A FIXED-POINT THEOREM FOR RECURSIVE-ENUMERABLE LANGUAGES AND SOME CONSIDERATIONS ABOUT FIXED-POINT SEMANTICS OF MONADIC PROGRAMS

Sorin Istrail
Computer Center, University "Al.I.Cuza"
Iaşi 6600, Romania

ABSTRACT

This paper generalizes the ALGOL-like theorem showing that every λ-free context-sensitive (recursive-enumerable) language is a component of the minimal solution of a system of equation $X=F(X)$, where $X=(X_1,\dots,X_t)$, $F=(F_1,\dots,F_t)$, $t\geq 1$ and F_i, $1\leq i\leq t$ are regular expressions over the alphabet of operations: $\{$concatenation, reunion, kleene "+" closure, nonereasing finite substitution (arbitrary finite substitution), intersection$\}$.

In the second part is presented a method which constructs for a monadic program a system of equations (in the above form) so that one of the components of the minimal solution of the system gives the partial function f computed by the program in a language form:

$$\left\{ a^{n+1} \# b^{f(n)+1} \mid n \in \mathrm{Dom}\ f \right\} .$$

1. PRELIMINARIES

Let V be a finite set of symbols, V^* the free monoid generated by V, λ the unit of V^*, $V^+ = V^* \setminus \{\lambda\}$

The elements of V^* are called words and the subsets of V^* are called languages. We suppose the reader familiar with the basic facts about formal language theory [7] and developmental systems [2]. Let us denote by R, CF, CS, CS_λ, RE the classes of regular, context-free, context-sensitive, λ-free context-sensitive and recursive-enumerable languages.

DEFINITION. A OL-system is a triple $S = \langle V,P,w \rangle$ where P is a finite set of pairs, $P \subset V \times V^*$ with the property that for every $a \in V$, there exists $u \in V^*$ so that $(a,u) \in P$; the elements of P are called rules and are usually denoted by $p \rightarrow q$, for $(p,q) \in P$; w is a word from V^*, called the axiom. The set P is called table, and the pair $S' = \langle V,P \rangle$ is sometimes called OL-scheme.

The binary relation $\underset{S}{\Longrightarrow} \subset V^* \times V^*$ is defined by $w_1 \underset{S}{\Longrightarrow} w_2$ if $w_1 = a_1 \dots a_t$, $w_2 = u_1 \dots u_t$, $t \geq o$, $a_j \in V$, $u_j \in V^*$, $1 \leq j \leq t$ and for every i, $1 \leq i \leq t$, $a_i \rightarrow u_i \in P$.

The relation $\overset{*}{\underset{S}{\Longrightarrow}}$ denotes the reflexive transitive closure of $\underset{S}{\Longrightarrow}$.

A language L is called OL language if there exists an OL-system S so that L(S) = L.

A generative device, which is a derivational restricted OL system is introduced in the following lines.

DEFINITION. A perturbant configuration for the OL-scheme $S = \langle V,P \rangle$ is a family $\Pi = (\pi_a)_{a \in V}$ where for every $a \in V$, $\pi_a = \langle n(a), E_a, F_a \rangle$ and

i) $n(a) \geq 1$

ii) $E_a = \{ E_a^{(1)}, \ldots, E_a^{(n(a))} \}$, $\bigcup_{i=1}^{n(a)} E_a(i) = V^+$,

$$E_a^{(i)} \cap E_a^{(j)} = \emptyset \text{ , } i \neq j, \ 1 \leq i, \ \ j \leq n(a)$$

iii) $F_a = \{ F_a^{(1)}, \ldots, F_a^{(n(a))} \}$, $\emptyset \neq F_a^{(i)} \subset (P \cap \{a\} \times V^*)$
$1 \leq i \leq n(a)$

Let be $\mathcal{L}$ a family of languages. A perturbant configuration is called $\mathcal{L}$-perturbant configuration for an OL scheme S if $\Pi = (\pi_a)_{a \in V}$ and for every $a \in V$ and i, $1 \leq i \leq n(a)$ we have $E_a^{(i)} \in \mathcal{L}$.

DEFINITION. A SICK-OL system is a triple $\mathcal{S} = (S, \Pi, w)$ where:

i) $S = \langle V,P,w \rangle$ is an OL-system

ii) Π is a perturbant configuration for the scheme $S' = \langle V,P \rangle$.

iii) w is the axiom of $\mathcal{S}$, $w \in V^*$.

We define now the following binary relation $\underset{\mathcal{S}}{\Longrightarrow}$, for $w = a_1 \ldots a_t$, $u = u_1, \ldots, u_t$ with $a_k \in V$, $u_k \in V^*$, $1 \leq k \leq t$ we put $w \underset{\mathcal{S}}{\Longrightarrow} u$ iff for every j, $1 \leq j \leq t$, $a_j \rightarrow u_j \in F_{a_j}^{(s)}$, where "s" is defined by $w \in E_{a_j}^{(s)}$.
(In words, we can apply for a letter "a" occuring in a word w_1 rules from those set in F_a corresponding to those set in E_a which contains w_1).

Let $\overset{*}{\underset{\mathcal{S}}{\Longrightarrow}}$ be the reflexive transitive closure of $\underset{\mathcal{S}}{\Longrightarrow}$.

The language generated by the SICK-OL system $\mathcal{S} = (S, \Pi, w)$ is defined by $L(\mathcal{S}) = \{ u \mid u \in V^*, w \overset{*}{\underset{\mathcal{S}}{\Longrightarrow}} u \}$, where $S' = \langle V,P \rangle$.

A language L is called SICK-OL language if there exists a SICK-OL system $\mathcal{S}$ so that $L(\mathcal{S}) = L$.

DEFINITION. An extended SICK-OL system is a 4-tuple $\mathcal{S}' = (S, \Pi, w, Z)$, where $\mathcal{S} = (S, \Pi, w)$ is a SICK-OL system, $S' = \langle V, P \rangle$ and $Z \subset V$.

The <u>language generated</u> by the extended SICK-OL system $\mathcal{S}' = (S, \Pi, w, Z)$ is given by $L(\mathcal{S}') = L(S, \Pi, w) \cap Z^{*}$.

Let us denote by SICK-OL the class of SICK-OL languages. If $\mathcal{L}$ is a family of languages, $\mathcal{L}$ SICK-OL denotes the class of languages obtained from those SICK-OL system with $\mathcal{L}$-perturbant configurations.

If the rules of a certain type of L systems do not erase, the L-system is called <u>propagating</u>.

We add the letters P and E (or both) to the abreviation of L-systems to denote the classes of corresponding Propagating and Extended L-systems.

2. TWO FIXED-POINT THEOREMS

In this section we present two fixed-point theorems, one for $\underline{CS}_{\lambda}$ and another for <u>RE</u>. They are generalizations of the well known ALGOL-like theorem.

In the following we are interested in P SICK-OL systems with <u>R</u>-perturbant configurations.

<u>THEOREM 1</u>. For every λ-free context-sensitive language L, there exists a propagating extended <u>R</u> SICK-OL system $\mathcal{S}'$ so that $L(\mathcal{S}')=L$.

<u>PROOF</u>. Let $G=(I_N, I_T, x_0, F)$ be a context-sensitive grammar so that $L(G)=L$ and suppose that $\lambda \notin L$. The rules of the grammars are in the form $pxq \rightarrow puq$ where $p,q \in V^{*}$, $x \in I_N$, $u \in V^{+}$ and $V = I_N \cup I_T$. Thus no rules in the form $x_0 \rightarrow \lambda$, belongs to F.

Let us consider a new alphabet $I_N = \{\bar{a} \mid a \in I_N\}$. We need some preliminary notations:

$$F(x) = \{x \rightarrow u \mid p,q \in V^{*}, \quad u \in V^{+}, \quad pxq \rightarrow puq \in F\}$$

If t_x is the number of elements of $F(x)$ then:

$$T_x = \{(p_i^x, r_i^x) \mid p_i^x \, x \, r_i^x \rightarrow p_i^x \, u \, r_i^x \in F, \quad 1 \leq i \leq t_x\}$$

(the set of all contexts for x, used in the rules of G).

$$Z(i,x) = \{\bar{x} \rightarrow u \mid p_i^x \, x \, r_i^x \rightarrow p_i^x \, u \, r_i^x \in F\} \cup \{\bar{x} \rightarrow \bar{x}\}$$

$$F(i,x) = \bigcup \{Z(j,x) \mid p_i^x = vp_j^x, \; r_i^x = r_j^x z, v,z \in V^{*}\}$$

$$E(i,x) = V^{*}p_i^x \, x \, r_i^x V^{*} \setminus \bigcup \{V^{*} p^x \, x \, r_j^x \, V^{*} \mid p_j^x = v \, p_i^x, \; r_j^x = r_i^x z, \quad v,z \in V^{*}, \quad vz \neq \lambda\}.$$

We notice that for $i \neq j$, $1 \leq i,j \leq t_x$, $E(i,x) \cap E(j,x) = \emptyset$.

We intend to construct a propagating extended SICK-OL system $\mathcal{S}' = (S, \Pi, x_0, I_T)$. So that $L(\mathcal{S}') = L(G)$.

We define $S = \langle V \cup \bar{I}_N, D \rangle$, where

$$D = \Big(\bigcup_{\substack{x \in I_N \\ i \leq i \leq t_x}} Z(i,x) \Big) \cup \{ x \longrightarrow x,\ \bar{x} \longrightarrow \bar{x}, x \longrightarrow \bar{x} \mid x \in I_N \} \cup \{ a \longrightarrow a \mid a \in I_T \}$$

We define a R-perturbant configuration $\Pi = (\pi_y)_{y \in V \cup \bar{I}_N}$ by

1) for $x \in I_N$, $\pi_x = \langle 2, E_x, F_x \rangle$, where

$E_x^{(1)} = V^+, E_x^{(2)} = (V \cup \bar{I}_N)^+ \setminus V^+$, $F_x^{(1)} = \{ x \longrightarrow x,\ x \longrightarrow \bar{x} \}$,

$F_x^{(2)} = \{ x \longrightarrow x \}$.

2) for $\bar{x} \in \bar{I}_N$, $\pi_{\bar{x}} = \langle t_x + 1,\ E_x,\ F_x \rangle$, where

$E_{\bar{x}}^{(i)} = E(i,x)$, $\quad F_{\bar{x}}^{(i)} = F(i,x)$, $\quad 1 \leq i \leq t_x$

$E_{\bar{x}}^{(t_x+1)} = V^+ \setminus \bigcup_{i=1}^{t_x} E_{\bar{x}}^{(i)}$, $\quad F_{\bar{x}}^{(t_x+1)} = \{ \bar{x} \longrightarrow \bar{x} \}$

3) for $a \in I_T$, $\quad \pi_a = \langle 1, (V \cup \bar{I}_N)^+, \{ a \longrightarrow a \} \rangle$.

DEFINITION. A Self-controled Tabled OL system (SC-TOL) is a 5-tuple $\mathcal{C} = (V, m(\mathcal{C}), D, C, w)$ where

i) V is the alphabet of $\mathcal{C}$;

ii) $m(\mathcal{C})$ is a positive integer;

iii) $D = \{ D_i \}_{i=1}^{m(\mathcal{C})}$, $\quad D_i \cap D_j = \emptyset$, $i \neq j$, $1 \leq i,j \leq n \ (\mathcal{C})$ $\quad \bigcup_{i=1}^{m(\mathcal{C})} D_i = V^+$

iv) $C = \{ C_i \}_{i=1}^{m(\mathcal{C})}$, $C_i \subset V \times V^*$, is a table, $1 \leq i \leq m \ (\mathcal{C})$

If $\mathcal{C}$ is a SC-TOL system, the following binary relation is introduced: for $w = a_1 \ldots a_t$, $u = u_1 \ldots u_t$ with the property that $a_k \in V$ and $u_k \in V^*$, $1 \leq k \leq t$ we put $w \underset{\mathcal{C}}{\Longrightarrow} u$ iff for every $j, 1 \leq j \leq t$, $a_j \longrightarrow u_j \in C_s$, where "s" is defined by $w \in D_s$. (In words, we can apply to w rules from a table C_s iff $w \in D_s$).

The definitions of $\overset{*}{\underset{\mathcal{C}}{\Longrightarrow}}$, language generated by $\mathcal{C}$, SC-TOL language, E SC-TOL, $\mathcal{L}$ SC-TOL can be obtained similarly.

Let us denote by $\hat{T}$ the finite substitution generated by a table T.

THEOREM 2. For every SC-TOL system $\mathcal{C}$ there is a SICK-OL system $\mathcal{S}$ so that $L(\mathcal{C}) = L(\mathcal{S})$.

PROOF. Let us suppose that we have an SC-TOL $\mathcal{C} = (V, m(\mathcal{C}), C, D, w)$. Then we define a perturbant configuration $\Pi = (\pi_a)_{a \in V}$ by

$$\pi_a = (m(\mathcal{C}), D, \{C_i \cap (a \times V^*)\}_{i=1}^{m(\mathcal{C})})$$

The SICK-OL system $\mathcal{S} = (V, \Pi, w)$ generates exactly $L(\mathcal{C})$.

The converse of Theorem 2 is also true.

THEOREM 3. For every SICK-OL system $\mathcal{S} = (S, \Pi, w)$ there exists an equivalent SC-TOL system $\mathcal{C} = (V, m(\mathcal{C}), D, C, w)$, i.e. $L(\mathcal{C}) = L(\mathcal{S})$.

PROOF. Let be $V = \{a_1, \ldots, a_s\}$ and Π detailed by

$$E_{a_j}^{(i)}, \; F_{a_j}^{(i)}, \quad 1 \leqslant i \leqslant n(a_j), \quad 1 \leqslant j \leqslant s.$$

For k_j variyng in $\{1, \ldots, n(a_j)\}$, $1 \leqslant j \leqslant s$, let us consider the sets:

$$E_{a_1}^{(k_1)} \cap E_{a_2}^{(k_2)} \cap \cdots \cap E_{a_s}^{(k_s)} = T(k_1, \ldots, k_s)$$

Now we have a partition of V^* given by the collection

$$\Delta = \{T(k_1, \ldots, k_s) \mid T(k_1, \ldots, k_s) \neq \emptyset,$$
$$k_j \in \{1, \ldots, n(a_j)\}, \quad 1 \leqslant i \leqslant s\}.$$

If m_o is the number of sets in Δ we define a SC-TOL

$$\mathcal{C} = (V, m_o, \{T(k_k, \ldots, k_s) \mid T(k_1, \ldots, k_s) \neq \emptyset\},$$
$$\{Z(k_1, \ldots, k_s) \mid T(k_1, \ldots, k_s) \neq \emptyset\}, w), \text{ where}$$
$$Z(k_1, \ldots, k_s) = \bigcup_{i=1}^{s} F_{a_i}^{(k_i)}$$

It is easy to see that

$$L(\mathcal{S}) = L(\mathcal{C}).$$

COROLLARY 1. SICK-OL = SC-TOL

EP R SICK-OL = EP R SC-TOL $\supseteq$ CS_λ

The inclusion presented in the Corollary 1 is in fact equality.

THEOREM 4. Every propagating R SC-TOL system generates a context-sensitive language.

COROLLARY 2.

EP R SICK-OL = EP R SC - TOL = CS_λ

THEOREM 5. For every SC-TOL system $\mathcal{C} = (V, m(\mathcal{C}), P, Q, w)$ there exists a system of equations

$$(*)\qquad \begin{cases} X_1 = F_1(X_1,\dots,X_t) \\ \dots\dots\dots\dots\dots\dots \\ X_t = F_t(X_1,\dots,X_t) \end{cases}$$

so that $L(\mathcal{G}) = \bigcup_{n=1}^{t} X_n^{MIN}$ where $(X_1^{MIN},\dots,X_t^{MIN})$ is the minimal solution of $(*)$.

PROOF. Let be the system of equations

$$(1)\qquad \begin{cases} X_1 = \hat{Q}_1\ (P_1 \cap (X_1 \cup \dots \cup X_t \cup \{w\})) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ X_t = \hat{Q}_t\ (P_t \cap (X_1 \cup \dots \cup X_t \cup \{w\})) \end{cases}$$

with $t = m(\mathcal{G})$ and let us denote $F_i(X_1,\dots,X_t) = \hat{Q}_i(P_i \cap (X_1 \cup \dots \cup X_t \cup \{w\}))$.

The minimal solution of the system (1) $(X_1^{MIN},\dots, X_t^{MIN})$ is given by

$$X_i^{MIN} = \bigcup_{n=0}^{\infty} X_i^{(n)}, \qquad 1 \leqslant i \leqslant t$$

and

$$X_i^{(n+1)} = F_i(X_1^{(n)},\dots,X_t^{(n)}), \quad n \geqslant 0,$$

We observe that $X_i^{(n)}$ is the set of all words from $L(\mathcal{G})$ with the property that are obtained in n steps of derivation in $\mathcal{G}$, and the last table used is Q_i. Of course X_i^{MIN} is the set of all words in $L(\mathcal{G})$ with the property that the last table used is Q_i.

Now it is manifest that

$$L(\mathcal{G}) = \bigcup_{i=1}^{t} X_i^{MIN}$$

THEOREM 6. Every E SC-TOL L is a component of the minimal solution of a system of equations in the form $(*)$.

PROOF. Let us consider $\mathcal{G}' = (V, m(\mathcal{G}'), P, Q, w, M)$ and a copy of $\mathcal{G}'$ with all letters a in V in the form $\bar{a}$: $\bar{\mathcal{G}}' = (\bar{V}, m(\mathcal{G}'), \bar{P}, \bar{Q}, \bar{w}, \bar{M})$.

Let us define now a SC-TOL $\mathcal{G}_1$.

We consider an alphabet $V' = \bar{V} \cup M \cup \{\sigma\}$, σ a new symbol.

Let us define a finite substitution h on V' by $h(a) = \{a, \bar{a}\}$, $\bar{a} \in \bar{M}$; $h(\bar{b}) = \{b\}$, $b \in \bar{V} - \bar{M}$; $h(c) = \{c\}$, $c \in M \cup \{\sigma\}$

1) For i, $1 \leqslant i \leqslant m(\mathcal{G})$ take

$$R_i = h(\bar{P}_i) \setminus M^+ \quad \text{and}$$

$$T_i = \{u \rightarrow v \mid u \in h(\bar{a}),\ v \in h(\bar{z}),\ \bar{a} \rightarrow \bar{z} \in \bar{Q}_i\} \cup \{\sigma \rightarrow \sigma\}$$

2) $R_{m(\mathcal{G}')+1} = M^+$, $T_{m(\mathcal{G}')+1} = \{x \rightarrow x \mid x \in V'\}$

3) $R_{m(\mathcal{C}')+2} = \{\sigma\}$, $T_{m(\mathcal{C}')+2} = \{\sigma \to u \mid u \in h(\bar{w})\} \cup \{x \to x \mid x \in V' - \{\sigma\}\}$

4) $(V')^+ - \bigcup_{i=1}^{m(\mathcal{C}')+2} R_i = R_{m(\mathcal{C}')+3}$

$T_{m(\mathcal{C}')+3} = \{x \to x \mid x \in V'\}$

We define the SC-TOL $\mathcal{C}_1$ by

$$\mathcal{C}_1 = (V', m(\mathcal{C}')+3, R, T, \sigma)$$

and we associate to $\mathcal{C}_1$ the system of equations:

$$\begin{cases} X_1 = \hat{T}_1(R_1 \cap (X_1 \cup \ldots \cup X_t \cup \{\sigma\})) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ X_t = \hat{T}_t(R_t \cap (X_1 \cup \ldots \cup X_t \cup \{\sigma\})) \end{cases}$$

where $t = m(\mathcal{C}')+3$.

We have

$$X_{t-2}^{MIN} = X_{m(\mathcal{C}')+1}^{MIN} = (\bigcup_{i=1}^{t} X_i^{MIN}) \cap R_{m(\mathcal{C}')+1}$$

(because $\hat{T}_{m(\mathcal{C}')+2}$ is the identity) $= (\bigcup_{i=1}^{t} X_i^{MIN}) \cap M^+ = L(\mathcal{C}_1) \cap M^+$.

It is easy to see that $\bar{u} \in L(\bar{\mathcal{C}}')$ iff $u \in L(\mathcal{C}')$ iff $u \in L(\mathcal{C}_1) \cap M^+$.

<u>THEOREM 7</u>. Let us consider the following data:

i) V an alphabet;

ii) $T_1, \ldots, T_p$, λ-free tables on V;

iii) $R_1, \ldots, R_p$, a partition of V^+ with each R_i regular;

iv) w a word over V.

Then, each component of the minimal solution of the system

$$\begin{cases} X_1 = \hat{T}_1(R_1 \cap (X_1 \cup \ldots \cup X_p \cup \{w\})) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ X_p = \hat{T}_p(R_p \cap (X_1 \cup \ldots \cup X_p \cup \{w\})) \end{cases}$$

is a context-sensitive language.

<u>PROOF</u>. The system of equations defines a SC-TOL $\mathcal{C} = (V, p, \{R_1, \ldots, R_p\}, \{T_1, \ldots, T_p\}, w)$ and we have that $L(\mathcal{C}) = \bigcup_{i=1}^{p} X_i^{MIN}$, where $X^{MIN} = (X_1^{MIN}, \ldots, X_t^{MIN})$ is the minimal solution of the system.

It can be proved that $X_i^{MIN} = \hat{T}_i(L(\mathcal{C}) \cap R_i)$, for all i, $1 \leq i \leq p$.

By theorem 4 it follows that $L(\mathcal{C})$ is in $\underline{CS}_\lambda$, and so is

$\hat{T}_i(L(\mathcal{G}) \cap R_i) = X_i^{MIN}$, $1 \leq i \leq p$.

COROLLARY 3. A language $L \subseteq V^+$ is in CS if and only if it is a component of the minimal solution of a system of equations in the form fulfiling the conditions i) - iv) from Theorem 7.

CORCLLARY 4. Every CS language $L \subseteq V^+$ is a component of the minimal solution of a system of equations in the form:

$$(\substack{*\\*}) \quad \begin{cases} X_1 = F_1(X_1,\dots,X_t) \\ \dots\dots\dots\dots\dots\dots \\ X_t = F_t(x_1,\dots,X_t) \end{cases}$$

where $F_1,\dots,F_t$ are regular espressions over the alphabet $\{".", "\cup", "+", "h_\lambda", "\cap"\} \cup V \cup \{),(\}$. ($h_\lambda$ denotes the λ -free finite substitution).

CONJECTURE 1. The converse of the Corollary 4 is also true.

If the above conjecture holds, we have a fixed-point characterization of CS_λ languages using the set of operations: $\{ ., \cup, h_\lambda, \cap, +\}$.

The essential point seems to be the use of intersection, because without " $\cap$ " a system of equations of type ($\substack{*\\*}$) has CF languages as components of the minimal solution.

CONJECTURE 2. A language is in CS_λ iff it is a component of the minimal solution of a system ($\substack{*\\*}$) using only $\{ ., \cup, \cap \}$.

THEOREM 8. A language $L \subseteq V^*$ is recursive-enumerable iff is a component of the minimal solution of a system of equations in the form

$$\begin{cases} X_1 = F_1 \ (X_1,\dots,X_t) \\ \dots\dots\dots\dots\dots\dots \\ X_t = F_t \ (X_1,\dots,X_t) \end{cases}$$

where $F_1,\dots,F_t$ are regular expressions over the alphabet: $\{ ".", "\cup", "*", "h", "\cap" \} \cup \{),(\} \cup V \cup \{\wedge\}$ where $\wedge$ stands for the empty word λ .

REMARK 1. The result of the Theorem 8 can be extended to the case when instead of letters of the alphabet V we consider a finite set of recursive-enumerable languages over V.

3. SOME CONSIDERATIONS ABOUT FIXED-POINT SEMANTICS OF MONADIC PROGRAMS

We work in this section with programs in the formalism presented by J.A. Goguen in [1].

Speaking heuristically now, in this section we consider programs consisting of operation and tests, each performed directly on values stored in memory. These tests and operations will appear as (labels) of edges in a graph, with all of the partial functions representing the several alternatives of a test emanating from the same node. Thus a path in this graph represents an execution sequence for the instructions of the program. It should be noted that these flow diagram programs are not purely syntactic entities: a specific interpretation is assumed to be already given for each operation and test instruction.

One of the question of greatest interest for such a program is semantic: What function does it compute?

We give now the formal definitions.

A (directed) graph is a pair, $G = (V,E)$ where V is a finite set of nodes, E is a set of edges $E \subset V \times V$.

An exit node v' is a node with the property that there are no edges in G with source v'.

We denote by $\mathcal{N}$ the class of sets in the form N^r, $r \geqslant 0$, and $\mathcal{PFN}$ the class of partial functions between sets in $\mathcal{N}$.

A program is a pair (G,P) where $|P| : V \longrightarrow \mathcal{N}$,

$P : E \longrightarrow \mathcal{PFN}$ with the property that for every $(v_1,v_2) \in E$,

$P(v_1,v_2) : |P|(v_1) \rightarrow |P|(v_2)$

A program (G,P) is called deterministic if whenever e,e' are edges with same source node, the partial functions Pe, Pe' have disjoint sets of definition.

If we denote by $Pa(G) = \{(v,v') \mid$ there exists a path in G from v to $v'\}$ we can define the behavior of a program. We can extend the functions $P : E \rightarrow \mathcal{PFN}$ to $\hat{P} : Pa(G) \rightarrow \mathcal{PFN}$. In fact, if $(v_o,v_1,\ldots,v_t)$ is the sequence of nodes which describes a path in G from v_o to v_t we have

$$\hat{P}(v_o,\ldots,v_t) = P(v_o,v_1) \circ \ldots \circ P(v_{t-1},v_t).$$

Also we have the following result stated as Proposition 5 in [1]:
If (G,P) is a deterministic program and if f, f' are path in G with same source, such that neither is an initial segment for the other, then $P(f)$ and $P(f')$ have disjoint sets of definition.

DEFINITION. The behavior or complete partial function computed by the program (G,P) with entry at v and exit at v' is

$$\hat{P}(v,v') = \bigcup \{\hat{P}(f) \mid f \text{ a path from } v \text{ to } v' \text{ in } G\}.$$

It is easy to see that if (G,P) is deterministic and v' is an exit node, then P(v,v') is also a partial function (Corollary 6 [1]).

Let us consider RelN the class of relations over N. We use three symbols "a", "b", "#" in order to define the function S: Rel N $\longrightarrow$ $\mathcal{P}(a^+ \# b^+)$ given by $S(R) = \{a^{n+1} \# b^{m+1} \mid (n,m) \in R\}$. (Note that $\mathcal{P}(A)$ is the power-set of A).

For a partial function f : N $\longrightarrow$ N, if Dom f is the definition domain of f, we have

$$S(f) = \{a^{n+1} \# b^{f(n)+1} \mid n \in \mathrm{Dom} f\}$$

We notice that the language S(f) encodes the association realized by f.

Our intention is to work with such type of languages instead of functions, in the definition of monadic programs, i.e. programs which use only one-variable functions.

In fact, if (G,P) is a monadic deterministic program we can consider the diagram

$$E \xrightarrow{P} N \xrightarrow{S} \mathcal{P}(a^+ \# b^+)$$

We observe that the function S is bijective, and its reverse F: $\mathcal{P}(a^+ \# b^+) \longrightarrow$ Rel N can be interpreted as a "forgetful" operator, i.e. forgets the language encoding of relations over N.

If "o" stands for the relation composition, we have:

$$S(R_1 \circ R_2) = S(FS(R_1) \circ FS(R_2)).$$

The above equality defines an operator which beginning with two languages $S(R_1)$ and $S(R_2)$ gives a new languages $S(R_1 \circ R_2)$.

More formally, the operation can be expressed with classical operators.

Let be c, $\#_1$ new symbols, and the languages:

$L_1 = \{a^m \# b^n \mid (m-1,n-1) \in R_1\}$, $L_2 = \{b^k \#_1 c^s \mid (k-1,s-1) \in R_2\}$.

We consider the language $L_3 = L_1 \#_1 c^+ \cap a^+ \# L_2$.

We have:

$$L_3 = \{a^m \# b^n \#_1 c^t \mid (m-1, n-1) \in R_1, (n-1, t-1) \in R_2\}.$$

The homomorphism h, defined by $h(b) = h(\#_1) = \lambda$, $h(a)=a$, $h(c)=b$ maps L_3 into $S(R_1 \circ R_2)$, i.e.

$$h(L_3) = \{a^m \# b^n \mid (m-1, n-1) \in R_1 \circ R_2\} = S(R_1 \circ R_2)$$

Therefore, if h' is a new homomorphism given by $h'(a)=b$, $h'(\#)=\#_1$, $h'(b)=c$ we have the following representation

(I) $S(R_1 \circ R_2) = h(L_1 \#_1 c^+ \cap a^+ \# L_2)$
$= h(S(R_1) \#_1 c^+ \cap a^+ \# h'(S(R_2)))$

We denote by φ this new operator, i.e.

$$\varphi : \mathcal{P}(a^+ \# b^+) \times \mathcal{P}(a^+ \# b^+) \longrightarrow \mathcal{P}(a^+ \# b^+)$$

given by

$$\varphi(E_1, E_2) = S(F(E_1) \circ F(E_2))$$

The operator can be extended for any $t \geqslant 2$ to

$$\underbrace{\mathcal{P}(a^+ \# b^+) \times \dots \times \mathcal{P}(a^+ \ b^+)}_{t}$$

Suppose that we have already defined the operator for s; now the extension to s+1 is defined by

$$\varphi(E_1, \dots, E_{s+1}) = \varphi(\varphi(E_1, \dots, E_s), E_{s+1})$$

In the rest of this section we consider monadic deterministic programs with one memory location only.

The extension to monadic nondeterministic programs with a finite number of locations requires a little bit more complicated notational apparatus.

Let (G,P) be a monadic deterministic program with one location. If G = (V,E), for every $e \in E$, by the way of P and S we have associate a language, i.e.

$$P(e) : |P|(v_1) \longrightarrow |P|(v_2), \quad e = (v_1, v_2)$$

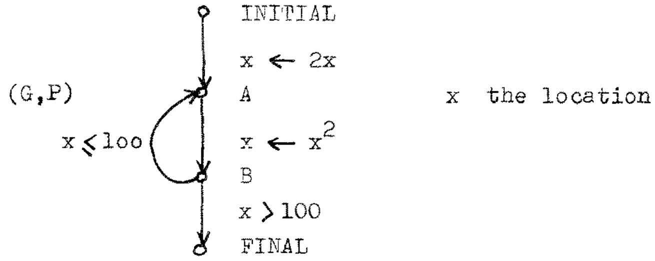
and $S(P(e)) \in \mathcal{P}(a^+ \# b^+)$.

To a path from Pa(G), say $\mu : (v_{i_1}, v_{i_2}, \dots, v_{i_k})$ we associate the language

$$S(\mu) = S(P(v_{i_1}, v_{i_2}) \circ P(v_{i_2}, v_{i_3}) \circ \dots \circ P(v_{i_{k-1}}, v_{i_k}))$$

$$= \varphi(P(v_{i_1}, v_{i_2}), \dots, P(v_{i_{k-1}}, v_{i_k}))$$

<u>EXAMPLE</u>



We have

$$S(x \leftarrow 2x) = \{a^{n+1} \# b^{2n+1} \mid n \geqslant o\}$$
$$S(x \leftarrow x^2) = \{a^{n+1} \# b^{n^2+1} \mid n \geqslant o\}$$
$$S(x \leqslant loo) = \{a^{n+1} \# b^{n+1} \mid n \leqslant loo\}$$
$$S(x > loo) = \{a^{n+1} \# b^{n+1} \mid n > loo\}$$

Let us consider the path μ: (A, B, A).
We have

$$S(\mu) = S((x \longleftarrow x^2) \circ (x \leqslant loo)) =$$
$$= S(F(\{a^{n+1} \# b^{n^2+1} \mid n \geqslant o\}) o \quad F(\{a^{n+1} \# b^{n+1} \mid n \leqslant loo\}))$$
$$= \{a^{n+1} \# b^{n^2+1} \mid n^2+1 \leqslant loo\}.$$

Now, for such a program we intend to construct a system of equations with variables in the power-set of a finite generated free monoid so that one of the components of its minimal solution gives its behavior as a function encoded with S.

Let be (G,P) a program with the location x, and G=(V,E). Suppose that v_I and v_F are the entry and the exit nodes.

If $V = \{v_I = v_o, v_1, \ldots, v_t = v_F\}$ then we associate a variable X_i (varying in $\mathcal{P}(a^+ \# b^+)$) to each node v_i, $o \leqslant i \leqslant t$.

For a node v_i, let be $(v_{j_1,1}, v_1), \ldots, (v_{j_{k(i)},i}, v_i)$ the collection of all edges in G which enter in v_i, and $f_1^{(i)}, \ldots, f_{k(i)}^{(i)}$ the corresponding partial functions associated by P.

For every i, $1 \leqslant i \leqslant t$ we consider the equation

$$X_i = \bigcup_{s=1}^{k(i)} S(F(X_{j_s,i}) \circ f_s^{(i)}) =$$
$$\bigcup_{s=i}^{k(i)} \varphi(X_{j_s,i}, S(f_s^{(i)}))$$

To the node $v_I = v_o$ we associate a constant equation

$$X_o = \{a^{n+1} \# b^{n+1} \mid n \geqslant o\}$$

Putting together, we obtain the system

$$(+) \qquad \begin{cases} X_o = \{a^{n+1} \# b^{n+1} \mid n \geqslant o\} \\ X_i = \bigcup_{s=1}^{k(i)} \varphi(X_{i_s,i}, S(f_s^{(i)})), \quad 1 \leqslant i \leqslant t \end{cases}$$

which plays a major role in the sequel.

Because of the representation of φ given in the formula (!), the equations of X_i, $1 \leqslant i \leqslant t$ have the form presented in the Theorem 9 with the addition of Remark 1.

So, at this moment, such a system has a minimal solution, with all components recursive - enumerable languages: $X^{MiN} = (X_o^{MiN},\dots,X_t^{MiN})$.

We intend to show the following

THEOREM 9.

$$S(\hat{P}(v_I,v_F)) = X_t^{MiN}$$

I.e., for every monadic deterministic program with one location, there exists a system of equations in the form (+) so that its semantics - in some encoded form - is a component of the minimal solution of the system.

PROOF. We have $\hat{P}(v_I,v_F) = \bigcup\{\hat{P}(\mu) \mid \mu$ path in G from v_I to $v_F\}$ and

$$S(\hat{P}(v_I,v_F)) = \bigcup\{S(\hat{P}(\mu)) \mid \mu \text{ path in G from } v_I \text{ to } v_F\}.$$

On the other side, $X_t^{MiN} = \bigcup_{n=o}^{\infty} X_t^{(n)}$, where $X_t^{(n+1)} = F_t(X_o^{(n)},\dots$ $\dots X_t^{(n)})$ and

$$F_t(X_o,\dots,X_t) = \bigcup_{s=1}^{k(i)} \varphi\,(X_{j_s,i}, S(f_s^{(i)}))$$

We intend to show that for every i and p, with $1 \leqslant i \leqslant t$, $p \geqslant 1$ we have

(A) $X_i^{(p)} = \bigcup\{S(\hat{P}(\mu)) \mid \mu$ path in G of length p from v_I to $v_i\}$.

We denote by Path $(v_i,v_j;m)$ the set of all paths of length m in G from v_i to v_j, and by Path $(v_i, v_j;-)$ the set of all path in G from v_i to v_j.

For p=o, $X_i^{(p)} = \emptyset$, $1 \leqslant i \leqslant t$.

We take first p=1. If $a^m \# b^n \in X_i^{(1)}$, we have for

$$X_i^{(1)} = \bigcup_{s=1}^{k(i)} \varphi(X_{j_s,i}^{(o)}, S(f_s^{(i)}))$$

a number r, so that $X_{j_r,i}^{(o)} = X_o^{(o)}$ and $a^m \# b^n \in S(f_r^{(i)})$.

Hence $(v_{j_r,i},v_i)$ is the edge (v_I,v_i), and it follows that $a^m \# b^n$ $S(\hat{P}(v_I,v_i))$ and so the inclusion $X_i^{(1)} \subset \bigcup\{S(\hat{P}(\mu)) \mid \mu \in \text{Path}(v_I,v_i;1)\}$ holds.

Conversely, $S(P(v_I,v_i)) = S(F(X_o^{(o)})o\ F(S(P(v_I,v_i)))) = \varphi\,(X_o^{(o)},$ $S(P(v_I,v_i))) \subset X_i^{(1)}$, because $(v_I,v_i) \in E$ implies that in the equation of X_i there exists a r so that $X_{j_r,1} = X_o$.

Now it is manifest that (A) holds for p=1. Suppose that it is true for $p \leq q$. Then we have

$$X_i^{(q+1)} = \bigcup_{s=1}^{k(i)} S(F(X_{j_s,i}^{(q)} \circ f_s^{(i)}) =$$

$$= \bigcup_{s=1}^{k(i)} S(F[\bigcup \{S(\hat{P}(\mu)) | \mu \in Path(v_I, v_{j_s,i}; q)\}] \circ f_s^{(i)})$$

$$= \bigcup_{s=1}^{k(i)} S(\bigcup \{FS(\hat{P}(\mu)) | \mu \in Path(v_I, v_{j_s,i}; q)\} \circ f_s^{(i)})$$

$$= \bigcup_{s=1}^{k(i)} S(\bigcup \{\hat{P}(\mu) | \mu \in Path(v_I, v_{j_s,i}; q)\} \circ f_s^{(i)}$$

$$= \bigcup_{s=1}^{k(i)} S(\bigcup \{\hat{P}(\mu) \circ f_s^{(i)} | \mu \in Path(v_I, v_{j_s,i}; q)\})$$

$$= \bigcup_{s=1}^{k(i)} S(\bigcup \{\hat{P}(\mu') | \mu' \in Path(v_I, v_i : q+1)\})$$

$$= \bigcup_{s=1}^{k(i)} (\bigcup S(\hat{P}(\mu')) | \mu' \in Path(v_I, v_i, q+1)\}).$$

Because of the simple observation that

$$\hat{P}(v_I, v_F) = \bigcup \{\hat{P}(\mu) | \mu \in Path(v_I, v_F : -)\}$$

$$= \bigcup_{m=0}^{\infty} \{\hat{P}(\mu) | \mu \in Path\ (v_I, v_F; m)\}$$

it follows that $S(\hat{P}(v_I, v_F)) = X_t^{MiN}$.

REFERENCES

1. J.A.G o g u e n - "On Homomorphism, Correctness, Termination, Unfoldments and Equivalence of Flow Diagram Programs". Journal of Comp.System Sci., vol.8, nr.3 (1974).
2. G.T.H e r m a n , G.R o z e n b e r g - "Developmental Systems and Languages" North-Holland (1975).
3. S.I s t r a i l - "Context-sensitive Languages: Recursivity, Fixed-point theorems and applications to program semantics and number theory".Ph.D.Thesis,Univ. Bucharest, March 1979.
4. S.I s t r a i l - "A fixed-point approach of context-sensitive languages using context-free grammars with choice" (submitted for publication).

5. S.I s t r a i l - "On the weak equivalence problem of SICK-OL systems with some generative devices". Annalles Univ.Iaşi, T.XXIII, S.I.a, f.2 (1977).

6. S.I s t r a i l - "SICK OL systems and simulating ability" (submitted for publication).

7. A.S a l o m a a - "Formal Languages" Academic Press, New York and London (1973).