Analele ştiinţifice ale Universităţii „Al. I. Cuza" din Iaşi
Tomul XXVII, s. I a, 1981 f. 2

A DEPTH-SPACE ANALYSIS OF ALGORITHMS

BY

C. CAZACU

Let us consider a *memory* with $\aleph_0$ registers which are identified by natural numbers. Every such register may contain any natural number. A *state* of memory is an infinite string

$$s = (s_0, s_1, s_2 \ldots\ldots)$$

of natural numbers in the registers 0, 1, 2 .... respectively. By $c_s(x)$ we denote the content of the x-register in a state s. Further, we shall consider only the set $\mathbf{S}$ of states s so that all but finitely-many $c_s(x)$ are equal to 0. Such states will sometimes be codified by the natural numbers

$$\gamma s = p_0^{s_0} p_1^{s_1} p_2^{s_2} \ldots\ldots$$

where p_i is the $i+1$-th prime number.

An *algorithm with registers* (R-algorithm) is a flowchart whose nodes represent assignment and decision statements of the form $x := x+1$, $x := = x \dot{-} 1$ and $x > 0$ for some x-registers. We denote these statements simply by x, $\bar{x}$ and x ?. The *space* $\mathrm{sp}(A)$ of the R-algorithm A is the finite set of all registers which appear in the flowchart A.

Two operations with algorithms, the sequence and the iteration will be used. The sequence $A_1 A_2 \ldots A_n$ or $A_1 ; A_2 ; \ldots ; A_n$ has the same meaning as **begin** $A_1 ; A_2 ; \ldots ; A_n$ **end**. A^n is the sequence $A_1 A_2 \ldots A_n$ for $A_i = A$. The iterations $(\bar{x}A)^*$ and $(\bar{x})^*$ are the algorithms **while** $x > 0$ **do** $x := x \dot{-} 1$; A **end** and **while** $x > 0$ **do** $x := x \dot{-} 1$ **end**.

The *normal R-algorithms* (*NR-algorithms*) are built up from statements of the form x, $\bar{x}$, using only the above operations. The *depth* $\mathrm{dp}(A)$ of a NR-algorithms A is an inductively defined number as follows: $\mathrm{dp}(x) = = \mathrm{dp}(\bar{x}) = 0$, $\mathrm{dp}(AB) = \max(\mathrm{dp}(A), \mathrm{dp}(B))$, $\mathrm{dp}((\bar{x}A)^*) = 1 + \mathrm{dp}(A)$, $\mathrm{dp}((\bar{x})^*) = = 1$ for every x-register and NR-algorithms A, B. Clearly, the NR-algorithms may be viewed as R-algorithms defined by flowcharts having a particular structure.

Any R-algorithms A defines a partial function $\bar{A} : \mathbf{S} \to \mathbf{S}$. Two R-algorithms A and B are *equivalent* ($A \sim B$) if $\bar{A} = \bar{B}$. Let s, t be states and $\mathbf{Z} \subseteq \mathbf{N}$ ($\mathbf{N}$ being the set of natural numbers). We write $s = t(\mathbf{Z})$ if $c_s(x) = = c_t(x)$ for all $x \in \mathbf{Z}$. The R-algorithms A and B are **Z**-*equivalent* ($A \sim B(\mathbf{Z})$) if for every two states s and t, the relation $s = t(\mathbf{Z})$ implies a) $\bar{A}(s)\downarrow$ iff $\bar{B}(t)\downarrow$ and b) if $\bar{A}(s)\downarrow$ then $\bar{A}(s) \sim \bar{B}(t)$ $(\mathbf{Z})$, For a state s and an algorithm A, the expression $s.A.t$ means a) if t is a defined state then $(\overline{A})s) \downarrow$, $\overline{A}(s) = t$ and b) if t is an undefined state then $\overline{A}(s) \uparrow$ (A never halts starting from the state s). We consider a state is undefined if at least one register has undefined content. The notation

$$x_1, x_2, \ldots, x_n : (a_1, a_2, \ldots, a_n) . A . (b_1, b_2, \ldots, b_n)$$

has an analogous sense. Here we emphasize some registers $x_1, x_2, \ldots, x_n$ so that $\mathrm{sp}(A) \subseteq \{x_1, x_2, \ldots, x_n\}$. If $x_1, x_2, \ldots, x_n$ are the registers $0, 1, 2, \ldots, n-1$ then

$$(a_1, a_2, \ldots, a_n) . A . (b_1, b_2, \ldots, b_n)$$

stands in the place of the previous expression.

Let f be a n-ary partial arithmetical function ($f : \mathbf{N}^n \to \mathbf{N}$). The R-algorithm A *computes* the function f using the initial distinct registers $x_1, x_2, \ldots, x_n$ and the final register y if, for all the natural numbers $a_1, a_2, \ldots, a_n$ and any state s having $c_s(x_i) = a_i (i = 1, 2, \ldots, n)$, the following holds:

a) if $f(a_1, a_2, \ldots, a_n) \downarrow$ then $\overline{A}(s) \downarrow$, $\overline{A}(s) = t$ and $c_y(t) = f(a_1, a_2, \ldots, a_n)$;

b) if $f(a_1, a_2, \ldots, a_n) \uparrow$ then $\overline{A}(s) \uparrow$.

We shall denote this relation by $[A, x_1, x_2, \ldots, x_n, y, f]$. The set of computable by R-algorithms functions is just the set of all partial recursive functions. Further, $\mathbf{RF}^n$ is the set of all n-ary partial recursive functions and $\mathbf{NRF}^n$ (d,s) is the set of all n-ary recursive functions computable by NR-algorithms A with $\mathrm{dp}(A) \leqslant d$, and $\mathrm{sp}(A) \leqslant s$. $\mathbf{RA}(k)$ is the set of R-algorithms A having $\mathrm{sp}(A) \subseteq \{0, 1, \ldots, k-1\}$.

The following theorems give some answers to the question about the minimal depth and space needed by R-algorithms for the computation of all partial recursive function. These theorems improve some results of Börger [2] and Baredin [1]. In the proofs we use, among other, the general recursive functions $x \dot{-} y$ (arithmetical difference), $\lceil x/y \rceil$ (the least natural number z such that $x \leqslant zy$ if $y > 0$ and 0 if $y = 0$) and $e(x, y)$ (the greatest natural number z such that p_x^z / y if $y > 0$ and 0 if $y = 0$). The NR-algorithms

$$R_k(u, v, w) = (\overline{w}(\overline{u}v)^*(\overline{v}\, u^k)^*)^* \text{ and } T_k(u, v, w) = (\overline{u}(\overline{u}v)^*(\overline{v}^k u)^* w)^* \overline{w}$$

will be also used. Clearly,

$$u, v, w : (a, 0, b) . R_k(u, v, w) . (ak^b, 0, 0)$$

for all a, b, k and

$$u, v, w : (k^a, 0, 0) . T_k(u, v, w) . (a, 0, 0),$$

for every a and $k \geqslant 2$. The statements of the form $y : = f(x_1, x_2, \ldots, x_n)$ and some structured programs are used in common sense.

Theorem 1 (see also [1, 2]). *For every R-algorithm A, there exists a R-algorithm $B \in \mathbf{RA}(2)$ such that*

$$\forall s \quad (\gamma s, 0) . B . (\gamma A(s), 0).$$

Proof. Let's replace in A all fragments of the form (a) by (b) (see fig. 1). Thus, an algorithm C equivalent to A is obtained. Then B results from C by

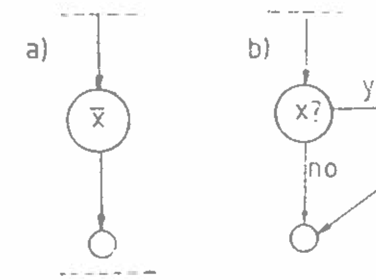

Fig. 1

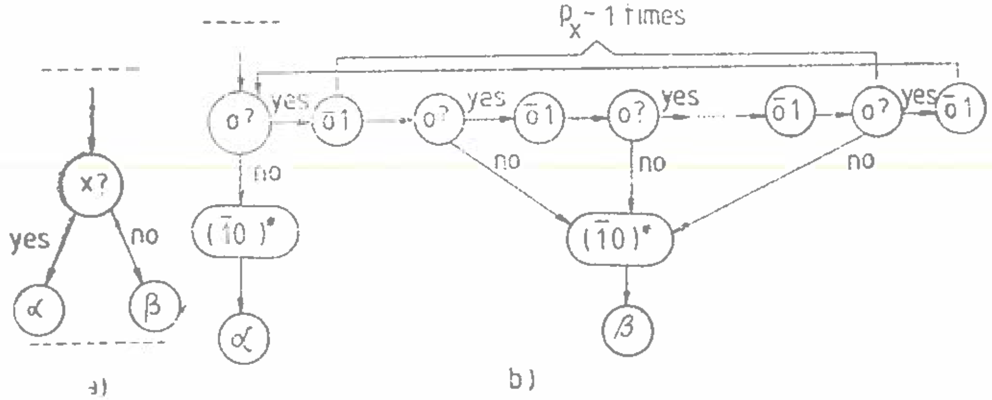

Fig. 2

replacing every node x by $(\overline{0}1)^* (\overline{1}0^{p_x})^*$, every node $\overline{x}$ by $(\overline{0}1)^* (\overline{1}^{p_x}0)^*$ and every fragment of the form (a) by (b) as it is showed in fig. 2.

Theorem 2. *For any R-algorithm A and the distinct registers $u, v, w, y, z \in \mathrm{sp}(A)$, there exists:*

1. *An algorithm $B_1 = (\overline{u})^*(\overline{y})^* y(\overline{y} U_1 U_2 \ldots U_t)^*$ with U_i of the form $a, \overline{a}, (\overline{a}bc)^*, (\overline{a}\overline{b}c)^*, (\overline{a}\overline{a}c)^*$ or $(\overline{a}cc)^*$, so that $B_1 \sim A(\mathrm{sp}(A))$ and $\mathrm{sp}(B_1) = \mathrm{sp}(A) \cup \{u, y\}$.*

2. *An algorithm $B_2 = (\overline{y})^* y(\overline{y} V_1 V_2 \ldots V_s)^*$ with V_i of the form $a : = 0$, $a : = a + 1$, $a : = a + b$ or $a : = a \dot{-} b$, so that $B_2 \sim A(sp(A))$ and $\mathrm{sp}(B_2) = sp(A) \cup \{u, v, y\}$.*

3. *An algorithm $B_3 = (\overline{u})^*(\overline{v})^*(\overline{y})^*(\overline{z})^* yz(\overline{y} W_1 W_2 \ldots W_t)^*$ with W_i of the form* **begin** $a : = a + c$; $b : = b \dot{-} c$ **end**, *so that $B_3 \sim A(\mathrm{sp}(A))$ and $\mathrm{sp}(B_3) = \mathrm{sp}(A) \cup \{u, v, w, y, z\}$.*

Proof. Let A be an R-algorithm. Clearly, it is possible to find an R-algorithm C so that $C \sim A$ and

a) for every statement $x : = x \dot{-} 1$ of C, in the moment of its work, the content of x is greater than 0;

b) if a, b, c are three nodes so that (a, c) and (b, c) are vertices of C then a and b contain no decision statements.

Let $n+1$ be the number of nodes in C, without *Start*. Under the above assumptions, we may fix an $1-1$ map of the nodes of C into the set of numbers $0, 1, 2, \dots, n$, so that *Stop* and the initial node (the successor of *Start*) have the numbers $0, 1$ respectively and for every decision statement with the number i, the „yes-" and „no-" successors have the numbers $i+1$ and $i+2$ respectively.

For every node $i = 1, 2, \dots, n$, it is possible to meet in C one of the situations from fig. 3.

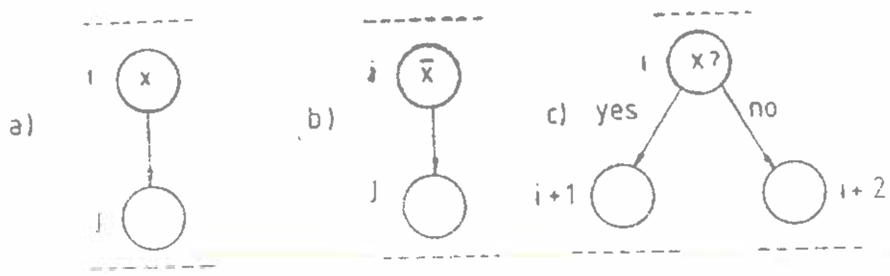

Fig. 3

According to these three cases; the algorithms B_1, B_2, B_3 are achieved by some different simulation of the algorithms

(a) **begin** $x: = x + \varepsilon(i, y)$; $y: = y \pm h\varepsilon(i, y)$ **end**,

(b) **begin** $x: = x - \varepsilon(i, y)$; $y: = y \pm h\varepsilon(i, y)$ **end**,

(c) $y: = \varepsilon(i, y) + (\varepsilon(i, y) \dot{-} x)$,

where $\varepsilon(i, b) = 1$ if $i = b$, $\varepsilon(i, b) = 0$ if $i \neq b$ and $i \pm h = j$.

1. Let q and k be two natural numbers so that $q = 2^{2k} \geq n+2$. The algorithms

$$F(u, v) = ((\bar{u}vv)^{*}\,(\bar{v}uu)^{*})^{k} \text{ and } G(u, v) = ((\bar{u}\bar{u}v)^{*}\,(\bar{v}\bar{v}u)^{*})^{k}$$

satisfy the relations

$$u, v : (a, 0)\,.\,F(u, v)\,.\,(qa, 0) \text{ and}$$

$$u, v : (a, 0)\,.\,G(u, v)\,.\,(\lceil a/q \rceil, 0)$$

for all $a \in \mathrm{N}$. Using these and other simple algorithms, we receive two new algorithms

$$E_i = F^3(x, u)\, u^i(\bar{y}\bar{u}x)^{*} F(u, y)\, (\bar{u}\bar{y}x)^{*}(\bar{x}yu)^{*}\, uG(u, x)\bar{u}$$
$$F(u, x)(\bar{u}\bar{y}x)^{*}\, G(x, u)\, F(y, u)\, (\bar{y}xu)^{*}\, G(u, y)^{*}\bar{u}^i$$
$$(\bar{u}\bar{y}x)^{*}\, (\bar{x}yu)^{*}\, u\, G(u, x)\bar{u}F(u, x)(\bar{u}\bar{y}x)^{*}\, u(\bar{y}\bar{u}x)^{*}xG(x, y)\bar{x}$$

and

$$H = (\bar{x}yu)^{*}u\, G(u, x)\bar{u}F(u, x)(\bar{u}\bar{y}x)^{*}\, G(x, u)$$

so that the relations

$$x, y\, u : (a, b, 0)\,.\,E_i\,.\,(qa+b, 0, \varepsilon(i, b)) \text{ and}$$
$$x, y, u : (qa+b, 0, 0)\,.\,H\,.\,(a, b, 0)$$

hold for every a and $b \leq n$. We also need the algorithm

$$I_i = ((\bar{y}xu)^{*}(\bar{u}xy)^{*})^{l}((\bar{y}xu)^{*})^{m}\,\bar{y}\bar{u} \quad \textit{if} \quad i+h=j \text{ or}$$
$$I_i = ((\bar{y}\bar{x}u)^{*}\,(\bar{u}\bar{x}y)^{*})^{l}((\bar{y}\bar{x}u)^{*})^{m}\,\bar{y}\bar{u}, \quad \textit{if} \quad i-h=j,$$

where $h = 2l+m$ and $0 \leq m \leq 1$.

The last algorithm satisfies the relation

$$x, y, u : (a, b, 0)\,.\,I_i\,.\,(a \pm hb, 0, 0), \text{ for every } a \text{ and } b \leq 1.$$

According to the three above emphasized cases, let us consider the algorithms A_i :

a) $A_i = E_i F(u, y)\, (\bar{u}xy)^{*}\, G(y, u)\, I_i H$,

b) $A_i = E_i F(u, y)(\bar{u}\bar{x}y)^{*}\, G(y, u) I_i H$ and

c) $A_i = E_i(\bar{u}xy)^{*}\, F(y, u)(\bar{x}\bar{y}u)^{*}\, F(u, x)x(\bar{y}\bar{x}u)^{*}\, y(\bar{x}\bar{y}u)^{*}\, F(y, x)(\bar{y}\bar{x}u)^{*}$
$G(u, y)\, (\bar{u}\bar{y}x)^{*}H$,

which satisfy the relations

a) $x, y, u : (a, b, 0)\,.\,A_i\,.\,(a+\varepsilon(i, b),\ b \pm h\varepsilon(i, b), 0)$,

b) $x, y, u : (a, b, 0)\,.\,A_i\,.\,(a-\varepsilon(i, b),\ b \pm h\varepsilon(i, b), 0)$,

c) $x, y, u : (a, b, 0)\,.\,A_i\,.\,(a, b+\varepsilon(i, b)+(\varepsilon(i, b) \dot{-} a), 0)$

for every a and $b \leq n$. Then

$$B_1 = (\bar{u})^{*}(\bar{y})^{*}\, y(\bar{y}yA_1A_2 \dots A_n)^{*}.$$

2. For the algorithm

$$E_i = \textbf{begin}\ u := u+y\ ;\ (v: = v+1)^i;\ u: = u \dot{-} v\ ;\ v: v \dot{-} y\ ;\ v: = v+u\ ;$$
$$u: = 0\ ;\ u: = u+1\ ;\ u := u \dot{-} v\ ;\ v := 0\ \textbf{end}$$

we have

$$y, u, v : (b, 0, 0)\,.\,E_i\,.\,(b, \varepsilon(i, b), 0)$$

for all b and i. Now, let us consider the algorithms A_i as it follows ;

a) $A_i = \textbf{begin}\ E_i\ ;\ x: = x+u\ ;\ (y: = y \pm u)^h\ ;\ u := 0\ \textbf{end}$,

b) $A_i = \textbf{begin}\ E_i\ ;\ x: = x \dot{-} u\ ;\ (y: = y \pm u)^h\ ;\ u := 0\ \textbf{end}$,

c) $A_i = \textbf{begin}\ E_i\ ;\ y: = y+u\ ;\ u: = u \dot{-} x\ ;\ y := y+u\ ;\ u: = 0\ \textbf{end}$.

It results

$$B_2 = (\bar{y})^{*}y(\bar{v}\ ;\ y: = y+1\ ;\ u: = 0\ ;\ v: = 0\ ;\ A_1A_2 \dots A_n)^{*}.$$

3. Let us denote the algorithm

$$\textbf{begin}\ a: = a+c\ ;\ b: = b \dot{-} c\ \textbf{end}$$

by $[a, b, c]$. If

$$E_i = [u, w, z]^i [v, u, y] [w, v, z]^i [v, w, u] [w, u, z] [u, w, z] [w, u, v] [w, v, z]^n$$

then, for every a and $b \leqslant n$, there is a number c so that

$$y, z, u, v, w : (b, 1, 0, 0, a) \,.\, E_i \,.\, (b, 1, \varepsilon(i, b), 0, 0, c).$$

The algorithms A_i are

a) $A_i = E_i[x, w, u]\, I_i[w, u, z]$,
b) $A_i = E_i[w, x, u]\, I_i[w, u, z]$,
c) $A_i = E_i[y, w, u]\, [w, u, x][y, w, u]\, [w, u, z]$

where $I_i = [y, w, u]^h$ if $i+h=j$ and $I_i = [w, y, u]^h$ if $i-h=j$. Then

$$B_3 = (\overline{u})^* (\overline{v})^* (\overline{y})^* (\overline{z})^* yz(\overline{y}[y, w, z]\, A_1 A_2 \ldots A_n)^*.$$

Theorem 3. *If A is a R-algorithm then there is a NR-algorithm B such that $B \in \mathbf{RA}(3)$, $dp(B) \leqslant 3$ and $(\gamma s, 0, 0).\, B \,.\, (\gamma A(s), 0, 0)$ for all s.*

Proof. Let A be a R-algorithm. By theorem 1, we have an algorithm C so that $C \in \mathbf{RA}(2)$ and $(\gamma s, 0) \,.\, C \,.\, (\gamma A(s), 0)$ for every state s. By theorem 2, it must exist an algorithm

$$D = (\overline{2})^* 2\, (\overline{2} V_1 V_2 \ldots V_s)^*$$

with V_i of the form $a := 0$, $a := a+1$, $a := a+b$ or $a := a \dot{-} b$, so that $D \in \mathbf{RA}(5)$ and $D \sim C(\{0, 1\})$. Therefore

$$(\gamma s, 0, 0, 0, 0) \,.\, D \,.\, (\gamma A(s), 0, 0, 0, 0)$$

for every s. Now, in strong concordance with the structure of D, we shall construct a NR-algorithm $F \in \mathbf{RA}(3)$ with the depth 3, so that if

$$(a, 0, 0, 0, 0) \,.\, D \,.\, (u, v, w, x\, y),$$

then $(2^a, 0, 0) \,.\, F \,.\, (2^u 3^v 5^w 7^x 11^y, 0, 0)$, for all a, u, v, w, x, y.

To this end, let us consider the algorithms

$$E_p = (\overline{0}12)^* (\overline{2^p 0})^* (\overline{02^p})^* (\overline{1}\overline{2}0)^*\, 1(\overline{2}\overline{1})^*,$$
$$T_{p,q} = E_p(\overline{1}(\overline{0^p}1)^* (\overline{1}0^q)^* E_p)^*,$$
$$P_{p,q} = T_{q,13}\, T_{13,pq} \text{ and}$$
$$M_{p,q} = P_{q.17}\, E_{17p}\, (\overline{1}(\overline{0^{17p}}1)^* (\overline{1}0)^* E_{17p})^*\, T_{17,1}$$

where $p \in \{2, 3, 7, 5, 11\}$, $q \in \{1, 2, 3, 5, 7, 11\}$ and $p \neq q$.
These algorithms have the following properties:

—for every b, $(b, 0, 0) \,.\, E_p \,.\, (b, \eta(p, b), 0)$,

where $\eta(p, b) = 1$ if p/b and $\eta(p, b) = 0$ if $\rceil p/b$;

— for all a and b, if $\rceil p/b$ then $(p^a b, 0, 0) \,.\, T_{p,q} \,.\, (q^a b, 0, 0)$;
— for all a, b, c if $\rceil 13\ pq/c$ then $(p^a q^b c, 0, 0) \,.\, P_{p,q} \,.\, (p^{a+b} q^b c, 0, 0)$;
— for all a, b, c, if $\rceil 13 \,.\, 17\ pq/c$ then $(p^a q^b c, 0, 0) \,.\, M_{p,q} \,.\, (p^{a \dot{-} b} q^b c, 0, 0)$.

Then

$$F = (\overline{0}1^s)^* (\overline{1}0)^* 1(\overline{1}(\overline{0}1)^* (\overline{1^5}0)^* L_1 L_2 \ldots L_s E_s)^*$$

where $L_i (i = 1, 2, \ldots, n)$ is $T_{p_a, 1}$, $(\overline{0}1^{p_a})^* (\overline{1}0)^*$, P_{p_a, p_b} or M_{p_a, p_b}, according to V_i has one of the four above specified forms.

Taking into account the connection between D and F, we conclude

$$(2^{\gamma s}, 0, 0) \,.\, F \,.\, (2^{\gamma A(s)}, 0, 0)$$

for every state s. Finally, the algorithm we looked for is

$$B = (\overline{0}2)^* 0\, R_2(0, 1, 2)\, F T_2(0, 1, 2)\,.$$

Theorem 4. $\mathbf{RF}^n = \mathbf{NRF}^n(2, 2 + \max(2, n))$ *for all* $n > 0$.

Proof. Let $f \in \mathbf{RF}^n$. Then there is an R-algorithm A so that for all the natural numbers $a_1, a_2, \ldots, a_n$,

$$(0, a_1, a_2, \ldots, a_n, 0, 0, \ldots) \,.\, A \,.\, (f(a_1, a_2, \ldots, a_n), 0, 0, \ldots).$$

From the theorem 2 it follows that there is an algorithm $B \in \mathbf{RA}(2)$ for which $(\gamma s, 0) \,.\, B \,.\, (\gamma A(s), 0)$ holds for every state s and therefore

$$(p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n}, 0) \,.\, B \,.\, (2^{f(a_1, a_2, \ldots, a_n)}, 0)$$

for all $a_1, a_2, \ldots, a_n$. By theorem 2 we have a NR-algorithm C, so that $\mathrm{dp}(C) = 2$, $\mathrm{sp}(C) = \{0, 1, 2, 3\}$ and

$$(p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n}, 0, 0, 0) \,.\, C \,.\, (2^{f(a_1, a_2, \ldots, a_n)}, 0, 0, 0)$$

for arbitrary $a_1, a_2, \ldots, a_n$. If

$$D = (\overline{0})^* (\overline{1})^* 0 R_{p_1}(0, 1, 2)\, R_{p_2}(0, 1, 3) \ldots R_{p_n}(0, 1, n+1) C T_2(0, 1, 2),$$

then $\mathrm{dp}(D) = 2$, $D \in RA(2 + \max(2, n))$ and $[D, 2, 3, \ldots, n+1, 0, f]$.
Therefore $f \in \mathbf{NRF}^n(2, 2 + \max(2, n))$.

Theorem 5. $\mathbf{RF}^n = \mathbf{NRF}^n(3, 1 + \max(2, n))$ *for all* $n > 0$.

Proof. Let $f \in \mathbf{RF}^n$ and let us consider the functions

$$h(x) = [\lceil x / (p_1^{e(1, x)} p_2^{e(2, x)} \ldots p_{n-1}^{e(n-1, x)}) \rceil / q] \dot{-} 1.$$

where $q = p_1 p_2 \ldots p_{n-1}$ and

$$g(x) = f(e(1, x), e(2, x), \ldots, e(n-1, x), h(x)).$$

Clearly, if

$$x = p_1^{x_1} p_2^{x_2} \ldots p_{n-1}^{x_{n-1}} (q x_n + 1)$$

then $g(x) = f(x_1, x_2, \ldots, x_n)$.

The function g belongs to $\mathbf{RF}^1$ and therefore a R-algorithm A must exist, so that

$$(x, 0, 0, \ldots) \,.\, A \,.\, (g(x), 0, 0, \ldots)$$

for arbitrary x. By theorem 3, there is a NR-algorithm B such that $B \in \mathbf{RA}(3)$, $\mathrm{dp}(B) \leqslant 3$ and $(2^x, 0, 0) \,.\, B \,.\, (2^{g(x)}, 0, 0)$. Then, for

$$C = (\overline{0})^* (\overline{n}0^q)^*\, 0\, R_{p_1}(0, n, 1)\, R_{p_2}(0, n, 2) \ldots$$
$$\ldots R_{p_{n-1}}(0, n, n-1)\, (\overline{0}2)^* 0 R_2(0, 1, 2)\, B T_2(0, 1, 2),$$

the relations $dr(C) \leq 3$, $C \in \mathbf{RA}(1+\max(2, n))$ and $[C, 1, 2, \dots, n, 0, f]$ holds. Therefore $f \in \mathbf{NRF}^n(3, 1+\max(2, n))$.

The $n+1$-ary function φ is *universal* for $\mathbf{RF}^n$ if, for every $f \in \mathbf{RF}^n$, there exists a natural number a such that

$$\forall x_1 x_2 \dots x_n\ \varphi(a, x_1, x_2, \dots, x_n) = f(x_1, x_2, \dots, x_n),$$

and conversely, for every a there exists a function $f \in \mathbf{RF}^n$ satisfying the above relation.

The theorem 2 allows to give an easy proof on the existence of recursive universal functions. Indeed, by this theorem, for any $f \in \mathbf{RF}^n$ we may construct an algorithm

$B=$ **while** $y>0$ **do** $[u_1, v_1, w_1]\ [u_2, v_2, w_2] \dots [u_t, v_t, w_t]$ **end** where y is the register 0, so that for arbitrary $a_1, a_2, \dots, a_n$ the relation $(1, a_1, a_2, \dots, a_n, 1, 0, 0, \dots) . B . (0, f(a_1, a_2, \dots, a_n), 0, 0, \dots)$ holds. Now, look to the algorithm U:

while $e(0, m) > 0$ **do** $i := 1$;
  while $i < e(0, q)$ **do**
  end $m := f(m, q, i)$; $i := i+3$
end

where $f(m, q, i) = [m / p_{e(i+1,q)}^{e(e(i+1,q),m)}]\, p_{e(i,q)}^{e(e(i+2,q),m)} \cdot p_{e(i+1,q)}^{e(e(i+1,q),m) \dot{-} e(e(i+2,q),m)}$

We observe that

$$m, q, i : (2 p_1^{a_1} p_2^{a_2} \dots p_n^{a_n} p_{n+1}, \rho(B), 0) . U . (3^{f(a_1, a_2, \dots, a_n)} \dots, \rho(B), 3t+1)$$

if

$$\rho(B) = p_0^{3t} p_1^{u_1} p_2^{v_1} p_3^{w_1} \dots p_{3t-2}^{u_t} p_{3t-1}^{v_t} p_{3t}^{w_t}$$

and therefore the algorithm

begin $m := 2 p_1^{x_1} p_2^{x_2} \dots p_n^{x_n} p_{n+1}$; U ; $y := e(1, m)$ **end**

computes in $x_1, x_2, \dots, x_n$ and y an universal recursive function for $\mathbf{RF}^n$.

By means of the universal recursive functions and previous theorems, some new results may be obtained. For example, from theorem 2 it results

Corollary 6. If $n>0$ then there is an algorithm $B=(\bar{y})^{\cdot} y(\bar{y} V_1 V_2 \dots V_s)^{\cdot}$ with V_i of the form $a := 0$, $a := a+1$, $a := a+b$ or $a := a \dot{-} b$ and so that all recursive n-ary functions are computable by algorithms from the set $\{$**begin** $u := c$; B **end** $\mid c \in \mathrm{N}\}$.

REFERENCES

1. B a r e d i n, Y a. M. — *Ob odnom klasse maşin Tiuringa*, Algebra i logica, 1, 1963.
2. B ö r g e r, E. — *Recursively unsolvable algorithmic problems and related questions reexamined.* Lect. Notes in Math., 499, Springer-Verlag, Berlin, 1975.
3. C a z a c u, C. and C a z a c u, M. — *Teoria algoritmilor*, University of Iassy, 1977.
4. M a l t e v, A.I. — *Algoritmi i rekursivnie funkţii*, Izd, Nauka, Moskva, 1965.
5. R o g e r s, H. jr. — *Theory of recursive functions and effective calculability*, Mc. Graw-Hill Book Comp., New-York, 1967.

Received 25. VII. 1980

Faculty of Mathematics University of Iaşi, R.S. Romania

Analele ştiinţifice ale Universităţii „Al. I. Cuza" din Iaşi
Tomul XXVII, s. I a, 1981 f. 2

SUR UN INVARIANT DANS UN TOPOS ÉLÉMENTAIRE

PAR

MIRCEA GEANĂU

Soit G un groupe qui opère sur un ensemble A. Les éléments $a \in A$ ayant la propriété $\sigma a = a$, $\forall \sigma \in G$, sont appelés éléments invariants de l'ensemble A et on les désigne par G-ensembles et on les note A^G. L'association $A \to A^G$ définit un foncteur $(\ )^G$: Act $G \to$ Ens, où Act G est la catégorie dont les objets sont les G-ensembles et les morphismes sont les homomorphismes des G-ensembles.

Dans cet ouvrage on essaye de traiter cette question dans un topos élémentaire et on lui donne un caractère fonctoriel.

Définition 1. *Un topos élémentaire est une catégorie $\mathcal{E}$ qui vérifie les conditions suivantes :*

(*i*) $\mathcal{E}$ a des limites projectives finies c.à.d. $\mathcal{E}$ a des produits fibrés et un objet final 1.

(*ii*) pour chaque objet X on a un foncteur exponentiel $(-)^X : \mathcal{E} \to \mathcal{E}$ qui est adjoint à droite au foncteur $(-)\ x X$.

(*iii*) $\mathcal{E}$ a un sousobjet classifiant Ω et un morphisme $1 \xrightarrow{v} \lambda$ tel que $\forall A$ et $A' \xrightarrow{i} A$ monomorphisme, il y a un seul $\varphi_i : A \to \lambda$ tel que le diagramme suivant soit un produit fibré :

$$\begin{tikzcd} A' \arrow[r] \arrow[d, "i"'] & 1 \arrow[d, "v"] \\ A \arrow[r, "\varphi_i"'] & \lambda \end{tikzcd}$$

Exemple. Le topos des ensembles, le topos des faisceaux sur un espace topologique, sont des catégories $\mathcal{E}$.

Soit (G, m) un objet groupe de $\mathcal{E}$ et soit A un objet de $\mathcal{E}$.

Définition 2. *Une opération à gauche de l'objet G sur A est un morphisme $W : G \times A \to A$ qui vérifie les conditions suivantes :*

(*i*) le diagramme suivant est commutatif :

$$\begin{tikzcd} G \times G \times A \arrow[r, "m \times 1A"] \arrow[d, "1_G \times W"'] & G \times A \arrow[d, "W"] \\ G \times A \arrow[r, "W"'] & A \end{tikzcd}$$

(*ii*) $w < \varepsilon\eta, 1_A > = 1_A$

où $\eta : A \to 1$ est le seul morphisme, et $\mathcal{E} : 1 \to G$ est le morphisme ayant la propriété que quel que soit l'objet Y et le morphisme $\alpha : Y \to 1$, $\mathcal{E}\alpha$ est l'élément neutre du groupe Hom (Y, G).