Electronic Notes in Theoretical Computer Science 69 (2003)
URL: http://www.elsevier.nl/locate/entcs/volume69.html 15 pages

Joyal's arithmetic universes via type theory

Maria Emilia Maietti

Dipartimento di Matematica Pura ed Applicata
University of Padova
via Belzoni n.7, 35100 Padova, Italy
maietti@math.unipd.it

Abstract

Andrè Joyal constructed arithmetic universes to provide a categorical proof of Gödel incompleteness results. He built them in three stages: he first took a Skolem theory, then the category of its predicates and finally he made the exact completion out of the latter.

Here, we prove that the construction of an initial arithmetic universe is equivalent to that of an initial list-arithmetic pretopos and also of an initial arithmetic pretopos. The initial list-arithmetic pretopos is built out of its internal language formulated as a dependent typed calculus in the style of Martin-Löf's extensional type theory. Analogously, we prove that the second stage of Joyal's construction is equivalent to taking an initial arithmetic lextensive category or an initial regular locos.

We conclude by proposing the notion of list-arithmetic pretopos as the general definition of arithmetic universe. We are motivated from the fact in any list-arithmetic pretopos we can show the existence of free internal categories and diagrams as in any of Joyal's arithmetic universes.

Key words: Dependent type theory, pretopoi, categorical logic.

1 Introduction

In category theory some kinds of category like topoi and pretopoi can be thought of as universes in which to develop mathematics (see [Joh77,JM95,MM92]). Arithmetic universes provide further examples of such universes.

The notion of arithmetic universe was introduced by Andrè Joyal in some lectures given in the seventies, all still unpublished. The main purpose was to provide a categorical proof of Gödel incompleteness theorems based on the fact that any arithmetic universe contains an initial one. Arithmetic universes are built in three stages: first, take a Skolem theory, namely a cartesian category with a parameterized natural numbers object where all the objects are finite products of the natural numbers object, then take the category of its predicates and finally add quotients by doing its exact completion. The outcome gives in particular a pretopos with parameterized list objects, here called list-arithmetic pretopos. An initial arithmetic universe is obtained by performing the described construction starting from an initial Skolem theory.

Now a question arises: what is the general categorical definition of arithmetic universe such that we can build free internal categories and diagrams as in any of Joyal's arithmetic universes? In [Mor96] and [Wra85] it is more or less said that it should be a pretopos with free internal categories and diagrams.

Here, we propose the notion of list-arithmetic pretopos as the general notion of arithmetic universe, for at least two reasons. First, we prove that the construction of an initial arithmetic universe amounts to be equivalent to that of an initial list-arithmetic pretopos. Second, we know that free internal categories and diagrams exist in any list-arithmetic pretopos [Mai99] as Joyal proved for any of his arithmetic universes. Actually, since we prove that the construction of an initial arithmetic universe is equivalent to that of an initial arithmetic pretopos as well, we could also choose the notion of arithmetic pretopos as the general definition of arithmetic universe. But, we do not do that since we think that list constructors are essential to build free internal categories and diagrams.

All these results are obtained by employing the internal language of the considered categories. In particular that of list-arithmetic pretopoi, already introduced in [Mai99,Mai01], turns out to be the internal language of any initial arithmetic universe. Knowing the internal language of a category means that we can treat the category as a syntactic one built out of its internal language. This is a stronger link between a calculus and a class of categories than that established by the soundness and completeness theorem.

The internal type theory of list-arithmetic pretopoi, and hence of general arithmetic universes as defined here, is based on the dependent typed calculus $\mathcal{A}u$ formulated in the style of extensional Martin-Löf's type theory [Mar84]. In designing the internal language of a category (at least lex) we are led to produce an extensional type theory since we view the equality between morphisms of the category as the definitional equality between terms. Extensional Martin-Löf's type theory differs from the intensional Martin-Löf's Constructive Type Theory [NPS90] for what regards its computational contents: while in the former the definitional equality between terms is in general undecidable, in the latter it is decidable and it enjoys all the computational properties good enough to reckon it as a paradigm of a functional programming language. Anyway, providing an internal dependent type theory, even if extensional, for an initial arithmetic universe is important to investigate its computational properties like, for example, the validity of a canonical normal form theorem for closed terms, which is the best one can hope for extensional dependent type theories and from which we can get a syntactic consistency proof of the theory.

Since the correspondence between type constructors and categorical properties is modular [Mai01], we can easily recognize the internal type theories both of arithmetic lextensive categories and of regular locoi as fragments of the calculus $\mathcal{A}u$. The novelty of the formulation of internal languages for categorical universes in terms of a dependent type theory, in contrast to traditional formulations in terms of a many sorted logic, is that they reveal that propositions interpreted as subobjects correspond to *mono* types, that is types with at most one proof. Hence, in designing an internal dependent type theory we do not need to add a new syntactic entity representing propositions with corresponding rules of formation, but we just consider dependent types. Thinking of the interpretation of a dependent typed calculus via fibrations [Ben85,Jac99], all this corresponds categorically to regain the properties of the subobject fibration of a category only by means of its codomain fibration [Mai01].

Here, by making use of the internal languages of the considered categories, we prove not only that the third stage in the construction of an initial arithmetic universe amounts to be equivalent to the construction of an initial list-arithmetic pretopos or even of an initial arithmetic pretopos, but also that the second stage is equivalent to the construction of an initial arithmetic lextensive category or of an initial regular locos.

In the future we aim to employ the internal type theory of an arithmetic universe to translate Joyal's categorical proof of Gödel incompleteness result into a type-theoretic one. Moreover, we intend to use the internal language to proceed in the constructions of theories generated from finite decidable sketches internally to a generic arithmetic universe as started in [Mai00].

2 The categorical definition of list-arithmetic pretopos

We recall the definitions of the categories we are going to use in the sequel.

Definition 2.1 A *lextensive category* is a finitely complete category with stable finite disjoint coproducts [CLW93].

Note that a lextensive category amounts to be a locally distributive lex category [1] .

Definition 2.2 A *parameterized natural numbers object* in a category with finite products is an object N together with maps $0 : 1 \rightarrow N, s : N \rightarrow N$ such that for every $b : B \rightarrow Y$ and $g : Y \rightarrow Y$ there is an unique $rec(b, g)$ making

[1] In previous versions of this paper or in [Mai01] we used the term "distributive" as in [Coc90] for what most of the authors call "lextensive" to reserve the word "distributive" for a more general concept (see [CLW93]).

the following diagrams commute

$$\begin{tikzcd}
B \arrow[r, "<id{,}0\cdot !_B>"] \arrow[dr, "b"'] & B\times N \arrow[d, "rec(b{,}g)"] & B\times N \arrow[l, "id\times s"'] \arrow[d, "rec(b{,}g)"] \\
& Y & Y \arrow[l, "g"]
\end{tikzcd}$$

with $!_B : B \to 1$ the unique map towards the terminal object.

It is worth to recall here that in the presence of function spaces, like in a cartesian closed category, this parameterized version of natural numbers object is equivalent to the usual natural numbers object.

Definition 2.3 An *arithmetic lextensive category* is a lextensive category with a parameterized natural numbers object.

Definition 2.4 A finitely complete category $\mathcal{U}$ has *parameterized list objects* if for any object $A \in Ob\mathcal{U}$, there is an object $List(A)$ with maps $r_o^A : 1 \to List(A), r_1^A : List(A) \times A \to List(A)$ such that for every $b : B \to Y$ and $g : Y \times A \to Y$ there is an unique $rec(b, g)$ making the following diagrams commute

$$\begin{tikzcd}
B \arrow[r, "<id{,}r_o^A\cdot !_B>"] \arrow[dr, "b"'] & B\times List(A) \arrow[d, "rec(b{,}g)"] & B\times(List(A)\times A) \arrow[l, "id\times r_1^A"'] \arrow[d, "(rec(b{,}g)\times id_A)\cdot\sigma"] \\
& Y & Y\times A \arrow[l, "g"]
\end{tikzcd}$$

where $\sigma : B\times(List(A)\times A) \to (B\times List(A))\times A$ is the associative isomorphism $<< \pi_1, \pi_1 \cdot \pi_2 >, \pi_2 \cdot \pi_2 >$.

In [Coc90] there is an equivalent definition of list-arithmetic finitely complete categories in terms of recursive objects and preservation of recursive objects by the pullback functor $!_D^* : \mathcal{C} \to \mathcal{C}/D$ sending an object B to $\pi_1 : D\times B \to D$.

Definition 2.5 A *locos* is a lextensive category with parameterized list objects.

Definition 2.6 A *regular locos* is a locos which is also a regular category, namely it has stable images.

Finally, we recall the categorical definition of a pretopos [MR77], [JM95]

Definition 2.7 A *pretopos* is a category equipped with finite limits, stable finite disjoint coproducts and stable effective quotients of equivalence relations.

Definition 2.8 A *list-arithmetic pretopos* is a pretopos with parameterized list objects.

Some general categorical properties are needed in order to extract the internal dependent type theory of a category, for example of a list-arithmetic pretopos. To start with, we need a list-arithmetic pretopos to be locally a list-arithmetic pretopos:

Proposition 2.9 *For every object $A \in Ob\mathcal{U}$ of a list-arithmetic pretopos $\mathcal{U}$ the slice category $\mathcal{U}/A$ is still a list-arithmetic pretopos, i.e. the notion of list-arithmetic pretopos is* local.

Having a local structure corresponds to the closure of the various constructors on dependent types. But this is not enough to interpret a dependent type theory. Indeed, since substitution for dependent types and terms is interpreted via the pullback functor, we also need that

Proposition 2.10 *For every morphism $f : A \to B$ of a list-arithmetic pretopos $\mathcal{U}$ the pullback functor $f^* : \mathcal{C}/B \to \mathcal{C}/A$ preserves the list-arithmetic pretopos structure of the corresponding slice categories.*

Proof. For what concerns the structure of a pretopos these propositions follow easily. The key point is then to prove that parameterized list-objects exist in the slice categories and they are stable under pullbacks. The proof of these propositions for the list-arithmetic structure, stated also in [Coc90] about locoi, can be done by means of the internal language of a locos. Indeed, a locos has an internal language given by that for Heyting pretopoi in [Mai98] without forall types and quotient types but with list types restricted to closed types. Then, by means of this internal language, we can build list types on arbitrary dependent types, corresponding to list objects in a slice category, and prove that they are stable under pullbacks [Mai].

Analogously, we can prove that

Proposition 2.11 *Arithmetic lextensive categories, regular locoi and arithmetic pretopoi are local in the sense of prop. 2.9 and the pullback functors preserve their structure in the sense of prop. 2.10.*

Remark 2.12 If we interpret the type theory using the codomain fibration and substitution via pullback, then we need to face some coherence problems that we can fix by using the split fibration [Ben85,Hof94,Jac99] associated to the codomain fibration (more details can be found in [Mai99,Mai01]).

Remark 2.13 Note that, for what just said about locoi in the proof of prop. 2.10, we can immediately deduce that the internal type theory of a list-arithmetic pretopos is that one in [Mai98] without forall type and natural numbers type, - corresponding to the internal language of a pretopos -, together with lists restricted to closed types - corresponding to parameterized list-objects -. However, since we can prove that lists on arbitrary dependent types are definable, in the next section we will present the version of the calculus with them included.

3 The internal language of list-arithmetic pretopoi

In a few words the internal language of a list-arithmetic pretopos corresponds to predicative coherent logic equipped with the set-theoretic constructions of lists and quotients. We recall that with respect to intuitionistic logic, predicative coherent logic lacks implication and universal quantification, namely it has only conjunction, falsum, disjunction and existential quantification. Hence negation as $A \rightarrow \perp$ is not definable in general, but we can have proofs from A to falsum at the metalanguage level of an inference rule. For example, we can not prove the proposition $0 \neq s(0)$ because we can not even express it as a proposition, but we can prove that from $0 = s(0)$ we get a contradiction.

Now, we recall the rules of the typed calculus $\mathcal{A}u$ for arithmetic universes introduced in [Mai99]. The typed calculus $\mathcal{A}u$ is equipped with types, which should be thought of as sets or data types, and with typed terms which represent proofs of the types to which they belong.

In the style of Martin-Löf's type theory, we have four kinds of judgements [NPS90]:

$$A \ type \ [\Gamma] \quad A = B \ [\Gamma] \quad a \in A \ [\Gamma] \quad a = b \in A \ [\Gamma]$$

that is the type judgement, the equality between types, the term judgement and the equality between terms of the same type. The contexts Γ of these judgements are telescopic [dB91], since types are allowed to depend on variables of other types. The contexts are generated by the following rules

$$1C) \ \emptyset \ \ cont \qquad 2C) \ \frac{\Gamma \ \ cont \qquad A \ type \ [\Gamma]}{\Gamma, x \in A \ \ cont} \ (x \in A \notin \Gamma)$$

plus the rules of equality between contexts [Str91], [Pit00]. In the following, we present the inference rules to construct type judgements and term judgements with their equality judgements by recursion. One should also add all the inference rules that express reflexivity, symmetry and transitivity of the equality between types and terms together with the following set equality rule and assumption of typed variables

$$set\ rule) \ \frac{a \in A \ [\Gamma] \quad A = B \ [\Gamma]}{a \in B \ [\Gamma]} \qquad var) \ \frac{\Gamma, x \in A, \Delta \ \ cont}{x \in A \ [\Gamma, x \in A, \Delta]}$$

We can derive then the structural rules of weakening and of a suitable exchange. In the following we give the formation rules for types specific to $\mathcal{A}u$ with the corresponding introduction, elimination and conversion rules of their terms. We omit the equality rules of all the type and term constructors that are necessary to derive the substitution rules. We adopt the usual definitions of bound and free occurrences of variables and we identify two terms under α-conversion. Note that the context common to all judgements involved in a rule will be omitted. The typed variable appearing in a context is meant to be added to the implicit context as the last one. The rules to generate $\mathcal{A}_u$'s types and terms are all present in the extensional version of Martin-Löf's type theory [Mar84] except for the disjointness axiom, the rules about quotients types restricted to *mono* equivalence relations and the effectiveness axiom. A type is called *mono* if it is inhabitated by at most one proof.

Supposing A *type* and $R(x,y)$ *type* $[x,y \in A]$, we will write $\mathsf{Equiv}(R)$ to mean the following three judgements: $\mathsf{refl}(x) \in R(x,x)\ [x \in A]$, $\mathsf{sym}(x,y,z) \in R(y,x)\ [x \in A, y \in A, z \in R(x,y)]$, $\mathsf{trans}(x,y,z,u,v) \in R(x,z)\ [x \in A, y \in A, z \in A, u \in R(x,y), v \in R(y,z)]$.
Moreover, we will write $\mathsf{Mono}(R)$ to mean

$$z = w \in R(x,y)\ [x \in A, y \in A, z \in R(x,y), w \in R(x,y)]$$

The $\mathcal{A}u$ dependent typed calculus

Terminal type

$$\text{Tr)}\ \ \top\ type \qquad \text{I-Tr)}\ \ \star \in \top \qquad \text{C-Tr)}\ \ \frac{t \in \top}{t = \star \in \top}$$

False type

$$\text{Fs)}\ \ \bot\ type \qquad \text{E-Fs)}\ \ \frac{a \in \bot \quad A\ type}{\mathsf{r_o}(a) \in A}$$

Indexed Sum type

$$\Sigma)\ \ \frac{C(x)\ type\ \ [x \in B]}{\Sigma_{x \in B} C(x)\ type} \qquad \text{I-}\Sigma)\ \ \frac{b \in B \quad c \in C(b)}{< b, c > \in \Sigma_{x \in B} C(x)}$$

$$\text{E-}\Sigma)\ \ \frac{d \in \Sigma_{x \in B} C(x) \quad m(x,y) \in M(< x, y >)\ [x \in B, y \in C(x)]}{El_\Sigma(d, m) \in M(d)}$$

$$\text{C-}\Sigma)\ \ \frac{b \in B \quad c \in C(b) \quad m(x,y) \in M(< x, y >)\ [x \in B, y \in C(x)]}{El_\Sigma(< b, c >, m) = m(b,c) \in M(< b, c >)}$$

Equality type

$$\text{Eq)}\ \ \frac{C\ type \quad c \in C \quad d \in C}{\mathsf{Eq}(C,c,d)\ type} \qquad \text{I-Eq)}\ \ \frac{c \in C}{\mathsf{eq_C}(c) \in \mathsf{Eq}(C,c,c)}$$

$$\text{E-Eq)}\ \ \frac{p \in \mathsf{Eq}(C,c,d)}{c = d \in C} \qquad \text{C-Eq)}\ \ \frac{p \in \mathsf{Eq}(C,c,d)}{p = \mathsf{eq_C}(c) \in \mathsf{Eq}(C,c,d)}$$

Disjoint Sum type

$$+)\ \ \frac{C\ type \quad B\ type}{C + B\ type} \qquad \text{I}_1\text{-+)}\ \ \frac{c \in C}{\mathsf{inl}(c) \in C + B} \qquad \text{I}_2\text{-+)}\ \ \frac{b \in B}{\mathsf{inr}(b) \in C + B}$$

$$\text{E-+)}\ \ \frac{w \in C + B \quad a_C(x) \in A(\mathsf{inl}(x))\ [x \in C] \quad a_B(y) \in A(\mathsf{inr}(y))\ [y \in B]}{El_+(w, a_C, a_B) \in A(w)}$$

$$\text{C}_1\text{-+)}\ \ \frac{c \in C \quad a_C(x) \in A(\mathsf{inl}(x))\ [x \in C] \quad a_B(y) \in A(\mathsf{inr}(y))\ [y \in B]}{El_+(\mathsf{inl}(c), a_C, a_B) = a_C(c) \in A(\mathsf{inl}(c))}$$

$$\text{C}_2\text{-+)}\ \ \frac{b \in B \quad a_C(x) \in A(\mathsf{inl}(x))\ [x \in C] \quad a_B(y) \in A(\mathsf{inr}(y))\ [y \in B]}{El_+(\mathsf{inr}(b), a_C, a_B) = a_B(b) \in A(\mathsf{inr}(b))}$$

Disjointness

$$\frac{c \in C \quad b \in B \quad \mathsf{inl}(c) = \mathsf{inr}(b) \in C + B}{\mathsf{dsj}(c,b) \in \bot}$$

Quotient type

$$\text{Q)}\ \frac{R(x,y)\ \mathit{type}\ [x \in A, y \in A] \quad \mathsf{Mono}(R) \quad \mathsf{Equiv}(R)}{A/R\ \mathit{type}}$$

$$\text{I-Q)}\ \frac{a \in A\ A/R\ \mathit{type}}{[a] \in A/R} \qquad \text{eq-}Q)\ \frac{a \in A \quad b \in A \quad d \in R(a,b)\ A/R\ \mathit{type}}{[a] = [b] \in A/R}$$

$$\text{E-Q)}\ \frac{p \in A/R \quad l(x) \in L([x])\ [x \in A] \quad l(x) = l(y) \in L([x])\ [x \in A, y \in A, d \in R(x,y)]}{El_Q(l,p) \in L(p)}$$

$$\text{C-Q)}\ \frac{a \in A \quad l(x) \in L([x])\ [x \in A] \quad l(x) = l(y) \in L([x])\ [x \in A, y \in A, d \in R(x,y)]}{El_Q(l,[a]) = l(a) \in L([a])}$$

Effectiveness

$$\frac{a \in A \quad b \in A \quad [a] = [b] \in A/R}{\mathsf{eff}(a,b) \in R(a,b)}$$

List type

$$\text{list)}\ \frac{C\ \mathit{type}}{\mathit{List}(C)\ \mathit{type}} \qquad \text{I}_1\text{-list)}\ \ \epsilon \in \mathit{List}(C) \qquad \text{I}_2\text{-list)}\ \frac{s \in \mathit{List}(C) \quad c \in C}{\mathsf{cons}(s,c) \in \mathit{List}(C)}$$

$$\text{E-list)}\ \frac{s \in \mathit{List}(C) \quad a \in L(\epsilon) \quad l(x,y,z) \in L(\mathsf{cons}(x,y))\ [x \in \mathit{List}(C), y \in C, z \in L(x)]}{El_{\mathit{List}}(a,l,s) \in L(s)}$$

$$\text{C}_1\text{-list)}\ \frac{s \in \mathit{List}(C) \quad a \in L(\epsilon) \quad l(x,y,z) \in L(\mathsf{cons}(x,y))\ [x \in \mathit{List}(C), y \in C, z \in L(x)]}{El_{\mathit{List}}(a,l,\epsilon) = a \in L(\epsilon)}$$

$$\text{C}_2\text{-list)}\ \frac{s \in \mathit{List}(C) \quad c \in C \quad a \in L(\epsilon) \quad l(x,y,z) \in L(\mathsf{cons}(x,y))\ [x \in \mathit{List}(C), y \in C, z \in L(x)]}{El_{\mathit{List}}(a,l,\mathsf{cons}(s,c)) = l(s,c,El_{\mathit{List}}(a,l,s)) \in L(\mathsf{cons}(s,c))}$$

Note that $\mathit{List}(\top)$ corresponds to the type of natural numbers represented as lists on a singleton. Hence, we put $N \equiv \mathit{List}(\top)$ with $0 \equiv \epsilon$ and $s(n) \equiv \mathsf{cons}(n,*)$ for $n \in \mathit{List}(\top)$.

4 Joyal's arithmetic universes

What we describe in this section can be read in unpublished notes by Gavin Wraith [Wra85] and also in [Mor96] and it is due to Andrè Joyal. Joyal built arithmetic universes in three stages:

(i) consider a Skolem theory $\mathcal{S}$;

(ii) take the category $\mathit{Pred}(\mathcal{S})$ of predicates in $\mathcal{S}$;

(iii) make the exact completion $(\mathit{Pred}(\mathcal{S}))_{ex}$ on regular categories;

Then, he proved Gödel incompleteness theorems based on the fact that any arithmetic universe contains an initial one obtained by performing the above construction starting from an initial Skolem theory. Here, we describe Joyal's construction in more detail.

Definition 4.1 A *Skolem category* is a cartesian category (i.e. a category with terminal object 1 and binary products) with a parameterized natural numbers object.

Definition 4.2 A *Skolem theory* is a Skolem category whose objects are finite products of the natural numbers object.

In [Wra85] an initial Skolem category is built and used to model a programming language which represents exactly primitive recursive functions.
Using type theory we can also build an initial Skolem category as follows.

Definition 4.3 Let $\mathcal{T}_{sk}$ be the fragment of the type theory $\mathcal{A}u$ in section 3 with terminal type, natural numbers type (namely $List(\top)$) and finite products (namely $\Sigma_{x \in A} B$ with the two projections [Mar84]).

Definition 4.4 Let $\mathcal{S}_{in}$ be the category having the types of $\mathcal{T}_{sk}$ as objects and the proof-terms $b(x) \in B\ [x \in A]$ derivable in $\mathcal{T}_{sk}$ as morphisms between types A and B.

Proposition 4.5 *$\mathcal{T}_{sk}$ provides the internal language of Skolem categories and $\mathcal{S}_{in}$ is an initial Skolem category (and theory).*

Here is an useful property of the natural numbers object in a Skolem theory:

Proposition 4.6 *In any Skolem theory $\mathcal{S}$ the natural numbers object N is equipped with a structure of parameterized list object over itself.*

The proof relies on a particular encoding trick based on the binary representation of natural numbers. Then, we are ready to define the category of predicates on a Skolem theory. First notice that we define $f \leq g \equiv f \dot{-} g = 0$ between morphisms in the Skolem category, where $\dot{-}$ is the usual truncated subtraction on natural numbers.

Definition 4.7 Given a Skolem theory $\mathcal{S}$ the category $Pred(\mathcal{S})$ of *predicates* has as

- $\mathsf{O}b(Pred(\mathcal{S}))$: $\mathcal{S}$-morphisms $P : N \to N$ such that $P * P = P$ (where $*$ is the usual multiplication between natural numbers);
- $\mathsf{Hom}(P, Q)$: $\mathcal{S}$-morphisms $f : N \to N$ such that $P \leq Q \cdot f$ and two such $\mathcal{S}$-morphisms $f : N \to N$ and $g : N \to N$ are equal iff $P * f = P * g$.

Note that, if $Pred(\mathcal{S})$ is embedded in the usual category $Sets$ of classical ZFC-sets, then we get a category having subsets of the set of natural numbers as objects, because a subset is characterized by the elements with value 1 through a predicate P (that is its characteristic function), and equivalence classes of functions mapping a subset to the codomain subset as morphisms. Finally, two maps are considered equal if they are equal on the domain subset. This category of predicates has nice properties as Joyal proved:

Proposition 4.8 *The category $Pred(\mathcal{S})$ of a Skolem theory $\mathcal{S}$ is regular with parameterized list objects and stable disjoint coproducts. Moreover, there is an epi-mono factorization where epimorphisms split.*

The key point to prove the existence of list-objects is proposition 4.6. Now, we are ready to perform the last step, namely to make the exact completion on a regular category (see for example [CV98] and loc. cit.).

Definition 4.9 *We call Joyal-arithmetic universe the exact completion* $(Pred(\mathcal{S}))_{ex}$.

Note that by the presence of split epimorphisms we can define the morphisms in $(Pred(\mathcal{S}))_{ex}$ as morphisms of $Pred(\mathcal{S})$ preserving the equivalence relations:

Definition 4.10 *The category* $(Pred(\mathcal{S}))_{ex}$ *has*

- $\mathsf{Ob}(Pred(\mathcal{S}))_{ex}$*:* (X, R) *where* X *is an object of* $Pred(\mathcal{S})$ *and* R *is an equivalence relation on* X*;*
- $\mathsf{Hom}((X, R), (Y, S))$*:* $Pred(\mathcal{S})$*-morphisms preserving the equivalence relations.*

Finally, Joyal also proved that

Proposition 4.11 $(Pred(\mathcal{S}))_{ex}$ *is a list-arithmetic pretopos*[2] *.*

Definition 4.12 *We call* $\mathcal{A}_{in}$ *the initial arithmetic universe* $(Pred(\mathcal{S}_{in}))_{ex}$ *where* $\mathcal{S}_{in}$ *is the initial Skolem category defined above.*

Indeed, it can be easily shown that $\mathcal{A}_{in}$ is an *initial* arithmetic universe among Joyal-arithmetic universes with functors induced from functors between Skolem theories with a fixed choice of their structure preserving such a structure strictly. Note that $Pred(\mathcal{S}_{in})$ turns out to be the category of primitive recursive predicates.

5 The category of primitive recursive predicates via type theory

Here we outline how the category of primitive recursive predicates is equivalent to an initial arithmetic lextensive category and also to an initial regular locos.

Throughout the paper when we talk about initial categories, for example among arithmetic lextensive categories, we refer to the category of arithmetic lextensive categories with a fixed choice of their structure and functors preserving such a structure strictly. Instead, when we talk about functors preserving some structure we mean preservation up to isomorphisms.

First, we recall what are the dependent type theories we use to build our initial categories:

Definition 5.1 $\mathcal{T}_{ad}$ is the dependent type theory equipped with the terminal type, extensional equality types, indexed sum types, disjoint sum types and the natural numbers type $List(\top)$ as those present in the calculus $\mathcal{A}u$ in section 3.

Definition 5.2 $\mathcal{T}_{rl}$ is the dependent type theory obtained by extending $\mathcal{T}_{ad}$ with quotient types of the kind $A/\top$ [3] and list types as those present in the

[2] Actually from [Wra85,Mor96] we only read that $(Pred(\mathcal{S}))_{ex}$ is a pretopos with list-objects but we noticed that they are also parameterized.

[3] After Steve Awodey's talk (presenting a joint work with Andrej Bauer) at the Mittag-

calculus $\mathcal{A}u$ in section 3.

Definition 5.3 $\mathcal{U}_{ad}$ is the syntactic category having the closed types of $\mathcal{T}_{ad}$ as objects and the proof-terms $b(x) \in B\ [x \in A]$ derivable in $\mathcal{T}_{ad}$ as morphisms.

Definition 5.4 $\mathcal{U}_{rl}$ is the syntactic category having the closed types of $\mathcal{T}_{rl}$ as objects and the proof-terms $b(x) \in B\ [x \in A]$ derivable in $\mathcal{T}_{rl}$ as morphisms.

With the technique used in [Mai01] we can prove

Theorem 5.5 *$\mathcal{T}_{ad}$ provides the internal type theory of arithmetic lextensive categories and $\mathcal{U}_{ad}$ is an initial arithmetic lextensive category.*

Theorem 5.6 *$\mathcal{T}_{rl}$ provides the internal type theory of regular locoi and $\mathcal{U}_{rl}$ is an initial regular locos.*

The proof of these theorems consists in providing a sort of bi-equivalence between the category of arithmetic lextensive categories (regular locoi) with a fixed choice of their structure and equipped with strict functors, i.e. functors preserving the considered categorical structure strictly, and the category of theories based on $\mathcal{T}_{ad}$ ($\mathcal{T}_{rl}$). In particular, $\mathcal{U}_{ad}$ ($\mathcal{U}_{rl}$) has $\mathcal{T}_{ad}$ ($\mathcal{T}_{rl}$) as its internal language and it is initial with respect to strict functors.

Now, we are ready to prove that

Theorem 5.7 *The syntactic categories $\mathcal{U}_{ad}$ and $\mathcal{U}_{rl}$ are equivalent to $Pred(\mathcal{S}_{in})$.*

Proof. First, we define two embeddings $\mathcal{E}_{ad} : Pred(\mathcal{S}_{in}) \longrightarrow \mathcal{U}_{ad}$ and $\mathcal{E}_{rl} : Pred(\mathcal{S}_{in}) \longrightarrow \mathcal{U}_{rl}$ both as follows:

- $\mathcal{E}_{ad}(P) \ \equiv\ \Sigma_{x\in N}P(x) =_N 1$
- $\mathcal{E}_{ad}(f) \ \equiv < \pi_1(z), \mathsf{eq} > \in \Sigma_{x\in N}Q(x) =_N 1\ \ [z \in \Sigma_{x\in N}P(x) =_N 1]$
  for $f : P \rightarrow Q$ in $Pred(\mathcal{S}_{in})$.
  The definition of $\mathcal{E}_{rl}$ is analogous.

Then, we can prove that these embeddings have nice properties:

Lemma 5.8 *The embedding $\mathcal{E}_{ad}$ is an arithmetic lextensive functor and $\mathcal{E}_{rl}$ is a regular locos functor.*

Moreover, since by proposition 4.8 we know that $Pred(\mathcal{S}_{in})$ is a regular locos, then there exist an arithmetic lextensive functor and a regular locos functor, respectively $\mathcal{J}_{ad} : \mathcal{U}_{ad} \longrightarrow Pred(\mathcal{S}_{in})$ and $\mathcal{J}_{rl} : \mathcal{U}_{rl} \longrightarrow Pred(\mathcal{S}_{in})$, defined on objects and morphisms of $\mathcal{U}_{ad}$ and $\mathcal{U}_{rl}$ through the interpretations of $\mathcal{T}_{ad}$ and $\mathcal{T}_{rl}$ into $Pred(\mathcal{S}_{in})$, that are in turn defined as in the section about the free list-arithmetic pretopos in [Mai99], in a way as $\mathcal{J}_{ad} \cdot \mathcal{E}_{ad} = id$ and $\mathcal{J}_{rl} \cdot \mathcal{E}_{rl} = id$. Then, we can show that, being $\mathcal{U}_{ad}$ and $\mathcal{U}_{rl}$ an initial arithmetic lextensive category and an initial regular locos respectively, $\mathcal{E}_{ad} \cdot \mathcal{J}_{ad} : \mathcal{U}_{ad} \longrightarrow \mathcal{U}_{ad}$ and $\mathcal{E}_{rl} \cdot \mathcal{J}_{rl} : \mathcal{U}_{rl} \longrightarrow \mathcal{U}_{rl}$ are naturally isomorphic to the corresponding identity functors and hence we conclude.

Leffler Institute I realized that in my formulation of the internal type theory of regular categories $A/Ker(f)$ for a term f can be derived from $A/\top$.

Corollary 5.9 *The initial arithmetic lextensive category $\mathcal{U}_{ad}$ is equivalent to the initial regular locos $\mathcal{U}_{rl}$.*

From this corollary we conclude that in an initial arithmetic lextensive category stable images and lists are definable. But, since the equivalence is built through the interpretation functors, defined in turn by induction on type and term constructors, we can not say to be able to construct images and lists internally to any arithmetic lextensive category.

Remark 5.10 At a first glance, we could have thought of $Pred(\mathcal{S}_{in})$ as an initial Skolem category closed under equalizers. But, we can not prove this, since, for example, the embedding of $Pred(\mathcal{S}_{in})$ in an initial finitely complete category does not preserve the terminal object of $Pred(\mathcal{S}_{in})$ unless we know that natural numbers can be decomposed disjointly in zero plus strictly positive numbers, namely unless we have disjoint finite coproducts.

6 The initial arithmetic universe via type theory

Here, we outline how the construction of an initial arithmetic universe is equivalent to that of an initial list-arithmetic pretopos and also of an initial arithmetic pretopos. To accomplish our task, first we recall some facts about internal type theories and related initial syntactic categories.

Definition 6.1 $\mathcal{T}_{pn}$ is the fragment of the dependent type theory $\mathcal{A}u$ in section 3 without list types but with the natural numbers type $List(\top)$.

Definition 6.2 $\mathcal{U}_{pn}$ is the syntactic category having the closed types of $\mathcal{T}_{pn}$ as objects and the proof-terms $b(x) \in B\ [x \in A]$ derivable in $\mathcal{T}_{pn}$ as morphisms.

Definition 6.3 $\mathcal{U}_{\mathcal{A}u}$ is the syntactic category having the closed types of $\mathcal{A}u$ as objects and the proof-terms $b(x) \in B\ [x \in A]$ derivable in $\mathcal{A}u$ in section 3 as morphisms.

With the technique used in [Mai99,Mai01] analogously to theorem's. 5.5 5.6 we can prove

Theorem 6.4 *$\mathcal{T}_{pn}$ provides the internal type theory of arithmetic pretopoi and $\mathcal{U}_{pn}$ is an initial arithmetic pretopos.*

Theorem 6.5 *$\mathcal{A}u$ provides the internal type theory of list-arithmetic pretopoi and $\mathcal{U}_{\mathcal{A}u}$ is an initial list-arithmetic pretopos.*

Moreover, we prove

Theorem 6.6 *The syntactic category $\mathcal{U}_{\mathcal{A}u}$ is equivalent to $\mathcal{A}_{in}$.*

Proof. Analogously to the embeddings in theorem 5.7 we define $\mathcal{E}_{au} : Pred(\mathcal{S}_{in}) \longrightarrow \mathcal{U}_{\mathcal{A}u}$ as follows:

- $\mathcal{E}_{au}(P) \ \equiv\ \Sigma_{x \in N} P(x) =_N 1$
- $\mathcal{E}_{au}(f) \ \equiv < \pi_1(z), \mathsf{eq} > \in \Sigma_{x \in N} Q(x) =_N 1 \ \ [z \in \Sigma_{x \in N} P(x) =_N 1]$

for $f : P \to Q$ in $Pred(\mathcal{S}_{in})$.

and we can prove that

Lemma 6.7 *The embedding $\mathcal{E}_{au}$ is regular and preserves disjoint coproducts and parameterized list objects.*

Therefore, by the above lemma, since $\mathcal{A}_{in}$ is the exact completion of $Pred(\mathcal{S}_{in})$, by its universal property there exists a functor $\widehat{\mathcal{E}_{au}} : \mathcal{A}_{in} \longrightarrow \mathcal{U}_{\mathcal{A}u}$ [CV98]. Moreover, $\widehat{\mathcal{E}_{au}}$ preserves all the list-arithmetic pretopos structure since it is an exact functor by the property of exact completion and also:

Lemma 6.8 *The functor $\widehat{\mathcal{E}_{au}} : \mathcal{A}_{in} \longrightarrow \mathcal{U}_{\mathcal{A}u}$ preserves disjoint finite coproducts and list objects.*

Then, since by proposition 4.11 we know that $\mathcal{A}_{in}$ is a list-arithmetic pretopos, there exists a list-arithmetic functor $\mathcal{J}_{au} : \mathcal{U}_{\mathcal{A}u} \longrightarrow \mathcal{A}_{in}$ defined on objects and morphisms of $\mathcal{U}_{\mathcal{A}u}$ through the interpretation of $\mathcal{A}u$ into $\mathcal{A}_{in}$, that is in turn defined as in the section about the free list-arithmetic pretopos in [Mai99]. Then, being $\mathcal{U}_{\mathcal{A}u}$ an initial list-arithmetic pretopos and by the property of exact completion we conclude that these functors establish an equivalence of categories.

From the equivalence between the initial arithmetic universe $\mathcal{A}_{in}$ and the initial list-arithmetic pretopos $\mathcal{U}_{\mathcal{A}u}$ we deduce that the internal language of the initial arithmetic universe $\mathcal{A}_{in}$ is the typed calculus $\mathcal{A}u$. Analogously, we can prove that

Theorem 6.9 *The syntactic category $\mathcal{U}_{pn}$ is equivalent to $\mathcal{A}_{in}$.*

Corollary 6.10 *The initial list-arithmetic pretopos $\mathcal{U}_{\mathcal{A}u}$ is equivalent to the initial arithmetic pretopos $\mathcal{U}_{pn}$.*

In other words, we can prove that parameterized list objects are definable in an initial arithmetic pretopos. But it is worth noticing also here, as after corollary 5.9, that because of the nature of the above equivalence, where one of the functors is defined by induction on type and term constructors, we can not directly deduce that any arithmetic pretopos is list-arithmetic.

7 Conclusions.

As the initial arithmetic universe $\mathcal{A}_{in}$ is equivalent to the initial list-arithmetic pretopos $\mathcal{U}_{\mathcal{A}u}$, *we propose the notion of list-arithmetic pretopos as the general definition of arithmetic universe.* Of course, being $\mathcal{A}_{in}$ equivalent to an initial arithmetic pretopos as well, we could also choose the notion of arithmetic pretopos for that. But we do not make this choice because, while we are able to define free internal categories and diagrams (see for example [Jac99] for the definitions) in any list-arithmetic pretopos [Mai99], as Joyal did in any of his arithmetic universes, we doubt that this is possible in any arithmetic pretopos.

Acknowledgement

My acknowledgements go first to Martin Hyland, who proposed me to work on arithmetic universes and helped me with many fruitful discussions during my staying in Cambridge. Many thanks also to Steve Vickers for providing me Gavin Wraith's unpublished notes [Wra85] with the master's thesis of his student [Mor96]. Finally, I wish to thank Pino Rosolini and Silvio Valentini for their generous promptness in discussing my research work and surely Andrè Joyal for his interest in this work.

References

[Ben85] J. Benabou. Fibred categories and the foundations of naive category theory. *Journal of Symbolic Logic*, 50:10–37, 1985.

[CLW93] A. Carboni, S. Lack, and R.F.C. Walters. Introduction to extensive and distributive category. *Journal of Pure and Applied Algebra*, 84:145–158, 1993.

[Coc90] J.R.B. Cockett. List-arithmetic distributive categories: locoi. *Journal of Pure and Applied Algebra*, 66:1–29, 1990.

[CV98] A. Carboni and E.M. Vitale. Regular and exact completions. *Journal of Pure and Applied Algebra*, 125:79–116, 1998.

[dB91] N.G. de Bruijn. Telescopic mapping in typed lambda calculus. *Information and Computation*, 91:189–204, 1991.

[Hof94] M. Hofmann. On the interpretation of type theory in locally cartesian closed categories. In Proceedings of CSL'94, September 1994.

[Jac99] B. Jacobs. *Categorical Logic and Type Theory.*, volume 141 of *Studies in Logic*. Elsevier, 1999.

[JM95] A. Joyal and I. Moerdijk. *Algebraic set theory*, volume 220 of *Lecture Note Series*. Cambridge University Press, 1995.

[Joh77] P. Johnstone. *Topos theory.* Academic Press, 1977.

[Mai] M.E. Maietti. Locality of list objects via internal language. Manuscript, 2001.

[Mai98] M.E. Maietti. The internal type theory of an Heyting Pretopos. In C.Paulin-Mohring E. Gimenez, editor, *Types for Proofs and Programs. Selected papers of International Workshop Types '96, Aussois*, volume 1512 of *LNCS*, pages 216–235. Springer Verlag, 1998.

[Mai99] M.E. Maietti. The typed calculus of arithmetic universes. Technical report, University of Birmingham, CSR-99-14, December 1999. also Technical Report-University of Padova n.5 Dec. 1999.

[Mai00] M.E. Maietti. Theories generated by finite lex decidable sketches in an arithmetic universe. Technical report, University of Padova, n.9, December 2000.

[Mai01] M.E. Maietti. Modular correspondence between dependent type theories and categorical universes. *Mittag-Leffler Preprint Series*, 44, 2001.

[Mar84] P. Martin-Löf. *Intuitionistic Type Theory, notes by G. Sambin of a series of lectures given in Padua, June 1980.* Bibliopolis, Naples, 1984.

[MM92] S. MacLane and I. Moerdijk. *Sheaves in Geometry and Logic. A first introduction to Topos theory.* Springer Verlag, 1992.

[Mor96] A. Morrison. Reasoning in arithmetic universes. Master's thesis, University of London - Imperial College of Science, Technology and Medicine, Advisor: S. Vickers, September 1996.

[MR77] M. Makkai and G. Reyes. *First order categorical logic.*, volume 611 of *Lecture Notes in Mathematics.* Springer Verlag, 1977.

[NPS90] B. Nordström, K. Peterson, and J. Smith. *Programming in Martin Löf's Type Theory.* Clarendon Press, Oxford, 1990.

[Pit00] A.M. Pitts. Categorical logic. In Oxford University Press, editor, *Handbook of Logic in Computer Science*, volume 5, pages 39–128, 2000.

[Str91] Th. Streicher. *Semantics of type theory.* Birkhäuser, 1991.

[Wra85] G. C. Wraith. Notes on arithmetic universes and Gödel incompleteness theorems. Unpublished manuscript, 1985.