Mathematical Problems of Computer Science 32, 5–13, 2009.

# The Basic Semantics of Untyped Functional Programs and Reduction Strategies

Gurgen G. Hrachyan

Yerevan State University
gurgen.hrachyan@gmail.com

**Abstract**

The paper is devoted to untyped functional programs, which are defined as equation systems with separating variables in the untyped $\lambda$-calculus. The semantics of such programs is usually defined by means of the fix-point combinator $Y$. Previously, it was proved that the semantics of such programs is invariant with respect to the fix-point combinator. However, in this paper, we prove that this invariance is no longer valid when the reduction strategy is fixed.

## 1. Introduction

An untyped functional program is defined as a system of equations with separating variables in the untyped $\lambda$-calculus [1, 2]. The semantics of such programs is usually defined by means of fix-point combinator $Y$. According to the theorem on the invariance of the basic semantics [3], the basic semantics of the untyped functional programs that are defined by means of two different fix-point combinators are equivalent in the following sense. If the result of the application of one of them to some terms, (usually closed normal forms) can be reduced to a normal form, then the result of the application of the other one to the same terms, also can be reduced to the same normal form. However, from the current paper follows, that if we fix a reduction strategy, then generally speaking this equivalence will not be valid. We define and consider the class of so called active reduction strategies and the class of so called refined active strategies. It is proved that for all refined active reduction strategies that are not normalizing (i.e., that do not always lead to the normal form of the term if it exists), there exist fix-point combinators for which the mentioned equivalence is not true. The same is proved for not normalizing active strategies that are satisfying to a certain natural condition.

In the last section of the paper we consider programs consisting of one equation. The basic semantics of such programs can be defined in more natural and simpler way without redundant terms that are needed in the case of programs with multiple equations. First, based on [3, 4] it is proved that the basic semantics defined in this way is equivalent (in the sense mentioned above) to the basic semantics defined in the usual way. Then, it is proved that this equivalence is not valid when the reduction strategy is fixed.

## 2. Preliminaries

In this section, definitions and results used in the following discussion are presented. All definitions and assertions (given here without proofs) can be found in [1]. First of all, we define the concept of a term. Let $V$ be a countable set of variables. The set of terms $\Lambda$ is the least set that satisfies the following conditions:

1. If $x \in V$, then $x \in \Lambda$;
2. If $M_1, M_2 \in \Lambda$, then $(M_1 M_2) \in \Lambda$;
3. If $x \in V$ and $M \in \Lambda$, then $(\lambda x M) \in \Lambda$.

We will use the abridged notation for terms: the term $(\cdots(M_1 M_2)\cdots M_k)$, where $M_i \in \Lambda$, $i = 1, \cdots, k,\ k > 1$, is denoted as $M_1 \cdots M_k$; term $(\lambda x_1(\lambda x_2(\cdots(\lambda x_m M)\cdots)))$, where $M \in \Lambda$, $x_j \in V, j = 1, \cdots, m,\ m > 0$, is denoted as $\lambda x_1 x_2 \cdots x_m.M$.
The notions of free and bound occurrences of a variable in a term and the notion of a free variable of a term are introduced in a conventional way. The set of all free variables of a term $M$ is denoted as $FV(M)$. A term that does not contain free variables is said to be closed. The set of all closed terms is denoted as $\Lambda^0$.
To show mutually different variables of interest $x_1, \cdots, x_m$, $m \geq 1$, of a term $M$, the notation $M[x_1, \cdots, x_m]$ is used. The notation $M[N_1, \cdots, N_m]$ (or $M[x_1 := N_1, \cdots, x_k := N_m]$) denotes the term obtained by simultaneous substitution of terms $N_1, \cdots, N_m$ into the term $M$ for all free occurrences of variables $x_1, \cdots, x_m$, respectively. A substitution is said to be admissible if all free variables of the term being substituted remain free after the substitution. In what follows, we will consider only admissible substitutions.
Terms $M$ and $N$ are said to be congruent (notation $M \equiv N$), if one term can be obtained from the other by renaming bound variables. In what follows, congruent terms are considered identical.
The notion of $\beta$-reduction is defined as the following set of pairs:

$$\beta = \{((\lambda x.P[x])Q,\ P[x := Q]) \mid P, Q \in \Lambda, x \in V\}$$

A term of the form $(\lambda x.P[x])Q$ is called a $\beta$-redex (further, simply redex), and the term $P[x := Q]$ is called its contractum. The relation of one-step $\beta$-reduction ($\longrightarrow_\beta$) is defined as follows:

1. if $(M_1, M_2) \in \beta$, then $M_1 \longrightarrow_\beta M_2$;
2. if $M_1 \longrightarrow_\beta M_2$, then for an arbitrary term $M$ and arbitrary variable $x$ we have:

1. $MM_1 \longrightarrow_\beta MM_2$
2. $M_1 M \longrightarrow_\beta M_2 M$
3. $\lambda x.M_1 \longrightarrow_\beta \lambda x.M_2$

It is easy to see, that $M_1 \longrightarrow_\beta M_2$ means, that the term $M_2$ is obtained from the term $M_1$ by replacing an occurrence of a redex in the term $M_1$ by its contractum. The relation of $\beta$-reduction ($\twoheadrightarrow_\beta$) is defined as the reflexive and transitive closure of the one-step $\beta$-reduction, and the relation of $\beta$-equality ($=_\beta$) is defined as follows:

1. if $M_1 \twoheadrightarrow_\beta M_2$, then $M_1 =_\beta M_2$;

2. if $M_1 =_\beta M_2$, then $M_2 =_\beta M_1$;

3. if $M_1 =_\beta M_2$ and $M_2 =_\beta M_3$, then $M_1 =_\beta M_3$;

In what follows, the one-step $\beta$-reduction will be referred to as simply one-step reduction, and the $\beta$-reduction, as simply reduction. We will also omit the symbol $\beta$ in the corresponding notations, i.e. the relation $\longrightarrow_\beta$ we will denote as $\longrightarrow$, the relation $\twoheadrightarrow_\beta$ as $\twoheadrightarrow$, and the relation $=_\beta$ as $=$.
Since the $\beta$-reduction is the reflexive and transitive closure of the one-step $\beta$-reduction, from $M \twoheadrightarrow N$ it follows that there exist terms $M_0, \cdots, M_k$ $(k \geq 0)$ such that $M_0 \equiv M$, $M_k \equiv N$ and $M_{i-1} \longrightarrow M_i$, $i = 1, \cdots, k$. Taking this into account, further, we will use the term reduction not only in the sense of relation $\twoheadrightarrow$ but also in the sense of a particular (possibly, empty) sequence of one-step reductions; i.e., saying that we have a reduction $M \twoheadrightarrow N$, we mean the particular sequence of one-step reductions from term $M$ to term $N$.
Term that does not contain redexes is referred to as a $\beta$-normal form (further, simply a normal form). The set of all normal forms is denoted by $NF$, and the set of all closed normal forms, by $NF^0$. A term $M$ is said to have a normal form, if there exists a term $N \in NF$ such that $M = N$.
It is known that every term $M$ has one of the following forms:

$$M \equiv \lambda x_1 \cdots x_k.xN_1 \cdots N_n,$$

$$M \equiv \lambda x_1 \cdots x_k.(\lambda xP)QN_1 \cdots N_n,$$

where $x, x_i \in V$, $i = 1, \cdots, k$, $k \geq 0$, $P, Q, N_j \in \Lambda$, $j = 1, \cdots, n$, $n \geq 0$. A term of the first form is called a head normal form. The set of all head normal forms is denoted as $HNF$. It is said that the term $M$ has a head normal form if there exists a term $N \in HNF$ such that $M = N$. It is known that $NF \subset HNF$, but $HNF \not\subset NF$.

**Church-Rosser Theorem.**

a. If $M \twoheadrightarrow M_1$ and $M \twoheadrightarrow M_2$, then there exists a term $N$ such that $M_1 \twoheadrightarrow N$ and $M_2 \twoheadrightarrow N$.

b. If $M_1 = M_2$, then there exists a term $N$ such that $M_1 \twoheadrightarrow N$ and $M_2 \twoheadrightarrow N$.

**Corollary of Church-Rosser Theorem.**

a. If $M = N$, where $N$ is a normal form, then $M \twoheadrightarrow N$.

b. If $M = N_1$ and $M = N_2$, where $N_1$ and $N_2$ are normal forms, then $N_1 \equiv N_2$.

**Fix-point Theorem.** There exists a term $Z$ such that, its application to an arbitrary term $M$, yields a fix-point of the term $M$, i.e., $M(ZM) = ZM$.

A term $Z$ that satisfies to the condition of fix-point theorem (i.e. for any term $M$, $M(ZM) = ZM$) is called a fix-point combinator.

To formulate the multiple fix-point theorem, we introduce the following notation:

$$< M_1, \cdots, M_n > \equiv \lambda x.xM_1 \cdots M_n,$$

where $x \in V$, $M_i \in \Lambda$, $x \notin FV(M_i)$, $i = 1, \cdots, n$, $n \geq 1$;

$$U_i^n \equiv \lambda x_1, ..., x_n.x_i,$$

where $x_j \in V$, $j \neq k \Rightarrow x_j \not\equiv x_k$, $k, j = 1, \cdots, n$, $n \geq 1$;

$$P_i^n \equiv \lambda x.xU_i^n,$$

where $x \in V$, $1 \leq i \leq n$, $n \geq 1$.

**Multiple Fix-point Theorem.** Let $M_1 \cdots M_n$ ($n \geq 1$) be arbitrary terms, $Z$ be a fix-point combinator. Then,

$$M_i L_1^Z \cdots L_n^Z = L_i^Z,$$

where

$$L_i^Z \equiv P_i^n(Z(\lambda x. < M_1(P_1^n x) \cdots (P_n^n x), \cdots, M_n(P_1^n x) \cdots (P_n^n x) >)),$$

$i = 1, \cdots, n$.

## 3. Untyped Functional Program

Untyped functional program $P$ (see [2]) is an equation system of the following form:

$$\begin{aligned} f_1 &= M_1[f_1, \cdots, f_n], \\ &\cdots \\ f_n &= M_n[f_1, \cdots, f_n], \end{aligned} \tag{1}$$

where $f_i \in V$, $i \neq j \Rightarrow f_i \not\equiv f_j$, $M_i[f_1 \cdots, f_n] \in \Lambda$, $FV(M_i[f_1 \cdots, f_n]) \subseteq \{f_1, \cdots, f_n\}$, $i, j = 1, \cdots, n$, $n \geq 1$. It is considered, that the main equation of the program $P$ is the first equation of the system. Let us fix a fixpoint combinator $Z$ and consider the following solution of the system:

$$(L_1^Z, \cdots, L_n^Z),$$

where

$$L_i^Z \equiv P_i^n(Z(\lambda x. < M_1[P_1^n x, \cdots, P_n^n x], \cdots, M_n[P_1^n x, \cdots, P_n^n x] >)),$$

$i = 1, \cdots, n$. The first (main) component of the solution, i.e. the term $L_1^Z$ is referred to as the fix-point semantics or the basic semantics of the program $P$, corresponding to the fix-point combinator $Z$. Let us define the set $Fix(P, Z)$, corresponding to the semantics $L_1^Z$ as follows:

$$Fix(P, Z) = \{(Q_1, \cdots, Q_k, M_0) \mid L_1^Z Q_1 \cdots Q_k \twoheadrightarrow M_0, Q_1, \cdots, Q_k, M_0 \in NF^0, k \geq 0\}.$$

In [3] the following lemma on the equivalence of fix-point combinators and theorem on invariance of the basic semantics are proved.

**Lemma on the Equivalence of Fix-point Combinators.** Let $Z$ and $Z'$ be arbitrary fix-point combinators, $M$ be a term with a fixed occurrence of a subterm $ZL$ ($L \in \Lambda$) and $M'$ be the term that is obtained from $M$ by replacing the above-mentioned occurrence of subterm $ZL$ by the term $Z'L$. Suppose also that free occurrences of variables in the corresponding occurrences of subterms $ZL$ and $Z'L$ are not bound in terms $M$ and $M'$, respectively. Then, $M_0$ is the normal form of term $M$ if and only if $M_0$ is the normal form of term $M'$.

**Theorem on Invariance of the Basic Semantics.** For any two fix-point combinators $Z_1$, $Z_2$ and any untyped functional program $P$ we have:

$$Fix(P, Z_1) = Fix(P, Z_2),$$

i.e. the set, corresponding to the basic semantics is invariant with respect to fix-point combinator. This means, that if $L_1$ and $L_2$ are the semantics of the Untyped functional program $P$, corresponding to the fix-point combinators $Z_1$ and $Z_2$, respectively, then for any terms $Q_1, \cdots, Q_k, M_0 \in NF^0$ $(k \geq 0)$:

$$L_1Q_1 \cdots Q_k \twoheadrightarrow M_0 \Leftrightarrow L_2Q_1 \cdots Q_k \twoheadrightarrow M_0 \quad (2)$$

## 4. Reduction Strategies

A reduction strategy is an arbitrary mapping $S : \Lambda \to \Lambda$ such that for any term $M$, $M \twoheadrightarrow S(M)$. According to the corollary of Church-Rosser theorem, if $M \in NF$, then $M \equiv S(M)$. Reduction strategy $S$ is said to be one-step, if for $M \in \Lambda \setminus NF$, $M \longrightarrow S(M)$. The reduction $M \twoheadrightarrow N$ is called an $S$ reduction, if $N \equiv S^n(M)$, where $n \geq 0$ and $S^n(M)$ is defined as follows: $S^0(M) \equiv M$, $S^{k+1}(M) \equiv S(S^k(M))$, $k \geq 0$. For $S$ reduction $M \twoheadrightarrow N$, we will use the notation $M \overset{S}{\twoheadrightarrow} N$, and for one-step $S$ reduction $M \longrightarrow N$ the notation $M \overset{S}{\longrightarrow} N$. Reduction strategy $S$ is said to be normalizing, if for each term $M$, having the normal form $M_0$, we have $M \overset{S}{\twoheadrightarrow} M_0$.

Let $S$ be a reduction strategy, $Z_1$, $Z_2$ be fix-point combinators, $P$ be a program of form (1) and $L_1$, $L_2$ be the basic semantics of the program $P$ corresponding to the fix-point combinators $Z_1$ and $Z_2$, respectively. Then, according to the theorem on the invariance of the basic semantics, for any terms $Q_1, \cdots, Q_k, M_0 \in NF^0$ $(k \geq 0)$:

$$L_1Q_1 \cdots Q_k \twoheadrightarrow M_0 \Leftrightarrow L_2Q_1 \cdots Q_k \twoheadrightarrow M_0$$

A question arises: does the same thing hold for the reduction strategy $S$, i.e. is the following valid?

$$L_1Q_1 \cdots Q_k \overset{S}{\twoheadrightarrow} M_0 \Leftrightarrow L_2Q_1 \cdots Q_k \overset{S}{\twoheadrightarrow} M_0 \quad (3)$$

Reduction strategy is defined as an arbitrary mapping with the only constraint that this mapping should be a subset of the $\beta$-reduction relation. Hence, it is easy to construct an artificial strategy, for which (3) is not valid. In the view of this fact, we will consider the mentioned question only for classes of natural strategies, which we are going to define now.

Let us define the class of active reduction strategies and the class of refined active reduction strategies. All strategies of these classes will be one-step strategies. Let $\Delta$ be an arbitrary set of terms, containing all normal forms, i.e. $NF \subseteq \Delta \subseteq \Lambda$. The active reduction strategy $AS_\Delta$ is defined as follows. Let $(\lambda x.P)Q$ be the leftmost redex of the term $M$. Then,

1. if $Q \in \Delta$, then $AS_\Delta(M)$ is defined as the result of the one-step reduction of the term $M$, upon which, the leftmost redex $(\lambda x.P)Q$ is contracted (i.e. replaced by its contractum);

2. otherwise, if $Q \notin \Delta$, then $AS_\Delta(M)$ is defined as the term that is obtained from $M$ by replacing the considered occurrence of the subterm $Q$ by $AS_\Delta(Q)$.

The refined active reduction strategy $ASR_\Delta$ is defined as follows. Let $(\lambda x.P)Q$ be the leftmost redex of the term $M$. Then,

1. if $Q \in \Delta$ or $x \notin FV(P)$, then $AS_\Delta(M)$ is defined as the result of the one-step reduction of the term $M$, upon which, the leftmost redex $(\lambda x.P)Q$ is contracted;
2. otherwise, if $Q \notin \Delta$, then $AS_\Delta(M)$ is defined as the term that is obtained from $M$ by replacing the considered occurrence of the subterm $Q$ by $AS_\Delta(Q)$.

The point in the second definition is that, if the subterm $P$ of the leftmost redex $(\lambda x.P)Q$, do not contain free occurrences of the variable $x$, then the term $Q$ can be ignored and there is no sense in reducing it. So in this case the redex $(\lambda x.P)Q$ is contracted. The active strategy $AS_{NF}$ we will denote by $ANF$, the refined active strategy $ASR_{NF}$ by $ANFR$, the active strategy $AS_{HNF}$ by AHNF and the refined active strategy $ASR_{HNF}$ by $AHNFR$.

## 5. Non-invariance of the Basic Semantics in the Case of Active and Refined Active Reduction Strategies

In this section we introduce theorems 1,2 and their corollaries that show the non-invariance of the basic semantics for the active and refined active reduction strategies.

**Theorem 1** *If the reduction strategy $AS_\Delta (NF \subseteq \Delta \subseteq \Lambda)$ is not normalizing and $\Delta$ does not contain any term of form $\lambda y.(\lambda x.P)Q$ $(x, y \in V,\ P, Q \in \Lambda)$, then there exist a program $P$, fix-point combinators $Z_1$, $Z_2$ and terms $N_1, \cdots, N_k \in NF^0$ $(k \geq 0)$ such that if $L_1$, $L_2$ are the basic semantics of program $P$ corresponding to the fix-point combinators $Z_1$, $Z_2$, respectively, then $L_1 N_1 \cdots N_k \xrightarrow{AS_\Delta} M_0 \in NF$, but the $AS_\Delta$ reduction of the term $L_2 N_1 \cdots N_k$ continues infinitely.*

**Corollary of Theorem 1.** For the reduction strategy $ANF$ ($AHNF$), there exist a program $P$, fix-point combinators $Z_1$, $Z_2$ and terms $N_1, \cdots, N_k \in NF^0$ $(k \geq 0)$ such that if $L_1$, $L_2$ are the basic semantics of program $P$ corresponding to the fix-point combinators $Z_1$, $Z_2$, respectively, then $L_1 N_1 \cdots N_k \xrightarrow{ANF} M_0 \in NF$ ($L_1 N_1 \cdots N_k \xrightarrow{AHNF} M_0 \in NF$), but the $ANF$ ($AHNF$) reduction of the term $L_2 N_1 \cdots N_k$ continues infinitely.

**Theorem 2** *If the reduction strategy $ASR_\Delta (NF \subseteq \Delta \subseteq \Lambda)$ is not normalizing, then there exist a program $P$, fix-point combinators $Z_1$, $Z_2$ and terms $N_1, \cdots, N_k \in NF^0$ $(k \geq 0)$ such that if $L_1$, $L_2$ are the basic semantics of program $P$ corresponding to the fix-point combinators $Z_1$, $Z_2$, respectively, then $L_1 N_1 \cdots N_k \xrightarrow{ASR_\Delta} M_0 \in NF$, but the $ASR_\Delta$ reduction of the term $L_2 N_1 \cdots N_k$ continues infinitely.*

**Corollary of Theorem 2.** For the reduction strategy $ANFR$ ($AHNFR$), there exist a program $P$, fix-point combinators $Z_1$, $Z_2$ and terms $N_1, \cdots, N_k \in NF^0$ $(k \geq 0)$ such that if $L_1$, $L_2$ are the basic semantics of program $P$ corresponding to the fix-point combinators $Z_1$,

$Z_2$, respectively, then $L_1N_1\cdots N_k \xrightarrow{ANFR} M_0 \in NF$ ($L_1N_1\cdots N_k \xrightarrow{AHNFR} M_0 \in NF$), but the $ANFR$ ($AHNFR$) reduction of the term $L_2N_1\cdots N_k$ continues infinitely.

To prove the theorems 1 and 2 we need the following lemmas 1 and 2, which are given here without proof for the sake of brevity.

**Lemma 1** *If the strategy $AS_\Delta$ ($NF \subseteq \Delta \subseteq \Lambda$) is not normalizing, then there exists a fix-point combinator $Z$ such that for any untyped functional program $P$ and any terms $N_1,\cdots,N_k \in \Lambda$ ($k \geq 0$), if $L$ is the semantics of the program $P$ corresponding to $Z$, then the $AS_\Delta$ reduction of the term $LN_1\cdots N_k$ continues infinitely.*

**Lemma 2** *If the strategy $ASR_\Delta$ ($NF \subseteq \Delta \subseteq \Lambda$) is not normalizing, then there exists a fix-point combinator $Z$ such that for any untyped functional program $P$ and any terms $N_1,\cdots,N_k \in \Lambda$ ($k \geq 0$), if $L$ is the semantics of the program $P$ corresponding to $Z$, then the $ASR_\Delta$ reduction of the term $LN_1\cdots N_k$ continues infinitely.*

**Proof of Theorem 1.** Let $AS_\Delta(NF \subseteq \Delta \subseteq \Lambda)$ be a not normalizing reduction strategy such that $\Delta$ does not contain any term of form $\lambda y.(\lambda x.P)Q$ ($x,y \in V$, $P,Q \in \Lambda$). Let $P$ be the program $f = I$, where $I \equiv \lambda x.x$ and let $Z_1 \equiv Y \equiv (\lambda h.(\lambda x.h(xx))(\lambda x.h(xx)))$. In this case, it can be straightforwardly verified that if $L_1$ is the semantics of the program $P$ corresponding to $Z_1$, then $L_1 \xrightarrow{AS_\Delta} I$. On the other hand, according to lemma 2, there exists a fix-point combinator $Z_2$ such that, if $L_2$ is the basic semantics of the program $P$ corresponding to $Z_2$, then the $AS_\Delta$ reduction of the term $L_2$ continues infinitely. Theorem 1 is proved.

**Proof of Theorem 2.** The proof of theorem 2 is similar to the proof of theorem 1 with the difference, that as $Z_1$ the fix-point combinator $Z_1 \equiv \Theta \equiv (\lambda xy.y(xxy))(\lambda xy.y(xxy)))$ is taken and instead of lemma 1 the lemma 2 is used .

## 6. The Basic Semantics of Programs with One Equation

Consider, the following program $P$ that consists of one equation:

$$f = M[f]. \tag{4}$$

Let $Z$ be an arbitrary fix-point combinator. In this case the term

$$K^Z \equiv Z(\lambda f.M[f]). \tag{5}$$

is a solution of the equation (4). Let us prove the following theorem 3 which states that the term $K^Z$ is equivalent to the basic semantics of the program $P$ in the sense of (2).

**Theorem 3** *Let $Z$ be an arbitrary fix-point combinator, $P$ be the program (4), $L^Z$ be the basic semantics of the program $P$ corresponding to $Z$, and $K^Z$ be the term (5). Then, $L^Z$ is equivalent to $K^Z$ in the sense of (2).*

**Proof.** In [4] it is proved that if $Z \equiv Y \equiv (\lambda h.(\lambda x.h(xx))(\lambda x.h(xx)))$, then the solution $K^Y$ is equivalent to the semantics $L^Y$ in the sense of (2). In the view of the lemma on equivalence of the fix-point combinators and of the theorem on the invariance of the basic semantics, it is easy to prove that this equivalence holds for arbitrary fix-point combinator $Z$. Indeed, according to the lemma on equivalence of the fix-point combinators, $K^Z$ is equivalent to $K^Y$

(in the sense of (2)). As we mentioned above, $K^Y$ is equivalent to $L^Y$, and finally, according to the theorem on invariance of the basic semantics, $L^Y$ is equivalent to $L^Z$. Theorem 3 is proved.

Taking into account this fact, it makes sense to define the basic semantics of programs consisting of one equation, as the term (5). Note, that from the theorem 3 immediately follows, that the semantics defined using (5) will also be invariant with respect to the fix-point combinator in the sense of the theorem on invariance of the basic semantics.

However, as it follows from the following theorem 4, these two definitions are no longer equivalent when the reduction strategy is fixed. This fact is proved for the reduction strategy $ANFR$.

**Theorem 4** *There exists an untyped functional program $P$, consisting of one equation, fix-point combinator $Z$ and terms $N_1, \cdots, N_k \in NF^0$ ($k \geq 0$) such that if $L$ is the basic semantics of the program $P$ corresponding to $Z$ defined in the usual way and $K$ is the basic semantics of the program $P$ corresponding to $Z$ defined according to (5), then $KN_1 \cdots N_k \xrightarrow{ANFR} M_0 \in NF$, but the $ANFR$ reduction of the term $LN_1 \cdots N_k$ continues infinitely.*

**Proof.** Before proving theorem 4 let us introduce some conventional notations.

Terms $T \equiv \lambda xy.x$ and $F \equiv \lambda xy.y$ are used to represent the logical values $true$ and $false$, respectively. Let $B$, $M$ and $N$ be arbitrary terms, then by the expression $if\ B\ then\ M\ else\ N$ is denoted the term $BMN$. It is easy to see, that if $B = T$, then $BMN = M$ and if $B = F$, then $BMN = N$. Let us define terms for representing numbers. For every number $n \geq 0$, the term $\ulcorner n \urcorner$ is defined as follows:

$$\ulcorner 0 \urcorner \equiv \lambda x.x, \ulcorner n+1 \urcorner \equiv \lambda x.xF\ulcorner n \urcorner$$

These terms are known as Church numerals. We will also use the following notations:

$$P^- \equiv \lambda x.xF,\ Zero \equiv \lambda x.x.T \text{ and } I \equiv \lambda x.x.$$

One can show, that for any number $n \geq 0$, $P^-\ulcorner n+1 \urcorner = \ulcorner n \urcorner$, if $n = 0$, then $Zero\ \ulcorner n \urcorner = T$, if $n > 0$, then $Zero\ \ulcorner n \urcorner = F$ and that for any term $M$, $IM = M$.

Let $Z \equiv \lambda h.I(\lambda x.h(xx))(\lambda x.h(xx))$ and let $P$ be the following program:

$$f = M[f] \equiv \lambda n.if\ (Zero\ n)\ then\ \ulcorner 0 \urcorner\ else\ f(P^- n).$$

In this case, it can be proved that if $L$ is the basic semantics of the program $P$ corresponding to $Z$ defined in the usual way and $K$ is the basic semantics of the program $P$ corresponding to $Z$ defined according to (5), then $K\ulcorner 0 \urcorner \xrightarrow{ANFR} \ulcorner 0 \urcorner$, but the $ANFR$ reduction of the term $L\ulcorner 0 \urcorner$ continues infinitely. Theorem 4 is proved.

## References

[1] H. P. Barendregt, "The Lambda Calculus, Its Syntax and Semantics", *North-Holland Pub. Comp.*, 1981.

[2] S. A. Nigiyan and S. A. Avetisyan, "On the Semantics of Untyped Functional Programs", *Programming and Computer Software*, Vol. 28, N 3, pp. 119-126, 2002.

[3] G. G. Hrachyan, "The Invariance of the Main Semantics of Type-Free Functional Programs Relatively to the Fixpoint Combinator", *Proceedings of the Conference CSIT-2007*, Yerevan, pp. 48-50.

[4] S. A. Avetisyan, "On the Semantics of Untyped Functional Programs", *PHD Thesis*, Yerevan, 2005.

## Առանց տիպերի ֆունկցիոնալ ծրագրերի հիմնական սեմանտիկան և ռեդուկցիոն ստրատեդիաները

Գ. Հրաչյան

**Ամփոփում**

Աշխատանքը նվիրված է առանց տիպերի ֆունկիոնակալ ծրագրերին, որոնք իրենցից ներկայացնում են հավասարումների համակարգեր առանց տիպերի $\lambda$ - հաշվում: Նման ծրագրերի հիմնական սեմանտիկան ինչպես կանոն սահմանվում է $Y$ անշարժ կետի կոմբինատորի միջոցով: Նախկինում ապացուցվել է , որ հիմնական սեմանտիկան ինվարիանտ է անշարժ կետի կոմբինատորի նկատմամբ: Սակայն, այս աշխատանքից հետևում է , որ տրված ռեդուկցիոն ստրատեգիայի համար նման ինվարիանտություն տեղի չունի: