Lambda definability is decidable for second order types and for regular third order types [1]

Marek Zaionc [2]
Department of Computer Science, State University of New York at Buffalo
226 Bell Hall, Buffalo, NY 14260
zaionc@cs.buffalo.edu

Abstract: It has been proved by Loader [1] that Statman-Plotkin conjecture (see [4] and [2]) 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 for second order types and regular third order types is decidable in any finite domain. Obviously λ definability is decidable for 0 and 1 order types. As an additional effect of the result described we may observe that for certain types there is no finite grammar generating all closed terms.

1. Syntax of simple typed λ calculus

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

Definition 1.1. Type τ is called regular if $rank(\tau) \leq 3$ and every component of τ has $arg \leq 1$. This implies that only components allowed for regular types are O, $O \rightarrow O$ and $(O, ..., O \rightarrow O) \rightarrow O$.

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 $\tau \rightarrow \mu$ 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 $\tau \rightarrow \mu$. 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 $T = \lambda x_1 ... x_n . y T_1 ... T_k$, 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 $y T_1 ... T_k$ is a type O term. Long normal forms exist and are unique for $\beta\eta$ conversions (compare [3]). 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

[1] Supported by KBN research grant 0384/P4/93

[2] Permanent address of the author is: Computer Science Department, Jagiellonian University, Nawojki 11, 30-072 Krakow, Poland E-mail `zaionc@ii.uj.edu.pl`

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 $T = \lambda x_1 ... x_n . x_i T_1 ... T_k$, 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 [5].

2. Model of simple typed lambda calculus

A *full type hierarchy* $\{D_\tau\}_{\tau \in TYPES}$ 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:

1. $V^\phi(x) = \phi(x)$
2. $V^\phi(TS) = V^\phi(T)(V^\phi(S))$
3. $V^\phi(\lambda x_\tau . T_\mu)$ 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 $f \in \bigcup_{\tau \in TYPES} D_\tau$ 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

1. $D_L^0 = \{a_1, ..., a_p\}$
2. $D_L^{i+1} = D_L^i \cup \bigcup_{1 \leq j \leq k} \{\alpha_j(x_1, ..., x_{p_j}) : x_1, ..., x_{p_j} \in D_L^i\}$

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 [6]. A more detailed treatment of term grammars can be found in [5] or [7]. 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 $y \xrightarrow{*} T$ 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 $L(G, y) = \{t \in Cl(\tau) : y \xrightarrow{*} t\}$. 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.

1. $y \to \lambda px.x$
2. $y \to \lambda px.p(\lambda vz.ypx)$
3. $y \to \lambda px.p(\lambda vz.ypv)$
4. $y \to \lambda px.p(\lambda vz.ypz)$

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

1. $\lambda px.x$
2. $\lambda px.p(\lambda vz.x)$
3. $\lambda px.p(\lambda vz.p(\lambda x_6 x_7.v)$
4. $\lambda px_1.p(\lambda x_2 x_3.p(\lambda x_4 x_5.p(\lambda x_6 x_7.x_4)))$

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 [6] or theorem 3.3).

Theorem 3.2. For every type τ such that $rank(\tau) \leq 2$ there is a finite grammar with one type τ nonterminal variable which generates all closed type τ terms.

Proof: Let $\tau = \tau[1], \tau[2], ..., \tau[n] \rightarrow O$. 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:

1. $y \rightarrow \lambda x_1...x_n.x_i$ if $arg(\tau[i]) = 0$
2. $y \rightarrow \lambda x_1...x_n.x_i(yx_1...x_n)...(yx_1...x_n)$ 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 $\lambda x_1...x_n.x_i K_1...K_k$ 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 (2).

Theorem 3.3. (see Zaionc [6]) For every regular rank 3 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 $\tau = \tau[1], \tau[2], ..., \tau[n] \rightarrow O$. Every $\tau[i]$ is one of the following types: O, $O \rightarrow O$ or $(O, ..., O \rightarrow O) \rightarrow O$. 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 $O \rightarrow O$ components and $1{+}arg(\tau[i])$ productions for each second order component $\tau[i] = (O, ..., O \rightarrow O) \rightarrow O$. The productions have the form:

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

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 $x_i S$ 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 [6]) For every regular rank 3 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 $\tau = \tau[1], \tau[2], ..., \tau[n] \to O$. Let us define $\tau' = \tau[1], \tau[2], ..., \tau[n], O \to O$. The 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:

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

The start 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 $\{D_\tau\}_{\tau \in TYPES}$ 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 2 τ or for any regular rank 3 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 3 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

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

Any type (1) production defines the unary function $\alpha : D_\tau \rightarrow D_\tau$ in the usual way (see theorem 4.1). Productions 2.1 - 2.p define p unary functions $\beta_1 : D'_\tau \rightarrow D_\tau, ..., \beta_p : D'_\tau \rightarrow D_\tau$. 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 0, 1, 2 types and for regular rank 3 types.

Proof. For the rank 2 and regular rank 3 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 1 types the problem is decidable since for such types there are only finite number of closed terms.

Example 4.5. 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 $(O \rightarrow O) \rightarrow (O \rightarrow O)$. Let us name 4 elements of the set $D_{O\rightarrow O}$ 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).

1. $y \rightarrow \lambda ux.x$
2. $y \rightarrow \lambda ux.u(yux)$

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 $y \rightarrow \lambda ux.u(yux)$.
Object a_0 is given by the true-table.

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

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

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

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

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

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.

Observation. As a side effect of the result described in theorem 4.1 we may observe that there is no finite grammar generating all closed terms for certain types. In Loader paper [1] we have seen that certain types have been used for encoding of the word problem. For those types the λ definability problem is undecidable. Therefore there is no finite grammar for those types. Otherwise the problem would be decidable.

Acknowledgments The author would like to thank Bharat Jayaraman for inspiration and discussion and Anna Nizegorodcew for her efforts in careful proofreading.

References

[1] Ralph Loader. The Undecidability of λ - Definability, private communication, will be published in forthcoming Alonzo Church Festschrift book.

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

[3] R. 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).

[4] R. 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.

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

[6] M. 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.

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