Probabilistic approach to the lambda definability for fourth order types [1]

Marek Zaionc [2]

Computer Science Department
Jagiellonian University
Krakow, Poland

Abstract

It has been proved by Loader [3] that Statman-Plotkin conjecture (see [7] and [5]) fails. The Loader proof was done by encoding the word problem in the full type hierarchy based on the domain with 7 elements. The aim of this paper is to show that the lambda definability problem limited to regular fourth order types is decidable in any finite domain. Obviously λ definability is decidable for $1, 2$ and 3 order types. As an additional effect of the result described we may observe that for certain types there is no finite context free grammar generating all closed terms. We prove also that probability that randomly chosen fourth order type (or type of the order not grater then 4) admits decidable lambda definability problem is zero.

1 Syntax of simple typed lambda calculus

We shall consider a simple typed lambda calculus with a single ground type O. The set $\mathbb{T}$ of types is defined as follows: O is a type and if τ and μ are types then $\tau \to \mu$ is a type. We will use the following notation: if $\mu, \tau_1, \tau_2, ..., \tau_n$ are types then by $\tau_1 \to \tau_2 \to ... \to \tau_n \to \mu$ we mean the type $\tau_1 \to (\tau_2 \to ... \to (\tau_n \to \mu)...)$. Therefore, every type τ has the form $\tau_1 \to ... \to \tau_n \to O$. By a $\tau^k \to \mu$ we mean the type $\tau... \to \tau \to O$ with k occurrences of type τ (with $\tau^0 \to \mu = \mu$). If τ has the form $\tau_1 \to ... \to \tau_n \to O$, then τ_i for $i \leq n$ are called components of type τ and are denoted by $\tau[i]$. For any type τ we define $rank(\tau)$ and $arg(\tau)$ as follows: $arg(O) = 0$, $rank(O) = 1$ and $arg(\tau_1 \to ... \to \tau_n \to O) = n$ and $rank(\tau_1 \to ... \to \tau_n \to O) = \max_{i=1...n} rank(\tau[i]) + 1$. We define inductively types $\tau[i_1, ..., i_k]$ by $(\tau[i_1, ..., i_{k-1}])[i_k]$ for $1 \leq i_k \leq arg(\tau[i_1, ..., i_{k-1}])$.

[1] Supported by the State Committee for Scientific Research in Poland (KBN), research grant 7T11C 022 21

[2] Email: zaionc@ii.uj.edu.pl

This is a preliminary version. The final version will be published in Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

Definition 1.1 *Type τ is called regular if $rank(\tau) \leq 4$ and every component of τ has $arg \leq 1$. This implies that only components allowed for regular types are O, $O \to O$ and $(O^k \to O) \to O$ for any k.*

For any type τ there is given a denumerable set of variables $V(\tau)$. Any type τ variable is a type τ term . If T is a term having type $\tau \to \mu$ and S is a type τ term , then TS is a term which has type μ. If T is a type μ term and x is a type τ variable , then $\lambda x.T$ is a term having type $\tau \to \mu$. The axioms of equality between terms have the form of $\beta\eta$ conversions and the convertible terms will be written as $T =_{\beta\eta} S$. Term T is in a long normal form if $T = \lambda x_1...x_n.yT_1...T_k$, where y is an x_i for some $i \leq n$ or y is a free variable, T_j for $j \leq k$ are in a long normal form and $yT_1...T_k$ is a type O term. Long normal forms exist and are unique for $\beta\eta$ conversions (compare [6]). A closed term is a term without free variables. By $Cl(\tau)$ we mean a set of all closed type τ terms. Let Y be a set of typed variables. By $Cl(\tau, Y)$ we mean a set of all type τ terms such that any free variable of such terms is in Y. Let us introduce a complexity measure π for closed terms. If T is a closed term written in a long normal form and $T = \lambda x_1...x_n.x_i$ then $\pi(T) = 1$. If $T = \lambda x_1...x_n.x_iT_1...T_k$, then $\pi(T) = \max_{j=1...k}(\pi(\lambda x_1...x_n.T_j)) + 1$. Complexity π is just the depth of the Böhm tree of a given term. If $y_1, ..., y_n$ are some occurrences of free variables in term T then $T[y_1/t_1, ..., y_k/t_k]$ is a term obtained from T by replacement of each y_i be a term t_i respectively. For more detailed treatment of typed lambda calculus see [8].

2 Model of simple typed lambda calculus

A *full type hierarchy* $\{D_\tau\}_{\tau\in\mathbb{T}}$ is a collection of finite domains, one for each type. The whole hierarchy is determined by D_O. We assume that the set D_O is a given finite set and $D_{\tau\to\mu}$ is a collection of all functions from D_τ to D_μ. Therefore all D_τ are finite. Since any function $f \in D_{\tau\to\mu}$ is a finite set, we assume some sensible encoding of those functions as Gödel numbers for their graphs. An assignment ϕ is a function associating an object from one of the domains with each variable of typed lambda calculus in such a way that the type τ variable is assigned an element from the domain D_τ . Any assignment ϕ can be extended to an interpretation V^ϕ of all terms in *full type hierarchy*. A function V^ϕ is defined by:

(i) $V^\phi(x) = \phi(x)$

(ii) $V^\phi(TS) = V^\phi(T)(V^\phi(S))$

(iii) $V^\phi(\lambda x_\tau.T_\mu)$ is such a function from D_τ to D_μ that for each element $X \in D_\tau$ the value of this function is $V^\psi(T)$, where ψ, is an assignment which is identical to ϕ except that $\psi(x) = X$

It can be noted that any type τ term is interpreted as an objects from D_τ. It is also easy to notice that interpretation V^ϕ on term T depends only on the assignment ϕ on free variables in T. Hence, closed terms have a fixed

interpretation in the *full type hierarchy*. Function $f \in D_\tau$ is called lambda definable if there is a closed type τ term T that is interpreted as f.

The *lambda definability problem* is a decision problem to determine whether or not the given object $f \in \bigcup_{\tau \in \mathbb{T}} D_\tau$ is lambda definable. It was proved by Loader that the problem is undecidable. For the particular type τ the *τ-lambda definability problem* is the decision problem to determine whether or not the given object $f \in D_\tau$ is lambda definable. In this case the type τ is not a part of the problem.

Definition 2.1 *A function $\alpha : {D_\tau}^n \to D_\tau$ is called λ preserving function if the following implication holds: if $x_1, ..., x_n$ are lambda definable objects in D_τ then $\alpha(x_1, ..., x_n)$ is also a lambda definable object in D_τ.*

Definition 2.2 *We call $\{a_1, ..., a_p, \alpha_1, ..., \alpha_k\}$ a λ system in D_τ if $a_1, ..., a_p$ are some λ definable objects in D_τ and $\alpha_1, ..., \alpha_k$ are some λ preserving functions in D_τ of various arities.*

Definition 2.3 *By the closure of the λ system $\{a_1, ..., a_p, \alpha_1, ..., \alpha_k\}$ in D_τ we mean the minimal set of objects in the domain D_τ containing elements $a_1, ..., a_p$ and closed for all functions $\alpha_1, ..., \alpha_k$. Obviously all objects in the closure are λ definable functions. The λ system $\{a_1, ..., a_p, \alpha_1, ..., \alpha_k\}$ in D_τ is called λ complete if the closure of the system consists of all λ definable objects in D_τ.*

Lemma 2.4 *For λ system $L = \{a_1, ..., a_p, \alpha_1, ..., \alpha_k\}$ the closure is effectively constructible.*

Proof. We are going to construct the closure just by iteration of λ preserving functions. Let us define recursively sets $D_L^0, D_L^1, ...$ by

(i) $D_L^0 = \{a_1, ..., a_p\}$

(ii) $D_L^{i+1} = D_L^i \cup \bigcup_{1 \le j \le k} \{\alpha_j(x_1, ..., x_{p_j}) : x_1, ..., x_{p_j} \in D_L^i\}$

We are having a monotonic chain of sets of λ definable objects in D_τ. $D_L^0 \subseteq D_L^1 \subseteq ... \subseteq D_\tau$. Since D_τ is finite, then there will be an index i_0 such that $D_L^{i_0} = D_L^{i_0+1}$. Obviously $D_L^{i_0}$ is a closure of the system L. In the case when the system is complete, the closure $D_L^{i_0}$ consists of all λ definable objects in D_τ. □

3 Term grammars

Term grammars have been introduced in [9]. A more detailed treatment of term grammars can be found in [8] or [10]. A term grammar G is a triple (V, P, S) where V is a finite set of nonterminals, each of which is a typed variable, P is a finite set of productions, each of which is a pair (y, t) denoted also as $y \to t$, where $y \in V$ and $t \in Cl(\tau, V)$, where τ is a common type for y and t. S is a special nonterminal symbol from V. A closed term T is obtained from closed terms $t_1 \in \tau_1, ..., t_k \in \tau_k$ by production $y \to t$ if there is

exactly k occurrences of nonterminal variables $y_1, ..., y_k$ in t which have types $\tau_1, ..., \tau_k$ respectively, such that $T =_{\beta\eta} t[y_1/t_1, ..., y_k/t_k]$. By a derivation of the closed term T we mean a finite sequence of closed terms $t_0, t_1, ..., t_n = T$, such that for every term t_i in the sequence either there is production $y \to t_i$ in grammar G (which means that t_i is a closed term) or t_i is obtained from previous terms in the sequence by some production. Let us write $y \stackrel{*}{\to} T$ to denote that T is obtained from some closed terms $t_1, ..., t_k$ derivable in G by production $y \to t$. By $L(G, y)$ where $y \in V$ we mean the set of all closed terms which are generated from y by grammar G, i.e. if y is type τ variable then $L(G, y) = \{t \in Cl(\tau) : y \stackrel{*}{\to} t\}$. Let us assume that $y \to t$ is production and $y_1, ..., y_n$ are all free occurrences of nonterminal variables in the term t. Let $y \in \tau$, $y_1 \in \tau_1$,..., $y_n \in \tau_n$. This production determines the function $\alpha : Cl(\tau_1) \times ... \times Cl(\tau_n) \to Cl(\tau)$ defined by: $\alpha(t_1, ..., t_n) = t[y_1/t_1, ..., y_k/t_k]$ for every closed terms $t_1 \in Cl(\tau_1), ..., t_n \in Cl(\tau_n)$. If there are no nonterminal variables in the term t, then the production $y \to t$ determines a 0-ary function (constant) t which belongs to $Cl(\tau)$.

Example 3.1 Let τ be a following type $((O \to O \to O) \to O) \to (O \to O)$. Let us consider the following grammar $(\{y\}, P, y)$. Types of auxiliary variables used are the following $p \in (O, O \to O) \to O$ and $x, v, z \in O$. There are 4 productions in the set P.

(i) $y \to \lambda px.x$

(ii) $y \to \lambda px.p(\lambda vz.ypx)$

(iii) $y \to \lambda px.p(\lambda vz.ypv)$

(iv) $y \to \lambda px.p(\lambda vz.ypz)$

Let $\alpha, \beta, \gamma, \delta$ be the functions determined by these productions respectively. The closed term $\lambda px_1.p(\lambda x_2x_3.p(\lambda x_4x_5.p(\lambda x_6x_7.x_4)))$ of type τ has the following derivation sequence

(i) $\lambda px.x$

(ii) $\lambda px.p(\lambda vz.x)$

(iii) $\lambda px.p(\lambda vz.p(\lambda x_6x_7.v)$

(iv) $\lambda px_1.p(\lambda x_2x_3.p(\lambda x_4x_5.p(\lambda x_6x_7.x_4)))$

or the term can be presented as $\beta(\gamma(\beta(\alpha)))$. It is easy to prove that this grammar generates all closed terms of type τ (compare [9] or theorem 3.3).

Theorem 3.2 *For every type rank 3 type τ there is a finite grammar with one type τ nonterminal variable which generates all closed type τ terms.*

Proof. Let $\tau = \tau[1], \tau[2], ..., \tau[n] \to O$. It will be proved that the following grammar $(\{y\}, P, y)$ produces all closed type τ terms. The grammar contains exactly n productions which have the form:

(i) $y \to \lambda x_1...x_n.x_i$ if $arg(\tau[i]) = 0$

(ii) $y \to \lambda x_1...x_n.x_i(yx_1...x_n)...(yx_1...x_n)$ if $arg(\tau[i]) = k > 0$

It is easy to see that any term produced by the grammar has the type τ. The reverse implication is by induction of the complexity π of closed terms. Let us suppose that T is a closed type τ term in a long normal form . If T is a projection $\lambda x_1...x_n.x_i$, then obviously it can be obtained directly by i-th production. Suppose T has a long normal form $\lambda x_1...x_n.x_iK_1...K_k$ for some $\tau[i]$ such that $arg(\tau[i]) = k$ where K_j are type O terms. Let us define $T_1, ..., T_k$ by $T_j = \lambda x_1...x_n.K_j$. Each term T_j is a closed type τ term and is simpler with respect to complexity π then the original term T. Therefore by induction closed terms $T_1, ..., T_k$ are to be generated by the grammar G. The term T can be obtained from $T_1, ..., T_k$ by production (ii). □

Theorem 3.3 *[see Zaionc [9]] For every regular rank 4 type, such that there is $i \leq arg(\tau)$ such $\tau[i] = O$, there is a finite grammar with one nonterminal variable generating all closed type τ terms. An illustration of this case is presented in the example 3.1*

Proof. Let $\tau = \tau[1], \tau[2], ..., \tau[n] \to O$. Every $\tau[i]$ is one of the following types: O, $O \to O$ or $(O, ..., O \to O) \to O$. Let us assume that $\tau[r] = O$ for some $r \leq n$. We will prove that the following grammar $(\{y\}, P, y)$ produces all closed type τ term. The grammar contains productions, one for each type O and type $O \to O$ components and $1 + arg(\tau[i])$ productions for each second order component $\tau[i] = (O, ..., O \to O) \to O$. The productions have the form:

p_i	$y \to \lambda x_1...x_n.x_i$	if $\tau[i]) = O$
q_i	$y \to \lambda x_1...x_n.x_i(yx_1...x_n)$	if $\tau[i] = O \to O$
l_i	$y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.yx_1...x_n)$	if $\tau[i] = (O^p \to O) \to O$
$l^r_{i,1}$	$y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.yx_1...x_{r-1}z_1x_{r+1}...x_n)$	if $\tau[i] = (O^p \to O) \to O$
	.	
	.	
	.	
$l^r_{i,arg(\tau[i])}$	$y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.yx_1...x_{r-1}z_px_{r+1}...x_n)$	if $\tau[i] = (O^p \to O) \to O$

We can see that for any term $\lambda x_1...x_n.T$ in a long normal form where $T \in Cl(O, \{x_1...x_n\})$ which has a regular type τ, there is at the most one occurrence of type O variable among all $x_1...x_n$ which occurs free in T. We are going to prove that every closed type τ term can be obtained by the grammar. Let $\lambda x_1...x_n.T$ be a closed type τ term.

(case 1) If $T = x_i$ for some type O variable x_i then $\lambda x_1...x_n.T$ can be obtained by production p_i.

(case 2) If T has the form x_iS for some term $S \in Cl(O, \{x_1...x_n\})$ where x_i has type $O \to O$, then the term $\lambda x_1...x_n.S$ has also type τ and is simpler than $\lambda x_1...x_n.T$ with respect to the complexity measure π. Therefore by induction $\lambda x_1...x_n.S$ can be obtained by the grammar. Hence our term $\lambda x_1...x_n.T$ can be obtained by production q_i from $\lambda x_1...x_n.S$.

(case 3) If T has the form $x_i(\lambda z_1...z_p.S)$ for some term $S \in Cl(O, \{x_1...x_n, z_1...z_p\})$ where x_i has type $(O^p \to O) \to O$ then $\lambda x_1...x_n z_1...z_p.S$ is the closed term with regular type $\tau[1], \tau[2], ..., \tau[n], O^p \to O$. According to what we note there is at the most one occurrence of one type O variable among $x_1...x_n, z_1...z_p$ which occurs free in S.

(case 3a) If the only type O variable which occurs free in S is x_s for some $s \leq n$, then $\lambda x_1...x_n.S$ is a closed type τ term and is simpler than T with respect to the complexity measure π. Therefore by induction $\lambda x_1...x_n.S$ can be obtained by the grammar. Hence our term $\lambda x_1...x_n.T$ can be obtained by production l_i from $\lambda x_1...x_n.S$.

(case 3b) If the only type O variable which occurs free in S is z_s for some $1 \leq s \leq p$, then $\lambda x_1...x_r z_s x_{r+1}...x_n.S$ is closed type τ term and is simpler then T with respect to the complexity measure π. Therefore by induction $\lambda x_1...x_r z_s x_{r+1}...x_n.S$ can be obtained by the grammar. Hence our term $\lambda x_1...x_n.T$ can be obtained by production $l^r_{i,s}$ from the term $\lambda x_1...x_r z_s x_{r+1}...x_n.S$.

□

Theorem 3.4 *[see Zaionc [9]] For every regular rank 4 type, such that there is no such $i \leq arg(\tau)$ that $\tau[i] = O$, there is a finite grammar with two nonterminal variables generating all type τ closed terms.*

Proof. Let $\tau = \tau_1, \ldots, \tau_n \to O$. Let us define $\tau' = \tau_1, \ldots, \tau_n, O \to O$. The new type τ' satisfies the assumption of theorem 3.3, therefore there exists a grammar with one nonterminal type τ' variable y', generating all closed terms of τ'. Let us add to the existing grammar the following additional productions:

$$
\begin{array}{ll}
y \to \lambda x_1...x_n.x_i(yx_1...x_n) & \text{if } \tau[i] = O \to O \\
y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.y'x_1...x_n, z_1) & \text{if } \tau[i] = (O^p \to O) \to O \\
 & . \\
 & . \\
 & . \\
y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.y'x_1...x_n, z_p) & \text{if } \tau[i] = (O^p \to O) \to O
\end{array}
$$

The starting symbol for the new grammar is y. The proof of correctness of this construction is similar to the proof of theorem 3.3. □

4 Main result

Theorem 4.1 *Suppose there is a grammar with exactly one nonterminal, type τ variable y , generating all closed type τ terms. Then there exists effectively a complete λ system in D_τ for any full type hierarchy.*

Proof. Let $\{D_\tau\}_{\tau\in\mathbb{T}}$ be a *full type hierarchy* built up from some finite set D_O. First we identify all productions $y \to t$ such that there is no nonterminal variable in t. For each such production $p : y \to t$, term t is a closed type τ term. Let us evaluate t in the *full type hierarchy* to find an object $a_p \in D_\tau$. Then let us identify all productions $p : y \to t[y, ...y]$ such that there are some $k > 0$ occurrences of nonterminal variable y in t. For each such production we will define a function $\alpha_p : {D_\tau}^k \to D_\tau$. Let $p : y \to t[y, ...y]$ be a production with exactly $k > 0$ occurrences of nonterminal variable y in $t[y, ..., y]$. Let $t_1, ..., t_k$ be given objects from D_τ. Let us define the term $t[y_1, ..., y_k]$ obtained by replacement of each occurrence of the free variable y by the new type τ variables $y_1, ..., y_k$ respectively. Now let us define $\alpha_p(t_1, ..., t_k)$ as an object $V^\phi(t[y_1, ..., y_k])$ from D_τ where ϕ is an assignment associating y_i with t_i for all $i \leq k$. Therefore we have defined the function $\alpha_p : {D_\tau}^k \to D_\tau$. All such functions and such objects form the λ system in D_τ. We need to prove that such λ system is complete. Suppose an object $a \in D_\tau$ is λ definable. We are going to show that a belongs to the closure of the just constructed λ system. There is a closed λ term T_a interpreted as a. Term T_a has a finite derivation in the grammar G. If the length of the derivation is 1, it means that T_a is obtained from production $y \to T_a$. In this case interpretation of T_a is an $a \in D_\tau$ which belongs to the λ system. In the case the length of the derivation for T_a is longer than 1, T_a is obtained from some previous terms in the derivation sequence, let us say from $R_1, ..., R_k$, by some production $p : y \to t$ from the grammar. Let α_p be a function $\alpha_p : {D_\tau}^k \to D_\tau$ associated with the production $p : y \to t$. Let $b_1, ..., b_k$ be interpretations of closed terms $R_1, ..., R_k$ in D_τ. By induction we assume that $b_1, ..., b_k$ belong already to the closure of the λ system. From the construction of the λ system it follows that $a = f_p(b_1, ..., b_k)$. So $a \in D_\tau$ also belongs to the closure. □

Theorem 4.2 *For any rank 3 type τ or for any regular rank 4 τ for which there is $i \leq arg(\tau)$ such that $\tau[i] = O$ the λ definability problem is decidable in full type hierarchy built up from any finite set D_O.*

Proof. Given is an object $f \in D_\tau$. For type τ there is effectively a finite grammar with exactly one nonterminal variable generating all closed type τ terms . The following procedure applies. First we find the grammar (theorem 3.2 , 3.3) which generates all closed type τ terms . Then having the grammar we construct the λ system using procedure described in theorem 4.1. The λ system formed from such grammar must be complete (theorem 4.1). Finally, we find the closure of this system (lemma 2.4) which consists of all λ definable objects in D_τ. Then we check whether or not the given function

$f \in D_\tau$ belongs to this closure. □

Theorem 4.3 *Let τ be a regular rank 4 type such that there is no such $i \leq arg(\tau)$ that $\tau[i] = O$. The λ definability problem for τ is decidable in full type hierarchy built up from any finite set D_O.*

Proof. Given is an object $f \in D_\tau$. Suppose $\tau = \tau[1], ..., \tau[n] \to O$. Let us define a new type τ' as $\tau[1], ..., \tau[n], O \to O$. Type τ' satisfies assumptions of theorem 3.3 and therefore there is a finite grammar with exactly one nonterminal variable y' generating all closed type τ' terms. Using theorem 4.2 we construct effectively the finite set $D^\lambda_{\tau'}$ of all λ definable objects in $D_{\tau'}$. To generate closed type τ terms we add additional productions

(1)	$y \to \lambda x_1...x_n.x_i(yx_1...x_n)$	if $\tau[i] = O \to O$
(2.1)	$y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.y'x_1...x_n, z_1)$	if $\tau[i] = (O^p \to O) \to O$
		.
		.
		.
(2.p)	$y \to \lambda x_1...x_n.x_i(\lambda z_1...z_p.y'x_1...x_n, z_p)$	if $\tau[i] = (O, ..., O \to O) \to O$

Any type (1) production defines the unary function $\alpha : D_\tau \to D_\tau$ in the usual way (see theorem 4.1). Productions 2.1 - 2.p define p unary functions $\beta_1 : D'_\tau \to D_\tau, ..., \beta_p : D'_\tau \to D_\tau$. The following procedure applies. First find the image I of $D^\lambda_{\tau'}$ by all functions $\beta_1, ..., \beta_p$.

$$I = \bigcup_{1 \leq i \leq p} \beta_i(D^\lambda_{\tau'})$$

Then let us find the closure of the set I in $D\tau$ with respect to all functions α (see lemma 2.4). This closure is a set of all λ definable functions in D_τ. Finally, we check whether the given function $f \in D_\tau$ belongs to this closure. The proof of correctness of this procedure is identical with the proof of theorem 4.2. □

Theorem 4.4 *λ definability problem is decidable for all rank 1, 2, 3 types and for regular rank 4 types.*

Proof. For the rank 3 and regular rank 4 types see theorems 4.2 and 4.3. For the type O of rank 0 the λ definability problem is trivial since there is no λ definable objects in D_O. For rank 2 types the problem is decidable since for such types there are only finite number of closed terms. □

Theorem 4.5 *λ definability problem is undecidable for any non regular rank 4 type.*

Proof. Proof is based on the observation that the type $\mathbb{L} = ((O \to O) \to O) \to ((O \to (O \to O) \to O)$ is the simplest non regular type of rank 4. Therefore by simple lambda definable coding $\mathbb{L}$ can be embedded to any non

regular type of rank 4. But the λ definability problem is undecidable for $\mathbb{L}$ (see [1]) . □

Example 4.6 Let us consider the *full type hierarchy* built from the set $D_O = \{0, 1\}$. The problem is to find all λ definable elements in the set D_τ for type τ of Church's numerals $(O \to O) \to (O \to O)$. Let us name 4 elements of the set $D_{O\to O}$ respectively by a, b, c and d, where those are given by the following true table:

	0	1
a	0	0
b	0	1
c	1	0
d	1	1

The set D_τ consists of 256 functionals. The grammar for generating all Church's numerals is the following: (see theorem 3.2).

(i) $y \to \lambda ux.x$

(ii) $y \to \lambda ux.u(yux)$

The λ system $\{a_0, \alpha\}$ associated with the grammar consists of one lambda definable object a_0, which is an interpretation of the term $\lambda ux.x$, and one unary function α which is an interpretation of the production $y \to \lambda ux.u(yux)$. Object a_0 is given by the true-table.

$$a_0 = \{a \to b,\ b \to b,\ c \to b,\ d \to b\}$$

It can be found easily that the value of α on the object a_0 is another object a_1 given by:

$$a_1 = \{a \to a,\ b \to b,\ c \to c,\ d \to d\}$$

The value of α on a_1 is a new object, let us say a_2 given by:

$$a_2 = \{a \to a,\ b \to b,\ c \to b,\ d \to d\}$$

Since the value $\alpha(a_2) = a_1$ we have found all lambda definable functionals $\{a_0, a_1, a_2\}$ among all 256 elements of D_τ. In fact the object a_1 is an interpretation of all odd Church's numerals and a_2 is an interpretation of all positive even Church's numerals. The object a_0 is the interpretation of Church's zero.

Theorem 4.7 *As a trivial side effect of the result described in theorems 4.1 and 4.5 we may observe that there is no finite grammar generating all closed terms for any of non regular rank 4 types.*

5 Probabilistic approach

In this section we will consider probability of the fact that randomly chosen 4 order type has decidable lambda definability problem. So we investigate the size of the fraction of number of types of the given length n in which the λ definability problem is decidable in *full type hierarchy* against the number of all types of length n. We are specially interested in asymptotic behavior of this fraction. Our interest lays in finding limit of that fraction when $n \to \infty$. If the limit exists it represents the real number between 0 and 1 which we may call *the density of decidability*. We prove that rank 4 types with decidable λ definability problem are asymptotically empty meaning that the limit of that fraction is 0.
First we have to establish the way the length of types are measured.

Definition 5.1 *By $\|\tau\|$ we mean the length of type τ which we define as the total number of occurrences of atomic type O in the given type. Parenthesis which are sometimes necessary and arrow sign itself are not included in the length of type. Formally, $\|a_i\| = 1$ and $\|\phi \to \psi\| = \|\phi\| + \|\psi\|$.*

Definition 5.2 *We associate the density $\mu(\mathcal{A})$ with a subset $\mathcal{A} \subset \mathbb{T}$ of types as:*

$$\mu(\mathcal{A}) = \lim_{n\to\infty} \frac{\#\{\tau \in \mathcal{A} : \|\tau\| = n\}}{\#\{\tau \in \mathbb{T} : \|\tau\| = n\}} \tag{1}$$

if the limit exists.

The number $\mu(\mathcal{A})$ if exists is an asymptotic probability of finding type from the class $\mathcal{A}$ among all types or it can be interpreted as the asymptotic density of the set $\mathcal{A}$ in the set of types. It can be seen immediately that the density μ is finitely additive so if $\mathcal{A}$ and $\mathcal{B}$ are disjoint classes of formulas such that $\mu(\mathcal{A})$ and $\mu(\mathcal{B})$ exist then $\mu(\mathcal{A} \cup \mathcal{B})$ also exists and $\mu(\mathcal{A} \cup \mathcal{B}) = \mu(\mathcal{A}) + \mu(\mathcal{B})$. It is straightforward to observe that for any finite set $\mathcal{A}$ the density $\mu(\mathcal{A})$ exists and is 0. Dually for co-finite sets $\mathcal{A}$ the density $\mu(\mathcal{A}) = 1$. Nevertheless the density μ is not countably additive.

5.1 Elementary Counting of Types

In this section we present some properties of numbers characterizing the amount of types of different ranks. We may observe that many results and methods could be rephrased purely in terms of binary trees with given properties. Obviously any type of size n can be seen as a binary tree with n leaves. More advanced type counting can be found Moczurad, Tyszkiewicz, Zaionc [4]. See also Kostrzycka and Zaionc [2] and [11].

Definition 5.3 *By F_k^n and G_k^n we mean respectively the total number of types of rank k and size n and the total number of types of rank $\leq k$ and size n so:*

$$F_k^n = \#\{\phi \in \mathbb{T} : \|\phi\| = n\}, \tag{2}$$
$$G_k^n = \#\{\phi \in \mathbb{T} : \|\phi\| \leq n\}. \tag{3}$$

Lemma 5.4 *F_k^n and G_k^n are given by the following mutual recursion:*

$$F_1^n = \text{ if } n = 1 \text{ then } 1 \text{ else } 0 \tag{4}$$

$$G_1^n = \text{ if } n = 1 \text{ then } 1 \text{ else } 0 \tag{5}$$

$$F_{k+1}^n = \sum_{i=0}^{n} F_k^i G_k^{n-i} + \sum_{i=0}^{n} G_k^i F_{k+1}^{n-i} \tag{6}$$

$$G_{k+1}^n = G_k^n + F_{k+1}^n \tag{7}$$

Proof. The equality 4 and 5 are obvious since the type O is the only one of size 1. Recursive equation 7 is obtained directly from the definition of rank in section 1. The type $\tau \to \mu$ has rank $k+1$ either when τ is exactly of rank k and rank of μ is $\leq k$ or when rank of τ is $\leq k$ and rank of μ is exactly $k+1$. This leads to the recursive equation 6. □

5.2 Generating functions

We are going to use the generating function technique for proving the asymptotic behavior of appropriate fractions. For the purpose of counting for every $k \geq 1$ we define the pair of generating functions $f_k(z) = \sum_{i=0}^{\infty} F_k^i z^i$ and $g_k(z) = \sum_{i=0}^{\infty} G_k^i z^i$.

Lemma 5.5 *Functions f_k and g_k satisfy the following recursive definitions:*

$$f_{k+1}(x) = \frac{f_k(x) g_k(x)}{1 - g_k(x)} \tag{8}$$

$$g_{k+1}(x) = g_k(x) + f_{k+1}(x) \tag{9}$$

Proof. By simple encoding of 4, 5, 6 and 7 we get equations

$$f_1(x) = x \tag{10}$$

$$g_1(x) = x \tag{11}$$

$$f_{k+1}(x) = f_k(x) g_k(x) + g_k(x) f_{k+1}(x) \tag{12}$$

$$g_{k+1}(x) = g_k(x) + f_{k+1}(x) \tag{13}$$

Solving it proves the lemma □

Theorem 5.6 *Let $\mathcal{R} \subset \mathbb{T}$ be a set of all regular types of rank 4. Let R_n be the number of elements of $\mathcal{R}$ of size n. The generating function f_R for the sequence R_n is*

$$f_R(x) = \frac{x^3}{(1-2x)(1-x-x^2)}$$

Proof. We start with counting regular types of rank not greater then 3. Let L_n stands for the number of such types of size n. Obviously we get $L_0 = 0$, $L_1 = 1$ and moreover $L_n = L_{n-1} + L_{n-2}$ since any regular type of rank not greater then 3 must be of either of the form $O \to \tau$ or $(O \to O) \to \tau$ for some shorter regular type τ. As we can see L_n forms the sequence of Fibonacci

numbers. Therefore the generating function f_L for the sequence L_n is

$$f_L(x) = \frac{1}{1-x-x^2}.$$

By generic regular fourth order types we mean any type of the form $((O^k \to O) \to O) \to \tau$ for some regular type τ of rank not greater then 3 and for some $k \geq 1$. Let G_n stands for the number of such types of size n. Since for every $k \geq 1$ there is exactly one type of the form $((O^k \to O) \to O)$ the generating function f_G for the sequence G_n must be

$$f_G(x) = \frac{x^3}{1-x} \cdot f_L(x) = \frac{x^3}{(1-x)(1-x-x^2)}.$$

Finally we observe, that any regular fourth order type type is either generic regular forth order type or can be obtain from other shorter regular type τ of rank 4 by the construction $O \to \tau$ or $(O \to O) \to \tau$ or $((O^k \to O) \to O) \to \tau$. Let R_n stands for the number of all regular forth order types of size n. Therefore R_n has to satisfy the following recursive equation $R_n = G_n + \sum_{i=1}^{i=n-1} R_i$. This can be translated in to equation for the generating function f_R namely $f_R(x) = f_G(x) + x\frac{f_R(x)}{1-x}$. □

Theorem 5.7 *The density of rank* 4 *types with decidable* λ *definability problem among all rank 4 types is* 0.

Proof. It is enough to find the closed form for generating functions involved. Namely for all rank 4 types the function calculated from equality 8 is

$$f_4(x) = \frac{x^4}{1-5x+7x^2-2x^3}$$

Which can be easily turned into power serious $O\left(\left(\frac{3+\sqrt{5}}{2}\right)^n\right)$. The closed form term for the function $f_R(x)$ enumerating regular 4 order types returns rate of Fibonacci numbers namely $O\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$. □

It doesn't help much to add all third order types for which we know the lambda definability problem is decidable. Undecidability have again a dominating factor.

Theorem 5.8 *The density of types of rank* ≤ 4 *with the decidable* λ *definability problem among all types of rank* ≤ 4 *is again* 0.

Proof. It is enough to find the closed form for generation functions involved. For all rank ≤ 4 types the function is

$$g_4(x) = \frac{x-2x^2}{1-3x+x^2}$$

The respective power serious is again $O\left(\left(\frac{3+\sqrt{5}}{2}\right)^n\right)$. Function g_3 obtained from equality 9 is $\frac{x^3}{(1-2x)(1-x)}$. The closed form term for the function $f_R(x) + g_3(x)$ enumerating regular 4 order types plus all third order types returns rate of $O(2^n)$. □

References

[1] Thierry Joly, On the λ definability I, Problem at Fixed Model and Generalizations of the Matching Problem, to appear at Fundamentae Informaticae.

[2] Zofia Kostrzycka, Marek Zaionc, Statistics of intuitionistic versus classical logics, Studia Logica 76(3) (2004) 307 -328.

[3] Ralph Loader, The Undecidability of λ - Definability, *In Logic, Meaning and Computation: Essays in Memory of Alonzo Church,* 331-342, C.A. Anderson and Zeleny editors, Kluwer Academic Publishers, 2001.

[4] Małgorzata Moczurad, Jerzy Tyszkiewicz, Marek Zaionc, Statistical properties of simple types, Mathematical Structures in Computer Science 10 (2000), 575-594.

[5] Gordon D. Plotkin. λ definability and logical relations, Memorandum SAI-RM-4, School of Artificial Intelligence, University of Edinburgh, Ocober 1973

[6] Richard Statman. On the existence of closed terms in the typed λ-calculus. In: R. Hindley and J. Seldin, eds. Combinatory Logic, Lambda Calculus and Formalism (Academic Press, New York, 1980).

[7] Richard. Statman. Equality of functionals revisited, in L.A. Harrington et al. (Eds.), *Harvey Friedman's Research on the Foundations of Mathematics*, North-Holland, Amsterdam, 1985, 331-338.

[8] David A. Wolfram. The Clausual Theory of Types, Cambridge Tracts in Theoretical Computer Science 21, Cambridge University Press 1993.

[9] Marek Zaionc. The set of unifiers in typed λ calculus as regular expression, Rewriting techniques and Applications, Lecture Notes in Computer Science 202, Springer 1985, 430-440.

[10] Marek Zaionc. Word Operations Definable in the Typed λ calculus, Theoretical Computer Science 52, (1987) pp 1- 14.

[11] Marek Zaionc, On the asymptotic density of tautologies in logic of implication and negation, Reports on Mathematical Logic 39 (2004).