*Logica Trianguli*, 4, 2000, 41-51

# AN ALGEBRA FOR FINITARY ONTOLOGY OR A FUNCTIONALLY COMPLETE LANGUAGE FOR THE FINITARY THEORY OF TYPES

François LEPAGE

**Abstract**

This paper presents a generalization of a proposal of van Benthem's who has shown how to provide a canonical name for any object in propositional type theory. Van Benthem's idea is to characterize any function in the hierarchy by the Boolean values the function takes for any sequence of arguments. The recursive definition of canonical names uses only the λ abstraction, functional application, the identity operator and the fact that we have a name for the true and the false. We show that this result can be extended to the finite theory of types by the introduction of an algebra on individuals.

## 1. Introduction

In [4] and [5] a system was provided for the partial propositional theory of types and it was proved to be semantically complete. The completeness proof uses the fact that we have a canonical name for every object of the semantical space. A careful examination of the proof reveals that if we were able to provide a name for every object in partial type theory, the same completeness proof would apply to the system. So a preliminary question is the following: is it possible to provide such a name for every object in classical simple type theory? We will see that - for reasons of cardinality - the general answer is no. But what about type theory based on finite sets of individuals? This question deserves attention because - among other reasons - the simple theory of types is an extremely powerful tool for describing semantic universes, that is, for constructing models of sets of statements for languages containing operators and functions of all orders (as seems to be the case for the portion of natural languages for which it is possible to provide an explicit semantics). The basic idea is very simple.

Starting from sets of objects corresponding to a basic type, we define the universe of objects as being the set of functions definable on these basic sets. For example, if we begin with the set {0,1} - the two truth-values - and some set of individuals *E*, the set $\{0,1\}^E$ of the functions of *E* in {0,1} could be interpreted as the set of properties of individuals. So, the property 'being blue' is a function, which, for each individual, is associated either with 1, if the individual is blue, or 0, if the individual is not blue. Another example would be that of a property of a property, namely, an element of $\{0,1\}^{(\{0,1\}^E)}$ such as 'being a colour.' We can see the potential strength of this instrument.

**2. The simple theory of types**

Formally, the set of types *T* is the smallest set such that

(i) $t \in T$ (the truth values type),
(ii) $e \in T$ (the individuals type),
(iii) if $\sigma, \tau \in T$, then $<\sigma,\tau> \in T$

The set of objects for each type would thus be

(i) $D_t = \{0,1\}$ (the set of truth values)
(ii) $D_e = E$ (the set of individuals)
(iii) $D_{<\sigma,\tau>} = D_\tau^{D_\sigma}$ (the set of functions of $D_\sigma$ in $D_\tau$).

There exists a very simple language, which permits the formulation of a system the theorems of which corresponds exactly to its valid statements (in Henkin's general sense, see [3], [8] and [6]). This language, which contains but very few primitives, is the following:

Let $A_L$ (the alphabet of *L*) be such that $A_L = \{\lambda, \equiv, (, ), \{x_{\alpha_i}\}_{\alpha \in T, i \in \omega}\}$

$\{x_{\alpha_i}\}_{i \in \omega} = Var_\alpha$ is the denumerable set of the variables of type $\alpha$.

The set of well-formed formulas $WFF_\alpha$ for each type $\alpha$ of *L* is defined as follows:

(i) $\{ x_{\alpha_i} \}_{i \in \omega} \subseteq \mathrm{WFF}_\alpha$
(ii) if $A,B \in \mathrm{WFF}_\alpha$, then $(A \equiv B) \in \mathrm{WFF}_t$
(iii) if $A \in \mathrm{WFF}_\alpha$ and $x_\beta \in Var_\beta$, then $\lambda x_\beta A_\alpha \in \mathrm{EBF}_{<\beta,\alpha>}$
(iv) if $A \in \mathrm{EBF}_{<\beta,\alpha>}$ and $B \in \mathrm{EBF}_\beta$, then $(AB) \in \mathrm{EBF}_\alpha$.

The idea is very simple. Starting with a denumerable number of variables of each type, we can construct identity statements, we can construct the functions of one type into another, and we can apply a function to any argument of the right type. Explicitly, and formally, the semantics is as follows:

An assignment of value to the variables is a function

$$\chi : \bigcup_i Var_{\alpha_i} \rightarrow \bigcup_i D_{\alpha_i} ,$$

with $\chi(x_\alpha) \in D_\alpha$.

We define the denotation of a $\mathrm{WFF}_\alpha$ of type $\alpha$ as follows:

(i) $\|x_\alpha\|_\chi = \chi(x_\alpha)$
(ii) $\|(A \equiv B)\|_\chi = 1$ iff $\|A\|_\chi = \|B\|_\chi$ and
$= 0$ otherwise
(iii) $\|\lambda x_\beta A_\alpha\|_\chi$ is the function $h : D_\beta \rightarrow D_\alpha$ such that
$h(a) = \|A_\alpha\|_{\chi \frac{x_\beta}{a}}$ where $\chi \frac{x_\beta}{a}$ is like $\chi$ except that $\chi \frac{x_\beta}{a}(x_\beta) = a$.
(iv) $\|(A_{<\beta,\alpha>} B_\beta)\|_\chi = \|A_{<\beta,\alpha>}\|_\chi(\|B_\beta\|_\chi)$

This paper proposes to answer a very simple question. (I admit that I do not know whether this question has been asked or whether it has received a satisfying answer.) The question is the following: in a universe where the objects are either *individuals* of basic types, truth values, or functions belonging to the hierarchy definable in terms of the individuals and truth values, is it possible - supposing that we have a name for each individual of each basic type, a name for the true, and a name for the false - to define a *canonical name* for every object in the universe, (i.e., for every $a \in D$, to provide an expression $E_\alpha$ such that for every $\chi$, $\|E_\alpha\|_\chi = a$) with, as our only resource, the abstractor $\lambda$, its converse, the functional application, and identity?

**3. A name for every object in propositional type theory**

To simplify, let us suppose that, barring truth-values, there is only one type of basic individual and that $E$ designates the set of individuals as above. If the cardinality of $E$ is infinite, the answer to the question is, trivially, no for the simple reason that $2^E$, for example, is, in this case, non-denumerable and the number of expressions of the language is denumerable. But what about when $E$ is finite?

A very elementary version of this problem was stated and solved by Post in his thesis in 1920 [7]. The question was the following: is it possible to find a name for each $n$-ary truth function by using only $\vee$ and $\neg$? The well-known answer is yes and it consists of introducing the notion of disjunctive normal form after having defined $\wedge$ in terms of $\neg$ and $\vee$. As, moreover, we can define $\neg$ in $\lambda$-calculus by the following (see [6] and [1])

$$\neg =_{\text{def}} \lambda x_t(x \equiv \text{F})$$

where $x$ is a variable of type $t$ (the truth value type) and F the name of the false[1] and

$$\wedge =_{\text{def}} \lambda x \lambda y(\lambda z((zx)y) \equiv \lambda z((z\text{T})\text{T}))$$

where T is a name for the true, $x$ and $y$ are variables of the truth value type and $z$ is a variable of type $<t,<t,t>>$, that is, the type of function taking truth values as arguments and for values function taking truth values as argument and truth values for values. $((zx)y)$ is thus of type $t$. Introducing the '$\wedge$' by De Morgan's laws and writing $(x \vee y)$ for $((\vee x)y)$ and $(x \wedge y)$ for $((\wedge x)y)$, the canonical name for '$\supset$', for example, is

$$\lambda x \lambda y\left[(x \wedge y) \vee (\neg x \wedge y) \vee (\neg x \wedge \neg y)\right].$$

The technique is simple: it suffices to describe all and only those lines of the table where the statement is true by writing $x$ (or $y$) when $x$ (or $y$) is true and $\neg x$ (or $\neg y$) when $x$ (or $y$) is false.

---

[1] F and T are themselves definable: $\text{T} =_{\text{def}} (\lambda x_t x \equiv \lambda x_t x)$, $\text{F} =_{\text{def}} (\lambda x_t(x \equiv x) \equiv \lambda x_t x)$.

This can be generalized. The problem is to give a canonical name to $f: 2_n \rightarrow 2$ (where 2 = {0,1}).

Let $\mathbf{1}(f)$ be the set of lines of the truth table where $f$ is true. Then

$$\lambda p_0 \ldots \lambda p_{n-1} \bigvee_{j \in \mathbf{1}(f)} \left( \bigwedge_{k<n} l_{jk} \right)$$

(where $l_{jk}$ is $p_k$ if $p_k$ is true and $\neg p_k$ if $p_k$ is false on line *j*) is a name for *f*.

This method settles the question for functions of type <*t*,<*t*,<…<*t*,*t*>…>>>. But what about other types? For the moment, let us restrict ourselves to the hierarchy of functions definable starting with the single type *t*, that is, starting with the set {0,1} (we call these types 'propositional types'). For these types, we can generalize the approach described above [2]. To this end, we will need the following formal notions: a function *f* of type $\sigma = <\sigma_0,<\sigma_1,<\ldots<\sigma_{n-1},t>\ldots>>>$ (all types have this form, the degenerate case being *t*).

Let us call $\vec{f}_\sigma = < f_{\sigma_0}, \ldots, f_{\sigma_{n-1}} >$ a projector of $f_\sigma$ (because $f( f_{\sigma_0} ) \ldots ( f_{\sigma_{n-1}} ) \in \{0,1\}$).

Let us suppose, as the induction hypothesis, that we have a canonical name $(g_i)^c$ for each $g_i$ of a type simpler than σ and that for each variable $X_{\sigma_j}$, $[X_{\sigma_j}(\vec{g})^c]$ is $[\ldots[X_{\sigma_j}(g_1)^c] \ldots (g_k)^c]$ that is, the syntactic expression of type *t* constructed by applying $X_{\sigma_j}$ to the projector formed from the canonical name $(\vec{g})^c = <(g_1)^c,\ldots, (g_k)^c>$.

If $\vec{p}_i = <p_{i0},\ldots,p_{in-1}>$ is a projector of *f*, then let $\delta_{pij}$ be such that

$\delta_{pij}([x_{\sigma_j}(\vec{g})^c]) = [x_{\sigma_j}(\vec{g})^c]$ if $p_{ij}(\vec{g}) = 1$

$\delta_{pij}([x_{\sigma_j}(\vec{g})^c]) = \neg[x_{\sigma_j}(\vec{g})^c]$ if $p_{ij}(\vec{g}) = 0$

then

*Proposition:*

$$\lambda x_{\alpha_0}\ldots\lambda x_{\alpha_{n-1}}[\bigvee_{\vec{p_i}\in 1(f)}(\bigwedge_{\vec{g}\in Pr(\alpha_0)}\delta_{pi_0}([x_{\alpha_0}(\vec{g})^c])\wedge\ldots\wedge\bigwedge_{\vec{g}\in Pr(\alpha_{n-1})}\delta_{pi_{n-1}}([x_{\alpha_{n-1}}(\vec{g})^c]))]$$

is a canonical name for *f*. ($\bigvee$ and $\bigwedge$ are, respectively, the generalized disjunction and conjunction).

This method of constructing names for objects is not, however, directly exportable to the general case if, in addition to truth values, we have individuals.

**4. A generalization to type theory with finite sets of individuals**

This way of proceeding relies on the essential fact that we are dealing with truth functions: every function of the propositional theory of types is entirely characterized by the values that it attributes to its different projectors, namely, 0's or 1's. It is the existence of an algebra - in this case, Boolean algebra - on {0,1} and the fact that this algebra can be expressed in the language that permits us to reconstruct a formula which canonically designates a given function. What happens if the value of a function is not a truth value but an individual? Let us consider a very simple example. Let

$f : A \rightarrow B$ be an arbitrary function of a finite set *A* in a finite set *B*.

Suppose that we have a name for each object belonging to *A* and for each object belonging to *B*. The question is: is it possible to construct a canonical name for $f$ using abstraction, functional application and identity? The answer is clearly no. If $f(a) = b$ and if $f^c$ is a canonical name for $f$, then $f^c$ must be of the form $\lambda x_e Y_e$ (a variable does not have a constant value and formulas whose principal connective is ≡ are of type *t*) and $([\lambda x_e Y_e a^c]^c) = b^c$ (where $a^c$ and $b^c$ are canonical names for *a* and *b* respectively). A possible value for *Y* is $b^c$. But then *f* is the constant function such that *f*(*x*) = *b* for every *x*. Since *Y* is of type *e* and $f^c$ is of type <*e*,*e*>, the only other possibility is that *Y* is $[Z_{<\tau,e>}X_\tau]$ where τ is any type. $f^c$ is thus $\lambda x_e[Z_{<\tau,e>}X_\tau]$. Again, $Z_{<\tau,e>}$ cannot be a variable and so $f^c$ is $\lambda x_e[\lambda x_\tau W_e X_\tau]$. Once again, a possible value of *W* is $b^c$. But then *f* is the constant function such that *f*(*x*) = *b* for every *x*. We cannot regress this way indefinitely, the

number of λ appearing in $f^c$ being finite. Therefore, unless *f* is a constant function, there is no formula using abstraction, functional application and identity whose value is *f*.
The lesson is clear: we must enrich the language and the ontology, taking inspiration from the case where the objects are truth values, by introducing a kind of *algebra of individuals*.

Let *E* be a finite set of individuals. An algebra of individuals on *E* is a 5-uple
<*E*, ⊕, ⊗, ♦,♠>
where
(1) ♦ and ♠ are two new individuals distinct from each other and from all the elements in *E*.
(2) ⊕, ⊗ are the following partial operations on *E*:

$x \otimes y = \blacklozenge$ if $x = y$

$x \otimes y = \spadesuit$ if $x \neq y$

$x \oplus \blacklozenge = \blacklozenge \oplus x = x$ if $x \neq \spadesuit$ and $x \neq \blacklozenge$

$x \oplus \spadesuit = \spadesuit \oplus x = \blacklozenge$ if $x \neq \blacklozenge$ and $x \neq \spadesuit$

$\blacklozenge \oplus \spadesuit = \spadesuit \oplus \blacklozenge = \spadesuit$

$\spadesuit \oplus \spadesuit = \spadesuit$

$\blacklozenge \oplus \blacklozenge = \blacklozenge$

In other cases, the operations are not defined.

Let $f : A \rightarrow B$. Suppose that $a_0^c,...,a_n^c$ are the respective canonical names for $a_0,...,a_n$, that $(f(a_0))^c,...,(f(a_n))^c$ are the respective canonical names for $f(a_0),...,f(a_n)$, that
$\|X \oplus^c Y\|_\chi = \|X\|_\chi \oplus \|Y\|_\chi$ and that $\|X \otimes^c Y\|_\chi = \|X\|_\chi \otimes \|Y\|_\chi$
then
$\lambda x(((f(a_0))^c \oplus^c (x \otimes^c a_0^c)) \oplus^c \ldots \oplus^c ((f(a_n))^c \oplus^c (x \otimes^c a_n^c)))$
is a canonical name for $f$.

For example, let $f: \{a,b\} \rightarrow \{c,d\}$ be such that $f(a) = b$ and $f(c) = d$. We easily check that
$\|\lambda x(((f(a))^c \oplus^c (x \otimes^c a^c)) \oplus^c ((f(c))^c \oplus^c (x \otimes^c b^c)))\|_\chi(a) =$

$\|(b^c \oplus^c (x \otimes^c a^c)) \oplus^c (d^c \oplus^c (x \otimes^c b^c))\|_{\chi'} =$ (where $\chi'$ is $\chi_{\vec{a}}^{x}$ )

$\|b^c \oplus^c (x \otimes^c a^c)\|_{\chi'} \oplus \|d^c \oplus^c (x \otimes^c b^c)\|_{\chi'} =$

$(\|b^c\|_{\chi'} \oplus \|x \otimes^c a^c\|_{\chi'}) \oplus (\|d^c\|_{\chi} \oplus \|x \otimes^c b^c\|_{\chi'}) =$

$(b \oplus (\|x\|_{\chi'} \otimes \|a^c\|_{\chi'})) \oplus (d \oplus (\|x\|_{\chi'} \otimes \|b^c\|_{\chi'})) =$

$(b \oplus (a \otimes a)) \oplus (d \oplus (a \otimes b))$

$(b \oplus \blacklozenge) \oplus (d \oplus \spadesuit) =$

$(b \oplus \blacklozenge) =$

$b$

Note that we can trivially apply this method to the particular case of truth functions by taking care to distinguish $\spadesuit_t$ from $\spadesuit_e$ and $\blacklozenge_t$ from $\blacklozenge_e$ so that we can generalize to any finite number of basic types.

The generalization to the finite theory of types is straightforward.

*Proposition*. Let $f$ be a function of type $<\alpha_0, <\alpha_1, <\ldots<\alpha_n,t>...>>>$ or of type
$<\alpha_0, <\alpha_1, <\ldots<\alpha_n,e>...>>>$. The expression

$$\lambda x_{\alpha_0}...\lambda x_{\alpha_n} [ \bigoplus^c_{\vec{a}_i \in P(f)} [(f(\vec{a}_i))^c \oplus^c [\bigoplus^{n}_{j=0}{}^c [ \bigoplus^c_{\vec{g} \in P(\alpha_j)} (x_{\alpha_j}(\vec{g})^c) \otimes^c (a^c_{i_{\alpha_j}}(\vec{g})^c)]]]]$$

(where $\vec{a}_i = < a_{i\alpha_0}...a_{i\alpha_n} >$ and $P(f)$ is the set of projectors of $f$) is a canonical name for $f$.

*Proof*
To begin with, note that the value of

$\bigoplus_{j=0}^{n} x_j = x_0 \otimes \ldots \otimes x_n$, where every $x_j$ is either $\blacklozenge$ or $\spadesuit$, is $\spadesuit$ if there is at least one $x_j$ which is $\spadesuit$ and is $\blacklozenge$ if, and only if, every $x_j$ is $\blacklozenge$.

Let us calculate

$\| \lambda x_{\alpha_0} ... \lambda x_{\alpha_n} [ \bigoplus^c_{\vec{a}_i \in P(f)} [ f((\vec{a}_i)^c) \oplus^c [ \bigoplus^n_{j=0}{}^c [ \bigoplus^c_{\vec{g} \in P(\alpha_j)} (x_{\alpha_j} (\vec{g})^c) \otimes^c (a^c_{i\alpha_j} (\vec{g})^c)]]]] \|_\chi (\vec{a}_k)$

According to the definition of $\| \ \|_\chi$, this value is

$\| [ \bigoplus^c_{\vec{a}_i \in P(f)} [ (f(\vec{a}_i))^c \oplus_c [ \bigoplus^n_{j=0}{}^c [ \bigoplus^c_{\vec{g} \in P(\alpha_j)} (x_{\alpha_j} (\vec{g})^c) \otimes_c (a^c_{i\alpha_j} (\vec{g})^c)]]]] \|_{\chi'}$ where $\chi'$ is like $\chi$ except that $\chi'(x_{\alpha_j}) = a_{k\alpha_j}$.

Two cases are possible:

(i) $k = i$.

In this case, we have
$\| (x_{\alpha_j} (\vec{g})^c) \otimes^c (a^c_{i\alpha_j} (\vec{g})^c) \|_{\chi'} = \blacklozenge$ for every
$\vec{g} \in P(\alpha_j)$ and for every $j$, $0 \leq j \leq n$. Therefore,

$\| [ f((\vec{a}_k)^c) \oplus^c [ \bigoplus^n_{j=0}{}^c [ \bigoplus^c_{\vec{g} \in P(\alpha_j)} (x_{\alpha_j} (\vec{g})^c) \otimes^c (a^c_{i\alpha_j} (\vec{g})^c)]]] \|_{\chi'} =$
$\| f((\vec{a}_k)^c) \|_{\chi'} \oplus \blacklozenge = f(\vec{a}_k) \oplus \blacklozenge = f(\vec{a}_k)$.

(ii) ) $k \neq i$

In this case, we have $\| (x_{\alpha_j} (\vec{g})^c) \otimes (a^c_{i\alpha_j} (\vec{g})^c) \|_{\chi'} = \spadesuit$ for some
$\vec{g} \in P(\alpha_j)$ and some $j$, $0 \leq j \leq n$ and, for all the others,
$\vec{g} \in P(\alpha_j)$ and $j$, $0 \leq j \leq n$, $\| (x_{\alpha_j} (\vec{g})^c) \otimes (a^c_{i\alpha_j} (\vec{g})^c) \|_{\chi'} = \blacklozenge$. Therefore,

$\| [ (f(\vec{a}_i))^c \oplus^c [ \bigoplus^n_{j=0}{}^c [ \bigoplus^c_{\vec{g} \in P(\alpha_j)} (x_{\alpha_j} (\vec{g})^c) \otimes^c (a^c_{i\alpha_j} (\vec{g})^c)]]] \|_{\chi'} =$
$\| (f(\vec{a}_i))^c \|_{\chi'} \oplus \spadesuit = f(\vec{a}_i) \oplus \spadesuit = \blacklozenge$.

(i) and (ii) entail that

$$\|[\lambda x_{\alpha_0}...\lambda x_{\alpha_n} \bigoplus^{c}_{\vec{a}_i \in P(f)} [(f(\vec{a}_i))^c) \oplus^c [\bigoplus^{n}_{j=0}{}^c [\bigoplus^c_{\vec{g} \in P(\alpha_j)} (x_{\alpha_j}(\vec{g})^c) \otimes^c (a^c_{i_{\alpha_j}}(\vec{g})^c)]]]]\|_\chi(\vec{a}_k) =$$

$$f(\vec{a}_k) \oplus \blacklozenge = f(\vec{a}_k).$$

**5. Getting rid of special individuals**

This way of proceeding may seem *ad hoc*. If the idea of defining operations on individuals is acceptable, postulating the "existence" of individuals like ♦ and ♠ may seem a bizarre hypothesis. Note, however, that at the level of the object language we have no need for names denoting these strange individuals: the canonical names contain only the usual symbols of the language of the theory of types plus $\oplus^c$ and $\otimes^c$ This strongly suggests that these individuals are eliminable using an adequate redefinition of the operators on "normal" individuals. Here is one way to do so.

Let the following be the partial ternary operation on $E$:

$\partial_\otimes : E^3 \rightarrow E$

with

$\partial_\otimes(x,y,z) = x$ if $y = z$ and undefined if $y \neq z$.

and the following family of functions

$\partial_{\oplus_n}(x_0, ..., x_{n-1}) = a$ if for every $i$ such that $0 \leq i \leq n\text{-}1$, $x_i = a$ or is not defined
and
$\partial_{\oplus_n}(x_0, ..., x_{n-1})$ is not defined in other cases.

Taking once again our example of the trivial function $f$ with $f(a) = c$ and $f(b) = d$ and using our two special functions, we easily verify that

$\lambda x(\partial^c_{\oplus_2}(\partial^c_\otimes((f(a))^c, a^c, x), \partial^c_\otimes((f(b))^c, b^c, x)))$ is a canonical name for $f$.

The generalization to functions of any type is immediate.

*Proposition*. Let $f$ be a function of type $<\alpha_0, <\alpha_1, <\ldots<\alpha_n,t>...>>>$ or of

type $<\alpha_0, <\alpha_1, <\ldots<\alpha_n,e>\ldots>>>$. The expression

$$\lambda x_{\alpha_0} \ldots \ \lambda x_{\alpha_{n-1}} [\ \partial^c_{\oplus\ \vec{a}_i \in P(f)} [\partial^c_{\oplus\ j=0}{}^{n-1} [\ \partial^c_{\oplus\ \vec{g} \in P(\alpha_j)} (\partial^c_{\otimes}((f(\vec{a}_i))^c,\ a^c_{i_{\alpha_j}}(\vec{g})^c,\ x_{\alpha_j}(\vec{g})^c))]]]$$

is a canonical name for *f*.

Université de Montréal, lepagef@ere.umontreal.ca

## REFERENCES

[1] ANDREWS, P.B., "A Reduction of the Axioms for the Theory of Propositional Types", *Fundamenta Mathematicæ* LII, 1963, 345-350.

[2] VAN BENTHEM, J., *Language in Action*, North-Holland, Amsterdam/MIT Press, Cambridge, 1995.

[3] HENKIN, L., "Completeness in the Theory of Types", *The Journal of Symbolic Logic*, 15, 1950, 81-91.

[4] LEPAGE, F., AND LAPIERRE, S., *Logique partielle et savoir, Essai de philosophie formelle*, Vrin, Paris/Bellarmin, Montréal, 2000.

[5] LEPAGE, F., "Partial Monotonic Protothetics", *Partiality and Modality*, E. Thijsse, F. Lepage & H. Wansing (eds.) special issue of *Studia Logica*, Vol. 66, no. 1, 2000, 147-163.

[6] MONTAGUE, R., "Universal Grammar", in *Formal Philosophy*, Yale University Press, New Haven, 1974, 222-246.

[7] POST, E., "Introduction to a general theory of elementary propositions", in *From Frege to Gödel*, J. van Heijenoort (ed.), Harvard University Press, Cambridge (Mass.), 1967, 264-283.

[8] TARSKI, A., "Sur le terme primitif de la logistique", *Fundamenta Mathematicæ* IV, 1923, 59-74.