Journal of Engineering and Applied Sciences 14 (21): 7853-7856, 2019
ISSN: 1816-949X

On Notion Of βδ-Reduction for Main Canonical Notion of δ-Reduction

D.A. Grigoryan
Chair of Programming and Information Technologies, Youngstown State University (YSU),
Yerevan, Armenia

Abstract: In this study, the notion of βδ-reduction for main canonical notion of δ-reduction is considered. Typed λ-terms use variables of any order and constants of order ≤1 where constants of order 1 are strongly computable, monotonic functions with indeterminate values of arguments. The canonical notion of δ-reduction is the notion of δ-reduction that is used in the implementation of functional programming languages. For main canonical notion of δ-reduction the uniqueness of βδ-normal form of typed λ-terms is shown.

Key words: Typed λ-terms, main canonical notion, uniqueness of βδ-normal form, monotonic functions, canonical notion, δ-reduction

INTRODUCTION

The canonical notion of δ-reduction as well as the main canonical notion of δ-reductions are introduced by Nigiyan and Khondkaryan (2017) where also shown that the main canonical notion of δ-reduction is a canonical notion of δ-reduction. In this research, we examine the uniqueness of βδ-normal form of typed λ-terms for main canonical notion of δ-reduction.

MATERIALS AND METHODS

Definitions of this study are take from the researches of Nigiyan and Khondkaryan (2017), Nigiyan (1992, 1993) and Budaghyan (2002). Let, M be a partially ordered set which has a least element ⊥ which corresponds to the indeterminate value and each element of M is comparable only with ⊥ and itself. Let us define the set of types (denoted by types):

- M∈Types,
- If β, α_1, …, α_k∈Types (k>0), then, the set of all monotonic mappings from $\alpha_1\times, \ldots, \times\alpha_k$ into β (denoted by $[\alpha_1\times, \ldots, \times\alpha_k\to\beta]$) belongs to types

Let α∈Types, then, the order of type α (denoted by ord (α)) will be natural number which is defined in the following way: if α = M then ord (α) = 0, if $\alpha = [\alpha_1 \times, \ldots, \times \alpha_k\to\beta]$ where, β, α_1, …, α_k∈Types, k>0 then ord (a) = 1+max (ord (α_1), …, ord (α_k), ord (β)). If x is a variable of type α and constant c∈α, then, ord (x) = ord (c) = ord (α).

Let α∈Types and V_α be a countable set of variables of type α, then, $V = \cup_{\alpha\in Types} V_\alpha$ is the set of all variables. The set of all terms, denoted by $\Lambda = \cup_{\alpha\in Types}$ where, Λ_α is the set of terms of type α is defined the following way:

- If c∈α, α∈Types, then $c\in\Lambda_\alpha$
- If $x\in V_\alpha$, α∈Types, then $x\in\Lambda_\alpha$
- If $\tau\in\Lambda_{[\alpha_1\times, \ldots, \times\alpha_k\to\beta]}$, $t_i\in\Lambda_{\alpha_i}$ where, β, α_i∈Types, i = 1, …, k, k≥1, then, τ (t_1, …, t_k)$\in\Lambda_\beta$ (the operation of application, (t_1, …, t_k) is the scope of the applicator τ)
- If $\tau\in\Lambda_\beta$, $x_i\in V_{\alpha_i}$ where, β, α_i,∈ Types, i≠j, $x_i\neq x$, i, j = 1, …, k, k≥1, then, $\lambda x_1, \ldots, x_k[\tau]\in\Lambda_{[\alpha_1\times\ldots\times\alpha_k\to\beta]}$ (the operation of abstraction, τ is the scope of the abstractor λ x_1, …, x_k)

The notion of free and bound occurrences of variables as well as free and bound variable are introduced in the conventional way. The set of all free variables in the term t is denoted by FV (t). Terms t_1 and t_2 are said to be congruent (which is denoted by ($t_1\equiv t_2$), if one term can be obtained from the other by renaming bound variables. The free occurrence of a variable in the term is called internal, if it does not enter in the applicator, the scope of which contains a free occurrence of some variable. The free occurrence of a variable in the term is called external, if it does not enter in the scope of the applicator that contains a free occurrence of some variable.

Let, $t\in\Lambda_\alpha$, α∈Types and FV(t)⊂{y_1, …, y_n} $\bar{y}_0 = (y_1^0, \ldots, y_n^0)$ where, $y_i\in V_{\beta i}$, $y_i^0\in\beta_i$, β_i∈Types, i = 1, …, n, n≥0. The value of the term t for the values of the variables y_1, …, y_n equal to $\bar{y}_0 = (y_1^0, \ldots, y_n^0)$ is denoted by $Val_{\bar{y}_0}(t)$ and is defined in the conventional way.

Let, terms t_1, $t_2\in\Lambda_\alpha$, α∈Types, FV(t_1)∪FV(t_2) = {y_1, ..., y_n}, $y_i\in V_{\beta_i}, y_i^0\in\beta_i$, β_i∈Types, i = 1, ..., n, n≥0 then, terms t_1 and t_2 are called equivalent (denoted by $t_1\sim t_2$), if for any $\bar{y}_0 = (y_1^0, \ldots, y_n^0)$ where, $y_1^0\in V_\beta, i = 1, \ldots, n$ we have the following: $Val_{\bar{y}_0}(t_1) = Val_{\bar{y}_0}(t_2)$. A term $t\in\Lambda_\alpha$, α∈Types is called a constant term with value a∈α, if t~a.

Further, we assume that M is a recursive set and considered terms use variables of any order and constants of order ≤ 1 where constants of order 1 are strongly computable, monotonic functions with indeterminate values of arguments. A function $f: M^k \rightarrow M$, $k \geq 1$ with indeterminate values of arguments is said to be strongly computable, if there exists an algorithm which stops with value $f(m_1, \ldots, m_k) \in M$ for all $m_1, \ldots, m_k \in M$ (Nigiyan, 2015).

To show mutually different variables of interest $x_1, \ldots, x_k$, $k \geq 1$ of a term t, the notation $t(x_1, \ldots, x_k)$ is used. The notation $t(t_1, \ldots, t_k)$ denotes the term obtained by the simultaneous substitution of the terms $t_1, \ldots, t_k$ for all free occurrences of the variables $x_1, \ldots, x_k$, respectively where $x_i \in V_{\alpha_i}$, $i \neq j \Rightarrow x_i \neq x_j$, $t_i \in \Lambda_{\alpha_i}$, $\alpha_i \in Types$, $i, j = 1, \ldots, k$, $k \geq 1$.

A substitution is said to be admissible, if all free variables of the term being substituted remain free after the substitution. We will consider only admissible substitutions.

A term t with a different fixed occurrences of subterms τ_1, τ_2 where, τ_1 is not a subterm of τ_2 and τ_2 is not a subterm of τ_1 and $\tau_i \in \Lambda_{\alpha_i}$, $\alpha_i \in Types$, $i = 1, 2$ is denoted by t_{τ_1, τ_2}. A term with the fixed occurrences of the terms τ'_1, τ'_2 replaced by the terms $\tau'_1 \tau'_2$, respectively is denoted by $t_{\tau'_1, \tau'_2}$ where, $\tau'_i \in \Lambda_{\alpha_i}$, $i = 1, 2$.

A term of the form $\lambda x_1, \ldots, x_k [\tau [x_1, \ldots, x_k]] (t_1, \ldots, t_k)$ where, $x_i \in V_\alpha$, $i \neq j \Rightarrow x_i \neq x_j$, $\tau \in \Lambda$, $t_i \in \Lambda_{\alpha_i}$, $\alpha_i \in Types$ $i, j = 1, \ldots, k$, $k \geq 1$ is called a β-redex, its convolution is the term $\tau_1[t_1, \ldots, t_k]$. The set of all pairs (τ_0, τ_1) where, τ_0 is a β-redex and τ_1 is its convolution is called a notion of β-reduction and is denoted by β. A one-step β-reduction ($\rightarrow\beta$) and β-reduction ($\rightarrow\rightarrow\beta$) are defined in the conventional way. A term containing no β-redexes is called a β-normal form. The set of all β-normal forms is denoted by β-NF.

δ-redex has a form $f(t_1, \ldots, t_k)$ where, $f \in [M^k \rightarrow M]$, $t_i \in \Lambda_M$, $i = 1, \ldots, k$, $k \geq 1$ its convolution is either $m \in M$ and in this case $f(t_1, \ldots, t_k) \sim m$ or a subterm t_i and in this case $f(t_1, \ldots, t_k) \sim t_i$, $i = 1, \ldots, k$. A fixed set of term pairs (τ_0, τ_1) where, τ_0 is a δ-redex and τ_1 is its convolution is called a notion of δ-reduction and is denoted by δ. A one-step δ-reduction ($\rightarrow\delta$) and δ-reduction ($\rightarrow\rightarrow\delta$) are defined in the conventional way.

A one-step $\beta\delta$-reduction ($\rightarrow$) and $\beta\delta$-reduction ($\rightarrow\rightarrow$) defined in the conventional way. A term containing no $\beta\delta$-redexes is called normal form. The set of all normal forms is denoted by NF.

A notion of δ-reduction is called a single-valued notion of δ-reduction, if δ is a single-valued relation, i.e., if $(\tau_0, \tau_1) \in \delta$ and $(\tau_0, \tau_1) \in \delta$, then, $\tau_1 \equiv \tau_2$ where $\tau_0, \tau_1, \tau_2 \in \Lambda_M$. A notion of δ-reduction is called an effective notion of δ-reduction if there exists an algorithm which for any term $f(t_1, \ldots, t_k)$ where, $f \in [M^k \rightarrow M]$. $t_i \in \Lambda_M$, $i = 1, \ldots, k$, $k \geq 1$, gives its convolution, if $f(t_1, \ldots, t_k)$ is a δ-redex and stops with a negative answer otherwise (Barendregt, 1981).

Definition 1; Nigiyan and Khondkaryan (2017): An effective, single-valued notion of δ-reduction is called a canonical notion of δ-reduction if:

- $t \in \beta$-NF, $t \sim m$, $m \in M \setminus \{\perp\} \Rightarrow t \rightarrow\rightarrow_\delta m$
- $t \in \beta$-NF, $FV(t) = \varnothing$, $t \sim \perp \Rightarrow t \rightarrow\rightarrow_\delta \perp$

Main canonical notion of δ-reduction, the uniqueness of the βδ-normal form

Definition 2: Let, C be a recursive set of strongly computable, monotonic functions with indeterminate values of arguments. The following notion of δ-reduction is called main canonical notion of δ-reduction if for every $f \in C$, $f: M^k \rightarrow M$, $k \geq 1$, we have:

- If $f(m_1, \ldots, m_k) = m$ where $m, m_1, \ldots, m_k \in M$, $m \neq \perp$, then $(f((\mu_1, \ldots, \mu_k) \in \delta$ where, $\mu_i = m_i$, if $m_i \neq \perp$ and $\mu_i \equiv t_i$, $t_i \in \Lambda_M$ if $m_i = \perp$, $i = 1, \ldots, k$, $k \geq 1$
- If $f(m_1, \ldots, m_k) = \perp$ where, $m_1, \ldots, m_k \in M$, then $(f(m_1, \ldots, m_k), \perp) \in \delta$

Nigiyan and Khondkaryan (2017) showed that the δ is a canonical notion of δ-reduction.

Definition 3: The term $t \in \Lambda$ is said to be strongly normalizable, if the length of each $\beta\delta$-reduction chain from the term t is finit.

RESULTS AND DISCUSSION

Theorem 1; Budaghyan (2002): Every term is strongly normalizable.

Theorem 2; Budaghyan (2002): For every term $t \in \Lambda$, if $\rightarrow\rightarrow_\beta t'$, $t \rightarrow\rightarrow_\beta t''$ and $t', t'' \in \beta$-NF then, $t' \equiv t''$.

Definition 4: Let, $t \in \Lambda_\alpha$, $\alpha \in Types$ and $t \equiv t_1 \rightarrow, \ldots, \rightarrow t_n$ $n \geq 1$ where, $t_i \in \Lambda_\alpha$, $i = 1, \ldots, n$, then, the sequence $t_1, \ldots, t_n$ is called the inference of the term t_n from the term t and n is called the length of that inference.

Definition 5: The inference tree of the term t is an oriented tree with the root t and if a term τ is some node of the tree and $\tau_1, \ldots, \tau_k$, $k \geq 0$ are all $\beta\delta$-redexes of τ, then, $\tau_{\tau'1}, \ldots, \tau_{\tau'k}$, $k \geq 0$ are all descendants of the node τ where, τ'_i is the convolution of τ_i, $i = 1, \ldots, k$.

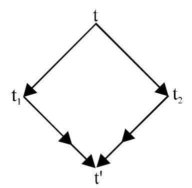
It is easy to see that each node in the inference tree of the term t has finite number of descendants and if τ is a leaf of that tree then $\tau \in NF$.

Theorem 3: For the main canonical notion of δ-reduction δ and for every term $t \in \Lambda$, if $t \rightarrow\rightarrow t'$, $t \rightarrow\rightarrow t''$ and $t', t'' \in NF$, then, $t' \equiv t''$. To prove theorem 3 let us prove lemma 1-3.

Lemma 1: Let, δ be the main canonical notion of δ-reduction, t_τ be a term with a fixed occurence of the term τ. If t is a δ-redex, τ is a $\beta\delta$-redex then there exists $m \in M$, $m \neq \perp$ such that $t \to_\delta m$ and $t_{\tau'} \to_\delta m$ where, τ' is the convolution of the τ which is a $\beta\delta$-redex.

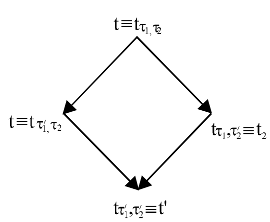
Proof: Let, $t_\tau \equiv f(t_1, ..., t_k, ..., t_k)$, $f \in [M^k \to M]$ $t_i \in \Lambda_M$, i = 1, ..., k, $1 \leq j \leq k$. Since, is a $\beta\delta$-redex, then, $t_k \notin M$ and since, t is a δ-redex then from the definition 2 follows that there exists $m \in M$, $m \neq \perp$ such that $(t, m) \in \delta$. Therefore, $t \to_\delta m$. Since, $t_k \notin M$ and $(t, m) \in \delta$ where, $m \neq \perp$, then, from the definition 2 follows that $f(t_1, ..., \mu, ..., t_k) \in \delta$ for every term $\mu \in \Lambda_M$. Therefore, $f(t_1, ..., t_{j\tau'}, ..., t_k), m) \in \delta$ and $t_{\tau'} \to_\delta m$.

Lemma 2: For the main canonical notion of δ-reduction δ and for every term $t \in \Lambda_\alpha$, $\alpha \in$ Types, if $t \to t_1$, $t \to t_2$, t_1, $t_2 \in \Lambda_\alpha$ then, there exists a term $t' \in \Lambda_\alpha$ such that $t_1 \to\to t'$ and $t_2 \to\to t'$:



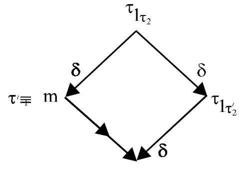
(1)

Proof: If $t_1 \equiv t_2$, then, $t' \equiv t_1 \equiv t_2$. If $t_1 \neq t_2$ then, there exist $\beta\delta$-redexes τ_1, $\tau_2 \in \Lambda$ such that $t \equiv t_{\tau_1} \equiv t_{\tau_2}$, $Wt_1 \equiv t_{\tau'_1}$ and $t_2 \equiv t_{\tau'_2}$ where terms τ'_1, τ'_2 are the convolutions of τ_1 and τ_2 accordingly. If τ_1 is not a subterm of τ_2 and τ_2 is not a subterm of τ_1, then from Eq. 2 follows that $t' \equiv t_{\tau'_1 \tau'_2}$:



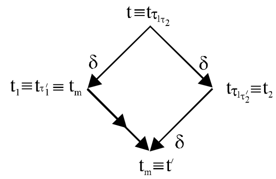
(2)

If τ_2 is a subterm of τ_1 or τ_1 is a subterm of τ_2, then, the following cases are possible: τ_1 and τ_2 are both δ-redexes. Without loss of generality we suppose that δ_2 is a subterm of $\tau_1 = (t_1 \equiv t_{1_{\tau_2}})$. From lemma Eq. 1 follows Eq. 3:



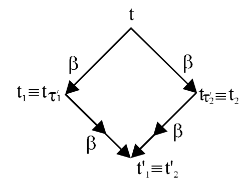
(3)

where, $m \in M$, $m \neq \perp$ and m is the convolution of the term τ_1. Therefore, from Eq. 4 follows that $t' \equiv t_m$:



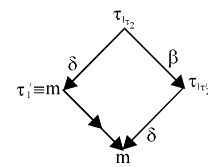
(4)

τ_1 and τ_2 are both β-redexes. From theorem 2 and Eq. 5 follows that $t' \equiv t'_1 \equiv t'_2$:



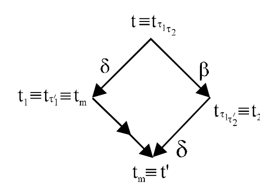
(5)

τ_1 is δ-redex and τ_2 is β-redex or τ_1 is β-redex and τ_2 is δ-redex. Without lose of generality, we suppose that τ_1 is δ-redex and τ_2 is β-redex. Let, $\tau_2 \equiv \lambda x_1, ..., x_n$ $[\tau [x_1, ..., x_n]]$ $(\mu_1, ..., \mu_n)$ $\tau \in \Lambda$, $x_i \in \Lambda$, x_i, V_{α_i}, $\alpha_i \in$ Types, i = 1, …, n. If $\tau_1 \equiv \tau_{1\tau_2}$ the from Lemma 1 follows Eq. 6:



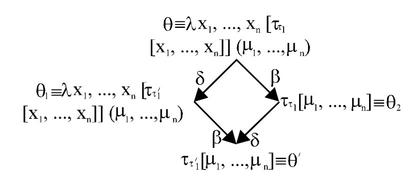
(6)

where $m \in M$, $m \neq \perp$ and m is the convolution of the term τ_1. Therefore, from Eq. 7 follows that $t' \equiv t_m$:



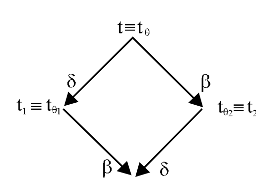
(7)

If $\tau_2 \equiv \lambda x_1, ..., x_n$ $[\tau_{\tau 1}$ $[x_1, ..., x_n]]$ $(\mu_1, ..., \mu_n)$, then, it is easy to see that if τ_1 $[x_1, ..., x_k] \to_\delta \tau'_1$, then, τ_1 $[\mu_1, ..., \mu_n] \to_\delta \tau'_1$ and we have:



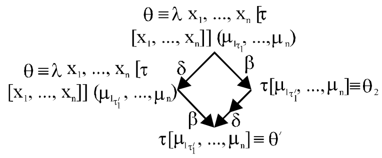
(8)

Therefore, Eq. 9 follows that $t' \equiv t_{\tau'_1[\mu_1, ..., \mu_n]}$:

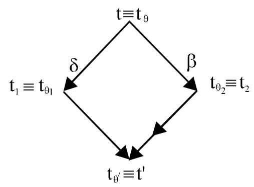


(9)

If $\lambda x_1, ..., x_n$ $[\tau$ $[x_1, ..., x_n]]$ $(\mu_1, ..., \mu_{i\tau 1}, ..., \mu_n)$, then, without loss of generality, we suppose that i = 1 and Eq. 10 and 11 follows that $t' \equiv t_{\tau[\mu_{1_{\tau'_1}}, ..., \mu_n]}$:



(10)



(11)

In conclusion, we showed that in all cases there exists a term t' such that $t_1 \to\to t'$ and $t_2 \to\to t'$.

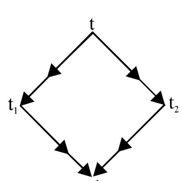
Lemma 3: For every term, the number of inferences to normal forms from that term is finite.

Proof: We consider the inference tree of the term t. Let us suppose that the number of inferences to normal forms from the term t is infinite which means that the number of paths from root t to leafs is also infinite. Since, every node in the inference tree has finite number of descendants, then from the Konig's lemma follows that there exists an infinite path that starts from the root t which contradicts to the theorem 1. Therefore, the number of paths from the root t to leafs is finite which means that the number of inferences to normal forms from the term t is also finite.

It follows from lemma 3 that for every term t the inference tree of the term t is a finite tree. The height of an inference tree of the term t is the length of the longest path from the root t to a leaf.

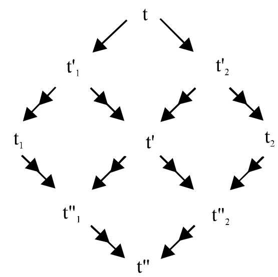
Definition 6: The set of all terms the height of the inference tree of which is equial to n-1 is denoted by $\Lambda^{(n)}$, $n \geq 1$.

Lemma 4: For every term $t \in \Lambda$, if $t \to\to t_1$ and $t \to\to t_2$, $t_1, t_2 \in \Lambda$, then, there exists a term $t' \in \Lambda$ such that $t_1 \to\to t'$ and $t_2 \to\to t'$:



(12)

Proof: Let, $t \in \Lambda^{(1)}$, then, $t \in NF$ and $t \equiv t_1 \equiv t_2 \equiv t'$. Now, let us suppose that the lemma 4 holds for every term $\tau \in \Lambda^{(k)}$, $k \leq n-1$, $n \leq 2$ and show that it holds for every term $t \in \Lambda^{(n)}$. If $t \equiv t_1$, then, $t_1 \to\to t_2$ and $t' \equiv t_2$. If $t \equiv t_2$ then $t_2 \to\to t_1$ and $t' \to t_1$. If $t_1 \neq t$ and $t_2 \neq t$, then, there exist terms t'_1, $t'_2 \in \Lambda$ such that $t \to t'_1 \to\to t_1$ and $t \to t'_2 \to\to t_2$. Therefore, from lemma 2 follows that there exists a term t' such that $t'_1 \to\to t'$ and $t'_2 \to\to t'$. Since, $t'_1 \to\to t_1$, $t'_1 \to\to t'$, and $t'_1 \in \Lambda^{(k_1)}$, $1 \leq k_1 \leq n-1$, then, from the induction hypothesis follows that there exists a term t"1 such that $t'_1 \to\to t'$ and $t' \to\to t''_1$. Since, $t'_2 \to\to t_2$, $t'_2 \to\to t'$ and $t'_2 \in \Lambda^{(k_2)}$, $1 \leq k_2 \leq n-1$, then, from the induction hypothesis follows that there exists a term t_2'' such that $t_2 \to\to t'_2$ and $t' \to\to t_2''$. Since, $t' \to\to t_1''$, $t' \to\to t''_2$ and $t' \in \Lambda^{(k_3)}$, $1 \leq k_3 \leq n-1$, then, from the induction hypothesis follows that there exists a term t" such that $t_1'' \to\to t''$ and $t_2'' \to\to t''$. Therefore, $t_1 \to\to t''$ and $t_2 \to\to t''$:



(13)

Proof of theorem 3: Let us suppose that the original statement is false and $t' \neq t''$ It follows from lemma 4 that there exists a term $t''' \in \Lambda$ such that $t' \to\to t'''$ and $t'' \to\to t'''$. Since, t', $t'' \in NF$ then $t' \equiv t'' \equiv t'''$. Therefore, we have a contradiction and the original statement is true.

CONCLUSION

In this study, the uniqueness of βδ-normal form of typed λ-terms for main canonical notion of δ-reduction has shown.

REFERENCES

Budaghyan, L.E., 2002. Formalizing the notion of δ-reduction in monotonic models of typed λ-calculus. Algebra, Geom. Appl., 1: 48-57.

Nigiyan, S.A. and T.V. Khondkaryan, 2017. On canonical notion of δ-reduction and on translation of typed λ-terms into untyped λ-terms. Proc. Yerevan State Univ. Phys. Math. Sci., 51: 46-52.

Nigiyan, S.A., 1992. Functional languages. Program. Comput. Software, 17: 77-86.

Nigiyan, S.A., 2015. On non-classical theory of computability. Proc. Yerevan State Univ. Phys. Math. Sci., 1: 52-60.