FEEDBACK THEORIES (A CALCULUS FOR ISOMORPHISM CLASSES OF FLOWCHART SCHEMES)*

GH. ŞTEFĂNESCU

A simple representation of multi-entry/multi-exit flowchart schemes is given. This shows that the basic operations on flowchart schemes are : separated sum, composition with 'empty flowcharts,' and feedback. The main technical point is giving a calculus for isomorphism classes of flowchart schemes. This calculus is similar to that of polynomials and may be considered as a framework of our calculi for deterministic and nondeterministic flowchart schemes presented in [5].

1. INTRODUCTION

The reasons for the present note is twofold (a) we are trying to prove that feedback is more natural than iteration; (b) we give a calculus for isomorphism classes of flowchart schemes. This paper may be seen as a natural extension of the last paper of Elgot [4]. The characteristic feature of [4] is the attempt to weaken the 'algebraic theory' structure (in the sense of Lawvere), widely used in semantics of flowchart algorithms.

(a) The feedback is 'scalar' and all usual flowchart schemes can be built up from atomic flowchart schemes and trivial ones (which may be thought of as redirecting flow of control) by means of sum, composition and feedback. This is no longer true for scalar iteration (cf. [4], only flowcharts fulfiling 'for every closed path C there is a vertex v_C of C such that every begin path to a vertex of C meets v_C' can be obtained).

(b) There is some interest in axiomatization of isomorphism classes of flowchart schemes [2, 3, 4]. Our calculus extends these; its characteristic features are

— the operations on flowcharts are defined by simple formulae rather by some 'verbal descriptions' as in [2, 4];

— the restriction to sum and composition of our algebraic structure is more general than 'algebraic theories' used in [2, 3] and essentially corresponds to flow theories in [4], but instead of surjective functions we need only bijective ones;

— our calculus work in a more general (and useful) case, e.g. instead of trivial flowchart schemes we can use arbitrary known flowchart algorithms in which a change of memory state may accompany redirecting flow of control.

* This paper has been circulated as Preprint Series in Mathematics, No. 24/1986, INCREST, Bucharest, April 1986.

2. ON THE CHOICE OF OPERATIONS

Every multi-entry/multi-exit flowchart scheme can be ordered as it is shown in Fig. 1.

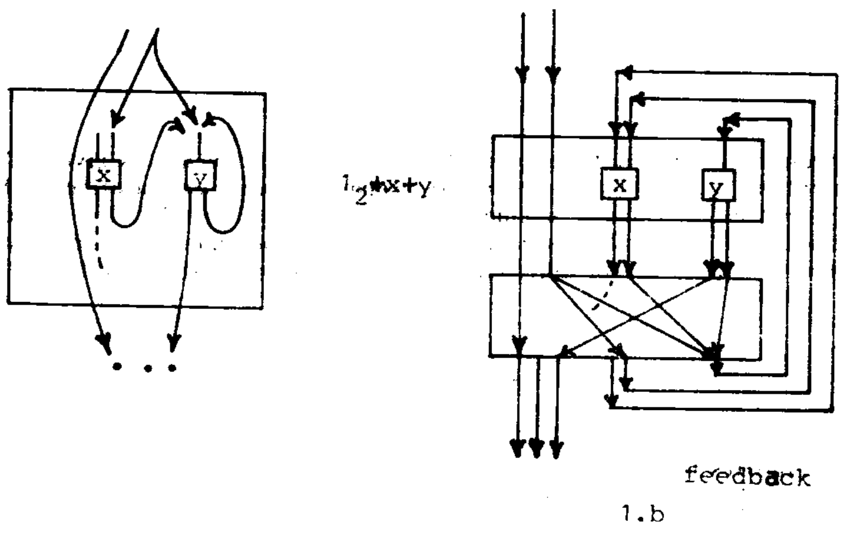

Fig. 1. — The standard form of a flowchart scheme.

This shows that the basic operations on flowchart schemes are : *separated sum* (or *parallel composition*) $+$, *composition* (or *serial composition*) and *feedback* $\uparrow$; these have the intuitive meaning given in Fig. 2.

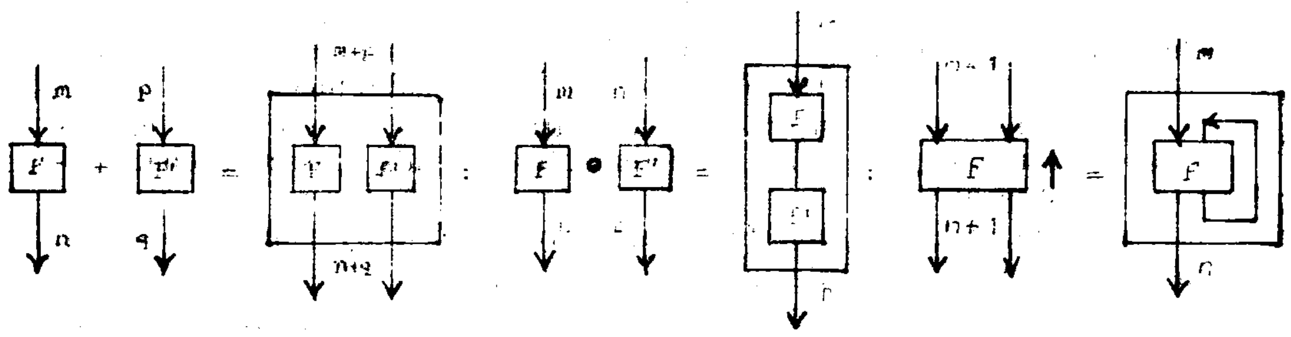

Fig. 2. — " $+$ " denotes separated sum ; " $\cdot$ " denotes composition ; " $\uparrow$ " denotes feedback.

In the sequal $\uparrow^k$ denotes k-times application of $\uparrow$. By Fig. 1, composition can be restricted to composition with 'empty flowcharts', that is flowchart schemes without internal vertices. Generally, we define the composition as shown in Fig. 3.

3. THE CATEGORIES Fn AND Bi

The category **Fn** has the set of natural numbers $\mathbb{N}$ as its class of objects. The set of morphisms of **Fn** with source n and target p is the set

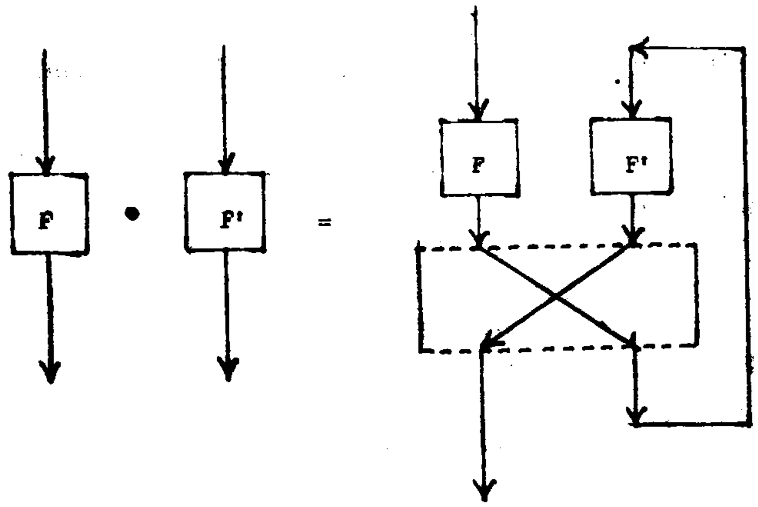

Fig. 3. — Composition in terms of sum, feedback and "scalar" composition.

of all functions $f : [n] = \{1, \ldots, n\} \to [p]$. If $x \in [n]$ we write xf for the value of f applied to x and if $g : [p] \to [q]$ is a function, we write $f \circ g : [n] \to \to [q]$ for the composite.

Given a pair of functions $f_i : [n_i] \to [p_i]$, $i \in [2]$ we define the function $f_1 + f_2 : [n_1 + n_2] \to [p_1 + p_2]$ as

$$x(f_1 + f_2) = \text{'if } x \in [n_1] \text{ then } xf_1 \text{ else } (x - n_1)f_2 + p_1\text{'}$$
$$\text{for } x \in [n_1 + n_2].$$

The class of bijective functions in **Fn** is closed under $\circ$ and contains the identities, hence it gives a subcategory **Bi** of **Fn**. Moreover, **Bi** is closed under $+$. We write $m \leftrightarrow n$ for the *block permutation function* $m \leftrightarrow n$: $[m + n] \to [n + m]$ given by

$$x(m \leftrightarrow n) = \text{'if } x \in [m] \text{ then } n + x \text{ else } x - m\text{' for } x \in [m + n].$$

Given a bijective function $f : [n + 1] \to [p + 1]$ we define the bijective function $f\uparrow : [n] \to [p]$ by

$$xf\uparrow = \text{'if } xf \neq p + 1 \text{ then } xf \text{ else } (n + 1)f\text{' for } x \in [n].$$

4. FEEDBACK THEORIES

Our basic algebraic structure is defined as follows.

4.1. A *biflow* $(T, +, \circ, \uparrow)$ is an extension of $(\mathbf{Bi}, +, \circ, \uparrow ; I_m, m \leftrightarrow n)$ such that

(4.1.1) $(T, +, I_0)$ is a monoid;

(4.1.2) Block permutation axiom : for $f_i \in T(m_i, n_i)$, $i \in [2]$

$$(f_1 + f_2) \circ (n_1 \leftrightarrow n_2) = (m_1 \leftrightarrow m_2) \circ (f_2 + f_1);$$

(4.1.3) $(T, \circ, I_m)$ is a category, having the same objects as **Bi**;
(4.1.4) Composition and sum are related by: for $f_i \in T(m_i, n_i)$, $g_i \in T(n_i, p_i)$, $i \in [2]$

$$(f_1 + f_2) \circ (g_1 + g_2) = (f_1 \circ g_1) + (f_2 \circ g_2);$$

(4.1.5) Feedback is context free
 (4.1.5.1) $f + g\uparrow^p = (f + g)\uparrow^p$;
 (4.1.5.2) $f\uparrow^p + g = ((I_m + m' \leftrightarrow p) \circ (f + g) \circ (I_n + p \leftrightarrow n'))\uparrow^p$ for $f \in T(m + p, n + p)$, $g \in T(m', n')$;
 (4.1.5.3) $f\uparrow^p \circ g = (f \circ (g + I_p))\uparrow^p$;
 (4.1.5.4) $f \circ g\uparrow^p = ((f + I_p) \circ g)\uparrow^p$;
(4.1.6) Shifting blocks on feedback: for $f \in T(m + p, n + q)$, $g \in T(q, p)$

$$(f \circ (I_n + g))\uparrow^p = ((I_m + g) \circ f)\uparrow^q.$$

4.2. *Remark.* (i) According to [4], this should be called a (scalar) feedback flow theory over **Bi**.

(ii) The axioms are not independent. In fact, (4.1.5.2) follows from (4.1.5.1) using (4.1.2), (4.1.5.3), (4.1.5.4).

(iii) For $f \in T(m, n)$, $g \in T(n, p)$ we have
(4.2.1) $f \circ g = (f \circ (n \leftrightarrow n)\uparrow^n \circ g =)\ ((f + g) \circ (n \leftrightarrow p))\uparrow^n$.

4.3. *Examples.* **Bi** with the operations defined in **3** is a biflow. All iteration theories cf. [1, 2], strong iteration theories cf. [5] and theories with iterate cf. [3], naturally are biflows (as feedback we take: $(1_m + 0_p)(f(1_n + 0_m + 1_p))^\dagger$, for $f \in T(m + p, n + p)$ — † is the right iteration).

4.4. In practice, it is useful to have a simpler characterization of this algebraic structure. Such a simplification can be obtained using only composition with morphisms in **Bi** and taking (4.2.1) as a definition for general composition. More precisely, if T endowed with sum, left and right composition with morphisms in **Bi**, and feedback, extends **Bi** and fulfils:

—(4.1.1), (4.1.2), (4.1.5.1);
— $(T, \circ, I_m)$ is a bimodule over **Bi**, i.e.
 $f \circ I_n = I_m \cdot f = f$, for $f \in T(m, n)$, and
 $f \circ (g \circ h) = (f \circ g) \circ h$, whenever two morphisms are in **Bi**;
— (4.1.4), (4.1.5.3), (4.1.5.4), (4.1.6) whenever the g morphisms are in **Bi**; (4.1.4), (4.1.5.3), (4.1.5.4) whenever the f morphisms are in **Bi**; then T is a biflow (composition being extended using (4.2.1)).

4.5. *Definition.* The category **BFl** has as objects biflows and as morphisms functors which preserve morphisms in **Bi**, sum and feedback.

5. ABSTRACT THEORIES OF FLOWCHART SCHEMES

Such a theory is given by
— a double indexed set X of variables for atomic flowchart schemes (that is, every $x \in X$ has a number of entries $\cdot x$ and a number of exits $.x$ — another way to specify this is $x \in X\ (\cdot x,\ .x)$);

— a 'support theory' T consisting of a family of sets $T(m, n)$ $m, n \in \mathbb{N}$ (an element $c \in T(m, n)$ is considered as a known computation process with m entries and n exits).

Note. The type of T corresponds to the type of flowchart schemes we consider. While in the case of deterministic flowcharts the basic support theory is **Pfn** given by $\mathbf{Pfn}(m, n)$ = 'the set of all partial functions from $[m]$ to $[n]$', in the nondeterministic case this is **Rel** given by $\mathbf{Rel}(m, n) =$ = 'the set of all relations included in $[m] \times [n]$'. Actually, Elgot & Shepherdson [4] use **Sur**, the subtheory of all surjective functions in **Pfn.** In this paper we use **Bi**. All these theories model only redirecting flow of control. Note that more complicated theories can also be used, e.g. $\mathbf{Pfn}_D$ given by $\mathbf{Pfn}_D(m, n)$ = 'the set of all partial functions from $D \times [m]$ to $D \times [n]$' in which a change of memory state $d \in D$ may accompany redirecting flow of control. ■

A flowchart scheme is abstracted to an *X-flownomial over T* defined as an expression

$$((I_m + \Sigma x_i) \circ c)\uparrow^{\Sigma^{\cdot} x_i}$$

where $x_i \in X$, the sum is finite and $c \in T(m + \Sigma.x_i,\ n + \Sigma^{\cdot}x_i)$; denote by $\mathbf{Fl}_{X,T}(m, n)$ their set.

The *interpretation* of an X-flownomial over T in a structure Q in which I_m, sum, composition and feedback have sense, is specified by a rank-preserving function $\varphi_X : X \to Q$ (i.e. $\varphi_X(x) \in Q(^{\cdot}x, .x)$) and a 'morphism' $\varphi_T : T \to Q$ (i.e. it preserves I_m and operations); it is

$$\varphi^{\#}(((I_m + \Sigma x_i) \circ c)\uparrow^{\Sigma^{\cdot} x_i}) = ((I_m + \Sigma\varphi_X(x_i)) \circ \varphi_T(c))\uparrow^{\Sigma^{\cdot} x_i}.$$

T and X can naturally be embedded in $\mathbf{Fl}_{X,T}$ as follows

$$c = ((I_m) \circ c)\uparrow^0 \text{ and } x = ((I_{\cdot x} + x) \circ (^{\cdot}x \leftrightarrow .x))\uparrow^{\cdot x}$$

(the latter follows by (4.2.1)).

In the sequel we shall frequently write $\mathbf{x}, {}^{\cdot}\mathbf{x}, .\mathbf{x}$ instead of Σx_i, $\Sigma^{\cdot}x_i$ $\Sigma.x_i$, respectively.

6. OPERATIONS ON FLOWNOMIALS

Sum, composition and feedback in T extend themselves to X-flownomials over T

(i) for $F = ((I_m + \mathbf{x}) \circ c)\uparrow^{\cdot\mathbf{x}} : m \to n$ and $F' = ((I_{m'} + \mathbf{x}') \circ c')\uparrow^{\cdot\mathbf{x}'} : m' \to n'$ we define $F + F' : m + m' \to n + n'$ as

$$F + F' = ((I_{m+m'} + \mathbf{x} + \mathbf{x}') \circ (I_m + m' \leftrightarrow .\mathbf{x} + I_{.\mathbf{x}'}) \circ (c + c') \circ$$
$$\circ (I_n + {}^{\cdot}\mathbf{x} \leftrightarrow n' + I_{\cdot\mathbf{x}'}))\uparrow^{\cdot\mathbf{x} + \cdot\mathbf{x}'};$$

(ii) for $F = ((I_m + \mathbf{x}) \circ c) \uparrow^{\cdot \mathbf{x}} : m \to n$, $f \in T(p, m)$, **and** $g \in T(n, q)$ we **define**

$$F \circ g = ((I_m + \mathbf{x}) \circ c \circ (g + I_{\cdot \mathbf{x}}))\uparrow^{\cdot \mathbf{x}}, \text{ and}$$

$$f \circ F = ((I_p + \mathbf{x}) \circ (f + I_{. \mathbf{x}}) \circ c)\uparrow^{\cdot \mathbf{x}};$$

(iii) for $F = ((I_{m+1} + \mathbf{x}) \circ c)\uparrow^{\cdot \mathbf{x}} : m + 1 \to n + 1$ we define $F\uparrow : m \to n$ as

$$F\uparrow = ((I_m + \mathbf{x}) \circ ((I_m + .\mathbf{x} \leftrightarrow 1) \circ c \circ (I_n + 1 \leftrightarrow \cdot \mathbf{x}))\uparrow)\uparrow^{\cdot \mathbf{x}}.$$

Suppose T is a biflow. From these basic operations we derive the general feedback and composition, namely for $F = ((I_{m+k} + \mathbf{x}) \circ c) \uparrow^{\cdot \mathbf{x}} : m + k \to n + k$ the flownomial $F\uparrow^k : m \to n$ is

$$F\uparrow^k = ((I_m + \mathbf{x}) \circ ((I_m + .\mathbf{x} \leftrightarrow k) \circ c \circ (I_n + k \leftrightarrow \cdot \mathbf{x}))\uparrow^k)^{\cdot \mathbf{x}}$$

and for $F = ((I_m + \mathbf{x}) \circ c)\uparrow^{\cdot \mathbf{x}} : m \to n$, $F' = ((I_n + \mathbf{x}') \circ c')\uparrow^{\cdot \mathbf{x}'} : n \to p$ **the** flownomial $F \circ F' : m \to p$ is

$$F \circ F' = ((I_m + \mathbf{x} + \mathbf{x}') \circ (c + I_{.\mathbf{x}'}) \circ (I_n + \cdot\mathbf{x} \leftrightarrow .\mathbf{x}') \circ (c' + I_{\cdot \mathbf{x}}) \circ$$

$$\circ (I_p + \cdot\mathbf{x}' \leftrightarrow \cdot\mathbf{x}))\uparrow^{\cdot \mathbf{x} + \cdot \mathbf{x}'}.$$

Remark that all these formulae are rules of computation in a biflow, namely their instances obtained by replacing $\mathbf{x}s$ with elements in T are identities in T. This gives a half of the main theorem, i.e.

UNIQUE EXTENSION LEMMA. *For every morphism* $\varphi_T : T \to Q$ *in* **BFl** *and every rank-preserving function* $\varphi_X : X \to Q$ *the extension* $\varphi^{\#} : \mathbf{Fl}_{X,T} \to Q$ *preserves the operations. Moreover, this is the unique extension of* (φ_X, φ_T) *with respect to this property.* ■

$$\begin{tikzcd} T \arrow[r, hook] \arrow[dr, "\varphi_T"'] & \mathbf{Fl}_{X,T} \arrow[d, dashed, "\varphi^{\#}"] & X \arrow[l, hook'] \arrow[dl, "\varphi_X"] \\ & Q & \end{tikzcd}$$

7. ISOMORPHIC FLOWNOMIALS

Given $(x_1, \ldots, x_k)$ and $(x'_1, \ldots, x'_{k'})$ a function $y : [k] \to [k']$ such that $x_i = x'_{iy}$, $\forall\, i \in [k]$ has a unique 'block extension' to entries $\cdot y : [\cdot x_1 + + \ldots + \cdot x_k] \to [\cdot x'_1 + \ldots + \cdot x'_{k'}]$ and a unique block extension to exits $.y : [.x_1 + \ldots + .x_k) \to [.x'_1 + \ldots + .x'_{k'}]$ (see [4] for more details).

We say two flownomials $F = ((I_m + x_1 + \ldots + x_k) \circ c)\uparrow^{\Sigma \cdot x_i} : m \to n$ and $F' = ((I_m + x'_1 + \ldots + x'_k) \circ c')\uparrow^{\Sigma \cdot x'_i} : m \to n$ are *isomorphic* if there is a bijection $y : [k] \to [k]$ such that

(i) $x_i = x'_{iy}$, $\forall\, i \in [k]$;

(ii) $c \circ (I_n + \cdot y) = (I_m + .y) \circ c'$.

The isomorphism relation $\approx$ is a congurence relation, hence the operations are well defined in the quotient structure $\mathbf{Fl}_{X,T}/\approx$. On the other hand, two isomorphic flownomials have the same interpretation in a biflow, hence the interpretation $\varphi^{\#} : \mathbf{Fl}_{X,T} \to Q$ induces one $\varphi^{\wedge} : \mathbf{Fl}_{X,T}/\approx \to Q$ on isomorphism classes of flownomials.

8. THE ALGEBRAIC STRUCTURE OF $\mathbf{Fl}_{X,T}/\approx$

Suppose T is a biflow. A simple computation shows that $(\mathbf{Fl}_{X,T}, +, I_0)$ is a monoid, $(\mathbf{Fl}_{X,T}, \circ, I_m)$ is a bimodule over $\mathbf{Bi}$ and (4.1.5.1) holds in $\mathbf{Fl}_{X,T}$. The identities (4.1.4) (4.1.5.3), (4.1.5.4), and (4.1.6) hold, whenever the gs morphisms are in T. Moreover, (4.1.4), (4.1.5.3), (4.1.5.4) hold whenever fs morphisms are in $\mathbf{Bi}$. In addition, the two sides in (4.1.2) give isomorphic flownomials. By 4.4 these give the other half of the main result, i.e.

STRUCTURE PRESERVING LEMMA. *If T is a biflow, then $\mathbf{Fl}_{X,T}/\approx$ is a biflow.* ■

9. THE MAIN RESULT

This shows why we have asserted that this calculus is similar to that of polynomials. It follows from the above lemmas and the last sentence in **7**.

THEOREM. *If T is a biflow, then $\mathbf{Fl}_{X,T}/\approx$ is the coproduct of T and the biflow freely generated by X in* **BFl**. ■

$$\begin{tikzcd}
T \arrow[r, hook] \arrow[dr, "\varphi_T"'] & \mathbf{Fl}_{X,T}/\approx \arrow[d, dashed, "\varphi^\wedge"] & X \arrow[l, hook'] \arrow[dl, "\varphi_X"] \\
& Q &
\end{tikzcd}$$

This means that $\mathbf{Fl}_{X,T}$ is the biflow freely generated by adding X to T.

10. EXTENSIONS

The class of biflows T which extends $\mathbf{Fn}$ and fulfils:

(10.1) $0_m \circ f = 0_n$, for $f \in T(m, n)$

(10.2) $(m \vee m) \circ f = (f + f) \circ (n \vee n)$, for $f \in T(m, n)$

where 0_m is the unique function in $\mathbf{Fn}(0, m)$ and $m \vee m$ is the function in $\mathbf{Fn}(m + m, m)$ given by $x(m \vee m) =$ 'if $x \in [m]$ then x else $x - m$' for $x \in [m + m]$

equals the class of algebraic theories with iterate in [3].

Received October 18, 1988

INCREST, Dept. of Math.
Bucharest, Romania

REFERENCES

1. S. L. Bloom, C. C. Elgot and J. B. Wright, *Vector iteration in pointed iterative theories*. SIAM J. Comput. **9** (1980), 525–540.
2. S. L. Bloom and Z. Esik, *Axiomatizing schemes and their behaviours*. J. Comput. System Sci. **31** (1985), 375–393.
3. V. E. Căzănescu and Ş. Grama, *On definition of M- flowcharts*. INCREST Preprint Series in Mathematics, No. 56/1984.
4. C. C. Elgot and J. C. Shepherdson, *An equational axiomatization of the algebra of reducible flowchart schemes*. IBM Research Report RC-8221, April, 1980.
5. Gh. Ştefănescu, *An algebraic theory of flowchart schemes*. Proceedings CAAP' 86, pp. 60–73, Lecture Notes in Computer Science Vol. 214, Springer, Berlin, 1986.