Mathware & Soft Computing 15 (2008) 273-283

Infinitary Simultaneous Recursion Theorem

D. Vaggione
FaMAF, Universidad Nacional de Córdoba
Argentina

Abstract

We prove an infinitary version of the Double Recursion Theorem of Smullyan. We give some applications which show how this form of the Recursion Theorem can be naturally applied to obtain interesting infinite sequences of programs.

Suppose we have a formalized programming language and suppose the programs of this language are certain words on a finite alphabet Σ. Given a program $\mathcal{P}$ let $\mathcal{P}^{\#}$ be the 1-ary numerical function computed by $\mathcal{P}$. For a word $\alpha \in \Sigma^*$, let $|\alpha|$ denote the length of α. Suppose we are looking for programs $\mathcal{P}_0, \mathcal{P}_1$ such that

$$\begin{aligned} \mathcal{P}_0^{\#}(x) &= |\mathcal{P}_1|\, x^2 + |\mathcal{P}_0| \\ \mathcal{P}_1^{\#}(x) &= |\mathcal{P}_0|\, x^2 + |\mathcal{P}_1| \end{aligned}$$

for any natural number x. Although it is not clear how to define the programs $\mathcal{P}_0, \mathcal{P}_1$, the Double Recursion Theorem of Smullyan [5] guarantees that they can be effectively constructed.

Now suppose that we are looking for programs $\mathcal{P}_0, \mathcal{P}_1, \mathcal{P}_2, ...$ such that

$$\begin{aligned} \mathcal{P}_0^{\#}(x) &= |\mathcal{P}_0| + |\mathcal{P}_x| \\ \mathcal{P}_1^{\#}(x) &= |\mathcal{P}_0| + |\mathcal{P}_1|\, x + |\mathcal{P}_{2x}| \\ \mathcal{P}_2^{\#}(x) &= |\mathcal{P}_0| + |\mathcal{P}_1|\, x + |\mathcal{P}_2|\, x^2 + |\mathcal{P}_{3x}| \\ &\vdots \end{aligned}$$

for any natural number x. In this note we prove an infinitary version of the Double Recursion Theorem from which we obtain that such infinite sequence of programs does exist.

For a proof of the Recursion Theorem and many of its variants see [2], [3] and [6].

We conclude the paper with some applications which show how this form of the Recursion Theorem can be naturally applied to different situations. In particular we obtain the Double Recursion Theorem as a direct corollary.

1 Notation

We use $\mathbf{N}$ to denote the set of natural numbers and ω to denote the set $\mathbf{N} \cup \{0\}$.

Given sets $A_1, ..., A_n$ we use $A_1 \times ... \times A_n$ to denote the *Cartesian product* of $A_1, ..., A_n$, that is to say the set of all n-tuples $(a_1, ..., a_n)$ such that $a_1 \in A_1, ..., a_n \in A_n$. If $A_1 = ... = A_n = A$, then we write A^n in place of $A_1 \times ... \times A_n$. We use $\Diamond$ to denote the unique 0-tuple. Thus $A^0 = \{\Diamond\}$.

An *alphabet* is a finite set of symbols. If Σ is an alphabet, then we use Σ^* to denote the set of all words on Σ. We use $|\alpha|$ to denote the length of a word α. The unique word of length 0 is denoted by ε. If $\alpha_1, ..., \alpha_n \in \Sigma^*$, we use $\alpha_1 ... \alpha_n$ to denote the *concatenation* of the words $\alpha_1, ..., \alpha_n$.

We use Num to denote the alphabet formed by the *numerals*, i.e. the symbols

0 1 2 3 4 5 6 7 8 9

We assume that the natural numbers (that is the elements of ω) are purely abstract entities and hence we have $\omega \cap Num = \emptyset$.

Given an alphabet Σ, we use $\omega^n \times \Sigma^{*m}$ to abbreviate the expression

$$\overbrace{\omega \times ... \times \omega}^{n \text{ times}} \times \overbrace{\Sigma^* \times ... \times \Sigma^*}^{m \text{ times}}$$

For example, when $n = m = 0$, we have that $\omega^n \times \Sigma^{*m}$ denotes the set $\{\Diamond\}$ and if $m = 0$, then $\omega^n \times \Sigma^{*m}$ denotes the set ω^n. Also, we write $(\vec{x}, \vec{\alpha})$ in place of $(x_1, ..., x_n, \alpha_1, ..., \alpha_m)$.

Given a function f, we use $\operatorname{Dom} f$ and $\operatorname{Im} f$ to denote the domain and the image of f, respectively. A function f is called Σ-*mixed* if there exist $n, m \geq 0$, such that $\operatorname{Dom} f \subseteq \omega^n \times \Sigma^{*m}$ and either $\operatorname{Im} f \subseteq \omega$ or $\operatorname{Im} f \subseteq \Sigma^*$. We write $f : S \subseteq \omega^n \times \Sigma^{*m} \to O$ to express that f is a function such that $\operatorname{Dom} f = S \subseteq \omega^n \times \Sigma^{*m}$ and $\operatorname{Im} f \subseteq O$.

2 A theoretical programming language

In order to express the result in a more suggestive form we shall use a theoretical programming language, which is a mixed version of the basic programming language used in [1] to develop computability theory. For the remainder of the paper we assume Σ is an alphabet containing the following alphabet

$$Num \cup \{\leftarrow, +, \dot{-}, \neq, ^\frown, \mathrm{N}, \mathrm{W}, \mathrm{B}, \mathrm{L}, \mathrm{I}, \mathrm{F}, \mathrm{E}, \mathrm{S}, \mathrm{G}, \mathrm{O}, \mathrm{T}\}$$

Define $S : Num^* \to Num^*$ as follows

$S(\varepsilon) = 1$
$S(\alpha 0) = \alpha 1$
$S(\alpha 1) = \alpha 2$
$S(\alpha 2) = \alpha 3$
$S(\alpha 3) = \alpha 4$
$S(\alpha 4) = \alpha 5$
$S(\alpha 5) = \alpha 6$

$S(\alpha 6) = \alpha 7$
$S(\alpha 7) = \alpha 8$
$S(\alpha 8) = \alpha 9$
$S(\alpha 9) = S(\alpha)0$

Now define $- : \mathbf{N} \to Num^*$ as follows

$\overline{1} = 1$
$\overline{n+1} = S(\bar{n})$

We note that the word $\bar{n}$ is the usual decimal notation of n. The words of the form

$\mathrm{N}\bar{k}$

with $k \in \mathbf{N}$, are called *numerical variables*. The words of the form

$\mathrm{W}\bar{k}$

with $k \in \mathbf{N}$, are called *alphabetical variables*. The words of the form

$\mathrm{L}\bar{k}$

with $k \in \mathbf{N}$, are called *labels*. An *instruction* is an element of Σ^* which is of one of the following forms

$\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} + 1$
$\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} \dot{-} 1$
IF $\mathrm{N}\bar{k} \neq 0$ GOTO $\mathrm{L}\bar{m}$
$\mathrm{W}\bar{k} \leftarrow \mathrm{W}\bar{k}a$
$\mathrm{W}\bar{k} \leftarrow {}^\frown\mathrm{W}\bar{k}$
IF $\mathrm{W}\bar{k}$ BEGINS a GOTO $\mathrm{L}\bar{m}$
$\mathrm{L}\bar{n}$ $\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} + 1$
$\mathrm{L}\bar{n}$ $\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} \dot{-} 1$
$\mathrm{L}\bar{n}$ IF $\mathrm{N}\bar{k} \neq 0$ GOTO $\mathrm{L}\bar{m}$
$\mathrm{L}\bar{n}$ $\mathrm{W}\bar{k} \leftarrow \mathrm{W}\bar{k}a$
$\mathrm{L}\bar{n}$ $\mathrm{W}\bar{k} \leftarrow {}^\frown\mathrm{W}\bar{k}$
$\mathrm{L}\bar{n}$ IF $\mathrm{W}\bar{k}$ BEGINS a GOTO $\mathrm{L}\bar{m}$

where $a \in \Sigma$ and $k, n, m \in \mathbf{N}$[1]. We use Ins^Σ to denote the set of all instructions. When the instruction I is of the form $\mathrm{L}\bar{n}J$ for some $J \in Ins^\Sigma$, we call $\mathrm{L}\bar{n}$ the *label* of I and we say that I is *labeled* $\mathrm{L}\bar{n}$.

We give below the intuitive interpretation associated with some instructions:

INSTRUCTION: $\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} \dot{-} 1$

[1]As can be noted, in order to make the instructions more readable, we use spaces between some of the symbols. For example, we shall write

L124 IF N55 ≠ 0 GOTO L10

in place of

L124IFN55≠0GOTOL10

INTERPRETATION: If the value of N$\bar{k}$ is 0 leave it unchanged; otherwise decrease by 1 the value of N$\bar{k}$

INSTRUCTION: W$\bar{k} \leftarrow {}^\frown$W$\bar{k}$

INTERPRETATION: If the value of W$\bar{k}$ is ε leave it unchanged; otherwise remove the first symbol of the value of W$\bar{k}$

INSTRUCTION: W$\bar{k} \leftarrow$ W$\bar{k}a$

INTERPRETATION: Place the symbol a to the right of the word which is the value of W$\bar{k}$

INSTRUCTION: IF W$\bar{k}$ BEGINS a GOTO L$\bar{m}$

INTERPRETATION: If the value of W$\bar{k}$ begins with the symbol a, execute next the first instruction labeled L$\bar{m}$

A *program* is a word of the form

$I_1 I_2 ... I_n$

where $n \geq 1$, $I_1, ..., I_n \in Ins^\Sigma$ and for every $i = 1, ..., n$, we have that

(*) if GOTO L$\bar{m}$ is a suffix of I_i, then there exists j such I_j is labeled L$\bar{m}$.

We use Pro^Σ to denote the set of all programs. As usual, in order to give a program, we shall write it line by line. For example, we shall write

L2 N12 $\leftarrow$ N12$\dot{-}$1
W1 $\leftarrow$ ${}^\frown$W1
IF N12 $\neq$ 0 GOTO L2

in place of

L2N12$\leftarrow$N12$\dot{-}$1W1$\leftarrow{}^\frown$W1IFN12$\neq$0GOTOL2

The following lemma says that programs can be uniquely parsed.

Lemma 1 *If $I_1 ... I_n = J_1 ... J_m$, with $I_1, ..., I_n, J_1, ..., J_m \in Ins^\Sigma$, then $n = m$ and $I_j = J_j$ for each $j \geq 1$.*

Proof: Suppose I_n is a proper suffix of J_m. It is easy to check that $J_m = \text{L}\bar{u}I_n$ for some $u \in \mathbf{N}$. Thus

$I_1 ... I_n = J_1 ... J_{m-1} \text{L}\bar{u} I_n$

which says that $n > 1$ and

(1) $I_1 ... I_{n-1} = J_1 ... J_{m-1} \text{L}\bar{u}$.

Hence L$\bar{u}$ is a suffix of I_{n-1} and therefore GOTO L$\bar{u}$ is a suffix of I_{n-1}. By (1), GOTO is a suffix of $J_1 ... J_{m-1}$, which is impossible. We have arrived to a contradiction and hence we have that I_n is not a proper suffix of J_m. By symmetry we have that $I_n = J_m$ and so $n = m$ and $I_j = J_j$ for each $j \geq 1$. ■

Let

$$\omega^{[\mathbf{N}]} = \left\{(s_1, s_2, ...) \in \omega^{\mathbf{N}} : \text{ there is } n \in \mathbf{N} \text{ such that } s_i = 0, \text{ for } i \geq n\right\}$$

$$\Sigma^{*[\mathbf{N}]} = \left\{(\sigma_1, \sigma_2, ...) \in \Sigma^{*\mathbf{N}} : \text{ there is } n \in \mathbf{N} \text{ such that } \sigma_i = \varepsilon, \text{ for } i \geq n\right\}.$$

We shall always assume that at any moment of a computation using a program, almost all (i.e. all but a finite number) of the numerical variables have the value 0 and almost all of the alphabetical variables have the value ε. Thus, it is natural to define a *state* to be a pair

$$(\vec{s}, \vec{\sigma}) = ((s_1, s_2, ...), (\sigma_1, \sigma_2, ...)) \in \omega^{[\mathbf{N}]} \times \Sigma^{*[\mathbf{N}]}.$$

If $i \geq 1$, then s_i is the *value* of the variable $\mathrm{N}\overline{i}$ at the state $(\vec{s}, \vec{\sigma})$ and σ_i is the *value* of the variable $\mathrm{W}\overline{i}$ at the state $(\vec{s}, \vec{\sigma})$. The reader will have no problem to give a formal definition of the following predicates:

- $\mathcal{P}$ *halts, starting from the state* $(\vec{s}, \vec{\sigma})$
- $(\vec{e}, \vec{\gamma})$ *is the state obtained when* $\mathcal{P}$ *halts, starting from the state* $(\vec{s}, \vec{\sigma})$.

Let $\mathcal{P} \in Pro^{\Sigma}$ and let $n, m \in \omega$. We define functions $\mathcal{P}^{n,m,\#}$ and $\mathcal{P}^{n,m,*}$ as follows

(1) $\mathrm{Dom}\, \mathcal{P}^{n,m,\#} = \mathrm{Dom}\, \mathcal{P}^{n,m,*}$ is the set

$$\{(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} : \mathcal{P} \text{ halts, starting from the state } ((x_1, ..., x_n, 0, ...), (\alpha_1, ..., \alpha_m, \varepsilon, ...))\}$$

(2) $\mathcal{P}^{n,m,\#}(\vec{x}, \vec{\alpha})$ = value of N1 at the state obtained when $\mathcal{P}$ halts, starting from $((x_1, ..., x_n, 0, ...), (\alpha_1, ..., \alpha_m, \varepsilon, ...))$

$\mathcal{P}^{n,m,*}(\vec{x}, \vec{\alpha})$ = value of W1 at the state obtained when $\mathcal{P}$ halts, starting from $((x_1, ..., x_n, 0, ...), (\alpha_1, ..., \alpha_m, \varepsilon, ...))$

A Σ-mixed function f is called Σ-*computable* if there exist $\mathcal{P} \in Pro^{\Sigma}$ and $n, m \in \omega$ such that either $f = \mathcal{P}^{n,m,\#}$ or $f = \mathcal{P}^{n,m,*}$.

As usual, once we have proved that a function $f : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ is Σ-computable, we can use macros like

$$[\mathrm{V} \leftarrow f(\mathrm{V}_1, ..., \mathrm{V}_n, \mathrm{U}_1, ..., \mathrm{U}_m)]$$

where $\mathrm{V}, \mathrm{V}_1, ..., \mathrm{V}_n$ can be any numerical variables and $\mathrm{U}_1, ..., \mathrm{U}_m$ any alphabetical variables. (In particular V might be one of $\mathrm{V}_1, ..., \mathrm{V}_n$.). Of course, if the above macro is encountered in a program where $\mathrm{V}_1, ..., \mathrm{V}_n, \mathrm{U}_1, ..., \mathrm{U}_m$ have values for which f is not defined, the main program will never terminate. Similarly if $f : S \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ is Σ-computable, we shall use macros like

$$[\mathrm{U} \leftarrow f(\mathrm{V}_1, ..., \mathrm{V}_n, \mathrm{U}_1, ..., \mathrm{U}_m)]$$

where $V_1, ..., V_n$ can be any numerical variables and $U, U_1, ..., U_m$ any alphabetical variables.

A set $S \subseteq \omega^n \times \Sigma^{*m}$ is called Σ-*recursively enumerable* if $S = \emptyset$ or there exist Σ-computable functions

$f_i : \omega \to \omega$, $i = 1, ..., n$
$g_i : \omega \to \Sigma^*$, $i = 1, ..., m$

such that

$$S = \{(f_1(x), ..., f_n(x), g_1(x), ..., g_m(x)) : x \in \omega\}$$

Using macros it can be proved the following

Lemma 2 *If $f : D \subseteq \omega^n \times \Sigma^{*m} \to \omega$ is Σ-computable and $S \subseteq \omega^n \times \Sigma^{*m}$ is Σ-recursively enumerable, then $f \mid_{D \cap S}$ (i.e. the restriction of f to the set $D \cap S$) is Σ-computable.*

3 Infinitary simultaneous recursion theorem

Let E_1 and E_2 be expressions whose values are in the set ω (resp. Σ^*) and depend on values assigned to numerical variables $x_1, ..., x_n$ and alphabetical variables $\alpha_1, ..., \alpha_m$. Then we will write

$E_1 \approx E_2$

to express the fact that for any given values of $x_1, ..., x_n$ in ω and $\alpha_1, ..., \alpha_m$ in Σ^*, either E_1 and E_2 are undefined or both are defined and the corresponding values are equal.

Theorem 3 *(a) Let $f : S \subseteq \omega \times \omega^n \times \Sigma^{*m} \times Pro^{\Sigma} \to \omega$ be Σ-computable. There exists $\mathcal{F} \in Pro^{\Sigma}$ such that*

(1) $\mathcal{F}^{1,0,*}(i)$ is defined for every $i \in \omega$ and

$$\mathcal{F}^{1,0,*}(0), \mathcal{F}^{1,0,*}(1), \mathcal{F}^{1,0,*}(2), ... \in Pro^{\Sigma}$$

(2) For each $i \in \omega$:

$$\left(\mathcal{F}^{1,0,*}(i)\right)^{n,m,\#}(\vec{x}, \vec{\alpha}) \approx f(i, \vec{x}, \vec{\alpha}, \mathcal{F})$$

(b) Let $f : S \subseteq \omega \times \omega^n \times \Sigma^{*m} \times Pro^{\Sigma} \to \Sigma^*$ be Σ-computable. There exists $\mathcal{F} \in Pro^{\Sigma}$ such that

(1) $\mathcal{F}^{1,0,*}(i)$ is defined for every $i \in \omega$ and

$$\mathcal{F}^{1,0,*}(0), \mathcal{F}^{1,0,*}(1), \mathcal{F}^{1,0,*}(2), ... \in Pro^{\Sigma}$$

(2) For each $i \in \omega$:

$$\left(\mathcal{F}^{1,0,*}(i)\right)^{n,m,*}(\vec{x}, \vec{\alpha}) \approx f(i, \vec{x}, \vec{\alpha}, \mathcal{F})$$

Proof: We only will prove (a). As usual we need a particular case of the parameter theorem. Let $S_{n,m} : \Sigma^* \times Pro^\Sigma \to Pro^\Sigma$ be defined by

$$S_{n,m}(\alpha, \mathcal{P}) = \mathcal{Q}\mathcal{P}$$

where $\mathcal{Q}$ is the program

$$\mathrm{W}\overline{m+1} \leftarrow \mathrm{W}\overline{m+1}a_1\mathrm{W}\overline{m+1} \leftarrow \mathrm{W}\overline{m+1}a_2 ... \mathrm{W}\overline{m+1} \leftarrow \mathrm{W}\overline{m+1}a_k$$

with $a_1, ..., a_k \in \Sigma$ such that $\alpha = a_1 ... a_k$. We note that $S_{n,m}$ is Σ-computable and that for every $\mathcal{P} \in Pro^\Sigma$ we have

$$\begin{array}{lrcl} \text{(i)} & \mathcal{P}^{n,m+1,\#}(\vec{x}, \vec{\alpha}, \alpha) & \approx & S_{n,m}(\alpha, \mathcal{P})^{n,m,\#}(\vec{x}, \vec{\alpha}) \\ & \mathcal{P}^{n,m+1,*}(\vec{x}, \vec{\alpha}, \alpha) & \approx & S_{n,m}(\alpha, \mathcal{P})^{n,m,*}(\vec{x}, \vec{\alpha}) \end{array}$$

Let $g : Pro^\Sigma \to Pro^\Sigma$ be defined by

$$g(\mathcal{P}) = \mathcal{P}\,[\mathrm{W2} \leftarrow \mathcal{P}]\,[\mathrm{W1} \leftarrow S_{n,m}(\mathrm{W2}, \mathrm{W1})]$$

where $[\mathrm{W2} \leftarrow \mathcal{P}]$ and $[\mathrm{W1} \leftarrow S_{n,m}(\mathrm{W2}, \mathrm{W1})]$ are adequate macro expansions. Although we have not given a complete description of g, we observe that g can be defined in such a way that it is algorithmic and hence, by Church´s Thesis, we can assume that g is Σ-computable.

Note that[2]

$$\text{(ii)}\ g(\mathcal{P})^{1,0,*}(i) \approx S_{n,m}(\mathcal{P}, \mathcal{P}^{1,0,*}(i))$$

Let F be the following function

$$\begin{array}{rcl} \{(i, \vec{x}, \vec{\alpha}, \mathcal{P}) : (i, \vec{x}, \vec{\alpha}, g(\mathcal{P})) \in S\} & \to & \omega \\ (i, \vec{x}, \vec{\alpha}, \mathcal{P}) & \to & f(i, \vec{x}, \vec{\alpha}, g(\mathcal{P})) \end{array}$$

Since f and g are Σ-computable, so is F. Let $\mathcal{Q} \in Pro^\Sigma$ be a program which computes F. Thus we have

$$\text{(iii)}\ f(i, \vec{x}, \vec{\alpha}, g(\mathcal{P})) \approx \mathcal{Q}^{n+1,m+1,\#}(i, \vec{x}, \vec{\alpha}, \mathcal{P})$$

Let k be the least number satisfying

- the variables that occur in $\mathcal{Q}$ are all included in the list $\mathrm{N1}, ..., \mathrm{N}\bar{k}$, $\mathrm{W1}, ..., \mathrm{W}\bar{k}$
- $k \geq n+1, m+1$

Thus we have

$$\mathcal{Q} = \mathcal{Q}(\mathrm{N1}, ..., \mathrm{N}\bar{k}, \mathrm{W1}, ..., \mathrm{W}\bar{k})$$

Let

$$\tilde{\mathcal{Q}} = \mathcal{Q}(\mathrm{N}\bar{k}, \mathrm{N1}, ..., \mathrm{N}\overline{k-1}, \mathrm{W1}, ..., \mathrm{W}\bar{k})$$

Define $G : \omega \to Pro^\Sigma$ by

$$G(i) = \mathcal{T}\tilde{\mathcal{Q}}$$

where

[2] $g(\mathcal{P})^{1,0,*}(i)$ is defined iff $\mathcal{P}^{1,0,*}(i)$ is defined and belongs to Pro^Σ, since the macro $[\mathrm{W1} \leftarrow S_{n,m}(\mathrm{W2}, \mathrm{W1})]$ terminates iff the initial value of W1 is a program.

$$\mathcal{T} = \overbrace{\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k}+1 \mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k}+1 \ldots \mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k}+1}^{i \text{ times}}$$

Note that

(iv) $G(i)^{n,m+1,\#}(\vec{x},\vec{\alpha},\mathcal{P}) \approx \mathcal{Q}^{n+1,m+1,\#}(i,\vec{x},\vec{\alpha},\mathcal{P})$

Let $\mathcal{R}$ be the following program

$[\mathrm{W1} \leftarrow G(\mathrm{N1})]$

Obviously, we have

(v) $\mathcal{R}^{1,0,*}(i) \approx G(i)$

Let

$$\mathcal{F} = g(\mathcal{R}) = \mathcal{R}\,[\mathrm{W2} \leftarrow \mathcal{R}]\,[\mathrm{W1} \leftarrow S_{n,m}(\mathrm{W2},\mathrm{W1})]\,.$$

Since $\mathcal{R}^{1,0,*}(i)$ is defined and belongs to Pro^{Σ}, for every $i \in \omega$, we have that $\mathcal{F}$ satisfies (1). We conclude the proof by showing that $\mathcal{F}$ also satisfies (2).

$$\begin{aligned}
\left(\mathcal{F}^{1,0,*}(i)\right)^{n,m,\#}(\vec{x},\vec{\alpha}) &\approx \left(g(\mathcal{R})^{1,0,*}(i)\right)^{n,m,\#}(\vec{x},\vec{\alpha}) \\
&\approx \left(S_{n,m}(\mathcal{R},\mathcal{R}^{1,0,*}(i))\right)^{n,m,\#}(\vec{x},\vec{\alpha}) && \text{by (ii)} \\
&\approx \left(\mathcal{R}^{1,0,*}(i)\right)^{n,m+1,\#}(\vec{x},\vec{\alpha},\mathcal{R}) && \text{by (i)} \\
&\approx G(i)^{n,m+1,\#}(\vec{x},\vec{\alpha},\mathcal{R}) && \text{by (v)} \\
&\approx \mathcal{Q}^{n+1,m+1,\#}(i,\vec{x},\vec{\alpha},\mathcal{R}) && \text{by (iv)} \\
&\approx f(i,\vec{x},\vec{\alpha},g(\mathcal{R})) && \text{by (iii)} \\
&\approx f(i,\vec{x},\vec{\alpha},\mathcal{F}). \quad \blacksquare
\end{aligned}$$

We conclude the paper with some applications.

Application 1 (Double Recursion Theorem [5]). *Let*

$f_0 : S_0 \subseteq \omega^n \times \Sigma^{*m} \times Pro^{\Sigma} \times Pro^{\Sigma} \to \omega$
$f_1 : S_1 \subseteq \omega^n \times \Sigma^{*m} \times Pro^{\Sigma} \times Pro^{\Sigma} \to \omega$

be Σ*-computable. There exist* $\mathcal{P}_0, \mathcal{P}_1 \in Pro^{\Sigma}$ *such that*

$\mathcal{P}_0^{n,m,\#}(\vec{x},\vec{\alpha}) \approx f_0(\vec{x},\vec{\alpha},\mathcal{P}_0,\mathcal{P}_1)$
$\mathcal{P}_1^{n,m,\#}(\vec{x},\vec{\alpha}) \approx f_1(\vec{x},\vec{\alpha},\mathcal{P}_0,\mathcal{P}_1)$

Proof: Define

$$f(i,\vec{x},\vec{\alpha},\mathcal{P}) = f_i\left(\vec{x},\vec{\alpha},\mathcal{P}^{1,0,*}(0),\mathcal{P}^{1,0,*}(1)\right).$$

By Church´s Thesis f is Σ-computable. Thus we can take

$\mathcal{P}_0 = \mathcal{F}^{1,0,*}(0)$

$$\mathcal{P}_1 = \mathcal{F}^{1,0,*}(1)$$

where $\mathcal{F}$ is the program given by the above theorem. ■

We remember that $|\alpha|$ denotes the length of the word $\alpha \in \Sigma^*$.

Application 2. *There are programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that*

$$\begin{array}{lcl} \mathcal{P}_0^{1,0,\#}(x) & \approx & |\mathcal{P}_1|\, x + |\mathcal{P}_0| \\ \mathcal{P}_1^{1,0,\#}(x) & \approx & |\mathcal{P}_2|\, x^2 + |\mathcal{P}_1|\, x + |\mathcal{P}_0| \\ \mathcal{P}_2^{1,0,\#}(x) & \approx & |\mathcal{P}_3|\, x^3 + |\mathcal{P}_2|\, x^2 + |\mathcal{P}_1|\, x + |\mathcal{P}_0| \\ & \vdots & \end{array}$$

Proof: Take $f(i,x,\mathcal{P}) = \sum_{j=0}^{i+1} \left|\mathcal{P}^{1,0,*}(j)\right| x^j$. By Church´s Thesis f is Σ-computable and hence we can apply Theorem 3. ■

If we restrict the domain of f, we can add requirements on the domains of the functions $\mathcal{P}_i^{1,0,\#}$.

Application 3. *There are programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that*

$$\begin{array}{lcl} \mathrm{Dom}\, \mathcal{P}_0^{1,0,\#} & = & \omega \\ \mathrm{Dom}\, \mathcal{P}_1^{1,0,\#} & = & \mathrm{Im}\, \mathcal{P}_0^{1,0,\#} \\ \mathrm{Dom}\, \mathcal{P}_2^{1,0,\#} & = & \mathrm{Im}\, \mathcal{P}_1^{1,0,\#} \\ & \vdots & \end{array}$$

and

$$\begin{array}{lcl} \mathcal{P}_0^{1,0,\#}(x) & = & |\mathcal{P}_1|\, x + |\mathcal{P}_0| \\ \mathcal{P}_1^{1,0,\#}(x) & = & |\mathcal{P}_2|\, x^2 + |\mathcal{P}_1|\, x + |\mathcal{P}_0| \\ \mathcal{P}_2^{1,0,\#}(x) & = & |\mathcal{P}_3|\, x^3 + |\mathcal{P}_2|\, x^2 + |\mathcal{P}_1|\, x + |\mathcal{P}_0| \\ & \vdots & \end{array}$$

Proof: Let $f(i,x,\mathcal{P}) = \sum_{j=0}^{i+1} \left|\mathcal{P}^{1,0,*}(j)\right| x^j$. Let S be the following set

$$\{(i,x,\mathcal{P}) \in \mathrm{Dom}\, f : i = 0 \text{ or } \mathcal{P}^{1,0,*}(i-1) \in Pro^{\Sigma} \text{ and } x \in \mathrm{Im}\left(\mathcal{P}^{1,0,*}(i-1)\right)^{1,0,\#}\}$$

By Church´s Thesis S is Σ-recursively enumerable and hence $f\,|_S$ is Σ-computable. Thus, we can apply Theorem 3. ■

Application 4. *Let $g : Pro^{\Sigma} \to Pro^{\Sigma}$ be a Σ-computable function and let $\mathcal{P} \xrightarrow[sem]{g} \mathcal{Q}$ denote the fact that $g\,(\mathcal{P})^{n,m,\#} = \mathcal{Q}^{n,m,\#}$. Then*

(a) There are programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that

$$\cdots \xrightarrow[sem]{g} \mathcal{P}_2 \xrightarrow[sem]{g} \mathcal{P}_1 \xrightarrow[sem]{g} \mathcal{P}_0$$

(b) There are programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that

$$\cdots \xrightarrow[sem]{g\circ g\circ g} \mathcal{P}_2 \xrightarrow[sem]{g\circ g} \mathcal{P}_1 \xrightarrow[sem]{g} \mathcal{P}_0$$

Proof: (b) Let

$$f(i,\vec{x},\vec{\alpha},\mathcal{P}) = \left(\overbrace{g\circ ...\circ g}^{i+1}\left(\mathcal{P}^{1,0,*}(i+1)\right)\right)^{n,m,\#}(\vec{x},\vec{\alpha})$$

and take

$$\mathcal{P}_i = \mathcal{F}^{1,0,*}(i),\ i = 0,1,...$$

where $\mathcal{F}$ is the program given by Theorem 3. ■

In the following application we naturally obtain [4, Thm 11.VII].

Application 5. *There are pairwise different programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that for each $i \in \omega$, the function $\mathcal{P}_i^{0,1,\#}$ is the following*

$$\begin{array}{rcl} \{\mathcal{P}_{i+1}\} & \to & \omega \\ \mathcal{P}_{i+1} & \to & i \end{array}$$

Proof: Let

$$\begin{array}{rcl} f : \{(i,\alpha,\mathcal{P}) : \alpha = \mathcal{P}^{1,0,*}(i+1)\} & \to & \omega \\ (i,\alpha,\mathcal{P}) & \to & i \end{array}$$

and take

$$\mathcal{P}_i = \mathcal{F}^{1,0,*}(i),\ i = 0,1,...$$

where $\mathcal{F}$ is the program given by Theorem 3. We note that the programs $\mathcal{F}^{1,0,*}(i)$ are pairwise different since they compute pairwise different functions. ■

Application 6. *(a) There are pairwise different programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that for each $i \in \omega$, the function $\mathcal{P}_i^{0,1,*}$ is the following*

$$\begin{array}{rcl} \{\mathcal{P}_i, \mathcal{P}_{i+1}, \mathcal{P}_{i+2}, ...\} & \to & \{\mathcal{P}_{i^2}, \mathcal{P}_{i^2+1}, \mathcal{P}_{i^2+2}, ...\} \\ \mathcal{P}_{i+l} & \to & \mathcal{P}_{i^2+l} \end{array}$$

(b) Let $g : D \subseteq \omega \times \omega \to \omega$ be Σ-computable. There are pairwise different programs $\mathcal{P}_0, \mathcal{P}_1, ...$ such that for each $i \in \omega$, the function $\mathcal{P}_i^{0,1,*}$ is the following

$$\begin{array}{rcl} \{\mathcal{P}_l : (i,l) \in D\} & \to & \{\mathcal{P}_0, \mathcal{P}_1, ...\} \\ \mathcal{P}_l & \to & \mathcal{P}_{g(i,l)} \end{array}$$

Proof: We prove (b), since (a) is a particular case of (b). First we note that the construction of $\mathcal{F}$ in the proof of Theorem 3 depends only from the program $\mathcal{Q}$ and the integers n, m. Thus we have defined a function

$$\begin{array}{rcl} \omega^2 \times Pro^{\Sigma} & \rightarrow & Pro^{\Sigma} \\ (n, m, \mathcal{Q}) & \rightarrow & \mathcal{F}(n, m, \mathcal{Q}) \end{array}$$

We note that this function is algorithmic and hence, by Church´s Thesis it is Σ-computable. Also we note that

$$\mathcal{F}(n, m, \mathcal{Q})^{1,0,*}(i) = \mathcal{J}\overbrace{II...I}^{i}\mathcal{K}, \text{ for every } i \in \omega$$

where $\mathcal{J}, \mathcal{K} \in Pro^{\Sigma}$ and $I \in Ins^{\Sigma}$. Thus the programs

$$\mathcal{F}(n, m, \mathcal{Q})^{1,0,*}(0), \mathcal{F}(n, m, \mathcal{Q})^{1,0,*}(1), ...$$

are pairwise different. Let

$$Pro_0^{\Sigma} = \left\{\mathcal{F}(n, m, \mathcal{Q}) : (n, m, \mathcal{Q}) \in \omega^2 \times Pro^{\Sigma}\right\}$$

Note that Pro_0^{Σ} is Σ-recursively enumerable and therefore so is the set

$$S = \left\{(i, \alpha, \mathcal{P}) \in \omega \times \Sigma^* \times Pro_0^{\Sigma} : (i, \min_l \alpha = \mathcal{P}^{1,0,*}(l)) \in D\right\}$$

Hence the function

$$\begin{array}{rcl} f : S & \rightarrow & \Sigma^* \\ (i, \alpha, \mathcal{P}) & \rightarrow & \mathcal{P}^{1,0,*}(g(i, \min_l \alpha = \mathcal{P}^{1,0,*}(l))) \end{array}$$

is Σ-computable and we can take

$$\mathcal{P}_i = \mathcal{F}^{1,0,*}(i),\ i = 0, 1, ...$$

where $\mathcal{F}$ is the program given by Theorem 3. ■

References

[1] M. Davis and E. Weyuker, Computability, Complexity and Languages, Academic Press, 1983.

[2] S. B. Cooper, Computability Theory, Chapman & Hall/CRC Mathematical Series, Volume 26, 2004.

[3] P. Odifreddi, Classical Recursion Theory, North-Holland, 1989.

[4] H. Rogers, Jr., Theory of recursive functions and effective computability, McGraw-Hill, New York, 1967.

[5] R. M. Smullyan, Theory of formal systems, Ann. Math. Studies 47. Princeton University Press, 1961.

[6] ————, Recursion Theory for Metamathematics, Oxford Logic Guides 22. The Clarendon Press, Oxford University Press, New York, 1993.