we would construct H such that

$$(H\ x) = \neg(G\ x\ x)$$

and

$$A = (H\ H)$$

yielding

$$A = (H\ H) = \neg(G\ H\ H) = \neg\mathcal{T}(H\ H) = \neg\mathcal{T}(A)$$

hence

$$A \Leftrightarrow \neg A$$

which is contradictory.

In fact, these proofs do not go through as in the definition of G we need to give a domain to it. If we take, for instance,

$$G = P \in E \to O, x \in E \mapsto \exists y \in pr(P\ x)$$

and

$$H = x \in E \mapsto \neg(G\ x\ x)$$

applying H to H we get $((x \in E \mapsto \neg(G\ x\ x))\ (x \in E \mapsto \neg(G\ x\ x)))$ but to reduce this term to $\neg\exists y \in pr(H\ H)$ we first need to prove that $H \in E$ and this cannot be done.

Gödel's and Tarski's proofs rely on the existence of a proposition $Subst$ such that $Subst(\text{`}P\text{'}, \text{`}x\text{'}, \text{`}t\text{'}, \text{`}Q\text{'})$ is provable if and only if $Q = P[x \leftarrow t]$ thus the "function" $Subst$ permits to apply the "function" P to (the Gödel number of) any object. It is well-known that postulating the existence of functions defined on all the universe leads to paradoxes. In Gödel's and Tarski's proofs such a function can be defined because all the objects, predicates and propositions are replicated as numbers. Thus the problem is not with the axiom "truth = proof" but with the fact that internalization replicates anything as a number and permits to speak about all the expressions (i.e. all the objects) at the same time.

Here, interalization is much weaker. First, objects are not replicated (we do not have one term for the number 0 and another for the Gödel number of the term "0"). Then, propositions are replicated as object in the set O. Thus we have no way to construct a "function" $Subst$ applying any function to any object. This seems to permit to state the axiom "truth = proof" without contradiction.

It seems also that there is no contradiction between Gödel's second incompleteness theorem and the fact that, from the conecessitation axiom, we have $\exists y \in pr(\bot) \Rightarrow \bot$ and hence that the theory seems to prove its own consistency.

4 Judgements

Now we want to prove that each time a proposition $\varepsilon(P)$ is provable, there is a term t such that $t \in pr(P)$ is provable and moreover we want to build a proof-checking algorithm to recognize that the proposition $t \in pr(P)$ is provable.

Definition Let $\mathcal{T}$ be the set of axioms containing the universal closure of the propositions (1) to (33) and (a) to (f) above (the universal quantifier is the unbounded quantifier of the language, not the function symbol) and $\mathcal{T}^-$ the set of axioms containing the universal closure of the propositions (1) to (33).

4.1 Natural deduction on contents

First, we replace the axioms (a) to (f) by new deduction rules replicating natural deduction rules at the level of contents. e.g.

$$\frac{\Gamma \vdash \varepsilon(A \wedge B)}{\Gamma \vdash \varepsilon(A)}$$

keeping obvioulsy an equivalent theory.

These rules are called *internal rules*. The true natural deduction rules are called *external rules*.

Proposition (External cut elimination) In the system using external rules and internal rules, the elimination of external cuts terminates.

Proof We first define a translation on propositions, if P is a proposition, P' is the proposition P obtained by replacing every atomic proposition $\varepsilon(a)$ by the propositional constant E. If Γ is a set of proposition Γ' is obtained by translating every proposition of Γ and adding the proposition E to it.

We translate every proof in the system using external rules and internal rules of $\Gamma \vdash P$ to a proof in the system using external rules only of $\Gamma' \vdash P'$. By induction over the structure of the proof of $\Gamma \vdash P$.

- If the last rule of this proof is a external rule, for instance,

$$\frac{\dfrac{p}{\Gamma \vdash A \wedge B}}{\Gamma \vdash A} \wedge - elim$$

  we translate it into the proof

$$\frac{\dfrac{p'}{\Gamma' \vdash A' \wedge B'}}{\Gamma' \vdash A'} \wedge - elim$$

- If the last rule is a unary internal rule, for instance

$$\frac{\dfrac{p}{\Gamma \cup \{\varepsilon(A)\} \vdash \varepsilon(B)}}{\Gamma \vdash \varepsilon(A \Rightarrow B)}$$

  then by induction hypothesis p' is a proof of $\Gamma' \cup \{E\} \vdash E$. But $\Gamma' \cup \{E\} = \Gamma'$ thus p' is a proof of $\Gamma' \vdash E$, i.e. $\Gamma' \vdash \varepsilon(A \Rightarrow B)'$ we translate this proof into p'.

- If the last rule is a binary (or ternary) internal rule for instance

$$\frac{\dfrac{p_1}{\Gamma \vdash \varepsilon(A \Rightarrow B)} \quad \dfrac{p_2}{\Gamma \vdash \varepsilon(A)}}{\Gamma \vdash \varepsilon(B)}$$

  then we translate this proof into

$$\frac{\dfrac{\dfrac{p_1'}{\Gamma' \vdash E} \quad \dfrac{p_2'}{\Gamma' \vdash E}}{\Gamma' \vdash E \wedge E} \wedge - intro}{\Gamma' \vdash E} \wedge - elim$$

By induction over the structure of the proof p, if q is obtained by the elimination of some external cut in p then q' is obtained by the elimination of some external cut in p'. Thus, as by the cut elimination for first order logic, there is no infinite sequence of reductions starting from p', there is no infinite sequence of reductions starting from p and elimination of external cuts terminates.

Proposition Let Γ be a set of propositions that are either atomic propositions $\varepsilon(P)$ or universal closures of such atomic propositions. If the sequent $\Gamma \vdash \varepsilon(Q)$ has a proof free of external cuts and whose last rule is external, then $\varepsilon(Q)$ is an instance of some axiom $\forall x_1 ... \forall x_n\ \varepsilon(P)$.
Proof As the proved proposition is atomic, the last rule cannot be an external introduction rule. It is not an internal rule, thus it is either an axiom rule or an external elimination rule.

If it is an axiom rule, then the proposition $\varepsilon(Q)$ is an axiom.

If it is an external elimination rule, then the lowest rule that is not an external elimination rule cannot be an introduction rule (the proof contains no external cuts), it cannot be a internal rule (the proved proposition is not atomic) thus it is an axiom rule. Thus, all the elimination rules below are elimination of the universal quantifier and $\varepsilon(Q)$ is an instance of some axiom $\forall x_1 ... \forall x_n\ \varepsilon(P)$.

Corollary For sequents of the form $\mathcal{T}^- \vdash \varepsilon(A)$, the system is equivalent to the system with internal rules only and an axiom rule

$$\frac{}{\Gamma \cup \{\forall x_1 ... \forall x_n\ \varepsilon(P)\} \vdash \varepsilon(P[x_1 \leftarrow t_1, ..., x_n \leftarrow t_n])}\ axiom$$

4.2 An algorithm

Call $\mathcal{S}$ the set of propositions A such that $\mathcal{T} \vdash \varepsilon(A)$. In this section we define a subset $\mathcal{S}'$ of $\mathcal{S}$ such that $\mathcal{S}'$ is decidable and whenever a proposition A is in $\mathcal{S}$ there is a term t such that the proposition $t \in pr(A)$ is in $\mathcal{S}'$.

Definition

$$\frac{}{\Gamma \cup \{\forall x_1\ ...\ \forall x_n\ \varepsilon(t \in pr(A))\}\ \triangleright\ (t \in pr(A))[x_1 \leftarrow t_1, ..., x_n \leftarrow t_n]}$$

$$\frac{\Gamma \cup \{x \in pr(A)\}\ \triangleright\ t \in pr(B)}{\Gamma\ \triangleright\ (x \in pr(A) \mapsto t)) \in pr(A \Rightarrow B)}$$

$$\frac{\Gamma\ \triangleright\ f \in pr(A \Rightarrow B) \quad \Gamma\ \triangleright\ a \in pr(A)}{\Gamma\ \triangleright\ (f\ a) \in pr(B)}$$

$$\frac{\Gamma \cup \{x' \in pr(x \in A)\}\ \triangleright\ t \in (B\ x)}{\Gamma\ \triangleright\ (x \in A, x' \in pr(x \in A) \mapsto t) \in pr(\forall(A)\ B)}\ x, x'\ \text{fresh}$$

$$\frac{\Gamma\ \triangleright\ f \in pr(\forall(A)\ B) \quad \Gamma\ \triangleright\ p \in pr(a \in A)}{\Gamma\ \triangleright\ (f\ a\ p) \in pr(B\ a)}$$

$$\frac{\Gamma\ \triangleright\ a \in pr(A) \quad \Gamma\ \triangleright\ b \in pr(B)}{\Gamma \triangleright < a, b >_{pr(A), x \in pr(A) \mapsto pr(B)} \in pr(A \wedge B)}$$

$$\frac{\Gamma\ \triangleright\ t \in pr(A \wedge B)}{\Gamma\ \triangleright\ \pi(pr(A), x \in pr(A) \mapsto pr(B), pr(A), t, (x \in pr(A), y \in pr(B) \mapsto x)) \in pr(A)}$$

$$\frac{\Gamma\ \triangleright\ t \in pr(A \wedge B)}{\Gamma\ \triangleright\ \pi(pr(A), x \in pr(A) \mapsto pr(B), pr(B), t, (x \in pr(A), y \in pr(B) \mapsto y)) \in pr(B)}$$

$$\frac{\Gamma\ \triangleright\ p \in pr(a \in A) \quad \Gamma\ \triangleright\ b \in pr(B\ a)}{\Gamma \triangleright < a, p, b >_{A, x \in A \mapsto pr(x \in A), x \in A, q \in pr(x \in A) \mapsto pr(B\ x)} \in pr(\exists(A)\ B)}$$

$$\frac{\Gamma\ \triangleright\ t \in pr(\exists(A)\ B) \quad \Gamma\ \triangleright\ f \in pr(\forall(A)\ (x \in A \mapsto (B \Rightarrow C)))}{\pi'(A, x \in A \mapsto pr(x \in A), x \in A, y \in pr(x \in A) \mapsto pr(B\ x), pr(C), t, f) \in pr(C)}$$

$$\frac{\Gamma\ \triangleright\ t \in pr(A)}{\Gamma\ \triangleright\ i(A, B, t) \in pr(A \vee B)}$$

$$\frac{\Gamma\ \triangleright\ t \in pr(B)}{\Gamma\ \triangleright\ j(A, B, t) \in pr(A \vee B)}$$

$$\frac{\Gamma \rhd t \in pr(A \vee B) \quad \Gamma \cup \{x \in pr(A)\} \rhd t \in pr(C) \quad \Gamma \cup \{y \in pr(B)\} \rhd u \in pr(C)}{\Gamma \rhd \delta(pr(A), pr(B), pr(C), t, x \in pr(A) \mapsto t, x \in pr(B) \mapsto u) \in pr(C)}$$

Definition Let $\mathcal{A}$ the set containing for each proposition $\forall x_1 \ ... \ \forall x_n \ \varepsilon(P)$ of $\mathcal{T}^-$ the proposition

$$\forall x_1 \ ... \ \forall x_n \ \varepsilon(c(P) \in pr(P))$$

A proposition P is said to be provable in the system above if $\mathcal{A} \rhd P$

Proposition If P is provable in the system above (i.e. if $\mathcal{A} \rhd P$) then $\varepsilon(P)$ is provable in the theory $\mathcal{T}$ (i.e. $\mathcal{T} \vdash \varepsilon(P)$).

Remark Some propositions are provable in $\mathcal{T}$ but not in this system.

Proposition The set of sequents provable with the system above is decidable.
Proof By induction over the structure of t we compute a finite number of terms A such that $\Gamma \rhd t \in pr(A)$ is derivable.

4.3 From proofs to terms

Definition Let Γ be a set of atomic propositions, we let Γ^+ be the set containing for each proposition $\varepsilon(P)$ of Γ, the proposition $x \in pr(P)$ where x is a new variable.

Proposition If $\mathcal{T} \vdash \varepsilon(P)$ is intuitionistically provable then there is a term t such that $\mathcal{A} \rhd t \in pr(P)$.
Proof If $\mathcal{T} \vdash \varepsilon(P)$ is provable then there is a proof of $\mathcal{T}^- \vdash \varepsilon(P)$ in natural deduction with internal rules only. By induction over proof structure we prove that if $\mathcal{T}^- \cup \Gamma \vdash \varepsilon(P)$ in natural deduction using internal rules only then there is a term t such that $\mathcal{A} \cup \Gamma^+ \vdash t \in pr(P)$.

- If the proof has the form

  $$\frac{}{\mathcal{T}^- \cup \Gamma \vdash A} \, axiom$$

  then either A is an instance of some axiom of $\mathcal{T}^-$ of the form $\forall x_1 \ ... \ \forall x_n \ \varepsilon(P)$ $(A = \varepsilon(P[x_1 \leftarrow t_1, ..., x_n \leftarrow t_n]))$ and there is in $\mathcal{A}$ an axiom $\forall x_1 \ ... \ \forall x_n \ \varepsilon(c(P) \in pr(P))$ thus we have $\mathcal{T}' \cup \Gamma^+ \rhd c(P[x_1 \leftarrow t_1, ..., x_n \leftarrow t_n]) \in pr(A)$, or there is a proposition A in Γ and a proposition $x \in pr(A)$ in Γ^+ and thus $\mathcal{T}' \cup \Gamma^+ \rhd x \in pr(A)$. In both cases there is some term t such that $\mathcal{A} \cup \Gamma^+ \rhd t \in pr(A)$.

- If the proof has the form

  $$\frac{\dfrac{p_1}{\mathcal{T}^- \cup \Gamma \vdash A \Rightarrow B} \quad \dfrac{p_2}{\mathcal{T}^- \cup \Gamma \vdash A}}{\mathcal{T}^- \cup \Gamma \vdash B} \Rightarrow -elim$$

  Then by induction hypothesis there are terms t_1 and t_2 such that $\mathcal{A} \cup \Gamma^+ \rhd t_1 \in pr(A \Rightarrow B)$ and $\mathcal{T}' \cup \Gamma^+ \rhd t_2 \in pr(A)$. Then we have

  $$\frac{\mathcal{A} \cup \Gamma^+ \rhd t_1 \in pr(A \Rightarrow B) \quad \mathcal{A} \cup \Gamma^+ \rhd t_2 \in pr(A)}{\mathcal{A} \cup \Gamma^+ \rhd (t_1 \ t_2) \in pr(B)}$$

- If the proof has the form

  $$\frac{\dfrac{p}{\mathcal{T}^- \cup \Gamma \cup \{A\} \vdash B}}{\mathcal{T}^- \cup \Gamma \vdash A \Rightarrow B} \Rightarrow -intro$$

Then by induction hypothesis there is a term t such that $\mathcal{A} \cup \Gamma^+ \cup \{x \in pr(A)\} \ \triangleright \ t \in pr(B)$. Then we build the derivation

$$\frac{\mathcal{A} \cup (\Gamma \cup \{A\})^+ \ \triangleright \ t \in pr(B)}{\mathcal{A} \cup \Gamma^+ \ \triangleright \ (x \in pr(A) \mapsto t) \in pr(A \Rightarrow B)}$$

- etc.

Remark It is not the case that each time a proposition of the form $t \in pr(P)$ is provable, it is decided by the algorithm above (for instance a proposition of the form $c(P) \in pr(P)$ is almost never decided by this algorithm). But for each provable proposition, there is a term such that the proposition $t \in pr(P)$ is decided by this algorithm.

Definition Let $\mathcal{S}'$ be the decidable set of propositions $\varepsilon(P)$ such that $\mathcal{A} \ \triangleright \ P$.

Theorem A proposition $\varepsilon(P)$ is intuitionistically provable in the theory $\mathcal{T}$ if and only if it is provable in the system having all the propositions of $\mathcal{S}'$ as axioms and the single deduction rule

$$\frac{\varepsilon(t \in pr(P))}{\varepsilon(P)}$$

Conclusion

We have developed a first order theory that formalizes mathematics in such a way that proofs are objects, as are numbers, functions and sets. In this system proving a proposition can be replaced by providing a proof-term.

Having proofs as objects does not require to formalize mathematics in a typed language. But, it seems to require to formalize mathematics with weaker axioms than Zermelo-Fraenkel set theory, that together with the proofs-as-objects axioms permits to form too large sets and encode paradoxes. Even with a weaker theory, like that presented in section 2, paradoxes may be encoded when we have both the possibility to quantify over any set and the strong cartesian product. Thus we have to drop one or the other.

The truth of a proposition P and of a proposition $t \in pr(P)$ are defined differently as the latter needs to be decided by an algorithm and the former cannot. But these propositions are expressed in the same universal language and the two notions of truth are compatible: if $t \in pr(P)$ is true for the second definition of truth, then it is also true for the first.

When proofs are objects, equality splits into two symbols: equality of denotation and equality of meaning. The latter being the true equality and the former verifying a restricted Leibniz scheme.

In an extension of the theory presented here, we can add a rewriting system, and extend equality of meaning with the relation defined by this system. Proofs are shorter, because some equational reasoning are be erased from them. If this reduction eliminates cuts when applied to proof-terms, proving its termination may be a way to prove the consistency of the theory.

Waiting for a model, a normalization proof or a paradox, the consistency of this theory is open.

References

[1] P.B. Andrews, Resolution in type theory, *The Journal of Symbolic Logic*, 36, 3 (1971) pp. 414-432.

[2] M.J. Beeson, *Foundations of Constructive Mathematics*, Springer-Verlag (1985).

[3] G. Boolos, The logic of provability, *Cambridge University Press* (1993).

[4] N.G. de Bruijn, A Survey of the project automath, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, J.R. Hindley, J.P. Seldin (Eds.), Academic Press (1980).

[5] A. Church, A formulation of the simple theory of types, *The Journal of Symbolic Logic*, 5 (1940) pp. 56-68.

[6] Th. Coquand, An analysis of Girard's paradox, *Rapport de Recherche 531*, Institut National de Recherche en Informatique et en Automatique (1986).

[7] Th. Coquand, A new paradox in type theory, *Logic, Methodology and Philosophy of Science IX*, D. Prawitz, B. Skyrms and D. Westerståhl (Ed.), Elsevier (1994) pp. 555-570.

[8] Th. Coquand, G. Huet, The calculus of constructions, *Information and Computation*, 76 (1988) pp. 95-120.

[9] H.B. Curry, R.Feys, Combinatory logic, Vol. 1, *North Holland*, Amsterdam (1958).

[10] G. Dowek, Lambda-calculus, combinators and the comprehension scheme, *Typed Lambda Calculi and Applications*, Lecture Notes in Computer Science 902 (1995) pp. 154-170. *Rapport de Recherche* 2565, Institut National de Recherche en Informatique et en Automatique (1995).

[11] G. Dowek, Collections, sets and types, (abstract) *Logic, Methodology and Philosophy of Science X* (1995). *Rapport de Recherche* 2708, Institut National de Recherche en Informatique et en Automatique (1995).

[12] S. Feferman, Finitary inductively presented logics, *Logic Colloquium '88*, R. Ferro, C. Bonotto, S. Valentini and A. Zanardo (Ed.), North Holland (1989).

[13] S. Feferman, Reflecting on incompleteness, *The Journal of Symbolic Logic*, 56, 1, (1991) pp. 1-49.

[14] J.Y. Girard, Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieur, *Thèse de Doctorat d'État*, Université de Paris 7 (1972).

[15] K. Gödel, An interpretation of the intuitionistic propositional calculus, 1933, in K. Gödel collected works, S. Feferman, J.W. Dawson Jr., S.C. Kleene G.H. Moore, R.M. Solovay, J. van Heijenoort (Ed.), Oxford University Press (1986).

[16] V. Halbach, A system of complete and consistent truth, *Notre Dame Journal of Formal Logic*, 35, 3 (1994) pp. 311-327.

[17] W.A. Howard, The Formulæ-as-type notion of construction, 1969, *To H.B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism*, J.R. Hindley, J.P. Seldin (Ed.), Academic Press (1980).

[18] G. Huet, personal communication.

[19] J.L. Krivine, M. Parigot, *Programming with proofs*, J. Inf. Process. Cybern. EIK 26 (1990) pp. 149-167.

[20] V. McGee, How truthlike can a predicate be ? A negative result, *Journal of Philosophical Logic* 14 (1985) pp. 399-410.

[21] P. Martin-Löf, Constructive mathematics and computer programming, *Logic, Methodology and Philosophy of Science VI*, 1979, L.J. Cohen, J. Łoś, H. Pfeiffer, K.-P. Podewski (Ed.), North-Holland (1982) pp. 153-175.

[22] P. Martin-Löf, Intuitionistic type theory, *Bibliopolis*, Napoli (1984).

[23] Ch. Paulin-Mohring, Inductive definitions in the system COQ, Rules and properties, *Typed Lambda Calculi and Applications*, Lecture Notes in Computer Science 664 (1993) pp. 328-345.

[24] Ch. Paulin-Mohring, B. Werner, Synthesis of ML programs in the system Coq, *Journal of Symbolic Computation*, 15, 5-6 (1993) pp. 607-640.

[25] G.Plotkin. Building-in equational theories, *Machine Intelligence*, 7 (1972) pp. 73-90.

[26] W. W. Tait, Infinitely long terms of transfinite type, *Formal Systems and Recusrive Functions*, J.N. Crossley, M. Dummett (Ed.), North-Holland (1965).

[27] A.N. Whitehead, B. Russell, Principia mathematica, *Cambridge University Press*, (1910-1913, 1925-1927).

Unité de recherche Inria Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 Villers Lès Nancy
Unité de recherche Inria Rennes, Irisa, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Unité de recherche Inria Rhône-Alpes, 46 avenue Félix Viallet, 38031 Grenoble Cedex 1
Unité de recherche Inria Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex
Unité de recherche Inria Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 Sophia-Antipolis Cedex

Éditeur
Inria, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)
ISSN 0249-6399