ON FLOWCHART THEORIES I

by Gh. Ştefănescu

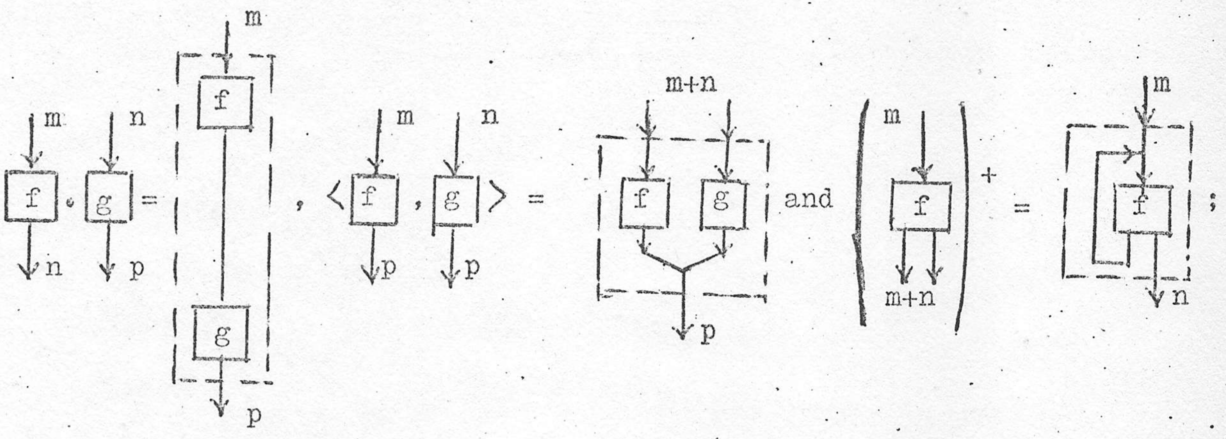
to the memory of C. C. ELGOT

Abstract. We give a calculus for the classes of deterministic flowchart schemes with respect to the strong equivalence relation, similar to the calculus of the classes of polynomials with respect to the reduction of similar terms.

1. Introduction

A. This paper is an attempt to develop Backus' ideas [6,7], namely to construct a mathematical theory in which program transformations are allowed, and in which correctness can be obtained using simple algebraic computations.

As program representation we use flowchart schemes. Many of their useful properties are known from [24, 27, 32], but with the above aim in mind we prefer the algebraic viewpoint of Elgot [17, 18]. It is well known that the operations of "structured programming" are not enough for representing all schemes [25, 26], essentially due to their one-input/one-exit feature. The basic Elgot's idea in [18] is to use many-input/many-exit schemes with composition, tupling and iteration as basic operations. Pictorially, these operations are



their intended interpretation at the semantical level will be given in Subsection B. Every scheme can be obtained from the atomic schemes using these operations.

We allow the program transformations which respect the strong equivalence relation in [18].

The main point of this paper is to give a calculus for the classes of deterministic flowchart schemes with respect to the strong equivalence relation, essentially similar to the calculus of the classes of polynomials with respect to the reduction of similar terms.

B. For the begining we fix the standard model for the interpretation of flowchart schemes. This model, proposed in [17] by Elgot, consists in the following. Fix a set D representing the set of all value-vectors for the registers in a computing device. A program scheme $F = \overset{\downarrow m}{\underset{\downarrow n}{\square}}$ is interpreted as a partial function $f : [m]\times D \dashrightarrow [n]\times D$ (in this paper $[m]$ denotes the set $\{1,\ldots,m\}$) with the meaning that if program execution begins at line $k\in[m]$ of the program with initial value-vector d and if $f(k,d) = (j,d')$, then d' is the new value-vector when program halts at line $j\in[n]$.

The set of all partial functions from $[m]\times D$ to $[n]\times D$ will be denoted by $Pfn_D(m,n)$; when D has exactly one element we shall write $Pfn_\bullet$ instead of Pfn_D.

The intended interpretation of the above informal operations is the following.

Composition: for $f\in Pfn_D(m,n)$, $g\in Pfn_D(n,p)$ we define $f\cdot_D g\in Pfn_D(m,p)$ as

$$(f\cdot_D g)(j,d) = g(f(j,d)) \ , \ \text{for } (j,d)\in[m]\times D.$$

Tupling: for $f\in Pfn_D(m,p)$, $g\in Pfn_D(n,p)$ we define $\langle f,g\rangle_D\in Pfn_D(m+n,p)$ as

$$\langle f,g\rangle_D(j,d) = \text{"if } j\in[m] \text{ then } f(j,d) \text{ else } g(j-m,d)\text{"} \ , \ \text{for } (j,d)\in[m+n]\times D.$$

Iteration: for $f\in Pfn_D(m,m+n)$ we define $f^{+D}\in Pfn_D(m,n)$ as

$$f^{+D}(j,d) = \begin{cases} (j_r-m,d_r), & \text{where } (j_r,d_r) \text{ is the last defined value (i.e. } j_r>m) \text{ in the} \\ & \text{sequence: } (j_0,d_0)=(j,d) \text{ and for } k\geqslant 0,\ (j_{k+1},d_{k+1}) = f(j_k,d_k); \\ \text{undefined, if such an } r \text{ does not exists,} & \end{cases}$$

for $(j,d)\in[m]\times D$.

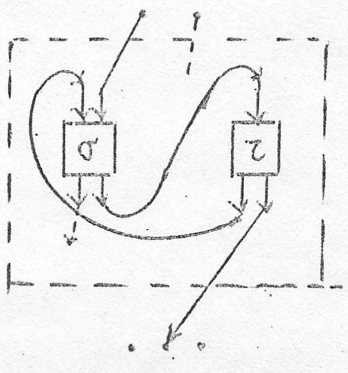
Our basic algebraic structure of strong iteration theories in Section 1 is obtained having this example in mind.

C. We come back now to the syntactical level in order to obtain a rigurous definition for flowchart schemes. The usual programs, written as flowchart schemes, allow two consecutive generalizations. The final result is partially similar with Petri nets [28] (where two kinds of elements appear: transitions and places), and differs from all representations of flowchart schemes listed in [4,ch.4].

The first generalization, appeared in 1970-1975 in the context of the study of the strong behaviour of program schemes [18, 24, 27], consist in replacing the concrete statements from vertices with symbols. More exactly, consider a set Σ of double-ranked variables, i.e. every $\sigma \in \Sigma$ has a number σ_{in} of inputs and another number σ_{out} of outputs -another writing is $\sigma \in \Sigma(\sigma_{in}, \sigma_{out})$ -; similarly, for a sequence e in the free monoid Σ^*, e_{in} (respectively, e_{out}) denotes the sum of inputs (respectively, the sum of outputs) of the letters of e.

The second generalization (which - as far as I know - appears only in [14]) considers an abstract "theory" T, whose morphisms $T(m,n)$, $m,n \geqslant 0$ are used for connections between vertices in a flowchart schemes. [For the usual flowchart schemes this theory is the theory of partial functions **Pfn**, and the morphisms can be seen as deterministic transsmisions of control between vertices, without change of memory.]

The usual flowchart scheme in Figure 1.a can be ordered as in Figure 1.b and can be briefly represented as in Figure 1.c (on the basis of connections).

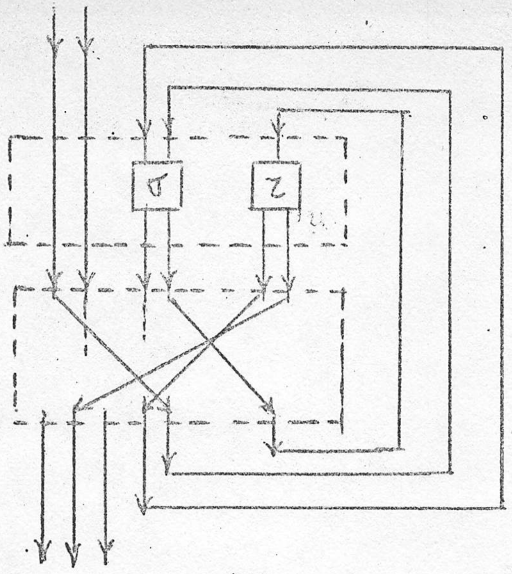


(a)

$$\begin{array}{cc} & 3 \\ 2 & \left[\begin{array}{ccc|ccc} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$$

with rows 3–4 marked $\sigma\{$ and rows 5–6 marked $\tau\{$.

(c)



(b)

Figure 1.

In conclusion, a program scheme $\begin{array}{c}\downarrow m\\ \boxed{}\\ \downarrow n\end{array}$ is abstracted to a <u>Σ-flowchart over T</u> with m inputs and n outputs, which is defined as a pair $F = (l,e)$, where

$e \in \Sigma^*$ is a sequence of its vertices, and

$l \in T(m+e_{out}, n+e_{in})$ gives the connections.

Denote this set by $Fl_{\Sigma,T}(m,n)$. A <u>flowchart theory</u> is $Fl_{\Sigma,T}$, for some Σ and T.

This idea of using an abstract theory T for the transmision of data between vertices in flowchart schemes is very important, even if the problem of the type of T is not completly solved. <u>In this paper we pled for the use of strong iteration theories as support for flowchart theories.</u>

The next step is to give a precise meaning to the above informal operations on flowchart schemes. Since every morphism in T can be seen as an "empty flowchart," composition, tupling and iteration must be defined in T. In Section 2 we shall see how these operations can be naturally extended from T to arbitrary flowcharts.

It remains to clarify the interpretation of a flowchart in a concrete computation theory. This is given by the Elgot formula (see the begining of Section 6).

D. Three advances of the present paper are worth mentioning here: .

(1) The introduction - in a constructive way - of a natural equivalence relation $\equiv$ on flowchatrs in $\mathrm{Fl}_{\Sigma,T}$. [Since in the case of usual flowcharts (i.e. $\mathrm{Fl}_{\Sigma,\mathrm{Pfn}}$) $\equiv$ is the strong equivalence relation in [18], we equally call $\equiv$ the strong equivalence relation.]

(2) The use of strong iteration theories as a necessary and sufficient semantical model for support theories in order to $\equiv$-equivalent flowcharts have the same interpretation.

(3) Our main technical result says that the classes of $\equiv$-equivalent Σ-flowcharts over a strong iteration theory T is the strong iteration theory freely generated by adding Σ to T.

Let us make some comments on these.

(1) It is well known the necessity of studying some equivalence relations on programs [18, 27, 35]. Since it is undecidible when two programs are functionally equivalent (i.e. when they define the same input-output partial function) for extremly poor classes of nonmonadic programs [27], we must restrict ourselfs to the strong equivalence relation (i.e. two programs are strongly equivalent if for each input they make the same steps in execution). It is known from Elgot's papers (for example [$\overset{16}{9}$]) that the classes of strongly equivalent flowchart schemes form an (iterative) algebraic theory, but this result is not constructive. We need a good flowchart representation and simple computing rules to check directly the algebraic structure of the classes of strongly equivalent flowchart schemes.

Beside the flowchart representation in Subsection C it is imperative to use an equivalence relation containing the following two basic relations (the pictures from [30] are also useful).

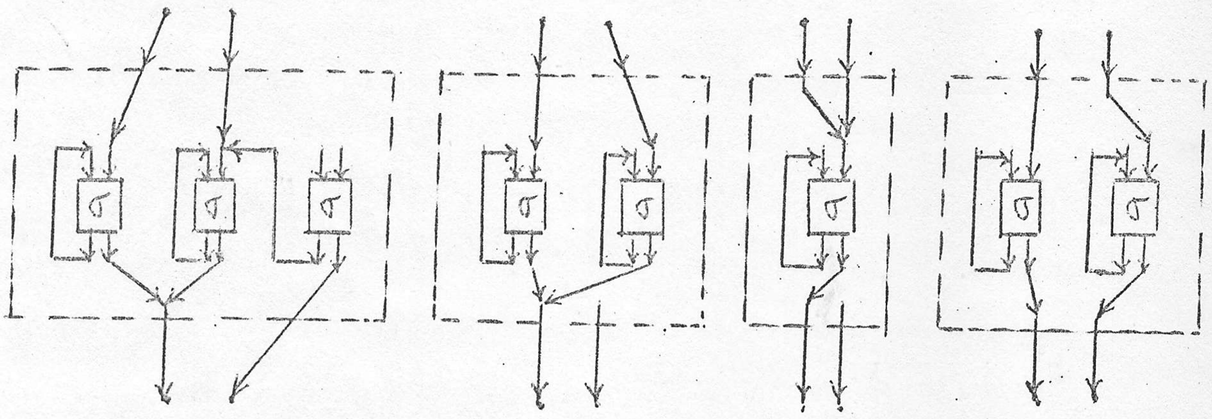
(i) $F \xrightarrow{i} F'$ iff F' can be obtained from F by adding some unaccessible vertices.

Example. $F2 \xrightarrow{i} F1$, with $F1$ and $F2$ from Figure 2. □

(s) $F \xrightarrow{s} F'$ iff F' can be obtained from F by identifying some vertices with the same label and whose output connections are equal after identification (or F can be obtained from F' by partially unfolding some vertices).

Example. $F2 \xrightarrow{s} F3$, with $F2$ and $F3$ from Figure 2. □

Nonexample. There is no F with only one σ-vertex such that $F4 \xrightarrow{s} F$; particularly, $F4 \not\xrightarrow{s} F3$ ($F4$ and $F3$ are those from Figure 2). □



(F1) (F2) (F3) (F4)

Figure 2.

Our equivalence relation $\equiv$ is the congruence relation generated by $\xrightarrow{i} \cup \xrightarrow{s}$. Theorem 4.1 gives a good and constructive characterization for this relation and Corollary 6.3 says that $\equiv$ is a generalization of the strong equivalence relation in the abstract context of $Fl_{\Sigma,T}$.

In conclusion, we belive that in the deterministic case the basic flowchart theory is that of the classes of $\equiv$-equivalent flowcharts.

(2) We need a semantical model in which $\equiv$-equivalent flowcharts have the same interpretation. Such model proves to be the strong iteration theories (see their axioms in Section 1). The strong iteration theories (as well as iteration theories in [11, 20; see 21, for their name] and theories with iterate in [13, 14]) are generalizations of both pointed iterative theories in [10] and rational theories in [3]. The new axiom (I4s) is crucial. Its necessity for our aim ($\equiv$-equivalent flowcharts have the same interpretation) is proved in Section 7, while its sufficiency is contained in the proof of our main theorem (Theorem 6.1). In conclusion, we take for the support theory a strong iteration theory.

(3) We can now try do describe the algebraic structure of $Fl_{\Sigma,T}/_{\equiv}$, when T is a strong iteration theory. The axioms (C1), (C2), (T1), (T2), (I0p), (I2) and (I3p) in Section 1 hold even in $Fl_{\Sigma,T}$. The left side of the equation in (T3) (respectively, in (T4), (I1)) is $\xrightarrow{i}$ (respectively, $\xrightarrow{s}$) equivalent with the right side. A technical analysis of (I4s), based on the characterization theorem for $\equiv$ (Theorem 4.1), completes the picture: $Fl_{\Sigma,T}/_{\equiv}$ is a strong iteration theory, too.

Moreover, the property that $\equiv$-equivalent flowcharts have the same interpretation in all strong iteration theories and the universality property (like in [9, 14]) show that $Fl_{\Sigma,T}/_{\equiv}$ is the strong iteration theory freely generated by adding Σ to the strong iteration theory T.

The first proof of the main result in this paper appeared in [33]. A partial similar result was obtained for the nondeterministic case in [34] using as support theory a matrix theory. In order to have a uniform representation for usual flowcharts we still represent morphisms in $Pfn_{\circ}$ also as matrices over $\{0,1\}$.

E. In Section 1 strong iteration theories are introduced. Section 2 gives the basic definitions on flowcharts. The basic congruence relation $\equiv$ is defined in Section 3, using as support theory T a paraiteration theory; it is shown that $FI_{\Sigma,T}/_{\equiv}$ is a paraiteration theory. A characterization theorem for $\equiv$ is given in Section 4; this is used in the next section to prove that $FI_{\Sigma,T}/_{\equiv}$ is a strong iteration theory, if T is so. The main theorem is proved in Section 6 (using all the above results). Section 7 gives a comparison of different types of theories. Some final remarks are given in Section 8.

1. STRONG ITERATION THEORIES

First of all we will define strong iteration theories. For the sake of simplicity we work in the nonsorted case, but all the results in this paper hold for sorted theories, too.

An algebraic theory T in [29, 17, 1] is given by : a family of morphisms $T(m,n)$, $m,n \geqslant 0$ [when T is known in context we allow the notation $f : m \to n$, instead of $f \in T(m,n)$]; two binary operations: composition $\cdot$ and tupling $\langle\, ,\rangle$; and some distinguished morphisms $1_n \in T(n,n)$, $0_n \in T(0,n)$, and $x_i^n \in T(1,n)$, for $i \in [n]$. Moreover $\cdot$ and $\langle\, ,\rangle$ have to verify the following six axioms. With the composition T has to be a category; this means that

(C1) $f\cdot(g\cdot h) = (f\cdot g)\cdot h$, for $f : m \to n$, $g : n \to p$ and $h : p \to q$;

(C2) $f\cdot 1_n = f = 1_m \cdot f$, for $f : m \to n$.

For axiomatizing the tupling we need two axioms for its extension to an arbitrary number of morphisms

(T1) $\langle f, \langle g,h\rangle\rangle = \langle\langle f,g\rangle, h\rangle$, for $f : m \to q$, $g : n \to q$ and $h : p \to q$;

(T2) $\langle f\rangle = \langle f,0_n\rangle = \langle 0_n,f\rangle = f$ and $\langle\,\rangle = 0_n$, for $f : m \to n$,

and two others for the unique source spliting of an arbitrary morphism into components,

(T3) $f_i = x_i^n \cdot \langle f_1,\dots,f_n\rangle$, for $f_1,\dots,f_n : 1 \to n$, and $i \in [n]$;

(T4) $\langle x_1^m \cdot f,\dots,x_m^m \cdot f\rangle = f$, for $f : m \to n$.

A preiteration theory T in [11] is an algebraic theory in which an iteration

$$^+ : T(m,m+n) \to T(m,n), \quad \text{for } m,n \geqslant 0.$$

is defined.

Let us agree on the following: 1_m+0_n and 0_m+1_n denotes $\langle x_1^{m+n},\dots,x_m^{m+n}\rangle$ and respectively $\langle x_{m+1}^{m+n},\dots,x_{m+n}^{m+n}\rangle$; the sum of $f\in T(p,m)$ and $g\in T(q,n)$ is defined by $f+g=\langle f(1_m+0_n),\ g(0_m+1_n)\rangle$; s_n^m denotes $\langle 0_m+1_n,\ 1_m+0_n\rangle$; every partial function $y\in \mathbf{Pfn}(m,n)$ can be seen as a morphism y_T in an arbitrary preiteration theory T (the interpretation of the nowhere defined function $\perp_{m,n}:[m]\multimap[n]$ is $1_m^+\cdot 0_n$), but we agree to dropp the index.

Before the main definition we denote three axioms for iteration.

(I0) $(f(1_n+g))^+ = f^+g$, for $f:m\to m+n$, $g:n\to p$;

(I3) $g(f(g+1_p))^+ = (gf)^+$, for $f:m\to n+p$, $g:n\to m$;

(I4) if $f(y+1_p)=yg$ then $f^+=yg^+$, for $f:m\to m+p$, $g:n\to n+p$ and a function $y:[m]\to[n]$.

DEFINITION 1.1. A preiteration theory is a strong iteration theory if the iteration fulfils the following five axioms:

(I0p), which is (I0) only for g of the form 0_r+1_n or 1_n+0_r;

(I1) $f\langle f^+,1_n\rangle = f^+$, for $f:m\to m+n$;

(I2) $(f(\langle 1_m,1_m\rangle+1_n))^+ = f^{++}$, for $f:m\to m+m+n$;

(I3p), which is (I3) only for g of the form 1_n+0_q;

(I4s), which is (I4) only for surjective functions y. $\square$

Comment 1.2. As we will see in Corollary 1.10 and Proposition 1.12 the restrictions in (I0), (I3) and (I4) was imposed only since we want an axiomatic system which can be easily verified.

The axiom (I0) says that the iteration has a "uniform behaviour" with respect to the last n variables of $f:m\to m+n$ (hence these last variables can be seen as parameters). By (I1), the iteration gives a "canonical" solution for the Elgot recursive system $x = f\langle x, 1_n\rangle$. In this context, the axioms (I2) and (I3p) show that the canonical solution of a system can be expresses in terms of the canonical solutionsof its components in a way similar to the Gauss substitution method for solving linear systems (Proposition 1.9, below). The last axiom expresses the preserving of the canonical solution when we rename the variables, not necessarily in an injective way, but in a consistent way, i.e. if two variables have the same rename then the right side of their equations are the same after renaming. $\square$

Essentially, the diference between strong iteration theories, iteration theories in [21] and theories with iterate in [13] consists in the power of their axioms of type (I4). In theories with iterate it is used

(I4t), which is (I4) only for transpositions $y = s^q_r$,

and it follows that (I4) holds for arbitrary bijective functions. In iteration theories it is used the following equational axiom, stronger than (I4t),

(I4p), which is (I4s) applied only if there are $h : n \longrightarrow m+p$ and $y_1,\dots,y_m : [m] \longrightarrow [m]$ with $y_i y = y$, $\forall i \in [m]$, such that $g = h(y+1_p)$ and $f = \langle x^m_1 yh(y_1+1_p),\dots, x^m_m yh(y_m+1_p)\rangle$.

Examples and nonexamples. It is well known that all "natural iteration" theories fulfil all the axioms (I0), (I1), (I2), (I3) and (I4) listed above. Particularly, all ω-continuous theories in [2], rational theories in [3], pointed iterative theories in [10], and metric iteration theories in [36] are strong iteration theories; also the iteration defined in a partially additive category in [5] fulfils these axioms.

The following three examples of ω-continuous (hence rational) theories are very important ones.

Example 1.4. (basic example) $\mathbf{Pfn}_D$ with the operations from Subsection B of Introduction is a strong iteration theory; particularly, $\mathbf{Pfn}_{\cdot}$ is the initial (in the sense of [11]) strong iteration theory. [When D has at least two elements $\mathbf{Pfn}_D$ is not ideal, hence is not a pointed iterative theory.] $\square$

Example 1.5. Let Ω be a ranked aphabet; the theory CT_{Ω}

$$CT_{\Omega}(m,n) = \{\text{m-tuples of partial } \Omega\text{-trees with leaves labeled in } \{x_1^n,\dots,x_n^n\}\}$$

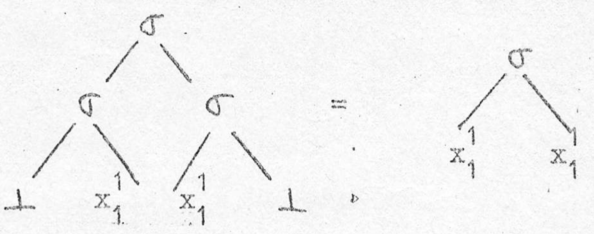
with the first order substitution (trees for leaves) in, [15] as composition, the natural tupling, and the iteration given by Kleene's fix point theorem is a strong iteration theory.

The subtheory R_{Ω} of rational Ω-trees in CT_{Ω} (i.e. of trees obtained by unfolding the usual, finite Ω-flowchart schemes [23]) is a strong iteration theory. [It follows from [20] that R_{Ω} is even the strong iteration theory freely generated by Ω.] $\square$

Example 1.6. The theory generated by the relations over a set D and denoted by $\mathbf{Rel}_D$ (more exactly, the morphisms in $\mathbf{Rel}_D(m,n)$ are the relations between $[m]\times D$ and $[n]\times D$, also represented as matrices of relations over D) with the composition of relations, the tupling given by $\langle[f],[g]\rangle = \begin{bmatrix} f \\ g \end{bmatrix}$ and the iteration given by $[f\ g]^{\dagger} = [f]^{*}[g]$, where $[f]^{*} = 1_m \cup [f] \cup [f]^2 \cup \dots$, for $f \in \mathbf{Rel}_D(m,m)$, $g \in \mathbf{Rel}_D(m,n)$, is a strong iteration theory. $\square$

Example 1.7. The theory $\mathbf{Fl}_{\Sigma,T}/\equiv$ of the classes of strongly equivalent flowcharts over a strong iteration theory T is a strong iteration theory (Theorem 5.1); particularly $\mathbf{Fl}_{\Sigma,\mathbf{Pfn}_{\cdot}}/\equiv$ is the strong iteration theory freely generated be Σ. [When $\sigma_{in} = 1$, $\forall \sigma \in \Sigma$ the theories R_{Σ} and $\mathbf{Fl}_{\Sigma,\mathbf{Pfn}_{\cdot}}/\equiv$ are isomorphic; the last theory has the advantage of consisting only in elements of finite structure.] $\square$

Nonexample 1.8. For Ω with only two elements: $\perp \in \Omega(1,0)$ and $\sigma \in \Omega(1,2)$ Esik proved in [22] that the quotient of R_Ω by the congruence generated by



is an iteration theory which is not a strong iteration theory. $\square$

More about the comparison of different types of theories and about the necessity of strong iteration theories will be said in Section 7. In the remainder of this section we will prove that the restrictions in (I0p), (I3p) and (I4s) are superfluous.

While strong iteration theories, iteration theories, and theories with iterate differ by their axioms of type (I4), in the presented axiomatic system there is another new part. The "pairing" axiom in [13, 21]

(P) $\langle f,g\rangle^+ = \langle f^+\langle (g\langle f^+,1_{n+p}\rangle)^+,1_p\rangle,(g\langle f^+,1_{n+p}\rangle)^+\rangle$

for $f : m \to m+n+p$ and $g : n \to m+n+p$,

is replaced by two simpler ones (I2) and (I3p). Before proving this we specify a technique for obtaining new identities.

(TECH) Suppose (P) and (I4t) hold in T. Given a system $f : m \to m+n+p$, $g : n \to m+n+p$ compute with (P) the f-component and the g-component of its solution $\langle f,g\rangle^+$, i.e. $f^+\langle h,1_p\rangle$, respectively h, where $h = (g\langle f^+,1_{n+p}\rangle)^+$. For the permuted system $\langle g(s^n_m+1_p), f(s^n_m+1_p)\rangle$, again with (P) compute the g-component and the f-component of its solution. Identify the f-component (respectively, the g-component) of the solutions of the given system and the permuted system. The conclusion is: these two identities hold in T. $\square$

PROPOSITION 1.9. Suppose that we have a preiteration theory T in which (I0p), (I1) and (I4t) hold; then the pairing axiom (P) holds in T iff (I2) and (I3p) hold in T.

Proof. (a) Suppose that (P) holds in T. (TECH) for the g-component of the system $\langle g(0_m+1_m+0_p), f(1_m+0_m+1_p)\rangle$ provides (I3); and for the f-component of the system $\langle f, 1_m+0_{m+p}\rangle$ provides (I2).

(b) Suppose that (I2) and (I3p) hold in T. It can be proved that

(α) $(\langle f,g\rangle(1_m+0_n+1_p))^+ = \langle f^+, g\langle f^+,1_p\rangle\rangle$, for $f: m \to m+p$ and $g: n \to m+p$.

Indeed, with (I3p) we have

$$(1_m+0_n)(\langle f,g\rangle(1_m+0_n+1_p))^+ = ((1_m+0_n)\langle f,g\rangle)^+ = f^+ ;$$

now (α) follows by using (I1),

$$(\langle f,g\rangle(1_m+0_n+1_p))^+ = \langle f,g\rangle(1_m+0_n+1_p)\langle f^+,\dots,1_p\rangle = \langle f^+, g\langle f^+,1_p\rangle\rangle.$$

Using (I4t) and (I3p) it follows that (I3) holds for g of the form 0_r+1_q. Indeed, for $f: r+q \to q+p$ we have

$$(0_r+1_q)(f(0_r+1_{q+p}))^+ = (0_r+1_q)s^q_r(s^r_q\, f(0_r+1_{q+p})(s^q_r+1_p))^+$$
$$= (1_q+0_r)(s^r_q\, f(1_q+0_r+1_p))^+ = ((1_q+0_r)s^r_q f)^+ = ((0_r+1_q)f)^+.$$

A similar computation as in (α), but with (I3) for $g = 0_m+1_n$ leads to

(β) $(\langle f,g\rangle(0_m+1_{n+p}))^+ = \langle f^+\langle g^+,1_p\rangle, g^+\rangle$, for $f: m \to n+p$ and $g: n \to n+p$.

For f, g as in (P), using in turn (I2), (α), (I0p) and (β) we have

$$\langle f,g\rangle^+ = (\langle f,g\rangle(1_m+0_m+1_{n+p})(1_m+0_n+1_{m+n+p})(\langle 1_{m+n},1_{m+n}\rangle+1_p))^+$$
$$= (\langle f,g\rangle(1_m+0_m+1_{n+p})(1_m+0_n+1_{m+n+p}))^{++}$$
$$= \langle (f(1_m+0_m+1_{n+p}))^+, g(1_m+0_m+1_{n+p})\langle (f(1_m+0_m+1_{n+p}))^+, 1_{m+n+p}\rangle\rangle^+$$
$$= (\langle f^+, g\langle f^+,1_{n+p}\rangle\rangle(0_m+1_{n+p}))^+$$
$$= \langle f^+\langle (g\langle f^+,1_{n+p}\rangle)^+, 1_p\rangle, (g\langle f^+,1_{n+p}\rangle)^+\rangle.$$

This proves (P). $\square$

COROLLARY 1.10. Suppose that (I0p), (I1), (I2), (I3p) and (I4t) hold in a preiteration theory T; then (I0) and (I3) hold in T.

Proof. We know from Proposition 1.9 that (P) holds in T, hence we can apply (TECH). For $f : m \to m+n$ and $g : n \to p$, this technique applied to the f-component of the system $\langle f(1_{m+n}+0_p), g(0_{m+n}+1_p)\rangle$ extends (I0p) (in other context this was proved in [21]). For $f : m \to n+p$ and $g : n \to m$ the technique (TECH) applied to the g-component of the system $\langle f(0_m+1_{n+p}), g(1_m+0_{n+p})\rangle$ extends (I3p). $\square$

In a slightly different form the following identities are known from [20, 14].

COROLLARY 1.11. In a theory with iterate the following hold.

(a) $\langle f, g(0_m+1_{n+p})\rangle^+ = \langle f^+, 1_n+0_p\rangle \langle g^+, 1_p\rangle$, for $f : m \to m+n+p$, $g : n \to n+p$;

(b) $\langle f(1_m+0_n+1_p), g(0_m+1_{n+p})\rangle^+ = \langle f^+, g^+\rangle$, for $f : m \to m+p$ and $g : n \to n+p$;

(c) $g^+\langle 1_n, (f\langle g^+, 1_{n+q}\rangle s^q_n)^+\rangle = (g\langle 1_{m+n}, (fs^q_{m+n})^+\rangle)^+$,

for $g : m \to m+n+q$, $f : q \to m+n+q$.

(Therefore (a), (b) and (c) hold in a strong iteration theory, too.)

Proof. For (c) use (TECH) for the g-component of $\langle f,g\rangle s^q_{m+n}$. $\square$

PROPOSITION 1.12. The axiom (I4) holds in a strong iteration theory.

Proof. The first step is to prove that (I4) holds for arbitrary injective functions y. Using (I4) for bijections, we can suppose that these injections have the particular form 1_r+0_q. Consequently, let $f : r \to r+p$, $g : r+q \to r+q+p$ be such that $f(1_r+0_q+1_p) = (1_r+0_q)g$, hence $g = \langle f(1_r+0_q+1_p), g'\rangle$. Using (P) we have the desired identity $(1_r+0_q)g^+ = f^+$.

For the second step, suppose that $f(y+1_p) = yg$, for $f : m \to m+p$, $g : n \to n+p$ and a function y. Take a decomposition of y as $y = y_s y_i$, where y_s is a surjective function and y_i is an injective function; take two functions $u : r \to m$ such that $uy_s = 1_r$ and $v : n \to r$ such that $y_i v = 1_r$; and set $f' = y_i g(v+1_p)$. It follows that $f(y_s+1_p) = y_s f'$ and $uf(y_s y_i+1_p) = y_i g$. From the first identity and (I4s) we have $f^+ = y_s f'^+$. From the second one we have $f'(y_i+1_p) = y_i g(vy_i+1_p) = uf(y_s y_i v y_i+1_p) = uf(y+1_p) = uyg = y_i g$, hence by the first step we obtain $f'^+ = y_i g^+$. Consequently, $f^+ = y_s y_i g^+ = yg^+$. $\square$

2. Flowchart theories

As it was shown in Subsection C of Introduction for a flowchart theory $Fl_{\Sigma,T}$ (see its definition there) we need a double-ranked set of variables for atomic flowcharts and a "theory" which gives morphisms for connections. In this section T is supposed to be a preiteration theory.

We will use some standard notation : a flowchart $F \in Fl_{\Sigma,T}(m,n)$ is $F = (l,e)$, where $e \in \Sigma^*$ and $l \in T(m+e_{out}, n+e_{in})$; q denotes e_{in} ; and $\langle i,t\rangle$ is the spliting of l into the input $i : m \to n+e_{in}$ and the transfer $t : e_{out} \to n+e_{in}$.

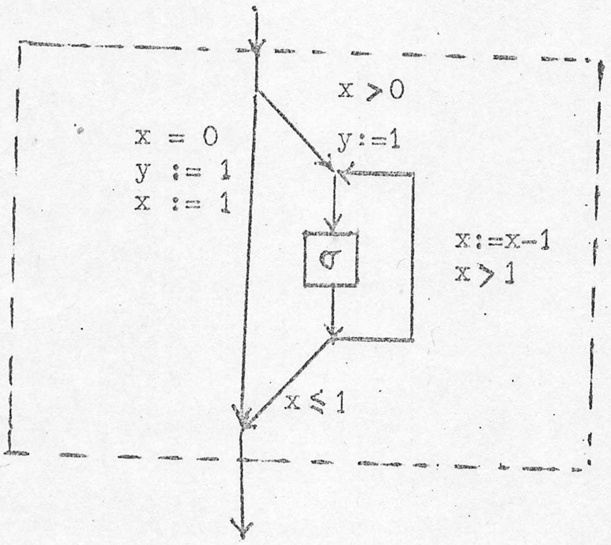


Figure 3.

Example 2.1. Like an X-polynomial over a concrete ring R, a Σ-flowchart over a concrete theory T is a mixture of concrete and abstract elements. Here is an example of a unusual flowchart (see Figure 3). The flowchart is $F = (\langle i,t\rangle, \sigma)$, where $i,t \in Pfn_{\omega^2}(1,2)$ (here ω denotes the set of natural numbers) are

$i(1,(x,y))=$ "if $x = 0$ then $(1,(1,1))$ else $(2,(x,1))$," $\quad \forall (x,y) \in \omega^2$;

$t(1,(x,y))=$ "if $x \leq 1$ then $(1,(x,y))$ else $(2,(x-1,y))$," $\quad \forall (x,y) \in \omega^2$. $\square$

The operations on flowcharts (informally defined in Subsection A of Introduction) have the following exact definitions.

DEFINITION 2.2 (basic operations on flowcharts).

Composition: for $F : m \to n$, $F' : n \to p$ we define $F \cdot F' : m \to p$ as

$$F \cdot F' = (\langle l(i'+1_q)(1_p+s^q_{q'}),\ t'(1_p+0_q+1_{q'})\rangle,\ ee').$$

Tupling: for $F : m \to p$, $F' : n \to p$, using $x = 1_{p+q}+0_{q'}$ and $x' = 1_p+0_q+1_{q'}$ we define $\langle F, F'\rangle : m+n \to p$ as

$$\langle F, F'\rangle = (\langle ix,\ i'x',\ tx,\ t'x'\rangle,\ ee').$$

Iteration: for $F : m \to m+n$ we define $F^+ : m \to n$ as

$$F^+ = (l\langle i^+, 1_{n+q}\rangle,\ e). \quad \square$$

Comment 2.3. The complications in the definitions of iteration of flowchart schemes in [19, 4], or more recently in [12], are due to the fact that it was not used a support theory, with a defined iteration. In a slightly different form the present definition of iteration appeared in [14]. $\square$

3.A flowchart equivalence

Here, and in Section 4, we will suppose that T is a paraiteration theory, i.e. a preiteration theory in which iteration fulfils the parameter axiom (I0). This axiom is necessary for the compatibility of iteration with the equivalence relation $\equiv$ (to be defined below).

The basic equivalence relation $\equiv$ can be introduced as the equivalence relation generated by the natural relations $\xrightarrow{i}$, $\xrightarrow{s}$ in Subsection D of Introduction. Roughly speaking, two flowcharts are $\equiv$-equivalent if they make the same steps in execution. The following claim (which holds true, see Corollary 6.3) gives some hints for the understanding of this notion.

CLAIM 3.1. When $T = Pfn$ and $\sigma_{in} = 1$, $\forall \sigma \in \Sigma$, two flowcharts are $\equiv$-equivalent iffthey unfold the same tree (i.e. $\equiv$ is the strong equivalence in [18]).$\square$

We now give the exact definitions of $\xrightarrow{i}$ and $\xrightarrow{s}$ in the abstract context of $Fl_{\Sigma,T}$. Before this, we need some notation :

For a string $e \in \Sigma^*$, $|e|$ denotes its length and $e_1,\dots,e_{|e|}$ its letters. For $e,e' \in \Sigma^*$ denote by $PStr_{\Sigma}(e,e')$ the set of all partial functions f from $[|e|]$ to $[|e'|]$ which preserves letters, i.e. if $f(k) = j$ then $e_k = e'_j$; clearly $PStr_{\Sigma}$ is a Σ-sorted algebraic theory. Every partial function $y \in PStr_{\Sigma}(e,e')$ has two naturally "block" extensions to inputs and to outputs denoted by

$$y_{in} : e_{in} \rightarrow e'_{in} \quad \text{and respectively,} \quad y_{out} : e_{out} \rightarrow e'_{out}.$$

[In fact, the in-extension is the unique functor from $PStr_{\Sigma}$ to Pfn given by $e \rightarrow e_{in}$ and $x^{|e|}_k \rightarrow 0_{(e_1 \dots e_{k-1})_{in}} + 1_{(e_k)_{in}} + 0_{(e_{k+1} \dots e_{|e|})_{in}}$; similarly, with the out-extension.] Str_{Σ} denotes the Σ-sotred algebraic subtheory of $PStr_{\Sigma}$, which consists of all totally defined functions.

DEFINITION 3.2 (basic relation $\rightarrow$). The flowchart F is simulated via y in F' (also denoted: $F \xrightarrow[y]{} F'$), if $y \in \mathbf{Str}_\Sigma(e,e')$ and

$$\mathrm{I}(1_n + y_{in}) = (1_m + y_{out})\mathrm{I}'. \quad \square$$

Example 3.3. Take T = Pfn. with its morphisms represented as matrices over $\{0,1\}$; denote by 1_m the identity $m \times m$ matrix and by $0_{m,n}$ the zero $m \times n$ matrix. Two generic examples of simulations are the following.

3.3.a (case when y is an injective function). The following

$$F = \begin{array}{c} \\ m \\ e \end{array}\!\!\begin{array}{c} n \\ \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \end{array} \xrightarrow[{[1_e \quad 0_{|e|,|e'|}]}]{} \begin{array}{c} \\ m \\ e \\ e' \end{array}\!\!\begin{array}{c} n \\ \left[\begin{array}{c|cc} A_1 & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right] \end{array} = F_1$$

is a simulation iff $\begin{bmatrix} A & B & 0 \\ C & D & 0 \end{bmatrix} = \begin{bmatrix} A_1 & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \end{bmatrix}$. In other words, $F \xrightarrow[{[1_{|e|} \quad 0_{|e|,|e'|}]}]{} F_1$ iff F_1 can be obtained from F by adding an inaccessible part e'.

3.3.b (case when y is a surjective function). The reader is invited to extend the following particular example in order to obtain exemples of simulations via arbitrary surjective functions. The following

$$F_1 = \begin{array}{c} \\ m \\ e \\ e' \end{array}\!\!\begin{array}{c} n \\ \left[\begin{array}{c|cc} A_1 & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right] \end{array} \xrightarrow[{\begin{bmatrix} 1_{|e|} \\ 1_{|e|} \end{bmatrix}}]{} \begin{array}{c} \\ m \\ e \end{array}\!\!\begin{array}{c} n \\ \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \end{array} = F$$

is a simulation iff $\begin{bmatrix} A_1 & B_1 \cup B_2 \\ C_1 & D_{11} \cup D_{12} \\ C_2 & D_{21} \cup D_{22} \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \\ C & D \end{bmatrix}$, where $\cup$ is the natural extension to matrices of the Boolean union in $\{0,1\}$. In other words, F_1 is simulated via $\begin{bmatrix} 1 & |e| \\ 1 & |e| \end{bmatrix}$ in F' iff F_1 can be obtained from F by making two copies of its vertices and sharing their internal connexions, i.e. by a partially unfolding of its vertices. $\square$

These examples show that we can take as the exact definitions for $\xrightarrow{i}$ and $\xrightarrow{s}$, in the abstract context of $\mathrm{Fl}_{\Sigma,T}$, the restrictions of the simulation $\longrightarrow$ to injective functions y and respectively to surjective functions y. Since $\to \subseteq \xrightarrow{s} \cdot \xrightarrow{i}$ (Lemma 4.2.a below) it follows that $\xrightarrow{i} \cup \xrightarrow{s}$ and $\to$ generate the same equivalence relation ; in this section we prefer to work with $\to$.

LEMMA 3.4. Suppose that the support theory is a paraiteration theory. If $F \xrightarrow[y]{} F'$ and $F_1 \xrightarrow[y_1]{} F'_1$, then we have (when the operations make sense) :

(a) $F \cdot F_1 \xrightarrow[y+y_1]{} F' \cdot F'_1$;

(b) $\langle F, F_1 \rangle \xrightarrow[y+y_1]{} \langle F', F'_1 \rangle$;

(c) $F^+ \xrightarrow[y]{} F'^+$.

Proof. Routine computations. For (c), we need (I0). $\square$

The simulation $\to$ is a reflexive and transitive relation, but not a symmetrical one. Denote $\to^{-1}$ by $\leftarrow$.

DEFINITION 3.5. We say that F is equivalent with F' (written $F \equiv F'$) if (F,F') is in the transitive closure of $\to \cup \leftarrow$. $\square$

Since $\to$, $\leftarrow$ are reflexive and transitive relations we have the following characterization:

$$F \equiv F' \text{ iff } \begin{cases} \text{there are } F_1,\dots,F_n,\ y_1,\dots,y_{n+1} \text{ such that} \\ F \xrightarrow[y_1]{} F_1 \xleftarrow[y_2]{} F_2 \dots F_{n-1} \xrightarrow[y_n]{} F_n \xleftarrow[y_{n+1}]{} F'. \end{cases}$$

PROPOSITION 3.6. If the support theory is a paraiteration theory, then the equivalence relation $\equiv$ is compatible with the composition, the tupling and the iteration (hence it is a congruence).

Proof. Since $\rightarrow$, $\leftarrow$ are reflexive relations we can suppose that two chains of simulations for $\equiv$ have the same length n. The proof now directly follows from Lemma 3.4. $\square$

THEOREM 3.7. If T is a paraiteration theory, then $Fl_{\Sigma,T}/_{\equiv}$ is a paraiteration theory.

Proof. By Proposition 3.6 the operations are well defined in $Fl_{\Sigma,T}/_{\equiv}$. Since every $f \in T(m,n)$ may be seen as the "empty" flowchart $(f,\lambda) \in Fl_{\Sigma,T}(m,n)$ (λ denotes the empty string) we can take as special morphisms 1_n, 0_n and x^n_k in $Fl_{\Sigma,T}$ the corresponding morphisms in T.

An easy computation shows that (C1), (C2), (T1) and (T2) hold in $Fl_{\Sigma,T}$, hence in $Fl_{\Sigma,T}/_{\equiv}$, too. The following computation shows that (T3) and (T4) hold in $Fl_{\Sigma,T}/_{\equiv}$, hence $Fl_{\Sigma,T}/_{\equiv}$ is an algebraic theory.

For (T3) we have to show that

$$F^k \equiv x^m_k \langle F^1,\dots,F^m\rangle, \quad \text{for } F^1,\dots,F^m : 1 \rightarrow m.$$

Let x^j denotes $1_n + 0_{q^1} + \dots + 1_{q^j} + \dots + 0_{q^m}$; then the right side is $(\langle i^k x^k, t^1 x^1, \dots, t^m x^m\rangle, e^1 \dots e^m)$. The following computation shows that F^k can be simulated via $0_{e^1} + \dots + 1_{e^k} + \dots + 0_{e^m}$ in $x^m_k \langle F^1,\dots,F^m\rangle$:

$$\langle i^k t^k\rangle\,(1_n + 0_{q^1} + \dots + 1_{q^k} + \dots + 0_{q^m}) = \langle i^k x^k, t^k x^k\rangle$$

$$= (1_1 + 0_{e^1_{out}} + \dots + 1_{e^k_{out}} + \dots + 0_{e^m_{out}})\,\langle i^k x^k, t^1 x^1, \dots, t^m x^m\rangle.$$

This proves (T3).

For (T4) we have to show that

$$\langle x^m_1 F,\dots,x^m_m F\rangle \cong F, \text{ for } F : m \to n.$$

Let x^k denotes $1_n + 0_{(k-1)q} + 1_q + 0_{(m-k)q}$; then the left side is $(\langle x^m_1 ix^1,\dots,x^m_m ix^m,\ tx^1,\dots,tx^m\rangle,\ e)$. The following verification shows that $\langle x^m_1 F,\dots,x^m_m F\rangle$ can be simulated via $\langle 1_e,\dots,1_e\rangle$ in F :

$$\langle x^m_1 ix^1,\dots,x^m_m ix^m, tx^1,\dots,tx^m\rangle (1_n + \langle 1_q,\dots,1_q\rangle)$$

$$= \langle x^m_1 i,\dots,x^m_m i,t,\dots,t\rangle = (1_m + \langle 1_{e_{out}},\dots,1_{e_{out}}\rangle)\langle i,t\rangle.$$

This proves (T4).

The following computation completes the proof, by verifying the parameter axiom (I0). For $F : m \to m+n$ and $F' : n \to p$ we have,

$$(F(1_m+F'))^+ = (\langle 1(1_m+i'+1_q)(1_{m+p}+s^q_{q'}),\ t'(0_m+1_p+0_q+1_{q'})\rangle \cdot$$
$$\langle i^+(i'+1_q)(1_p+s^q_{q'}),\ 1_{p+q+q'}\rangle,\ \ ee')$$

$$= (\langle 1\langle i^+,1_{n+q}\rangle(i'+1_q)(1_p+s^q_{q'}),\ t'(1_p+0_q+1_{q'})\rangle,\ ee') = F^+F'. \quad \square$$

4. A characterization theorem for the flowchart equivalence

Let $\approx$ denotes the restriction of $\longrightarrow$ to bijective functions y (when $F \approx F'$ we say that F is isomorphic with F'); then $\approx \subseteq \xrightarrow{s} \cap \xrightarrow{i} \cap \xleftarrow{s} \cap \xleftarrow{i}$. We say that the flowchart F is minimal if there is no $F' \equiv F$ with $|e'| < |e|$. The relations $\xrightarrow{s}$ and $\xleftarrow{i}$ are reductions (when $F \xrightarrow{s} F'$ or $F \xleftarrow{i} F'$ we have $|e'| \leq |e|$).

The following characterization theorem says that two flowcharts are $\equiv$-equivalent iff in two steps (firstly, identifying vertices and secondly, deleting inaccessible vertices) they can be reduced to the same flowchart.

THEOREM 4.1. $\quad \equiv \; = \; \xrightarrow{s} \cdot \xleftarrow{i} \cdot \xrightarrow{i} \cdot \xleftarrow{s}$.

$x \backslash y$	$\xrightarrow{i}$	$\xrightarrow{s}$	$\xleftarrow{i}$	$\xleftarrow{s}$
$\xrightarrow{i}$	yes	yes	yes	yes
$\xrightarrow{s}$	yes	yes	no	no
$\xleftarrow{i}$	yes	yes	yes	yes
$\xleftarrow{s}$	yes	yes	yes	yes

Table 1. Is $x \cdot y \subseteq y \cdot x$?

A complete picture of the commuting relations is given in Table 1 but for our aim (i.e. for a proof of Theorem 4.1) besides obvious commutations, we need only the commutations from the following lemma.

LEMMA 4.2. In $\mathbf{FI}_{\Sigma,T}$ the following commutations hold.

(a) $\xrightarrow{i} \cdot \xrightarrow{s} \subseteq \longrightarrow \subseteq \xrightarrow{s} \cdot \xrightarrow{i}$; $\qquad$ (a^0) $\xleftarrow{s} \cdot \xleftarrow{i} \subseteq \longleftarrow \subseteq \xleftarrow{i} \cdot \xleftarrow{s}$;

(b) $\xrightarrow{i} \cdot \xleftarrow{i} \subseteq \xleftarrow{i} \cdot \xrightarrow{i}$;

(c) $\xleftarrow{i} \cdot \xrightarrow{s} \subseteq \xrightarrow{s} \cdot \xleftarrow{i}$; $\qquad$ (c^0) $\xleftarrow{s} \cdot \xrightarrow{i} \subseteq \xrightarrow{i} \cdot \xleftarrow{s}$;

(d) $\xleftarrow{s} \cdot \xrightarrow{s} \subseteq \xrightarrow{s} \cdot \xleftarrow{s}$.

<u>Proof of Theorem 4.1 assuming Lemma 4.2.</u> Since $\xrightarrow{i}$, $\xrightarrow{s}$, $\xleftarrow{i}$ and $\xleftarrow{s}$ are reflexive, transitive relations and by Lemma 4.2 (the points (a), (a^o)) we have $\rightarrow \subseteq \xrightarrow{s} \cdot \xrightarrow{i}$, $\leftarrow \subseteq \xleftarrow{i} \cdot \xleftarrow{s}$ it follows that the congruence $\equiv$ can be written as

$$F \equiv F' \quad \text{iff} \quad \exists\, n \geqslant 0 \text{ such that } F \ \varrho^n \ F',$$

where $\varrho = \xrightarrow{s} \cdot \xleftarrow{i} \cdot \xrightarrow{i} \cdot \xleftarrow{s}$. From Lemma 4.2 easily follows that $\varrho \cdot \xrightarrow{s}$, $\varrho \cdot \xleftarrow{i}$, $\varrho \cdot \xrightarrow{i}$ and $\varrho \cdot \xleftarrow{s}$ are included in ϱ, hence ϱ is transitive. This proves the nonobvious inclusion $\equiv \subseteq \varrho$. $\square$

For a generic flowchart $F = (\langle i,t \rangle, e)$ and $j \in [|e|]$ we shall denote by t_j the component of t corresponding to the outputs of e_j, namely $t_j = (x_j^{|e|})_{out} \cdot t$. For a function $y \in \mathbf{Str}_\Sigma(e,e')$, $Ker(y) = \{(j,k) \mid j,k \in [|e|] \text{ and } y(j) = y(k)\}$ denotes its kernel and $Im(y)$ denotes its range.

<u>Remark 4.3.</u> Suppose we are given $F : m \rightarrow n$ and a surjective function $y \in \mathbf{Str}_\Sigma(e,e')$ such that

(α) $\quad (j,k) \in Ker(y)$ implies $t_j(1_n + y_{in}) = t_k(1_n + y_{in})$;

then there exists a unique F' such that $F \xrightarrow[y]{} F'$, namely $F' = ((1_m + v_{out})\, 1\, (1_n + y_{in}),\ e')$, where $v \in \mathbf{Str}_\Sigma(e,e')$ is an arbitrary left inverse of y (i.e. $vy = 1_e$).

Conversely, if $F \xrightarrow[y]{} F'$ for a (not necessary surjective) function y, then y fulfils (α) for F. $\square$

<u>Remark 4.4.</u> Suppose we are given $F : m \rightarrow n$ and an injective function $y \in \mathbf{Str}_\Sigma(e',e)$ such that

(β) $\quad (1_m + y_{out})\, 1\, (1_n + (y^{-1}y)_{in}) = (1_m + y_{out})\, 1,$

where $y^{-1} \in PStr_{\Sigma}(e,e')$ is the partial function defined only on $Im(y)$ and such that $yy^{-1} = 1_{e'}$; then there exists a unique F' such that $F' \xrightarrow[y]{} F$, namely $F' = ((1_m + y_{out}) I (1_n + y^{-1}_{in}), e')$.

Conversely, if $F \xrightarrow[y]{} F'$ for an injective function y, then y fulfils (β) for F. $\Box$

Lemma 4.5. (a) If $F \xrightarrow[y']{s} F'$, $F \xrightarrow[y]{} F''$, and there exists $y \in Str_{\Sigma}(e',e'')$ such that $y = y'y''$, then $F' \xrightarrow[y'']{} F''$.

(b) If $F' \xrightarrow[y']{i} F$, $F'' \xrightarrow[y]{} F$, and there exists $y'' \in Str_{\Sigma}(e'',e')$ such that $y = y''y'$, then $F'' \xrightarrow[y'']{} F'$.

Proof. (a) Let $v \in Str_{\Sigma}(e',e)$ be such that $vy' = 1_{e'}$; then $I' = (1_m + v_{out}) \cdot I(1_n + y'_{in})$. Hence

$$I'(1_n + y''_{in}) = (1_m + v_{out}) I (1_n + y_{in}) = (1_m + (vy)_{out}) I'' = (1_m + y''_{out}) I''.$$

(b) Dual. $\Box$

Proof of Lemma 4.2. Since (a^0), (c^0) directly follows from (a), respectively (c) we have to prove only the commutations (a), (b), (c) and (d).

(a) The first inclusion is obvious. For the second one suppose $F \xrightarrow[y]{} F'$; then by Remark 4.3 it follows that $(j,k) \in Ker(y)$ implies $t_j(1_n + y_{in}) = t_k(1_n + y_{in})$. Take a decomposition $y = y_s y_i$, where y_s is a surjective function and y_i is an injective function. Since $Ker(y) = Ker(y_s)$ and $y_i y_i^{-1} = 1$ the above implication holds for y_s; too. Remark 4.3 shows that y_s generates a simulation for F and by Lemma 4.5, y_i also gives a simulation.

(b) Suppose that $F' \xrightarrow[y']{i} F \xleftarrow[y'']{i} F''$. It is easy to find two injections $e' \xleftarrow[z']{} e^1 \xrightarrow[z'']{} e''$ in Str_{Σ}, such that $z'y' = z''y'' =: y$ and $y'^{-1}y'\, y''^{-1}y'' = y^{-1}y$ (the $^{-1}$ notation is that in Remark 4.4). The function y generates a simulation for F

because it fulfils the (β)-condition in Remark 4.4 : indeed, y' and y'' fulfil the (β)-condition for F ; hence

$$\begin{aligned}(1_m+y_{out})l &= (1_m+z''_{out})(1_m+y''_{out})\, l\, (1_n+(y''^{-1}y'')_{in})\\ &= (1_m+(z'y')_{out})\, l\, (1_n+(y''^{-1}y'')_{in})\\ &= (1_m+z'_{out})(1_m+y'_{out})\, l\, (1_n+(y'^{-1}y')_{in})(1_n+(y''^{-1}y'')_{in})\\ &= (1_m+y_{out})\, l'\, (1_n+(y^{-1}y)_{in}).\end{aligned}$$

In addition, by Lemma 4.5, z' and z'' are (obviously injective) simulations.

(c) Suppose that $F' \xleftarrow[y']{i} F \xrightarrow[y'']{s} F''$. Using an isomorphic representation of F' we may suppose that y' has the particular form $1_e+0_{e.}$. Set $F^1 = (<l''(1_n+(1_{e''}+0_{e.})_{in}),\ (0_e+1_{e.})_{out}t'(1_n+(y''+1_{e.})_{in}>,\ e''e.)$; then a routine computation shows that $F' \xrightarrow[y''+1_{e.}]{s} F^1 \xleftarrow[1_{e''}+0_{e.}]{i} F''$.

(d) Suppose that $F' \xleftarrow{s} F \xrightarrow{s} F''$. Let $\sim$ be the least equivalence relation in $[|e|]$ containing Ker(y') and Ker(y''). We shall use the following constructive definition

$$j\sim k \text{ iff } \begin{cases}\text{there is a sequence of elements in } [|e|] \text{ denoted } j=n_1,\dots,n_r=k\\ \text{such that } (n_s,n_{s+1})\in \mathrm{Ker}(y')\cup\mathrm{Ker}(y''), \text{ for } s<r.\end{cases}$$

The relation $\sim$, as well as Ker(y') and Ker(y'') does not identify elements $j,k\in[|e|]$ with $e_j \neq e_k$; hence it has a representation $\sim = \mathrm{Ker}(y)$, for a surjective function $y\in \mathrm{Str}_{\Sigma}(e,e^1)$. Let $e' \xrightarrow[z']{} e^1 \xleftarrow[z'']{} e''$ be the induced surjections in Str_{Σ} that fulfil $y'z' = y''z'' = y$. We shall show that y generates a simulation for F, namely that it fulfils the (α)-condition in Remark 4.3 : indeed, we know that $(j,k)\in \mathrm{Ker}(y')$ implies $t_j(1_n+y'_{in}) = t_k(1_n+y'_{in})$ and similarly for y''; these, the fact that $(j,k)\in\mathrm{Ker}(y)$ implies that there exist $j=n_1,\dots,n_r=k$ with $(n_s,n_{s+1})\in\mathrm{Ker}(y')\cup\mathrm{Ker}(y'')$, for $s<r$, and the equalities $y'z' = y''z'' = y$ lead to the desired relation: $t_j(1_n+y_{in}) = t_k(1_n+y_{in})$, for $(j,k)\in\mathrm{Ker}(y)$.

In addition, by Lemma 4.5, z' and z'' are (obviously surjective) simulations. $\square$

COROLLARY 4.6. Two equivalent minimal flowcharts are isomorphic. $\square$

COROLLARY 4.7. A flowchart F is minimal iff ($F \xrightarrow{s} F'$ or $F \xleftarrow{i} F'$) implies $F \approx F'$. $\square$

5. $Fl_{\Sigma,T}/_{\equiv}$ preserves strong iteration theory structure of T

THEOREM 5.1. If T is a strong iteration theory, then $Fl_{\Sigma,T}/_{\equiv}$ is a strong iteration theory.

<u>Proof.</u> By Theorem 3.8, $Fl_{\Sigma,T}/_{\equiv}$ is a paraiteration theory. It remains to verify that the axioms (I1), (I2), (I3p) and (I4s) in Section 1 hold in $Fl_{\Sigma,T}/_{\equiv}$.

For (I1) we have to show that

$$F\langle F^{+},1_n\rangle \equiv F^{+}, \text{ for } F: m \to m+n.$$

The left side is (A, ee), where $A = \langle l\,(\langle i\langle i^{+},1_{n+q}\rangle,1_n+0_q\rangle + 1_q)(1_n+s^q_q),$ $t(\langle i^{+},1_{n+q}\rangle(1_n+0_q+1_q)\rangle$. The following computation shows that $F\langle F^{+},1_n\rangle$ can be simulated via $\langle 1_e,1_e\rangle$ in F^{+}.

$$A(1_n+\langle 1_q,1_q\rangle) = (l\langle i^{+},1_{n+q}\rangle, t\langle i^{+},1_{n+q}\rangle) = (1_m+\langle 1_{e_{out}},1_{e_{out}}\rangle)\, l\langle i^{+},1_{n+q}\rangle.$$

This proves (I1).

The following two axioms hold even in $Fl_{\Sigma,T}$. For (I2) suppose that $F: m \to m+m+n$; then,

$$F^{++} = (l\langle i^{+},1_{m+n+q}\rangle\langle(i\langle i^{+},1_{m+n+q}\rangle)^{+},1_{n+q}\rangle,\ e) = (l\langle i^{++},i^{++},1_{n+q}\rangle, e)$$

$$= (\,l\,(\langle 1_m,1_m\rangle+1_{n+q})\,\langle(i(\langle 1_m,1_m\rangle+1_{n+q}))^{+},\ 1_{n+q}\rangle, e) = (F(\langle 1_m,1_m\rangle+1_n))^{+}.$$

For (I3) suppose that $F: m \to n+p$, $f \in T(n,m)$; then

$$f\,(F(f+1_p))^{+} = ((f+1_{e_{out}})\, l(f+1_{p+q})\langle(i(f+1_{p+q}))^{+},1_{p+q}\rangle,\ e)$$

$$= ((f+1_{e_{out}})\, l\langle(fi)^{+},1_{p+q}\rangle,\ e) = (f\,F)^{+}.$$

For the last axiom (I4s) we have to show that: if $F: m \to m+p$, $F': n \to n+p$ and the surjection $y: [m] \to [n]$ are such that $F(y+1_p) \equiv yF'$, then $F^{+} \equiv yF'^{+}$.

We shall use Theorem 4.1. Without loss of generality we can suppose that F' is a minimal flowchart. It follows that yF' is minimal too [suppose there is F'' with

$|e''| < |e'|$ and $(yF' \xrightarrow[z_1]{s} F''$ or $yF' \xleftarrow[z_2]{i} F'')$; using a $w : [n] \to [m]$ such that $wy = 1_n$ and Lemma 3.4.a it follows that $(F' = wyF' \xrightarrow[z_1]{s} wF''$ or $F' \xleftarrow[z_2]{i} wF'')$, but -cf. Corollary 4.7- this is contrary to the minimality of F']. By Theorem 4.1 the equivalence $F(y+1_p) \equiv yF'$ now can be replaced with two simulations, i.e. there exists F^1 such that

$$F(y+1_p) \xrightarrow[u]{s} F^1 \xleftarrow[v]{i} yF'.$$

The second simulation shows that $i^1 = yi'(1_{n+p}+v_{in})$; hence $F^1 = yF^2$, where $F^2 = (\langle i'(1_{n+p}+v_{in}), t^1\rangle, e^1)$. By using Lemma 3.4.a for $yF^2 \xleftarrow[v]{i} yF'$ and a w such that $wy = 1_n$, we obtain $F^2 = wyF^2 \xleftarrow[v]{i} wyF' = F'$; using Lemma 3.4.c this gives $F^{2+} \xleftarrow[v]{i} F'^{+}$.

Now (I4s) will be proved if we will show that

$$F(y+1_p) \xrightarrow[u]{s} yF^2 \text{ implies } F^{+} \xrightarrow[u]{s} yF^{2+}.$$

The hypothesis leads to the identity $I(y+1_p+u_{in}) = (y+u_{out})I^2$. Take the restriction of this identity to the inputs and apply (I4s) in T; it follows that $(i(1_{m+p}+u_{in}))^+ = yi^{2+}$. Using this fact, the desired implication easily follows, namely,

$$I\langle i^+, 1_{p+q}\rangle(1_p+u_{in}) = I\langle (i(1_{m+p}+u_{in}))^+, 1_p+u_{in}\rangle$$
$$= I(y+1_p+u_{in})\langle i^{2+}, 1_{p+q^2}\rangle = (1_m+u_{out})(y+1_{e^2_{out}})I^2\langle i^{2+}, 1_{p+q^2}\rangle.$$

The proof of the theorem is now completed. $\square$

6. The main result

For the begining we have to define the interpretation of a Σ-flowchart over T in an arbitrary strong iteration theory Q. The formula is known (for example [4]), and is similar with the Elgot representation of "normal descriptions" in [17].

Firstly, we have to interprete the variables using a rank-preserving function $\varphi_{\Sigma}: \Sigma \to Q$ (i.e. $\varphi_{\Sigma}(\sigma) \in Q(\sigma_{in}, \sigma_{out})$); secondly, the morphisms in T have to be interpreted using a strong iteration theory morphism $\varphi_T : T \to Q$ (namely, this is given by a family of functions $\varphi_T^{m,n} : T(m,n) \to Q(m,n)$ which preserves 1_n, 0_n, x_i^n, composition, tupling, and iteration); then the interpretation of $F = (\langle i,t\rangle, e)$ is

$$\varphi^{\#}(\langle i,t\rangle,e) = \varphi_T(i) \langle 1_n, (\varphi_{\Sigma}^{*}(e)\, \varphi_T(ts_n^q))^{\dagger}\rangle,$$

where $\varphi_{\Sigma}^{*} : \Sigma^{*} \to Q$ is the monoid extension of φ_{Σ}, considering Q as a monoid with the sum of morphisms.

Example 2.1 continued. Some interpretations for the flowchart in Example 2.1 are given in Table 2, varying φ_{Σ} and keeping fix $Q = T = \mathbb{P}\mathrm{fn}_{\omega^2}$ and $\varphi_T = 1_T$. $\square$

$\varphi_{\Sigma}(\sigma)$	the corresponding interpretation
$(1,(x,y)) \to (1,(x,y*x))$	$(1,(x,y)) \to (1,(1,x!))$
$(1,(x,y)) \to (1,(x,y+x))$	$(1,(x,y)) \to (1,(1, 1+x(x+1)/2))$
$(1,(x,y)) \to (1,(x+1,y))$	$(1,(x,y)) \to$ "if $x=0$ then $(1,(1,1))$ else undefined"

Table 2. Some interpretations of the unusual flowchart from Example 2.1 in its support theory.

Clearly, strong iteration theories with the above morphisms form a category. We can now state our main theorem.

THEOREM 6.1. $FI_{\Sigma,T}/_\equiv$ is the coproduct of T and the theory freely generated by Σ in the category of strong iteration theories.

Proof. The difficult part of the proof (namely, to show that $FI_{\Sigma,T}/_\equiv$ is a strong iteration theory) was shown in Theorem 5.1. It remains to prove the universality property. More exactly, we shall show that the disjoint union $\Sigma \amalg T$ can (injectively) be embedded in $FI_{\Sigma,T}/_\equiv$ by a rank-preserving function $I_\Sigma : \Sigma \to FI_{\Sigma,T}/_\equiv$ and a strong iteration theory morphism $I_T : T \to FI_{\Sigma,T}/_\equiv$ such that for every strong iteration theory Q, every rank-preserving function $\varphi_\Sigma : \Sigma \to Q$, and every strong iteration theory morphism $\varphi_T : T \to Q$, there exists a unique strong iteration theory morphism $\varphi^{\#} : FI_{\Sigma,T}/_\equiv \to Q$ such that $I_\Sigma \cdot \varphi^{\#} = \varphi_\Sigma$ and $I_T \cdot \varphi^{\#} = \varphi_T$.

$$\begin{tikzcd}
\Sigma \arrow[r, "I_\Sigma"] \arrow[dr, "\varphi_\Sigma"'] & FI_{\Sigma,T}/_\equiv \arrow[d, dashed, "\varphi^{\#}"'] & T \arrow[l, "I_T"'] \arrow[dl, "\varphi_T"] \\
 & Q &
\end{tikzcd}$$

In the interpretation of $(\langle i,t\rangle,e) : m \to n$ in Q we agree to omit the writing of φ_T and to replace $\varphi_\Sigma{}^*(e)\varphi_T(t)$ with a and as^q_n with b.

(1) At this point we will prove that two $\equiv$-equivalent flowcharts have the same interpretation in Q, hence $\varphi^{\#}$ is well-defined on $FI_{\Sigma,T}/_\equiv$. For this, it is enough to prove that $F \xrightarrow[y]{} F' : m \to n$ implies $\varphi^{\#}(F) = \varphi^{\#}(F')$. Since $\varphi^*_\Sigma(e)y_{out} = y_{in}\varphi^*_\Sigma(e')$ and $l(1_n+y_{in}) = (1_n+y_{out})l'$ it follows easily that $b(y_{in}+1_n) = y_{in}b'$. By Proposition 1.12, (I4) holds in Q; this leads to $b^+ = yb'^+$. Consequently,

$$\varphi^{\#}(F) = i\langle 1_n, b^+\rangle = i\langle 1_n, y_{in}b'^+\rangle = i'\langle 1_n, b'^+\rangle = \varphi^{\#}(F').$$

(2) Let $pr : FI_{\Sigma,T} \to FI_{\Sigma,T}/_\equiv$ denotes the canonical projection; then the embeddings I_Σ, I_T are: $I_\Sigma = I'_\Sigma \cdot pr$, where $I'_\Sigma : \Sigma \to FI_{\Sigma,T}$ is $I'_\Sigma(\sigma) = (s^{\sigma_{out}}_{\sigma_{in}}, \sigma)$,

and $I_T = I'_T \cdot pr$, where $I'_T : T \to Fl_{\Sigma,T}$ is $I'_T(f) = (f,\lambda)$. All the flowcharts in the range of $I_\Sigma \amalg I_T$ are nonisomorphic, minimal flowcharts, hence by Corollary 4.6 they are nonequivalent. Therefore $I_\Sigma \amalg I_T$ is really an embedding. In addition $I_T \cdot \varphi^{\#} = \varphi_T$, and $I_\Sigma \cdot \varphi^{\#} = \varphi_\Sigma$.

The remainder of the proof is similar to those from [9, 14] and is sketched here only for the sake of completeness.

(2.a) At this point we will show that $\varphi^{\#}$ is a strong iteration theory morphism, i.e. it preserves the operations.

For composition, take $F : m \to n$, $F' : n \to p$; then

$$\varphi^{\#}(F \cdot F') = i(i'+1_p)(1_p+s^q_{q'}) \langle 1_p, (a(i'+1_q)(1_p+s^q_{q'}), a'(1_p+0_q+1_{q'}) \rangle s_p^{q+q'})^+ \rangle .$$

Using Corollary 1.11.a the iterated part is $\langle b(1_q+i's_p^{q'}), b'(0_q+1_{q'+p}) \rangle^+ =$ $\langle b^+ i's_p^{q'}, 1_q+0_p \rangle \langle b'^+, 1_p \rangle$; hence

$$\varphi^{\#}(F \cdot F') = i(i'+1_q)(1_p+s^q_{q'}) \langle 1_p, b^+ i' s_q^{q'} \langle b'^+, 1_p \rangle, b'^+ \rangle$$

$$= i \langle i' \langle 1_p, b'^+ \rangle, b^+ i' \langle 1_p, b'^+ \rangle \rangle = \varphi^{\#}(F)\, \varphi^{\#}(F').$$

For tupling, take $F : m \to p$, $F' : n \to p$; using Corollary 1.11.b and the notations $x = 1_{p+q}+0_{q'}$, $x' = 1_p+0_q+1_{q'}$ we have

$$\varphi^{\#}(\langle F,F' \rangle) = \langle ix, i'x' \rangle \langle 1_p, (\langle ax, a'x' \rangle s_p^{q+q'})^+ \rangle$$

$$= \langle ix, i'x' \rangle \langle 1_p, b^+, b'^+ \rangle = \langle \varphi^{\#}(F), \varphi^{\#}(F') \rangle .$$

For iteration take $F : m \to m+n$; using (I1) and Corollary 1.11.c we have

$$\varphi^{\#}(F^+) = i \langle i^+, 1_{n+q} \rangle \langle 1_n, (a \langle i^+, 1_{n+q} \rangle s_n^q)^+ \rangle = (i \langle 1_{m+n}, b^+ \rangle)^+ = (\varphi^{\#}(F))^+ .$$

(2.b) At this point we will show that the extension $\varphi^{\#}$ is unique. Clearly, it is enough to show that an arbitrary $F = (\langle i,t \rangle, e) : m \to n$ can be represented as

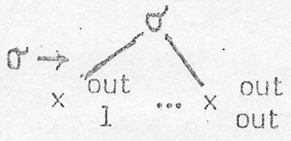
$$F = I_T(i) \langle 1_n, (I^*_\Sigma(e)\, I_T(ts_n^q))^+ \rangle .$$

To show this we remark that $(I^{*}_{\Sigma}(e)\, I_T(ts^q_n))^+ = (s_q^{e_{out}}(t+1_q)(s^q_n+1_q), e)^+ =$ $(\langle 0_n+1_q, t\rangle, e)$; hence the right part is equal to $(i,\lambda)(\langle 1_{n+q}, t\rangle, e) = (\langle i,t\rangle, e) = F$.

The proof of the theorem is now completed. $\square$

COROLLARY 6.2 (case of usual flowcharts). $Fl_{\Sigma,Pfn.}/_{\equiv}$ is the strong iteration theory freely generated by Σ. $\square$

COROLLARY 6.3. Suppose that $\sigma_{in} = 1$, $\forall \sigma \in \Sigma$; then two Σ-flowcharts over Pfn. are $\equiv$-equivalent iff they unfold the same tree.

Proof. It is known from [20] that the theory of rational Σ-trees in [23], denoted by R_Σ, is the iteration theory freely generated by Σ. Since R_Σ is also a strong iteration theory it is the strong iteration theory freely generated by Σ. Using Corollary 6.2 we check that R_Σ and $Fl_{\Sigma,Pfn.}/_{\equiv}$ are isomorphic theories; this says that the natural interpretation $\varphi^{\#} : Fl_{\Sigma,Pfn.}/_{\equiv} \to R_\Sigma$, which extend the function $\sigma \to$ 

and the unique morphism $\varphi_{Pfn.} : Pfn. \to R_\Sigma$ is an isomorphism. Now the corollary is proved using the observation that $\varphi^{\#}$ is the unfolding of flowcharts. $\square$

7. A comparison of different types of theories

A lack of the iterative theories in [17] consists in the definiton of iteration only for ideal morphisms. This was remedied in pointed iterative theories in [10] by extending the iteration to a totally defined operation. However, a pointed iterative theory still have to fulfil two very restrictive conditions: firstly, it must be an "ideal" theory (and $\mathbf{Pfn}_D$ is not ideal, when D has at least two elements); secondly, the Elgot iteration equation must have a unique solution for each ideal morphism (this is a very strong condition, indeed: it implies that (I4) holds if y is replaced by an arbitrary morphism, as far as f, g are assumed ideal).

All types of ordered theories, for example rational theories in [3], are not fitted since they are not pure algebraic objects.

All the above problems are overcome in strong iteration theories iteration thoeries, and theories with iterate: as a whole, they are pure algebraic objects and they have a total iteration which fulfils a few natural axioms. It remains to compare among these theories. The comparison will be done by studying the transformations of flowchart schemes which preserves interpretations in all theories of a given type.

Firstly, we look at the usual flowcharts. Theories with iterate are too weak (it seems that only the permutation of vertices respect the above mentionated condition). Iteration theories are good theories and satisfy the above condition for all transformations which respect the strong equivalence (in fact this was the reason for their consideration).

Secondly, we pass to the generalized flowcharts in $\mathbf{FI}_{\Sigma,T}$ and to the generalization of the strong equivalence in this context. We have the following

PROPOSITION 7.1. Suppose that $\Sigma(1,1) \neq \emptyset$. If strongly equivalent flowcharts in $Fl_{\Sigma,T}$ have the same interpretation in T, then (I4) holds in T.

Proof. Take f, g, y as in (I4) and $\sigma \in \Sigma(1,1)$. Clearly, the following is a simulation

$$F = (\langle 0_p + 1_m, fs^p_m \rangle, \sigma^m) \xrightarrow[y]{} (\langle 0_p + y, gs^p_n \rangle, \sigma^n) = F' .$$

The interpretations of F, F' in T when $\varphi_\Sigma(\sigma) = 1_1 \in T(1,1)$ and $\varphi_T = 1_T$ are f^+, respectively yg^+; hence the condition (I4) holds in T. $\square$

Consequently, the support theory must be a strong iteration theory. While this is clear, the us_fulness of iteration theories come from the following

Fact 7.2. Suppose that the support theory satisfies

(p) if $f : m \to m+p$, $g : n \to n+p$ and $y : [m] \to [n]$ are such that $f(y+1_p) = yg$, then there exist $h \in T(n, m+p)$ and $y_1, \ldots, y_m : [m] \to [m]$ with $y_i y = y$, $\forall\, i \in [m]$, such that $f = \langle x^m_1 yh(y_1 + 1_p), \ldots, x^m_m yh(y_m + 1_p) \rangle$ and $g = g(y + 1_p)$;

then two $\equiv$-equivalent flowcharts have the same interpretation in all iteration theories.

Hint for proof. Use (p) for a finer analysis than (1) in the proof of Theorem 6.1. $\square$

The above condition (p) holds when the support theory is (a "syntactical" theory) CT_Σ, R_Σ, or $Pfn_\cdot$, but does not holds when the support theory is (a "semantical" theory) Pfn_D with D having at least two elements.

8. Final remarks

(a) While at the theoretical level we have a calculus for the classes of strongly equivalent flowchart schemes, essentially similar to the calculus of polynomials, at the practical level the situation is much worse, namely we have not a good and powerful representation of finite partial functions in $\mathbf{Pfn}_{*}$ similar to the Arabian representation of natural numbers.

(b) When the algebraic structure for support theories in the deterministic case will be unanimous accepted, I propose to be called Elgot theory; moreover I propose strong iteration theories as a candidate to this notion.

Acknowledgements

I would like to thank V. E. Cazanescu for many useful discussions in several stages of this research. Also, I thank Gr. Arsene for the help in order to have a better presentation.

REFERENCES

1. ADJ(J. A. GOGUEN, J. W. THATCHER, E. G. WAGNER, AND J. B. WRIGHT), Initial algebra semantics and continuous algebras, J. Assoc. Comput. Mach. 24 (1977), 68-95.

2. ADJ(E. G. WAGNER, J. B. WRIGHT, J. A. GOGUEN, AND J. W. THATCHER), Some fundamentals of order algebraic semantics, in Lecture Notes in Computer Science 45, pp.153-168, Springer-Verlag, Berlin - New York, 1976.

3. ADJ(J. B. WRIGHT, J. W. THATCHER, E. G. WAGNER, AND J.A.GOGUEN), Rational algebraic theories and fixed-point solution, in "Proceedings 17th IEEE Symposium on Foundations of Computer Science, Huston Texas (1976)," pp.147-158.

4. ADJ(J. W. THATCHER, E. G. WAGNER, AND J. B. WRIGHT), Notes on algebraic fundamentals for theoretical computer science, in "Foundations of computer science III, Part 2: Language, logic, semantics," (J.W.de Bakker and J.van Leeuwen,Eds.), pp.83-164, Mathematical Centre Tracts 109, Amsterdam, 1979.

5. M. A. ARBIB, AND E. G. MANES, Partially additive categories and flow-diagram semantics, J. Algebra 62 (1980), 203-227.

6. J. W. BACKUS, Can programming be liberated from the von Neumann style? A functional style and its algebra of program, Comm. ACM 21 (1978), 613-641.

7. J. W. BACKUS, From function level semantics to program transformation and optimization, in "Mathematical Foundations of Software Development," Lecture Notes in Computer Science 185, pp.60-91, Springer-Verlag, Berlin - Heidelberg - New York - Tokio, 1985.

8. S. L. BLOOM, AND C. C. ELGOT, The existence and construction of free iterative theories, J. Comput. System Sci. 12 (1976), 305-318.

9. S. L. BLOOM, C. C. ELGOT, AND R. TINDELL, On the algebraic structure of rooted trees, J. Comput. System Sci. 16 (1978),

10. S. L. BLOOM, C. C. ELGOT, AND J. B. WRIGHT, Solutions of the iteration equations and extensions of the scalar iteration operation, SIAM J. Comput. 9 (1980), 25-45.

11. S. L. BLOOM, C. C. ELGOT, AND J. B. WRIGHT, Vector iteration in pointed iterative theories, SIAM J. Comput. 9 (1980), 525-540.

12. S. L. BLOOM, AND Z. ESIK, Axiomatizing schemes and their behaviours, to appear in J. Comput. System Sci.

13. V. E. CAZANESCU, On context-free trees, to appear in Theor. Comput. Sci. (1985).

14. V. E. CAZANESCU, AND C. UNGUREANU, "Again on advice on structuring compilers and proving them correct", INCREST Preprint Series in Mathematics No.75/1982.

15. B. COURCELLE, Fundamental properties of infinite trees, Theor. Comput. Sci. 25 (1983), 95-169.

16. C. C. ELGOT, Algebraic theories and program schemes, in "Semantic of algoritmic languages," (E. Engeler, Ed.), Lecture Notes in Mathematics 188, Springer-Verlag, 1971.

17. C. C. ELGOT, Monadic computation and iterative algebraic theories, in "Logic Colloquium," (H. E. Rose and J. C. Shepherdson, Eds.), pp.175-230, North-Holland, Amsterdam, 1975.

18. C. C. ELGOT, Structuted programming with and without GO TO statemants, IEEE Trans. Soft. Eng. SE-2 (1976), 41-53. Erratum an Corrigendum, idem : September 1976.

19. C. C. ELGOT, AND J. C. SHEPHERDSON, "A semantically meaningful characterization of reducible flowchart schemes," IBM Research Report RC 6656, July 1977.

20. Z. ESIK, Identities in iterative and rational theories, Comput. Linguistic Comput. Language 14 (1980), 183-207.

21. Z. ESIK, Algebras of iteration theories, J. Comput. System Sci. 27 (1983), 291-303.

22. Z.ESIK, "Independence of the equational axioms for iteration theories", draft paper, 1984.

23. S.GINALI, Regular trees and free iterative theory, J. Comput. System Sci. 18(1979), 228-242.

24. S. GREIBACH, Theory of program structures: schemes, semantics, verification, Srpinger-Verlag, Berlin - Heidelberg - New York, 1975.

25. D. E. KNUTH, AND R. W. FLOYD, Notes on avoiding "go to" statements, Inform. Proc. Letters 1 (1971), 23-31.

26. S. RAO KOSARAJU, Analysis of structured programs, J. Comput. System Sci. 9 (1974), 232-255.

27. V. E. KOTOV, Introduction to the theory of program schemes, (russian), Nauka, Novosibirsk, 1978.

28. V. E. KOTOV, Petri nets, (russian), Nauka, Moskva, 1984.

29. F. W. LAWVERE, Functorial semantics of algebraic theories, Proccedings National Academy of Science 50 (1963), 869-872.

30. R. J. LORENTZ, AND D. B. BENSON, Deterministic and nondeterministic flowchart interpretations, J. Comput. System. Sci. 27 (1983), 400-433.

31. E. G. MANES, Algebraic theories, Springer-Verlag, Berlin - New York, 1976.

32. Z. MANNA, Mathematical theory of computation, McGraw-Hill, New York, 1976.

33. Gh. STEFANESCU, "On flowchart theories (I)," INCREST Preprint Series in Mathematics, No.39/1984 ; "A completion of "On flowchart theories (I),"" idem : No.7/1983.

34. Gh. ŞTEFĂNESCU, "On flowchart theories II (nondeterministic case)," INCREST Preprint Series in Mathematics, No.32/1985.

35. J. E. STOY, Denotational semantics: the Scott-Strachey approach to programming language theory, The MIT Press, Cambridge - Massachusetts - London, 1977.

36. D. R. TROEGER, Metric iteration theories, Fund. Informaticae 5 (1982), 187-216.