Version 3
may 74

LECTURE NOTES

"ON THE INTERPRETATION OF PROGRAM SCHEMES: AN ALGEBRAIC APPROACH"

M. NIVAT

(IRIA & PARIS VII)

by

M. M. FOKKINGA

(T.H. DELFT)

28-3-1974

This manuscript is an elaboration of the notes made during the lectures of M. Nivat at the "Advanced Course on Semantics of Programming Languages" (Saarbrücken, Germany, febr 18 - march 1, 1974)

We would like to thank H. Goeman (R.U. Leiden) for stimulating discussions and the Netherlands Organization for Advancement of Pure Research (Z.W.O.) for financial support for attending the Course.

0. CONTENTS

1 INTRODUCTION

We will define the semantics of a recursive program scheme in three ways and show their relations (equivalence, inclusion). Thereafter we will discuss the decidability of the equivalence for a restricted class of recursive program schemes.

First of all we derive, by pure syntactic manipulations, some properties which are meaningfull under all the semantics of rec. pr. schemes. Thus these results need not be proved more than once and they are very strong because they are valid without any specification of the meaning of the symbols

The three different definitions of a semantics of rec. pr. schemes are the so called

- magma semantics, $Val_I \Sigma$: the common value of the interpretation I of the set of sequences of symbols produced by the rec pr. scheme Σ, considered as a rewriting system
- operational sem., $Comp_I \Sigma$: input-output behaviour defined by means of stepwise evaluation in the domain D_I of interpretation I
- fixed point sem., $Fix_I \Sigma$: least fixed point of a mapping $\tilde{\Sigma}_I$, induced by the scheme Σ, in the space $\tilde{D}_I$ of functions on the domain D_I of ~~function~~ interpretation I.

The sequences of symbols which will be considered frequently consist of well-known so called well formed terms, inductively built up from variable symbols and "known function"- and "unknown function" symbols, with parenthesis and comma's. Such a collection of terms is called the free F-magma generated by V, when F the set of all function symbols, V the set of variable symbols, and is denoted by $M(F, V)$.

2 BASIC DEFINITIONS, TERMINOLOGY, PROPERTIES

2.1. Free F-magma generated by V, M(F,V)

(21.1) First of all a precise description of the sequences of symbols we deal with.
Let F be a set of function symbols
Let V be a set of variable symbols
Let $\rho(f)$ for each $f \in F$ denote the arity of f (the arity of f is to be considered as a primitive notion), $\rho(f) \geq 1$ for all $f \in F$!!
The free F-magma generated by V, $M(F,V)$, is the least set of finite sequences of symbols
which – contains the sequence of v alone, for each $v \in V$
– contains the sequence $f(m_1, \dots, m_{\rho(f)})$ whenever $f \in F$ and m_1 and ... and $m_{\rho(f)}$ are in $M(F,V)$.

Remarks.
1. Each elt of $M(F,V)$ consist of symbols (and , and) and function symbols from F and variable symbols from V.
2. $M(F,V)$ can also be considered as the c.f. language generated by the c.f. grammar $\xi = \sum_{f \in F} f(\underbrace{\xi, \dots, \xi}_{\rho(f)}) + \sum_{v \in V} v$,
or in another notation
by the cf grammar

$$\xi \to \underbrace{f'(\underbrace{\xi, \dots, \xi}_{\rho(f')}) \mid f''(\underbrace{\xi, \dots, \xi}_{\rho(f'')}) \mid \dots \mid f'''(\underbrace{\xi, \dots, \xi}_{\rho(f''')})}_{\text{all } f \in F} \mid \underbrace{v' \mid v'' \mid \dots \mid v'''}_{\text{all } v \in V}$$

(21.2) Intermezzo.
We could have been somewhat more general by defining:
a F-magma is an ordered pair $\langle E, \sigma \rangle$
where E is a set, the domain, and
σ is a collection mappings, for each $f \in F$ one such mapping $\sigma(f) : E^{\rho(f)} \to E$

Hence a F-magma yields an interpretation for the system of function symbols F, viz. the domain of interpretation is E and σ is the association of function symbols in F with mappings in the domain.
A morphism φ from F-magma $\langle E, \sigma \rangle$ to F-magma $\langle E', \sigma' \rangle$ is a

mapping $\varphi : E \to E'$ with

$$\varphi(\sigma(f))(m_1, \ldots, m_{\rho(f)}) = \sigma'(f)(\varphi(m_1), \ldots, \varphi(m_{\rho(f)})) \quad \text{for each } f.$$

Now, we can state that
for each set V there exists a unique F-magma $\mathcal{M}$
such that V is contained in the domain of $\mathcal{M}$ and
for each F-magma $\langle E, \sigma \rangle$ and each mapping $\varphi_1 : V \to E$
there is an extension of φ_1 into a homomorphism $\varphi : \mathcal{M} \to \langle E, \sigma \rangle$
Because of this propertiy $\mathcal{M}$ is called free, as usual, and
this unique free F-magma generated by V is given by the pair
$\langle M(F,V), \sigma \rangle$
where $M(F,V)$ is defined as above in (21.1), and for each $f \in F$
$\sigma(f) : M(F,V)^{\rho(f)} \to M(F,V)$ is defined by: for all $m_i \in M(F,V)$
$\sigma(f)(m_1, \ldots, m_{\rho(f)}) = f(m_1, \ldots, m_{\rho(f)})$
Note that in the r.h.s. of the last equation there is one element of $M(F,V)$ which is a sequence of symbols to which f, the parenthesis (and), the comma's, belong and also the sequences of symbols named by $m_1, \ldots, m_{\rho(f)}$. But in the l.h.s. the parentheses and the commas are part of a notation for function application of $\sigma(f)$ on its arguments m_1 and ... and $m_{\rho(f)}$.
By identifying $M(F,V)$ with $\langle M(F,V), \sigma \rangle$ definition (21.1) is justified.

2.2 Substitution

(2.2.1) Definition of factors

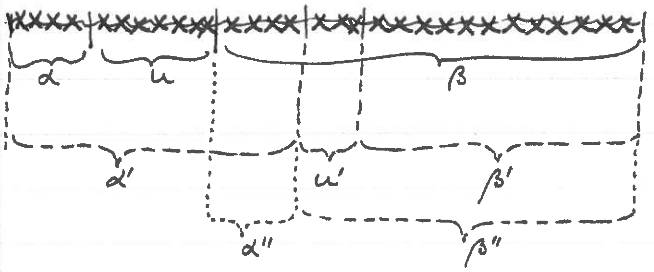
The elements of $M(F,V)$ are sequences of symbols and we can look at subsequences of them. In relation to substitution processes the following notions are of interest.

<u>An factor of $m \in M(F,V)$</u> is a threetuple $(\alpha; u; \beta)$
s.th. $m = \alpha . u . \beta$ and
$u \in M(F,V)$

Here $\alpha . u . \beta$ stands for the concatenation of sequences of symbols, also called product of α, u, β.

Factors $(\alpha; u; \beta)$ and $(\alpha'; u'; \beta')$ are disjoint
iff there exist α'', β'' such that
either $\alpha = \alpha' . u' . \alpha'' \wedge \beta' = \alpha'' . \beta''$ or $\alpha' = \alpha . u . \alpha'' \wedge \beta = \alpha'' . \beta''$,

informally, iff u and u' are disjoint substrings of m:



$\leftarrow m \in M(F,V)$

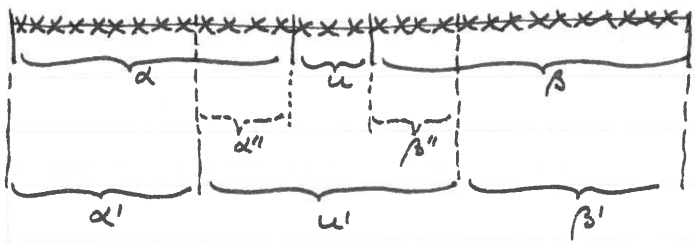
"$(\alpha;u;\beta)$ and $(\alpha';u';\beta')$ are disjoint."

Factor $(\alpha;u;\beta)$ of m is contained in $(\alpha';u';\beta')$, $(\alpha;u;\beta) \subseteq (\alpha';u';\beta')$

iff there exist α'', β'' such that

$\alpha = \alpha'.\alpha'' \wedge \beta = \beta''.\beta' \wedge u = \alpha''.u.\beta''$,

informally, iff u is a substring of u':



$\leftarrow m \in M(F,V)$

(2.2.2) Property of factors.

If $(\alpha;u;\beta)$ and $(\alpha';u';\beta')$ are factors of m, then one of (i), (ii), (iii) holds:

(i) they are disjoint

(ii) $(\alpha;u;\beta) \subseteq (\alpha';u';\beta')$

(iii) $(\alpha;u;\beta) \supseteq (\alpha';u';\beta')$

proof: By induction on the complexity; this is a well known property.

We denote substitution of m_i for n_i for $i = 1,\dots,n$ and $m_i, n_i \in M(F,V)$ in f by $f(m_1/n_1, \dots, m_n/n_n)$. It is strongly emphasized that the parentheses and commas displayed in $f(m_1/n_1, \dots, m_n/n_n)$ do not belong to the resulting expression in $M(F,V)$.

2.3 Rewriting systems

In the sequel we let $\Phi = \{\varphi_1, \dots, \varphi_N\}$ be a finite set of "unknown function" symbols, and F be a possibly infinite set of "known function" symbols. Moreover, we assume V to be countable, and fix the enumeration $v_1, v_2, \dots$ throughout the paper. Please note the difference between $M(F,V)$ and $M(F \cup \Phi, V)$.

(23.1) Definitions.

A rewriting system Σ over $M(F,V)$ is a collection equations

$$\begin{cases} \varphi_i(v_1,\dots,v_{\rho(\varphi_i)}) = \tau_i & \text{in which } \tau_i \in M(F\cup\Phi, \{v_1,\dots,v_{\rho(\varphi_i)}\}) \\ i=1,\dots,N \end{cases}$$

Example.

Let the rewriting system be given by

$$\Sigma = \begin{cases} \varphi_1(x,y) = h(x,y,h(y,x,\varphi_1(y,\varphi_2(x,y)))) \\ \varphi_2(x,y) = g(x,y,\varphi_2(f(x,y),x),x) \quad *) \end{cases}$$

then

$\Phi = \{\varphi_1,\varphi_2\}$ and $N=2$, $\rho(\varphi_1)=\rho(\varphi_2)=2$ and $V=\{x,y,\dots\}$. Furthermore f, g and h belong to F and $\rho(f)=2$, $\rho(g)=4$, $\rho(h)=3$.

For $f, f' \in M(F\cup\Phi, V)$ we say f derives immediately f' in Σ, $f \underset{\Sigma}{\Rightarrow} f'$,

iff there is a factor $(\alpha;u;\beta)$ of f such that

$u = \varphi_j(m_1,\dots,m_{\rho(\varphi_j)})$ and

$f' = \alpha\cdot\tau_j(m_1/v_1,\dots,m_{\rho(\varphi_j)}/v_{\rho(\varphi_j)})\cdot\beta$

Informally, iff f' can be obtained from f by a rewriting of a subterm according the j-th rewriting equation in which the variables $v_1,\dots,v_{\rho(\varphi_j)}$ are replaced by the current arguments $m_1,\dots,m_{\rho(\varphi_j)}$.

Similarly we say f derives f' in Σ, $f \overset{*}{\underset{\Sigma}{\Rightarrow}} f'$,

iff there are $f_1,\dots,f_{k+1}$ such that

$f=f_1$ and for $h=1\dots k$ $f_h \underset{\Sigma}{\Rightarrow} f_{h+1}$, and finally $f_{k+1}=f'$,

i.e. $\overset{*}{\underset{\Sigma}{\Rightarrow}}$ is the reflexive transitive slosure of $\underset{\Sigma}{\Rightarrow}$.

A derivation of f into f' in Σ is a $k{+}1$-Tuple $\langle f_1,\dots,f_{k+1}\rangle$ such that $f=f_1$ and for $h=1\dots k$ $f_h \underset{\Sigma}{\Rightarrow} f_{h+1}$, and finally $f_{k+1}=f'$.

By convention we denote by $(\alpha_h;u_h;\beta_h)$ the factor to be replaced in f_h in order to obtain f_{h+1}.

A derivation is called leftmost

iff for all h $|\alpha_h| \le |\alpha_{h+1}|$, where $|\alpha|$ = number of symbols in α.

(23.2) Theorem on leftmost derivations.

$f \overset{*}{\underset{\Sigma}{\Rightarrow}} f'$ if and only if there is a leftmost derivation of f into f' in Σ.

proof:

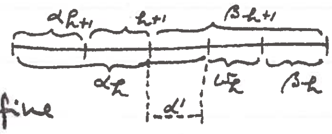
(Essentially the proof of M. Fischer given for macro-grammars)

---

*) which can be interpreted as computing the greatest common divisor, see ex (31.1).

Let $d = \langle f_1, \ldots, f_{k+1} \rangle$ be a derivation of f into f' in Σ.
Let $\pi(d) = \text{card}\{h \in \{1,\ldots,k\} : |\alpha_h| > |\alpha_{h+1}|\}$ = number of <u>not</u> l.most d. steps
We will give an algorithm such that a derivation d' of f into f' with $\pi(d') > 0$ is transformed into a derivation d'' of f into f' with $\pi(d'') < \pi(d')$. Thus by induction d can be transformed into a derivation d''' of f into f' with $\pi(d''') = 0$, i.e. d''' is leftmost.
Now,
let h be the smallest number s.th. $|\alpha_h| > |\alpha_{h+1}|$, then
$f_h = \alpha_h \cdot u_h \cdot \beta_h$
$f_{h+1} = \alpha_h \cdot w_h \cdot \beta_h$ where $u_h = \varphi_j(m_1, \ldots)$ and $w_h = \tau_j(m_1/v_1, \ldots)$
and looking at the next derivation step we can set
$f_{h+1} = \alpha_{h+1} \cdot u_{h+1} \cdot \beta_{h+1}$.
According to property (22.2) the factors $(\alpha_h; w_h; \beta_h)$ and $(\alpha_{h+1}; u_{h+1}; \beta_{h+1})$ either are contained or are disjoint.

<u>If disjoint</u>

then due to $|\alpha_{h+1}| < |\alpha_h|$ we have 

, so

$f_{h+1} = \alpha_{h+1} \cdot u_{h+1} \cdot \alpha' \cdot w_h \cdot \beta_h$, and we define
$f'_{h+1} = \alpha_{h+1} \cdot w_{h+1} \cdot \alpha' \cdot u_h \cdot \beta_h$, where $f_{h+2} = \alpha_{h+1} \cdot w_{h+1} \cdot \beta_{h+1} \Leftarrow_{\Sigma} \alpha_{h+1} \cdot u_{h+1} \cdot \beta_{h+1}$

Now, it is easy to see

$f_1 \overset{*}{\underset{\Sigma}{\Rightarrow}} f_h \underset{\Sigma}{\Rightarrow} f'_{h+1} \underset{\Sigma}{\Rightarrow} f_{h+2} \overset{*}{\underset{\Sigma}{\Rightarrow}} f_{k+1}$ and more over
$\pi(\langle f_1 \ldots f_h \; f'_{h+1} \; f_{h+2} \ldots f_{k+1} \rangle) < \pi(d)$

<u>If contained</u>

then due to $|\alpha_{h+1}| < |\alpha_h|$ we have $(\alpha_h; w_h; \beta_h) \subsetneq (\alpha_{h+1}; u_{h+1}; \beta_{h+1})$
and because $u_{h+1} \neq w_h$ we have (loosely formulated)
that w_h is contained in one of the factors of $u_{h+1} = \varphi_j(m_1, \ldots, m_{\rho(\varphi_j)})$, say in m_e, so $m_e = \bar{\alpha}\, w_h\, \bar{\beta}$.
Denote by $m'' = \bar{\alpha}\, u_h\, \bar{\beta}$, then we have

$f_{h+1} = \alpha_{h+1} \cdot u_{h+1} \cdot \beta_{h+1}$
$\quad = \alpha_{h+1} \cdot \varphi_j(m_1, \ldots m_{e-1}, m_e, m_{e+1} \ldots m_{\rho(\varphi_j)}) \cdot \beta_{h+1}$
$f_{h+2} = \alpha_{h+1} \cdot \tau_j(m_1/v_1, \ldots m_{e-1}/v_{e-1}, m_e/v_e, m_{e+1}/v_{e+1} \ldots\ ) \cdot \beta_{h+1}$
$f_h = \alpha_{h+1} \cdot \varphi_j(m_1, \ldots m_{e-1}, m'', m_{e+1}, \ldots m_{\rho(\varphi_j)}) \cdot \beta_{h+1}$

Define

$f'_h = \alpha_{h+1} \cdot \tau_j(m_1/v_1, \ldots m_{e-1}/v_{e-1}, m''/v_e, m_{e+1}/v_{e+1} \ldots\ ) \cdot \beta_{h+1}$

Then clearly

$f_h \underset{\Sigma}{\rightarrow} f'_{h+1}$ in one leftmost step, and to do $f'_h \underset{\Sigma}{\overset{*}{\Rightarrow}} f_{h+2}$ in a leftmost way we have to substitute m_e for m'' in each occurrence of m'' coming from an occurrence of v_e in τ_i. This indeed can be done in a leftmost way. Hence

$$\pi(f_1 \underset{\Sigma}{\overset{*}{\Rightarrow}} f_h \underset{\Sigma}{\rightarrow} f'_{h+1} \underset{\Sigma}{\overset{*}{\Rightarrow}} f_{h+2} \underset{\Sigma}{\overset{*}{\Rightarrow}} f_{k+1}) < \pi(f \underset{\Sigma}{\overset{*}{\Rightarrow}} f_h \underset{\Sigma}{\rightarrow} f_{h+1} \underset{\Sigma}{\rightarrow} f_{h+2} \underset{\Sigma}{\rightarrow} f')$$

end of the proof.

(23.3) Remark.

Let $d = \langle f_1 \cdots f_{k+1} \rangle$ be a leftmost derivation of f into $f' \in M(F,V)$, then it is easy to check that each α_h does not contain any $\varphi \in \Phi$. Call a factor $(\alpha ; u ; \beta)$ *replacable* iff $u = \varphi_j(\cdots)$ with $\varphi_j \in \Phi$. Call a replacable factor *maximal* iff it is not contained in an other one. Then in every leftmost derivation of $f \underset{\Sigma}{\overset{*}{\Rightarrow}} f' \in M(F,V)$, each factor $(\alpha_k : u_k ; \beta_k)$ is a maximal replacable factor. Hence leftmost often is called leftmost-outermost.

2.4 The symbol Ω, schematic variants of rewriting systems

(24.1) Definitions.

In the sequel we let V be $\{v_1, v_2, \ldots\} \cup \{\Omega\}$, thus Ω being a distinguished elt of V, not appearing in the enumeration $v_1, v_2, \ldots$!!

The schematic variant $\bar{\Sigma}$ of a rewriting system Σ is defined by $\bar{\Sigma}$: $\begin{cases} \varphi_i(v_1, \ldots, v_{\rho(\varphi_i)}) = \tau_i + \Omega \\ i = 1, \ldots, N \end{cases}$

where for $i = 1, \ldots, N$ $\varphi_i(v_1, \ldots, v_{\rho(\varphi_i)}) = \tau_i$ is in Σ.

We adjust the definition of f derives (immediately) f' in $\bar{\Sigma}$ as follows: iff $(\alpha ; u ; \beta)$ is a factor of f, and $u = \varphi_j(m_1, \ldots m_{\rho(\varphi_j)})$

then both $f' = \alpha \cdot \tau_j(m_1/v_1, \ldots, m_{\rho(\varphi_j)}/v_{\rho(\varphi_j)}) \cdot \beta$

and $f'' = \alpha \cdot \Omega \cdot \beta$

derive immediately from f in $\bar{\Sigma}$.

(24.2) The ordering relation $\prec$.

Due to the introduction of the symbol Ω with intended interpretation "the unspecified sequence of symbols", it is natural to express the interpretation of "being less specified than" by an partial ordering relation $\prec$.
We will define it and investigate its properties.

<u>The magma ordering</u>, $\prec$, on $M(F \cup \Phi, V)$ is defined

as the coarsest relation
compatible with the magma structure and
such that $\Omega \prec v$ for all $v \in V$,

i.e. for $m, m' \in M(F \cup \Phi, V)$, we have $m \prec m'$

iff $\exists\, \alpha_1, \alpha_2, \ldots, \alpha_{p+1}$ and $w_1, w_2, \ldots, w_p$, all $w_i \in M(F \cup \Phi, V)$ s.th.

$m = \alpha_1 \cdot \Omega \cdot \alpha_2 \cdot \Omega \cdot \ldots \cdot \Omega \cdot \alpha_{p+1}$

$m' = \alpha_1 \cdot w_1 \cdot \alpha_2 \cdot w_2 \cdot \ldots \cdot w_p \cdot \alpha_{p+1}$

Note that this defines $v' \prec v''$ for $v', v'' \in V$ iff $v' = \Omega$ or $v' = v''$.

(24.3) Theorem on the lattice structure of $\prec$.

<u>The magma-language</u> of f w.r.t. system Σ is defined as

$$L(\Sigma, f) = \{ f' \in M(F,V) : f \overset{*}{\underset{\Sigma}{\Rightarrow}} f' \} ,$$

for $f \in M(F \cup \Phi, V)$ and Σ a rewr. system or schematic variant.

<u>Theorem.</u>

The restriction of $\prec$ to $L(\Sigma, f)$ is a lattice order, i.e.
if $f \overset{*}{\underset{\Sigma}{\Rightarrow}} f_1$ and $f \overset{*}{\underset{\Sigma}{\Rightarrow}} f_2$ then there exist $f_3, f_4 \in L(\Sigma, f)$
s.th. a) both $f_3 \prec f_1$ and $f_3 \prec f_2$

b) both $f_1 \prec f_4$ and $f_2 \prec f_4$

and c) if for some $f_3' \in L(\Sigma, f)$ both $f_3' \prec f_1$ and $f_3' \prec f_2$ then $f_3' \prec f_3$

d) if for some $f_4' \in L(\Sigma, f)$ both $f_1 \prec f_4'$ and $f_2 \prec f_4'$ then $f_4 \prec f_4'$.

proof:

(Only part (b) is needed in the sequel). We only prove part (a) and (b) and leave (c) and (d) as an exercise.

Informally, look at leftmost derivations d_1, d_2 of f into f_1, f_2 and take for the derivations d_3, d_4 of f into f_3, f_4 the corresponding rewriting steps, if they are equal, and otherwise in d_3 the

rewriting step into Ω and in d_4 the rewriting into not-Ω.
Formally, by induction to the sum of lengths of leftmost derivations $d_1 = \langle g_1, \dots, g_{k+1} \rangle$ and $d_2 = \langle h_1, \dots, h_{L+1} \rangle$ for f into f_1 and f_2 respectively, in Σ.

If $k+L=0$

then $f = f_1 = f_2 \in M(F,V)$, so we can take $f_3 = f_4$ to be $f_1 = f_2 = f$.

If $k+L>0$

then $f = g_1 = \alpha_1 \cdot u_1 \cdot \beta_1$ with $g_2 = \alpha_1 \cdot w_1 \cdot \beta_1$ and $g_{k+1} = f_1 \in M(F,V)$
$f = h_1 = \alpha'_1 \cdot u'_1 \cdot \beta'_1$ with $h_2 = \alpha'_1 \cdot w'_2 \cdot \beta'_1$ and $h_{L+1} = f_2 \in M(F,V)$

Now,

if $(\alpha_1\, u_1\, \beta_1)$ and $(\alpha'_1\, u'_1\, \beta'_1)$ are disjoint, say $\alpha_1 = \alpha'_1 u'_1 \alpha''$
then α_1 contains a $\varphi \in \Phi$ and due to the leftmost property f_1 would contain that $\varphi \in \Phi$: contradiction ↯.

and

if $(\alpha'_1 ; u'_1 : \beta'_1)$ is proper contained in $(\alpha_1 ; u_1\, \beta_1)$, say $\alpha_1 = \alpha'_1 \alpha''$
then similarly α'' hence α'_1 contains a $\varphi \in \Phi$. ↯.

Hence

$(\alpha_1\, u_1\, \beta_1)$ equals $(\alpha'_1\, u'_1\, \beta'_1)$, so that we can say
$\alpha_1 = \alpha'_1 = \alpha$, $\beta_1 = \beta'_1 = \beta$ and $u_1 = u'_1 = \varphi_j(m_1, \dots, m_{\rho(\varphi_j)}) = u$.

Define $w = \tau_j(m_1/v_1, \dots, m_{\rho(\varphi_j)}/v_{\rho(\varphi_j)})$, and now there are four cases for g_2 and h_2:

(i) $\begin{cases} g_2 = \alpha \cdot \Omega \cdot \beta \\ h_2 = \alpha \cdot \Omega \cdot \beta \end{cases}$ (ii) $\begin{cases} g_2 = \alpha \cdot w \cdot \beta \\ h_2 = \alpha \cdot \Omega \cdot \beta \end{cases}$ (iii) $\begin{cases} g_2 = \alpha \cdot \Omega \cdot \beta \\ h_2 = \alpha \cdot w \cdot \beta \end{cases}$ (iv) $\begin{cases} g_2 = \alpha \cdot w \cdot \beta \\ h_2 = \alpha \cdot w \cdot \beta \end{cases}$

so that, due to the leftmost property, in each of the cases

$$\begin{cases} f_1 = \alpha \cdot w_1 \cdot \beta_1 & \text{with either } w_1 = \Omega \text{ or } w \overset{*}{\underset{\Sigma}{\Rightarrow}} w_1, \text{ and } \beta \overset{*}{\underset{\Sigma}{\Rightarrow}} \beta_1 \in M(F,V) \\ f_2 = \alpha \cdot w_2 \cdot \beta_2 & \text{with either } w_2 = \Omega \text{ or } w \overset{*}{\Rightarrow} w_2 \text{ and } \beta \overset{*}{\Rightarrow} \beta_2 \in M(F,V) \end{cases}$$

By using induction hypothesis, there exist w_3, w_4 and β_3, β_4:

(i) $\begin{cases} w_3 := \Omega \\ w_4 := \Omega \end{cases}$ (ii) $\begin{cases} w_3 := \Omega \\ w_4 := w_1 \end{cases}$ (iii) $\begin{cases} w_3 = \Omega \\ w_4 := w_2 \end{cases}$ (iv) $\begin{cases} w_3 \\ w_4 \end{cases}$ (by ind.)

$w_3 < w_1, w_2 < w_4$ $\quad$ $w_3 < w_1, w_2 < w_4$ $\quad$ $w_3 < w_1, w_2 < w_4$ $\quad$ $w_3 < w_1, w_2 < w_4$

$\beta_3 < \beta_1, \beta_2 < \beta_4$ $\quad$ $\beta_1 < \beta_1, \beta_2 < \beta_4$ $\quad$ $\beta_3 < \beta_2, \beta_2 < \beta_4$ $\quad$ $\beta_3 < \beta_1, \beta_2 < \beta_4$

Therefore, we can take
$f_3 = \alpha \cdot w_3 \cdot \beta_3$ and $f_4 = \alpha \cdot w_4 \cdot \beta_4$,
and clearly $f \overset{*}{\Rightarrow} f_3, f_4$ in Σ, and $f_3 < f_1, f_2 < f_4$.

end of proof.

(24.4) Kleene's sequences.
We now give the interesting notion of a Kleene's sequence of f w.r.t. Σ, being defined as

an increasing infinite sequence $\langle f^{(i)} \rangle_{i=0}^{\infty}$ of $f^{(i)} \in L(\Sigma, f)$ which majorizes all elements in $L(\Sigma, f)$,

i.e. $f^{(0)} \prec f^{(1)} \prec \dots \prec f^{(k)} \prec \dots$ where $f^{(i)} \in L(\Sigma, f)$, and for each $f' \in L(\Sigma, f)$ there is an k s.th. $f' \prec f^{(k)}$.

(24.5) Strong and weak derivations.
For the construction of Kleene's sequences and for other purposes, we need the following definitions.

$f \underset{\Sigma}{\Rightarrow} f'$ is a strong derivation step, $f \underset{\Sigma}{\Rightarrow} f'$, iff

$$f = \alpha \cdot \varphi_j(m_1 \dots m_{\rho(\varphi_j)}) \cdot \beta \quad \text{and} \quad f' = \alpha \cdot \tau_j(m_1/v_1, \dots, m_{\rho(\varphi_j)}/v_{\rho(\varphi_j)}) \cdot \beta$$

$f \underset{\Sigma}{\Rightarrow} f'$ is a weak derivation step, $f \underset{\Sigma}{\not\Rightarrow} f'$, iff

$$f = \alpha \cdot \varphi_j(m_1 \dots m_{\rho(\varphi_j)}) \cdot \beta \quad \text{and} \quad f' = \alpha \cdot \Omega \cdot \beta$$

With the obvious extensions for $\overset{*}{\Rightarrow}$ and $\overset{*}{\not\Rightarrow}$.

We will give the construction and the proof of the existence of Kleene's sequences after six lemma's, establishing among others that in derivations of $f \overset{*}{\Rightarrow} f'$ the strong derivation steps can be done before the weak ones (lemma 6).

Let $S : M(F \cup \Phi, V) \to M(F \cup \Phi, V)$ be a mapping, inductively defined as: $S(v) = v$

$$S(f(m_1, \dots, m_{\rho(f)})) = f(S(m_1), \dots, S(m_{\rho(f)}))$$
$$S(\varphi_j(m_1, \dots, m_{\rho(\varphi_j)})) = \tau_j(S(m_1)/v_1, \dots, S(m_{\rho(\varphi_j)})/v_{\rho(\varphi_j)})$$

thus performing at once all possible strong derivation steps simultaneously.

Let $W : M(F \cup \Phi, V) \to M(F, V)$ be a mapping, ind. defined by

$$W(v) = v$$
$$W(f(m_1, \dots, m_{\rho(f)})) = f(W(m_1), \dots, W(m_{\rho(f)}))$$
$$W(\varphi_j(m_1, \dots, m_{\rho(\varphi_j)})) = \Omega$$

thus performing at once all possible weak derivation steps simultaneously, by simply replacing all replacable factors by Ω.

lemma 1. $f \Rightarrow f'$ implies $f' \overset{*}{\Rightarrow} S(f)$

proof by induction on the number of $f^{\div}$ symbols in f.

lemma 2. $f \Rightarrow f'$ implies $S(f) \overset{*}{\Rightarrow} S(f')$

proof by induction on the number of $f^{\div}$ symbols in f.

lemma 3. $f \overset{*}{\Rightarrow} f'$ implies $S(f) \overset{*}{\Rightarrow} S(f')$

proof by induction on the length of derivation.

lemma 4. $f \overset{k}{\Rightarrow} f'$ implies $f' \overset{*}{\Rightarrow} S^k(f)$

proof by induction on the length of derivation.

lemma 5. $f \overset{*}{\nrightarrow} f_2 \Rightarrow f_3$ implies $f \Rightarrow f_4 \overset{*}{\nrightarrow} f_3$ for some f_4

proof: by induction on the length of the derivation $f \overset{*}{\nrightarrow} f_2$

length $= 0$: Trivial

length $= 1$: we have to make the diagram

$$\begin{tikzcd} f \arrow[r, Rightarrow] \arrow[d] \arrow[dr, phantom, "\circlearrowleft"] & f_4 \arrow[d, "*"] \\ f_2 \arrow[r, Rightarrow] & f_3 \end{tikzcd}$$

commutative, which can easily be done.

length > 1 : we have the situation (i), which by the preceding case reduces to (ii), which by ind. hyp. reduces to (iii).

$$\begin{tikzcd} f \arrow[d, "*"'] & \\ f' \arrow[d] & \\ f_2 \arrow[r, Rightarrow] & f_3 \end{tikzcd} \quad (i) \qquad \begin{tikzcd} f \arrow[d, "*"'] & \\ f' \arrow[r, Rightarrow] \arrow[d] \arrow[dr, phantom, "\circlearrowleft"] & f_4 \arrow[d, "*"] \\ f_2 \arrow[r, Rightarrow] & f_3 \end{tikzcd} \quad (ii) \qquad \begin{tikzcd} f \arrow[r, Rightarrow] \arrow[d, "*"'] \arrow[dr, phantom, "\circlearrowleft"] & f_4' \arrow[d, "*"] \\ f' \arrow[r, Rightarrow] \arrow[d] \arrow[dr, phantom, "\circlearrowleft"] & f_4 \arrow[d, "*"] \\ f_2 \arrow[r, Rightarrow] & f_3 \end{tikzcd} \quad (iii)$$

<u>lemma 6</u>: If $f \overset{*}{\underset{\Sigma}{\nrightarrow}} f' \in M(F,V)$ and in the derivation there are exactly k strong derivation steps,

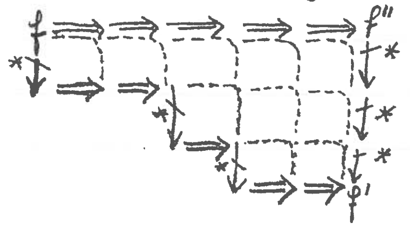
then $f \overset{k}{\Rightarrow} f''$ and $W(f'') = f'$ for some f''.

proof : by induction on k we show $f \overset{k}{\Rightarrow} f'' \overset{*}{\nrightarrow} f'$. Then, because $f' \in M(F,V)$, clearly $W(f'') = f'$ (no φ's are left!).

$k=0$: Trivial.

$k \geq 1$: $f \underset{\Sigma}{\rightarrow} f_2 \Rightarrow f_3 \overset{*}{\nrightarrow} f'$ reduces by ind. hyp to $f \overset{k-1}{\Rightarrow} f''' \overset{*}{\nrightarrow} f_2 \Rightarrow f_3 \overset{*}{\nrightarrow} f'$
and by lemma 5 this reduces to $f \overset{k-1}{\Rightarrow} f''' \Rightarrow f'' \overset{*}{\nrightarrow} f_3 \overset{*}{\nrightarrow} f'$.

Compare the following process in making the diagram commutative each step is done by applying lemma 5:



(24.6) **Theorem**

The sequence $\langle f^{(i)} \rangle_{i=0}^{\infty}$ where $f^{(k)} = W(S^k(f))$ is a kleene seq. of f w.r.t. $\bar{\Sigma}$.

Proof : (a) it is an increasing seq. w.r.t. $\prec$ (b) it majorizes $L(\bar{\Sigma}, f)$.

(a) by replacing in $S^k(f)$ all replacable factors by Ω we obtain $f^{(k)}$, whereas $f^{(k+1)}$ is obtained from $S^k(f)$ by replacing all replacable factors, say $\varphi_j(m, \dots)$, by $W(\tau_j(m_i/v_i, \dots))$, hence $f^{(k)} \prec f^{(k+1)}$.

(b) let $f' \in L(\bar{\Sigma}, f)$, i.e. $f \xrightarrow{*} f'$ say with exacty k strong steps, $f' \in M(F, V)$.
By lemma 6 $f \overset{k}{\Rightarrow} f''$ with $W(f'') = f'$ and
by lemma 4 $f'' \overset{*}{\Rightarrow} S^k(f)$, whereas $W(S^k(f)) = f^{(k)}$.
Hence $f^{(k)}$ and f' are the same except for some occurrences of Ω in f' which are replaced by something else in $f^{(k)}$: $f' \prec f^{(k)}$.

Example

Let $\Sigma : \begin{cases} \varphi_1(x,y) = h(x, \varphi_2(x,y)) \\ \varphi_2(x,y) = g(x, y, \varphi_1(x,y)) \end{cases}$

Let $f = \varphi_1(x,y)$.

Then we construct the kleene sequence of f w.r.t. Σ as follows:

$$
\begin{aligned}
S^0(f) &= \varphi_1(x,y) \\
S^1(f) &= h(x, \varphi_2(x,y)) \\
S^2(f) &= h(x, g(x,y,\varphi_1(x,y))) \\
S^3(f) &= h(x, g(x, y, h(x, \varphi_2(x,y)))) \\
&\vdots
\end{aligned}
$$

$$
\begin{aligned}
f^{(0)} &= W(S^0(f)) = \Omega \\
f^{(1)} &= W(S^1(f)) = h(x, \Omega) \\
f^{(2)} &= W(S^2(f)) = h(x, g(x,y,\Omega)) \\
f^{(3)} &= W(S^3(f)) = h(x, g(x, y, h(x, \Omega))) \\
&\vdots
\end{aligned}
$$

2.5 Characterization of the language of a rewriting system

(25.1) Informally.

In considering a recursive program scheme as a rewriting system Σ, the language of the system could be considered as yielding the meaning of the program. Therefore we look for a convenient characterization of that language. The definition will be given by means of the notion derivation, which corresponds in an obvious way to an operational approach for the semantics of the program. We prefer "let us say" a fixed point approach.

This means that we have to associate with Σ a mapping $\hat{\Sigma}$ such that the language can be a fixed point and is indeed the least one under an appropriate ordering. We take the domain to be sets t of terms in $M(F \cup \Phi, V)$ and we take $\hat{\Sigma}$ to map each set t of terms in $M(F, \Phi, V)$ into the set t' of terms which are derivable from the r.h.s of the i-th rewriting equation, according to rewritings induced by the given terms in t.
But as we have $i = 1, \dots, N$, we have to do all in N-tuples and coordinate wise.

(25.2) Definitions.

The language of the rewr. system Σ is $L = \langle L_1, \dots, L_N \rangle$,
where $L_i = L(\Sigma, \varphi_i(v_1, \dots, v_{\rho(\varphi_i)})) = \{ f \in M(F,V) : \varphi_i(v_1, \dots, v_{\rho(\varphi_i)}) \xrightarrow[\Sigma]{*} f \}$ for $i = 1 \dots N$.

Let $\mathcal{T}$ be the set of all N-tuples $t = \langle t_1 \dots t_N \rangle$ with $t_i \in M(F, \{v_1 \dots v_{\rho(\varphi_i)}\})$

Let $\mathcal{T}$ be ordered by coordinatewise settheoretic inclusion, i.e. $t \subseteq t'$ iff $t_i \subseteq t'_i$ for $i = 1, \dots, N$. Then $\mathcal{T}$ forms a complete lattice.

Let for each $t = \langle t_1, \dots, t_N \rangle \in \mathcal{T}$ λ_t be a mapping,
λ_t : set of subsets of $M(F \cup \Phi, V) \longrightarrow$ sets of subsets of $M(F,V)$,
by defining λ_t for singelton subsets inductively by

$\lambda_t(v) = \{v\}$

$\lambda_t(f(m_1, \dots, m_{\rho(f)})) = \{ f(m'_1, \dots, m'_{\rho(f)}) : m'_i \in \lambda_t(m_i) \text{ for } i = 1 \dots N \}$

$\lambda_t(\varphi_i(m_1, \dots, m_{\rho(\varphi_i)})) = \{ g(m'_1/v_1, \dots, m'_{\rho(\varphi_i)}/v_{\rho(\varphi_i)}) : g \in t_i \text{ and } m'_i \in \lambda_t(m_i) \}$

and by defining

$\lambda_t(S) = \bigcup_{s \in S} \lambda_t(s)$ for not-singleton $S \subseteq M(F \cup \Phi, V)$

and by coordinatewise application λ_t can deal with N-tuples.

Now we associate with Σ a mapping $\hat{\Sigma}$,

$\hat{\Sigma} : \mathcal{T} \longrightarrow \mathcal{T}$ defined by

$\hat{\Sigma}(t) = \lambda_t(\tau)$ where $\tau = \langle \tau_1, \dots, \tau_N \rangle$

Then $\hat{\Sigma}$ is increasing, i.e. for $t \subseteq t'$ we have $\hat{\Sigma}(t) \subseteq \hat{\Sigma}(t')$, and even $\hat{\Sigma}$ is continuous, i.e. for $t^{(1)} \subseteq t^{(2)} \subseteq \dots$ we have $\hat{\Sigma}(\bigcup t^{(i)}) \subseteq \bigcup \hat{\Sigma}(t^{(i)})$!

Hence by the Knaster-Tarski theorem,

there is a least fixed point s of $\hat{\Sigma}$ in $(\mathcal{T}, \subseteq)$, viz.
$s = \bigcup_{k\geq 0} \hat{\Sigma}^k(\emptyset)$ where $\emptyset = \langle \emptyset, \ldots, \emptyset \rangle \in \mathcal{T}$.
and intuitively this can be read as follows:
the least fixed point of $\hat{\Sigma}$ consist of the (N-tuples of sets of) terms obtained from the r.h.s. of the rewr. eq'ns by replacing the φ_j by "something from $\emptyset$" *) or by previously obtained terms for the j-th r.h.s.

(25.3) <u>Theorem</u>

L is the least fixed point of $\hat{\Sigma}$
(This is an extension of Schützenberger's theorem for c.f. languages)
<u>sketch</u> of the proof.
We are going to proof $L = s$ (see previous section) by use of
lemma 1: <u>if</u> $g' \in \lambda_t(g)$ for $g \in M(F \cup \Phi, V)$ <u>then</u> $g \overset{*}{\underset{\Sigma}{\Rightarrow}} g'$ (Exercise),
lemma 2: <u>if</u> $g \overset{*}{\underset{\Sigma}{\Rightarrow}} g'$ <u>then</u> $\lambda_t(g') \subseteq \lambda_{\hat{\hat{\Sigma}}*(t)}(g)$ (Proof ???),
where
$\hat{\hat{\Sigma}}(t) = \hat{\Sigma}(t) \cup t$ and $\hat{\hat{\Sigma}}^*(t) = \bigcup_{k\geq 0} \hat{\hat{\Sigma}}^k(t)$ (a clever trick, is'nt?)
Now we proof both $L \subseteq s$ and $L \supseteq s$.

$\subseteq$: take $g = \varphi_i(v_1, \ldots, v_{\rho(\varphi_i)})$ and $t = \emptyset \in \mathcal{T}$,
then for all $g' \in M(F,V)$ with $g \overset{*}{\underset{\Sigma}{\Rightarrow}} g'$, i.e. for all $g' \in L_i$,
by applying lemma 2:
$\lambda_\emptyset\, g' = \{g'\} \subset \lambda_{\hat{\hat{\Sigma}}*(\emptyset)}\, \varphi_i(v_1, \ldots, v_{\rho(\varphi_i)}) = [\hat{\hat{\Sigma}}^*(\emptyset)]_{i\text{-th ce.}}$
hence
$L_i \subseteq [\hat{\hat{\Sigma}}^*(\emptyset)]_{i\text{-th component}}$ for $i = 1, \ldots, N$, so
$L \subseteq \hat{\hat{\Sigma}}^*(\emptyset) = \hat{\Sigma}^*(\emptyset) = s$

$\supseteq$: we will show that L is a fixed point of $\hat{\Sigma}$, then
$L \supseteq s$ holds too, because s is the least fixed point.
Take $g = \tau_i$ and $t = L$,
then for all $g' \in \hat{\Sigma}(L)$, i.e. for all $g' \in \lambda_L(\tau_i)$,
by applying lemma 1: $g \overset{*}{\underset{\Sigma}{\Rightarrow}} g'$ i.e. $g' \in L_i$;
hence $\hat{\Sigma}(L) \subseteq L$, ... (1)
and now using the additional property $\hat{\hat{\Sigma}}^*(L) \subseteq L$ and
by applying lemma 2:
$\lambda_L(g') = \{g'\} \subseteq \lambda_{\hat{\hat{\Sigma}}*(L)}(\tau_i) \subseteq \lambda_L(\tau_i) = [\hat{\Sigma}(L)]_{i\text{-th comp.}}$
Hence

---

*): so that in this case the whole expression disappears, cfr. $\lambda_\emptyset(m)$.

$L_i \subseteq [\hat{\Sigma}(L)]_{ith\text{-}comp}$, so $L \subseteq \hat{\Sigma}(L)$ .....(2)

Together (1), (2) state $L = \hat{\Sigma}(L)$

end of sketch

3 REC. PR. SCHEMES, INTERPRETATION AND SEMANTICS.

We now try to use the previous results of the theory in the study of semantics of recursive program schemes.

3.1 Preliminary

(31.1) Informal introduction.

Suppose the following is an example of a recursive program:

$$\text{GCD}: \begin{cases} \varphi_1(x,y) = \underline{\text{if}}\ x=0\ \underline{\text{then}}\ y\ \underline{\text{else}}\ \underline{\text{if}}\ y=0\ \underline{\text{then}}\ x\ \underline{\text{else}}\ \varphi_1(y,\varphi_2(x,y)) \\ \varphi_2(x,y) = \underline{\text{if}}\ x<y\ \underline{\text{then}}\ x\ \underline{\text{else}}\ \varphi_2(x-y,y). \end{cases}$$

We will consider GCD as an instance of the following rewr. system Σ_1

$$\Sigma_1: \begin{cases} \varphi_1(x,y) = h(x,y,h(y,x,\varphi_1(y,\varphi_2(x,y)))) \\ \varphi_2(x,y) = g(x,y,\varphi_2(f(x,y),y)). \end{cases}$$

by the interpretation I, given by

$$I: \begin{cases} \text{Domain } D_I = \text{the set of natural numbers} \\ h_I(m,n,p) = \underline{\text{if}}\ m=0\ \underline{\text{then}}\ n\ \underline{\text{else}}\ p, \\ g_I(m,n,p) = \underline{\text{if}}\ m<n\ \underline{\text{then}}\ m\ \underline{\text{else}}\ p, \\ f_I(m,n) = m-n. \quad ^{*)} \end{cases}$$

Now, we want to define a semantics, assigning to each program $\langle\Sigma, I\rangle$ a function in the domain D_I which can be considered as the meaning for the program $\langle\Sigma, I\rangle$. We will do this in three ways: a magma semantics, an operational semantics, a fixed point semantics.

(31.2) Interpretation and valuation.

An interpretation I is given by

- a nonempty domain of interpretation $D_I \neq \emptyset$.
- for all $f \in F$ a partial mapping $f_I : D_I^{\rho(f)} \to D_I$

but as we will deal with total functions we extend the domain

- with a new element ω (the "undefined value")

and we extend the f_I into

$f_I' : (D_I \cup \{\omega\})^{\rho(f)} \longrightarrow D_I \cup \{\omega\}$ by defining

- for all $d_1, \dots, d_{\rho(f)} \in D_I$

$$f_I'(d_1,\dots,d_{\rho(f)}) = \begin{cases} f_I(d_1,\dots,d_{\rho(f)}) & \text{if this value is defined} \\ \omega & \text{otherwise} \end{cases}$$

*): so that GCD computes the Greatest common divisor

- for all $d_1, \ldots, d_{\rho(f)} \in D_I \cup \{\omega\}$, where exactly $d_{i_1} = \omega, \ldots, d_{i_k} = \omega$

$$f'_I(d_1, \ldots, d_{\rho(f)}) = \begin{cases} f_I((d_1, \ldots, d_{\rho(f)})(d'_1/d_{i_1}, \ldots, d'_k/d_{i_k})) & \text{if this value does not vary with } d'_1, \ldots, d'_k \in D_I \\ \omega & \text{otherwise} \end{cases}$$

That is, we use the call by name evaluation for the function f'_I.

By convention,
we assume from here onwards:

- D_I contains ω
- for each $f \in F$ f_I is a total mapping $f_I : D_I^{\rho(f)} \to D_I$ with for all $1 \le i_1 < i_2 < \ldots < i_k \le \rho(f)$ and all $\langle d_1, \ldots, d_{\rho(f)} \rangle \in D_I^{\rho(f)}$:
  if $f_I((d_1, \ldots, d_{\rho(f)})(\omega/d_{i_1}, \ldots, \omega/d_{i_k})) = d \neq \omega$ then $f_I(d_1, \ldots, d_{\rho(f)}) = d$.

The discrete ordering $\sqsubseteq$ on D_I is given by
$d_1 \sqsubseteq d_2$ iff $d_1 = \omega$ or $d_1 = d_2$,
and clearly the f_I are increasing wrt $\sqsubseteq$, i.e.
$d_1 \sqsubseteq d'_1, \ldots, d_{\rho(f)} \sqsubseteq d'_{\rho(f)}$ implies $f_I(d_1, \ldots, d_{\rho(f)}) \sqsubseteq f_I(d'_1, \ldots, d'_{\rho(f)})$.

A valuation is a mapping ν

$$\nu : V \to D_I$$

The mapping $(I,\nu) : M(F,V) \to D_I$ is inductively defined by

$$(I,\nu)(v) = \nu(v) \quad \text{for} \quad \Omega \neq v \in V$$
$$(I,\nu)(\Omega) = \omega$$
$$(I,\nu)(f(m_1, \ldots, m_{\rho(f)})) = f_I((I,\nu)(m_1), \ldots, (I,\nu)(m_{\rho(f)}))$$

3.2 The magma semantics

(32.1) Preparation.

lemma 1. (I,ν) is order preserving, i.e. $\forall m, m' \in M(F,V)$: $m \prec m' \to (I,\nu)m \sqsubseteq (I,\nu)m'$

proof: by induction on the number $\|m\|$ of fionsymbols in m.

$\|m\| = 0$: then $m = \Omega$ and $(I,\nu)\, m = \omega$ so for all m' $(I,\nu)\, m \sqsubseteq (I,\nu)\, m'$.
or $m = v$ and consequently $m' = v$ hence $(I,\nu)(m) \sqsubseteq (I,\nu)(m')$.

$\|m\| > 0$: $m = f(m_1, \ldots, m_{\rho(f)})$ and consequently for $m \prec m'$
$m' = f'(m'_1, \ldots, m'_{\rho(f)})$ with $f = f'$ and $m_\ell \prec m'_\ell$ for $\ell = 1 \ldots \rho(f)$.
Hence by induction $(I,\nu)\, m_\ell \sqsubseteq (I,\nu)\, m'_\ell$ for $\ell = 1 \ldots \rho(f)$ and
because f_I is increasing $f_I((I,\nu)\, m_1, \ldots, (I,\nu)\, m_{\rho(f)}) \sqsubseteq f_I((I,\nu)\, m'_1, \ldots, (I,\nu)\, m'_{\rho(f)})$
i.e. $(I,\nu)\, m \sqsubseteq (I,\nu)\, m'$.

lemma 2 For $m, m' \in L(\bar{\Sigma}, f)$: $(I,\nu)\, m \neq \omega \wedge (I,\nu)\, m' \neq \omega$ implies $(I,\nu)\, m = (I,\nu)\, m'$.
proof : By theorem (24.3) in $L(\bar{\Sigma}, f)$ the join w.r.t. $\prec$ exists, i.e. for $m, m' \in L(\bar{\Sigma}, f)$ there is an $m'' \in L(\bar{\Sigma}, f)$ with $m \prec m''$, $m' \prec m''$. Now, $\omega \neq (I,\nu)\, m \sqsubseteq (I,\nu)\, m''$ implies $(I,\nu)\, m = (I,\nu)\, m''$ and also $\omega \neq (I,\nu)\, m' \sqsubseteq (I,\nu)\, m''$ implies $(I,\nu)\, m' = (I,\nu)\, m''$, hence $(I,\nu)\, m = (I,\nu)\, m'$.

(32.2) The definition.
The magma semantics $Val_I\, \Sigma$ of a rec. program $\langle \Sigma, I \rangle$ is given by

$$Val_I\, \Sigma\, (\nu) = \begin{cases} \text{the common value of } (I,\nu)\, m \text{ for all } m \in L(\bar{\Sigma}, f) \\ \quad \text{for which } (I,\nu)\, m \neq \omega \quad \text{(if such } m \text{ exist)} \\ \omega \quad \text{(otherwise)} \end{cases}$$

3.3 The Operational Semantics

(33.1) Preparation.
a computation of Σ under I at point ν is a possibly infinite sequence $e_0, e_1, \dots$ of elements from $M(F \cup \Phi, D_I)$ such that

(i) $e_0 = \varphi_1(\nu_1, \dots, \nu_{\rho(\varphi_1)})$ and by ν_i we denote $\nu(v_i)$

(ii) either
e_{n+1} is obtained from e_n by a reduction step, that is to say:
$e_{n+1} = \alpha \cdot d \cdot \beta$ and $e_n = \alpha \cdot f(m_1, \dots, m_{\rho(f)}) \cdot \beta$ with
- $m_i \in M(F \cup \Phi, D_I)$ and
- for all $d_1, \dots, d_{\rho(f)} \in D_I$ s.th. $d_i = m_i$ whenever $m_i \in D_I$
  $f_I(d_1, \dots, d_{\rho(f)}) = d \neq \omega$

or
e_{n+1} is obtained from e_n by a rewriting step, that is to say
$e_{n+1} = \alpha \cdot \tau_i(m_1/v_1, \dots, m_{\rho(\varphi_i)}/v_{\rho(\varphi_i)}) \cdot \beta$ and $e_n = \alpha \cdot \varphi_i(m_1, \dots, m_{\rho(\varphi_i)}) \cdot \beta$

(iii) if there is a last element in the sequence, then no rewriting or reduction steps are applicable and consequently the last element is in D_I.
And then we say the computation terminates.

lemma 1. The results of two terminating computations of Σ under I at ν if both defined (i.e. $\neq \omega$), are the same.

proof : By the lemma in the next subsection, for each terminating computation $\varphi_1(\nu_1,\dots,\nu_{e(\varphi_1)}) = e_0,\ e_1,\ e_2,\ \cdots,\ e_n$
there exists a strong derivation in $\bar{\Sigma}$ $\varphi_1(v_1,\dots v_{e(\varphi_1)}) = \varepsilon_0 \Rightarrow \varepsilon_1 \cdots \Rightarrow \varepsilon_n$
with $(I,\nu)\ W(\varepsilon_n) = e_n$ (and of course $\varepsilon_n \in L(\bar{\Sigma},\varphi_1)$)
and by lemma 2 of (32·1) (I,ν) does not vary on $L(\bar{\Sigma},\varphi_1)$ whenever the value $\neq \omega$.

(33·2) <u>The definition</u>

The operational semantics $\mathrm{Comp}_I\, \Sigma_1$ of a rec. program $\langle \Sigma_1, I\rangle$ is given by

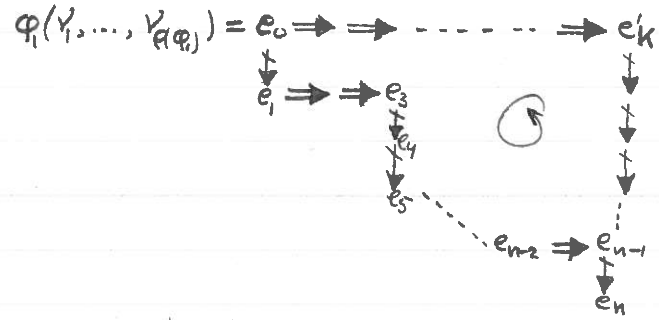
$$\mathrm{Comp}_I\, \Sigma_1(\nu) = \begin{cases} \text{the common value of all terminating comp's of } \Sigma \text{ under } I \text{ at } \nu & \\ \quad\text{whenever they are} \neq \omega & (\text{if such exist}) \\ \omega & (\text{otherwise}) \end{cases}$$

(33·3) Theorem on equivalence with magma semantics.

<u>lemma</u> If $e_0, e_1, \dots e_n$ is a terminating comp. seq. of Σ under I at ν with exactly k rewriting steps, then there exists a strong derivation of length k in $\bar{\Sigma}$ of $\varepsilon_0 = \varphi_1(v_1,\dots,v_{e(\varphi_1)})$ into some ε_n s.th. $(I,\nu)W(\varepsilon_n) = e_n$.

Sketch of proof:

(Due to B. Rosen's paper "Subtree manipulations and Church-Rosser properties") (Quite similar to the proof of Thm(24.6) and lemma 6 in (24.5)).
Denote a comp. step by $\rightarrow$, a rewriting step by $\Rightarrow$, a reduction step by $\nrightarrow$, then by B. Rosen's results we can make the following diagram commutative:



and corresponding to the top line we can make a strong derivation in $\bar{\Sigma}$: $\varphi_1(v_1,\dots,v_{e(\varphi_1)}) = \varepsilon_0 \Rightarrow\Rightarrow \cdots\cdots \Rightarrow \varepsilon_k$
with $\varepsilon_k \overset{*}{\nrightarrow} \varepsilon \in L(\bar{\Sigma},\varphi_1)$, i.e. $W(\varepsilon_k) = \varepsilon$ such that $(I,\nu)\,\varepsilon = e_n$

<u>Theorem</u>

The operational semantics and the magma semantics are equivalent.
proof :

We have to prove: for all Σ, I and for all ν $\quad Val_I \Sigma(\nu) = Comp_I \Sigma(\nu)$.
By the lemma

$$Comp_I \Sigma(\nu) = \begin{cases} (I,\nu)\, W(\varepsilon_k) & \text{if some term. comp. } e_0 \xrightarrow{*} e_n \neq \omega \text{ exists and} \\ & \varepsilon_0 \overset{*}{\underset{\Sigma}{\Rightarrow}} \varepsilon_k \text{ is constructed according the lemma} \\ \omega & \text{otherwise} \end{cases}$$

hence by lemma 2 of (32.1)

$$Comp_I \Sigma(\nu) = \begin{cases} (I,\nu)(\varepsilon) & \text{if some } \varepsilon \in L(\bar{\Sigma}, \varphi_1) \text{ exists with } (I,\nu)\,\varepsilon \neq \omega \\ \omega & \text{otherwise} \end{cases}$$

so that we conclude on basis of the definition of $Val_I \Sigma$

$Comp_I \Sigma(\nu) = Val_I \Sigma(\nu)$.

3.4 The fixed point semantics

(34.1) Preparation.

We have to associate with a rec. program $\langle \Sigma, I\rangle$ a mapping $\tilde{\Sigma}_I$, such that a fixed point of $\tilde{\Sigma}_I$ can be considered as the meaning of the program. Hence the $\tilde{\Sigma}_I$ must work on a space $\tilde{D}_I^{(N)}$ of (N-tuples of) functions and its arguments are to be considered as giving a value for the unknown function symbols in the τ_i.
Formally, we define the domain as follows:

$\tilde{D}_I = \bigcup_{n\geq 1} (D_I^n \to D_I)$,

$\sqsubseteq$ on $\tilde{D}_I$: a partial ordering induced by the $\sqsubseteq$ of D_I:

$\quad \psi_1 \sqsubseteq \psi_2$ iff $\psi_1, \psi_2 \in (D_I^n \to D_I)$ for some n, and

$\quad \psi_1(d_1, \ldots, d_n) \sqsubseteq \psi_2(d_1 \ldots d_n)$ for all $\langle d_1, \ldots, d_n\rangle \in D_I^n$.

Then, due to the discreteness of $\sqsubseteq$ on D_I we can define $\bigsqcup \psi^{(i)}$ for chains $\psi^{(1)} \sqsubseteq \psi^{(2)} \sqsubseteq \ldots \sqsubseteq \psi^{(k)} \sqsubseteq \ldots$ of $\psi^{(k)} \in \tilde{D}_I$, as

$$\bigsqcup \psi^{(i)}(d_1 \ldots d_n) = \begin{cases} \text{the common value of } \psi^{(k)}(d_1 \ldots d_n) \text{ for} \\ \quad \text{all } k \text{ with } \psi^{(k)}(d_1 \ldots d_n) \neq \omega \text{ (if such } k \text{ exist)} \\ \omega \quad \text{(otherwise)} \end{cases}$$

$\tilde{D}_I^N$ is just the set of N-tuples of functions and is a chain-closed set w.r.t. the componentswise ordering with $\sqsubseteq$ of $\tilde{D}_I$.

Secondly we define a mapping "subscript I" : $M(F,V)^n \to \tilde{D}_I$ by for each n (but we will omit the indication n when no confusion results)

$$\begin{cases} \Omega_I = \lambda d_1 \ldots d_n \cdot \omega \\ v_I = \lambda d_1 \ldots d_n \cdot d_i \quad \text{where } v = v_i \in V = \{v_1, v_2, v_3, \ldots\ldots\} \cup \{\Omega\} \\ f(m_1, \ldots, m_{\rho(f)})_I = f_I(m_{1I}, \ldots, m_{\rho(f)I}) \text{ (function composition).} \end{cases}$$

Now, $(I,\nu)\, m = m_I(\nu(v_1), \ldots, \nu(v_n))$, where $\{v_1, \ldots, v_n\}$ contains all variable symbols occurring in m.

Now, we associate with a rec. program $\langle \Sigma, I \rangle$ the mapping $\tilde{\Sigma}_I$,
$\tilde{\Sigma}_I : \tilde{D}_I^N \to \tilde{D}_I^N$ defined as
$\tilde{\Sigma}_I(\psi_1, \ldots, \psi_N) = \langle \tau'_1, \ldots, \tau'_N \rangle$, where
τ'_i is obtained from τ_i by replacing the unknown function symbols φ_j by the functions ψ_j and considering the expression thus obtained as a fnc: $D_I^{\rho(\varphi_i)} \to D_I$, i.e.
$\tau'_i = (\tau_i(\psi_1/\varphi_1, \ldots, \psi_N/\varphi_N))_I$ where
the ψ_i are function symbols with $\psi_{iI} = \psi_i$

Example

Let GCD: $\begin{cases} \varphi_1(x,y) = \underline{\text{if}}\ x=0\ \underline{\text{then}}\ y\ \underline{\text{else}}\ \underline{\text{if}}\ y=0\ \underline{\text{then}}\ x\ \underline{\text{else}}\ \varphi_1(y, \varphi_2(x,y)), \\ \varphi_2(x,y) = \underline{\text{if}}\ x<y\ \underline{\text{then}}\ x\ \underline{\text{else}}\ \varphi_2(x-y, y). \end{cases}$

Then

$$\tilde{\Sigma}_I(\psi_1, \psi_2) = \langle\ \underline{\text{if}}\ x=0\ \underline{\text{then}}\ y\ \underline{\text{else}}\ \underline{\text{if}}\ y=0\ \underline{\text{then}}\ x\ \underline{\text{else}}\ \psi_1(y, \psi_2(x,y)),$$
$$\underline{\text{if}}\ x<y\ \underline{\text{then}}\ x\ \underline{\text{else}}\ \psi_2(x-y, y)\ \rangle.$$

Now, we note that $\tilde{\Sigma}_I$ is continuous, i.e. for each chain $\psi_i^{(1)} \sqsubseteq \psi_i^{(2)} \sqsubseteq \ldots$
$\tilde{\Sigma}_I(\psi_1, \ldots, \bigsqcup_{k \geq 0} \psi_i^{(k)}, \ldots \psi_N) = \bigsqcup_{k \geq 0} \tilde{\Sigma}_I(\psi_1, \ldots, \psi_i^{(k)}, \ldots, \psi_N)$,
Hence
$\tilde{\Sigma}_I$ has a least-fixed point $\sigma = \langle \sigma_1, \ldots, \sigma_N \rangle \in \tilde{D}_I^N$, viz.
$\langle \sigma_1 \ldots \sigma_N \rangle = \bigsqcup_{k \geq 0} \tilde{\Sigma}_I^k(\omega_1, \ldots, \omega_N)$ where
ω_i is the "undefined function on $D_I^{\rho(\varphi_i)}$", i.e.
$\omega_i(d_1, \ldots, d_{\rho(\varphi_i)}) = \omega$ for all $d_1, \ldots, d_{\rho(\varphi_i)} \in D_I$.

(34.2) The Fixed Point Semantics $Fix_I \Sigma$ of a rec. program $\langle \Sigma, I \rangle$ is given by
$Fix_I \Sigma =$ the first component of the least fixed point of $\tilde{\Sigma}_I : \tilde{D}_I^N \to \tilde{D}_I^N$.

(34.3) Theorem

The Fixed Point Semantics is included in the Magma Semantics.

Proof:

We have to prove $Fix_I \Sigma \sqsubseteq Val_I \Sigma$ in $\tilde{D}_I$ for all rec. programs $\langle \Sigma, I \rangle$

For $t \in \mathcal{T}$ (see 25.2), if $(I,\nu)\, t \neq \{\omega\}$ then $(I,\nu)\, \hat{\Sigma}(t) = \tilde{\Sigma}_I (t_I)\, \nu$.
This should be clear from the definitions of $\hat{\Sigma}$ and $\tilde{\Sigma}_I$, but it is rather complicated to write down in a precise formulation.
Hence, if a fixed point in $(\tilde{D}_I^N, \sqsubseteq)$ of $\tilde{\Sigma}_I$ is of the form t_I for some $t \in \mathcal{T}$, then t is also a fixed point of $\hat{\Sigma}$ in $(\mathcal{T}, \subseteq)$, because $(I,\nu)\, t = (t_I)\,\nu = \tilde{\Sigma}_I (t_I)\nu = (I,\nu)\, \hat{\Sigma}(t) \Rightarrow (I,\nu) t = (I,\nu)\hat{\Sigma}(t)$.
And conversely, a fixed point t of $\hat{\Sigma}$ in $(\mathcal{T}, \subseteq)$ induces the fixed point t_I of $\tilde{\Sigma}_I$ in $(\tilde{D}_I^N, \sqsubseteq)$.
Now,
$Val_I\, \Sigma = \angle_1(\bar{\Sigma}, \varphi_1)_I = [\angle_I]_{\text{first comp}}$, whereas $\angle_1 \in \mathcal{T}$ is a fixed point of $\hat{\Sigma}$ in $(\mathcal{T}, \subseteq)$, so that $\angle_I$ is a fixed point of $\tilde{\Sigma}_I$ in $(\tilde{D}_I^N, \sqsubseteq)$, and consequently includes the least one.
Hence $Fix_I\, \Sigma \sqsubseteq [\angle_I]_{\text{first comp}} = Val_I\, \Sigma$.

Remark
We can strengthen the above result.
In spite of the fact that we did not prove $Val_I\, \Sigma$ to be the least fixed point of $\tilde{\Sigma}_I$, we can prove that it is the least one of $\tilde{\Sigma}_I$ with respect to all fixed points of the form $[\angle'_I]_{\text{first comp}}$, for $\angle' \in \mathcal{T}$:
Let $\angle'_I$ be a fixed point of $\tilde{\Sigma}_I$, then $\angle'$ is a fixed point of $\hat{\Sigma}$ and because $\angle_1$ is the least one (thm (25.2)), $\angle \subseteq \angle'$ (the ordering in $\mathcal{T}$) and because I is ordering preserving wrt $\subseteq$ and $\sqsubseteq$ (clear from def's) $(I,\nu)\, \angle_1 \sqsubseteq (I,\nu)\, \angle'$, i.e. $\angle_I \sqsubseteq \angle'_I$.
Hence
If we restrict $\tilde{D}_I$ to be the union of the magma-definable mappings: $D_I^n \to D_I$, then $Val_I\, \Sigma = Fix_I\, \Sigma$.

4 EQUIVALENCE OF RECURSIVE PROGRAM SCHEMES

B. Courcelle and J. Vuillemin have proved the decidability of the equivalence of rec. pr. schemes with one variable only and an additional restriction on the rec. pr. schemes, viz. the property of being "acceptable". We will try to formulate their proof in our terminology. *)

4.1 Preparation

(41.1) Definitions and Theorem.
$\Sigma \equiv \Sigma'$, Σ is equivalent with Σ', iff for all I $Val_I \Sigma = Val_I \Sigma'$.
We prefer to reason as syntactically as possible, therefore
$L \sim L'$, L is equivalent with L', iff both $L \prec L'$ and $L' \prec L$, where
$L \prec L'$, L majorizes L', iff $\forall m \in L \; \exists m \in L' : m \prec m'$.

Theorem
$\Sigma \equiv \Sigma'$ if and only if $L(\bar{\Sigma}, \varphi_1) \sim L(\bar{\Sigma}', \varphi_1)$
proof:
($\Leftarrow$) : immediately from def's $L \sim L'$, def $Val_I \Sigma$, lemma 2 (32.1).
($\Rightarrow$) : take Herbrand interpretation (I, ν) given by
$D_I = M(F,V)$ and $\nu(v) =$ "v" for all $v \in V$, $f_I =$ "f" for all $f \in F$.

(41.2) The method of Hopcroft and Korenjak.
We now sketch the way Hopcroft and Korenjak prove the decidability of the equivalence of simple deterministic grammars (s.d.g.). Our attempt will be an imitation of this one.
Sketch
S.d.g have production rules of the form $\xi_i \to a\, \xi_{i_1} \cdots \xi_{i_n}$, where $a \in$ terminal alphabeth and all $\xi \in$ nonterminal alpabeth, and for all ξ_i, a there exist at most one prod. rule of the form $\xi_i \to a \cdots$.
Equivalence is defined as $G \equiv G'$ iff $L(G, \xi_1) = L(G', \xi_1)$.
The deciding algorithm is sketched below.
First define for words w, w' $w \equiv w'$ iff $L(G, w) = L(G', w')$. Then $G \equiv G'$ is equivalent with $\xi_1 \equiv \xi_1$. Secondly, bring G and G' into a normal form, viz. the reduced form defined by: G is reduced iff $L(G, \xi) \neq \emptyset$ for all ξ in G. Thirdly let $d(\xi) = \min\{|w| : w \in L(G, \xi)\}$.

---

*): (added in may '74: see page 34 for further remarks)

Now, for deciding $\xi_i \xi_{i_1} \dots \xi_{i_n} \equiv \xi_j \xi_{j_1} \dots \xi_{j_m}$ proceed as follows:

<u>while</u> no negative tests are encountered and some equivalence relations still have to be tested

<u>do</u> either for $\xi_i \xi_{i_1} \dots \xi_{i_k} \equiv \xi_j$ and $\xi_{i_{k+1}} \dots \xi_{i_n} \equiv \xi_{j_1} \dots \xi_{j_m}$

or for $\xi_i \equiv \xi_j \xi_{j_1} \dots \xi_{j_k}$ and $\xi_{i_1} \dots \xi_{i_n} \equiv \xi_{j_{k+1}} \dots \xi_{j_m}$

(where both the k and the case are determined by means of δ; and this is based on a crucial lemma)

test $a = a'$, where $\xi_i \xrightarrow[\sigma]{} a\, \xi'_{i_1} \dots \xi'_{i_{n'}}$ and $\xi_j \xrightarrow[\sigma']{} a'\, \xi'_{j_1} \dots \xi'_{j_{m'}}$

and

go on with the relations

either $\xi'_{i_1} \dots \xi'_{i_{n'}} \xi_{i_1} \dots \xi_{i_k} \equiv \xi'_{j_1} \dots \xi'_{j_{m'}}$ and $\xi_{i_{k+1}} \dots \xi_{i_n} \equiv \xi_{j_1} \dots \xi_{j_m}$

or $\xi'_{i_1} \dots \xi'_{i_{n'}} \equiv \xi'_{j_1} \dots \xi'_{j_{m'}} \xi_{j_1} \dots \xi_{j_k}$ and $\xi_{i_1} \dots \xi_{i_n} \equiv \xi_{j_{k+1}} \dots \xi_{j_m}$.

<u>od</u>

and this proces terminates by virtue of the properties of s.d.g.'s.

4.2 <u>Imitation of the method of Hopcroft and Korenjak</u>

(42.1) Acceptability and standardform.

Let $\delta : M(F \cup \Phi, V) \longrightarrow \mathbb{N} \cup \{\infty\}$ be defined w.r.t. Σ by

$$\delta(v) = 0$$
$$\delta(f(m_1, \dots, m_{\rho(f)})) = 1 + \min\{\delta(m_1), \dots, \delta(m_{\rho(f)})\}$$
$$\delta(\varphi_j(m_1, \dots, m_{\rho(\varphi_j)})) = \delta(\tau_j(m_1/v_1, \dots, m_{\rho(\varphi_j)}/v_{\rho(\varphi_j)}))$$

Define

<u>Σ is acceptable</u> iff $\delta(\varphi_i) < \infty$ for $i = 1, \dots, N$.

Intuitively, $\delta(m)$ is finite iff there exists a derivation $m \xrightarrow[\Sigma]{*} m'$ such that in m' there is an occurrence of a variable symbol not lying in the scope of any $\varphi \in \Phi$. This property is decidable.

<u>Σ is in standard form</u> iff all τ_i in Σ are of the form

$$f(m_1, \dots, m_{\rho(f)}) \quad \text{with } m_\ell \in M(\Phi, V) \text{ for } \ell = 1 \dots \rho(f).$$

For each acceptable rewr. system Σ we effectively can construct another one which is in standard form, is acceptable and equivalent to the original one, by the following algorithm:

<u>while</u> there are some τ_i of the form $\varphi_j(m_1, \dots m_{\rho(\varphi_j)})$

<u>do</u> replace such a τ_i by $\tau_j(m_1/v_1, \dots, m_{\rho(\varphi_j)}/v_{\rho(\varphi_j)})$ <u>od</u>

{and this proces terminates by virtue of acceptability}
<u>while</u> there are proper subexpressions $f(---)$ of the r.h.s's
<u>do</u> add a <u>new</u> φ to the current Φ;
replace such an innermost $f(---)$ by $\varphi(v_1, \dots v_{\rho(\varphi)})$;
add a <u>new</u> equation $\varphi(v_1, \dots v_{\rho(\varphi)}) = f(---)$ to the current system
<u>od</u>
{and now the resulting system is in standard form and is equivalent to the original one}.

(42.2) The crucial lemma.
From here onwards we let Σ and Σ' be in standard form and one variable systems, i.e. $V = \{x; \Omega\}$ and for all $\varphi \in \Phi$, $\rho(\varphi) = 1$ and for $f \in F$ $\rho(f) \geq 1$.

<u>example</u>
$$\begin{cases} \varphi_1(x) = g_1(x, \varphi_1(\varphi_1(\varphi_3(x)))) \\ \varphi_2(x) = g_2(\varphi_1(x), \varphi_2(\varphi_3(x))) \\ \varphi_3(x) = g_3(\varphi_1(x), \varphi_3(x)) \end{cases}$$

Define for $m \in M(F \cup \Phi, V)$ and $m' \in M(F \cup \Phi', V)$
$m \equiv m'$, m is equivalent with m' with respect to Σ and Σ', iff $L(\bar{\Sigma}, m) \sim L(\bar{\Sigma}', m')$ (Note that we omit Σ and Σ'!).
$\Sigma \equiv \Sigma'$ iff $\varphi_1(--) \equiv \varphi_1'(---)$ is now immediately clear from def.'s

lemma 1: $f(m_1, \dots, m_{\rho(f)}) \equiv f'(m_1', \dots, m'_{\rho(f')})$ iff $f = f'$ and $m_i \equiv m_i'$ $i = 1 \dots \rho(f)$.
proof: ($\Leftarrow$) clear and for ($\Rightarrow$) we reason as follows:
by definition, for all $m_i^* \in L(\bar{\Sigma}, m_i)$ there are $m_i^{*\prime} \in L(\bar{\Sigma}, m_i')$ such that $f(m_1^*, \dots, m^*_{\rho(f)}) \prec f'(m_1^{*\prime}, \dots, m'^*_{\rho(f')})$. Hence by def. of $\prec$, $f = f'$ and $m_i^* \prec m_i'^*$.
But similarly for the reverse order $\succ$.
Hence $f = f'$ and $m_i \equiv m_i'$ for $i = 1 \dots \rho(f)$.

lemma 2: for all $m, n \in M(F \cup \Phi, V)$ and $m', n' \in M(F \cup \Phi', V)$
a) when $m \equiv m'$: $m(n/x) \equiv m'(n'/x)$ iff $n \equiv n'$
b) when $n \equiv n'$: $m(n/x) \equiv m'(n'/x)$ iff $m \equiv m'$
proof: a) $\Rightarrow$ by induction on the number $\|m\|$ of fcn symbols in m:

$\|m\| = 0$ then $x = m \equiv m'$, so $m' = x$ and $n = m(n|x) \equiv m'(n'/x) = n'$.

$\|m\| > 0$ if $m = \varphi_i(m^*)$ then $m \equiv \tau_i(m^*/x) = f(m_1, \ldots m_{\rho(f)})$, and otherwise $m = f(m_1, \ldots, m_{\rho(f)})$ so we have $m \equiv f(m_1 \ldots m_{\rho(f)})$.

Because m' can not be x, we similarly get $m' \equiv f'(m'_1 \ldots m'_{\rho(f')})$

By lemma 1, from $m \equiv m'$ we get $f = f'$ and $m_i \equiv m'_i$.

Now $f(m_1(n/x), \ldots, m_{\rho(f)}(n/x)) = f(m_1, \ldots, m_{\rho(f)})(n/x) \equiv m(n/x) \equiv$
$\equiv m'(n'/x) \equiv f'(m_1 \ldots m_{\rho(f')})(n'/x) = f'(m'_1(n'/x), \ldots m'_{\rho(f')}(n'/x))$.

hence by lemma 1, $f = f'$ (again) and $m_i(n/x) \equiv m'_i(n'/x)$.

Now by induction hypothesis $n \equiv n'$.

(a)($\Leftarrow$) Obvious, also provable by induction.

(b)($\Rightarrow$) by induction on $\|m\| + \|m'\|$.

$\|m\| + \|m'\| = 0$ then $m = x = m'$ hence they are equivalent.

$\|m\| + \|m'\| > 0$ then as in case (a), $m \equiv f(m_1, \ldots m_{\rho(f)})$, $m' \equiv f'(m'_1, \ldots, m'_{\rho(f')})$

Now as in case (a) from $m(n/x) \equiv m'(n'/x)$ we derive $f = f'$ and $m_i(n/x) \equiv m'_i(n'/x)$ for $i = 1 \ldots \rho(f)$.

By induction hypothesis $m_i \equiv m'_i$ hence by lemma 1 $m \equiv f(m_1 \ldots m_{\rho(f)}) \equiv f(m'_1, \ldots m'_{\rho(f)}) \equiv m'$.

(b)($\Leftarrow$) Obvious.

<u>lemma 3</u>: "CRUCIAL LEMMA"

« THE LEMMA PRESENTED AT THE LECTURES WAS ILL-FORMULATED AND EVEN NOT TRUE IN ITS PRESUMABLY INTENDED MEANING. MOREOVER, DUE TO THIS FAILURE THE COROLARY AND THE ALGORITHM WILL FAIL TOO »

IT IS ONLY FOR CURIOSITY THAT I PRESENT THE COROLARY AND THE ALGORITM. PLEASE SEE THE NEXT SECTION FOR FURTHER REMAKS.

<u>Corolary to lemma 3</u>: (omitting parentheses when no confusion results)

$\varphi_{i_1} \varphi_{i_2} \cdots \varphi_{i_m} x \equiv \varphi'_{j_1} \varphi'_{j_2} \cdots \varphi'_{j_n} x$ with $\delta(\varphi_{i_1}) \leq \delta'(\varphi'_{j_1})$

iff

$\vec{\varphi}_\ell \varphi_{i_2} \cdots \varphi_{i_k} x \equiv \vec{\varphi}'_\ell$ for $\ell = 1, \ldots, \rho(f)$ $\wedge$ $\varphi_{i_{k+1}} \cdots \varphi_{i_m} x \equiv \varphi'_{j_2} \cdots \varphi'_{j_n} x$ $\wedge$ $f = f'$ where
$\varphi_{i_1} x = f(\vec{\varphi}_1 x, \ldots, \vec{\varphi}_{\rho(f)} x)$ in Σ and $\varphi'_{j_1} x = f'(\vec{\varphi}'_1 x, \ldots, \vec{\varphi}'_{\rho(f')} x)$ in Σ'.

proof: by the lemma $\varphi_{i_1}\cdots\varphi_{i_m}x \equiv \varphi'_{j_1}\cdots\varphi'_{j_n}x$ iff for some k $1 \le k \le m$
$\varphi_{i_1}\cdots\varphi_{i_k}x \equiv \varphi'_{j_1} \;\wedge\; \varphi_{i_{k+1}}\cdots\varphi_{i_m}x \equiv \varphi'_{j_2}\cdots\varphi'_{j_n}x$ (provided $\delta(\varphi_{i_1}x) \le \delta(\varphi'_{j_1})$)
The conclusion follows by lemma 1.

(42.3) The algorithm.
Now we present the algorithm for deciding the equivalence of acceptable rewriting systems Σ and Σ'.
First bring Σ and Σ' into standard form, then perform

```
algorithm Decide:
begin list EAT of expressions #already tested# ;
      proc TEST = (expression #to be tested# φ_{i1}...φ_{im} x ≡ φ'_{j1}...φ'_{jn} x ):
           if expression does not appear in list EAT
           then write down φ_{i1}...φ_{im} x ≡ φ'_{j1}...φ'_{jn} x in list EAT ;
                if δ(φ_{i1} x) ≤ δ(φ'_{j1} x)
                then  search for k such that δ(φ_{i1}...φ_{ik} x) ≡ δ(φ_{j1} x)
                      with escape (print("NO"); leave algorithm Decide);
                      let φ_{i1} x = f(φ⃗_1 x, ..., φ⃗_{ρ(f)} x) be in Σ,
                          φ_{j1} x = f'(φ⃗'_1 x, ... φ⃗'_{ρ(f')} x) be in Σ';
                      test for f = f'
                      with escape (print("NO"); leave algorithm Decide);
                      (call TEST( φ⃗_l φ_{i2}...φ_{ik} x ≡ φ⃗'_l x) for l ∈ {1 ... ρ(f)} ,
                       call TEST( φ_{ik+1}...φ_{im} x ≡ φ'_{j2}...φ'_{jn} x)   )
                else  { δ(φ_{i1} x) ≥ δ'(φ'_{j1} x) and
                        do something similar with r.h.s <-> l.h.s }
                fi
           fi proc TEST;
      EAT := empty ;
      call TEST(φ_1 x ≡ φ'_1 x) ; print("YES") ;
end algorithm Decide.
```

Clearly the corolary implies the partial correctness of the algorithm and moreover the termination follows from the following argument:
Let $M = \max\{\, \|\vec{\varphi}_l x\| : f(\vec{\varphi}_1 x, \dots \vec{\varphi}_{\rho(f)} x)$ is r.h.s in Σ or Σ', $1 \le l \le \rho(f)\}$
let $\Delta = \max\{\, \delta(\varphi x) : \varphi \in \Phi$ or $\varphi \in \Phi' \} \le \infty$ (acceptability!)

Then initially the length of the expression to be tested is bound by $2\cdot M+\Delta$, and inductively a call of TEST with an expression of length less than $2\cdot M+\Delta$ induces call's of TEST with expressions of length less than $2\cdot M+\Delta$, because $\delta(\varphi_{i_1}\cdots\varphi_{i_k}x)=\delta(\varphi' x)$ implies $\|\varphi_{i_1}\cdots\varphi_{i_k}x\|<\Delta$.

(4.2.4) Example

We now present an example of an execution of the algorithm "Decide" for the following acceptable rewr. systems in standard form.

$$\Sigma=\begin{cases}\varphi_1 x = g_1(x,\ \varphi_1\varphi_2\varphi_3 x)\\ \varphi_2 x = g_2(\varphi_1 x,\ \varphi_2\varphi_3 x)\\ \varphi_3 x = g_3(\varphi_1 x,\ \varphi_3 x)\end{cases}\qquad \Sigma'=\begin{cases}\varphi_1 x = g_1(x,\ \varphi_4 x)\\ \varphi_2 x = g_2(\varphi_5 x,\ \varphi_6 x)\\ \varphi_3 x = g_3(\varphi_1 x,\ \varphi_3 x)\\ \varphi_4 x = g_1(\varphi_6 x,\ \varphi_4\varphi_6 x)\\ \varphi_5 x = g_1(x,\ \varphi_1\varphi_6 x)\\ \varphi_6 x = g_2(\varphi_1\varphi_3 x,\ \varphi_6\varphi_5 x)\end{cases}$$

hence

	φ_1	φ_2	φ_3
δ:	1	2	2

	φ_1	φ_2	φ_3	φ_4	φ_5	φ_6
δ':	1	2	2	5	1	4

Notation: for each actual parameter, we denote beneath it the tests performed and the new actual parameters for the recursive calls of TEST. For each depth in the tree, the eq. expr. above it belong to the list EAT, [eq. expr.] denotes the generation of an expr. already belonging to EAT.

Execution:

$\varphi_1\equiv\varphi_1$

$g_1=g_1,\ x=x,\quad \varphi_1\varphi_2\varphi_3\equiv\varphi_4$

$g_1=g_1,\ \varphi_2\varphi_3\equiv\varphi_6\qquad \varphi_1\varphi_2\varphi_3\varphi_2\varphi_3\equiv\varphi_4\varphi_6$

$g_2=g_2,\ \varphi_1\varphi_3\equiv\varphi_1\varphi_3,\ \varphi_2\varphi_3\varphi_3\equiv\varphi_6\varphi_5\qquad g_1=g_1\ [\varphi_2\varphi_3\equiv\varphi_6][\varphi_1\varphi_2\varphi_3\varphi_2\varphi_3\equiv\varphi_4\varphi_6][\varphi_2\varphi_3\equiv\varphi_6]$

$g_1=g_1,\ x=x\ [\varphi_1\varphi_2\varphi_3\equiv\varphi_4]\ \varphi_3\equiv\varphi_3\qquad g_2=g_2\ [\varphi_1\varphi_3\equiv\varphi_1\varphi_3]\ [\varphi_2\varphi_3\varphi_3\equiv\varphi_6\varphi_5]\ \varphi_3\equiv\varphi_5$

$g_3=g_3\ [\varphi_1\equiv\varphi_1][\varphi_3\equiv\varphi_3]\qquad \delta\varphi_3\neq\delta'\varphi_5\quad (g_3=g_1\quad \varphi_1\equiv x\quad \varphi_3\equiv\varphi_1\varphi_6)$

Thus $\Sigma\not\equiv\Sigma'$ because: "NO" is printed, k cannot be found.

4.3 Remarks on section 4.2

(43.1) The crucial lemma and the algorithm fail

The ill-formulated lemma presented at the lectures was

$$\begin{array}{lll} m(n/x) \equiv m'(n'/x) & \text{iff} & \exists m'' : \; n = (\text{or} \equiv ?)\, m''(n'/x) \\ \delta(m) \leq \delta'(m') & & \qquad m(m''/x) \equiv m' . \end{array}$$

Clearly the omitting of the respective Σ, Σ' in "$\equiv$" here has lead to a confusion: both the possibilities of $=$ and of $\equiv$ (w.r.t. Σ, Σ') and of $\equiv$ (w.r.t. Σ, Σ) and of $\equiv$ (w.r.t Σ', Σ') in the line $n = (\text{or} \equiv ?)\, m''(n'/x)$ lead to contradictions. Presumably the lemma should read

$$\begin{array}{lll} m(n/x) \equiv m'(n'/x) & \text{iff} & \exists m'', m''' : \; n = m''(m'''/x) \\ \delta(m) \leq \delta'(m') & & \text{and} \begin{cases} \text{both} & m(m''/x) \equiv m' \\ \text{and} & m''' \equiv n' \end{cases} \end{array}$$

because now the corolary follows by taking

$$\begin{cases} m' = \varphi_{i_1} x & , \; m' = \varphi_{j_1} x \\ n = \varphi_{i_2} \cdots \varphi_{i_m} x & , \; n' = \varphi_{j_2} \cdots \varphi_{j_n} x \end{cases} \quad \text{so that} \quad \begin{cases} m'' = \varphi_{i_2} \cdots \varphi_{i_k} x \\ m''' = \varphi_{i_{k+1}} \cdots \varphi_{i_m} x . \end{cases}$$

The proof of the lemma is given such that the failure becomes as clear as possible:

"proof": by induction on $\delta(m)$:

$\delta(m) = 0$: $m = x$ and if $n = x$ then trivially $m' = n' = x$ so $m'' = m''' = x$, but if $n \neq x$ we can not say anything. Here is the essential case where the lemma fails.

$\delta(m) > 0$: $m = \varphi(m_0)$ and $m' = \varphi'(m_0')$. Let $\varphi(x) = f(\dots)$, $\varphi'(x) = f'(\dots)$ in Σ, Σ'. Then $f(m_1(m_0/x)(n/x), \dots, m_{\rho(f)}(m_0/x)(n/x)) \equiv m(n/x) \equiv m'(n'/x) \equiv f'(m_1'(m_0'/x)(n'/x), \dots, m'_{\rho(f')}(m_0'/x)(n'/x))$ hence $f = f'$ and $m_\ell(m_0/x)(n/x) \equiv m_\ell'(m_0'/x)(n'/x)$ for all $\ell \in \{1 \dots \rho(f)\}$. Now, for ℓ s.th. $\delta(m_\ell(m_0/x)) = \min\{\delta(m_k(m_0/x)) : k = 1, \dots, \rho(f)\}$:

$$\delta(m_\ell(m_0/x)) = \min \delta(m_k(m_0/x)) = \delta(\varphi(m_0)) - 1 = \delta(m) - 1 \leq$$
$$\leq \delta'(m') - 1 = \delta'(\varphi'(m_0')) - 1 = \min \delta'(m_k'(m_0'/x)) \leq \delta'(m_\ell'(m_0'/x)).$$

hence we have

$$\begin{array}{l} m_\ell(m_0/x)(n/x) \equiv m_\ell'(m_0'/x)(n'/x) \\ \quad \delta(m_\ell(m_0/x)) \leq \delta'(m_\ell'(m_0'/x)) \end{array} \qquad \text{and more-over}$$

$\delta(m_e(m_0/x)) < \delta(m)$.
So by ind. hyp. there exist m'', m''' such that
$n = m''(m'''/x)$ and ... and $m''' \equiv n'$ ..... (1)
In addition, by associativety of substitution we get
$m(m''/x)(m'''/x) \equiv m(m''(m'''/x)/x) = m(n/x) \equiv m'(n'/x)$
So by lemma 2 and $m''' \equiv n'$ we get $m(m''/x) \equiv m'$ .... (2)
By (1) and (2) the induction step is proved.
end of "proof".

Due to the case $m = x$ and $n \neq x$ the lemma fails, but we can try to overcome this by formulating the premiss of the lemma as: $m(n/x) \equiv m'(n'/x)$ and $\delta(m+n) \leq \delta'(m')$, where $m+n$ is m whenever $m \neq x$ or $n = x$, and φx otherwise, where φ is the first symbol of n.
But again we cannot prove it, essentially by the same reason. Other formulations or inductions do not hold either.
Indeed:
<u>Counterexample</u> to the lemma, corolary, algorithm
Let $\Sigma \begin{cases} \varphi_1 x = f x \\ \varphi_2 x = g\,\varphi_3 x \\ \varphi_3 x = h\,x \end{cases}$ $\qquad \Sigma' \begin{cases} \varphi_1 x = f\,\varphi_2 x \\ \varphi_2 x = g\,x \\ \varphi_3 x = h\,x \end{cases}$
Then clearly
Σ and Σ' in standard form, and $\varphi_1\varphi_2 x \equiv \varphi_1\varphi_3 x$ (because $\{\Omega, f\Omega, fg\Omega, fgh\Omega\} \sim \{\Omega, f\Omega, fg\Omega, fgh\Omega\}$) and
$1 = \delta(\varphi_1 x) \leq \delta'(\varphi_1 x) = 2$,
but it is neither the case that $\varphi_1\varphi_2 x \equiv \varphi_1 x \wedge x \equiv \varphi_3 x$, nor the case that $\varphi_1 x \equiv \varphi_1 x \wedge \varphi_2 x \equiv \varphi_3 x$.

(43.2) Other attempts that failed.
Once we have demonstrated the incorrectness of the algorithm we can look for an improved version. Obviously the line
(7): <u>search for</u> k <u>such that</u> $\delta(\varphi_{i_1} \ldots \varphi_{i_k} x) = \delta'(\varphi'_{j_1})$
has to be changed, and accordingly lines (13), (14). We could try
$(\bar{7})$: <u>search for</u> minimal (k, L) <u>such that</u> $\delta(\varphi_{i_1} \ldots \varphi_{i_k} x) = \delta'(\varphi_{j_1} \ldots \varphi_{j_L} x)$
justified by the following lemma:

$$\varphi_{i_1}\cdots\varphi_{i_m}x \equiv \varphi'_{j_1}\cdots\varphi'_{j_n}x \quad \delta(\varphi_{i_1}) \le \delta(\varphi_{j_1}) \qquad \text{iff} \qquad \text{for some } (k,l) \text{ with } \delta(\varphi_{i_1}\cdots\varphi_{i_k}x)=\delta'(\varphi'_{j_1}\cdots\varphi'_{j_l}x)$$
$$\varphi_{i_1}\cdots\varphi_{i_k}x \equiv \varphi_{j_1}\cdots\varphi_{j_l}x$$
$$\varphi_{i_{k+1}}\cdots\varphi_{i_m}x \equiv \varphi_{j_{l+1}}\cdots\varphi_{j_n}x$$

which should be obviously true,
and may be there exists an argument (acceptability of Σ,Σ') by virtue of which the reviseted algorithm should terminate, that is an argument which garantees an bound for the length of the expressions to be tested.
However, for the revisited algorithm (line $(\bar{7})$ i.s.o. (7), and similar with line (13), (14)) we have the following <u>Counterexample</u> to the existence of such a bound:

Let $\Sigma = \begin{cases} \varphi_1 x = f(\varphi_1\varphi_2 x, x) \\ \varphi_2 x = f(\varphi_1 x, \varphi_2 x) \end{cases}$ $\qquad \Sigma' \begin{cases} \varphi_1 x = f(\varphi_2\varphi_1 x, x) \\ \varphi_2 x = f(\varphi_2\varphi_2 x, \varphi x) \\ \varphi x = f(x, \varphi x) \end{cases}$

Denote the states during the execution by the list EAT, denoted by $[\cdots]$, ~~and~~ the expressions to be tested (thus omitting the succesfull tests $f=f$ and $x=x$). Then there are infinitely many states of the form

$s(n)$: $[{}_0\bigwedge^n \varphi_1\varphi_2^i \equiv \varphi_2^i\varphi_1\ ,\ {}_0\bigwedge^n \varphi_2^i \equiv \varphi\varphi_2^{i-1}\varphi_1]\ ,\ \varphi_1\varphi_2^n \equiv \varphi_2^n\varphi_1$

where

${}_0\bigwedge^n \Phi_i$ denotes $\Phi_0 \wedge \cdots \wedge \Phi_{n-1}$,

so that there is no bound on the length of $\varphi_1\varphi_2^n \equiv \varphi_2^n\varphi_1$.
For
$s(0)$ is $\varphi_1 \equiv \varphi_1$, the initial state in testing $\Sigma \equiv \Sigma'$,
and from a state $s(n)$ we get $s(n+1)$ in some steps:

$s(n)$: $[{}_0\bigwedge^n \varphi_1\varphi_2^i \equiv \varphi_2^i\varphi_1\ ,\ {}_0\bigwedge^n \varphi_2^i \equiv \varphi\varphi_2^{i-1}\varphi]\ ,\ \varphi_1\varphi_2^n \equiv \varphi_2^n\varphi_1$

$\xrightarrow{(a)}$ $[{}_0\bigwedge^{n+1} \varphi_1\varphi_2^i \equiv \varphi_2^i\varphi_1\ ,\quad "\quad]\ ,\ \varphi_1\varphi_2^{n+1} \equiv \varphi_2^{n+1}\varphi_1\ ,\ \varphi_2^n \equiv \varphi\varphi_2^{n-1}\varphi_1$

$\xrightarrow{(b)}$ $[\quad "\quad ,\ {}_0\bigwedge^{n+1} \varphi_2^i \equiv \varphi\varphi_2^{i-1}\varphi_1]\ ,\quad "\quad ,[\cdots],[\cdots]$

$=$ $s(n+1)$.

where
in $\xrightarrow{(a)}$ due to $\delta(\varphi_1)=1$, $\delta(\varphi_2)=2$, $\delta'(\varphi_2)=2$, $\delta'(\varphi_1)=1$ the minimal k,l are $n+1, n+1$, so that we get the resulting line according to $\varphi_1 x = f(\varphi_1\varphi_2 x, x) \in \Sigma$ and $\varphi_2 x = f(\varphi_2\varphi_2 x, \varphi x) \in \Sigma'$.
in $\xrightarrow{(b)}$ due to the δ-values, again the minimal k,l yield the whole expressions. The genereted expressions are already

in the list EAT, so that they need not processed further.

Finally, it is even not true that equivalence*) of Σ and Σ' or more generally, equivalence of $\varphi_{i_1} \dots \varphi_{i_m} x$ with $\varphi'_{j_1} \dots \varphi_{j_n} x$ will garantee that minimal (k,l) are $\neq (m,n)$, so that with an additional test

(8b) <u>test for</u> $(k,l) \neq (m,n)$ <u>with escape</u> (print('NO'); <u>leave</u> <u>algorithm</u>)

a termination argument could hold.

This is shown by the first counterexample, where for the valid equivalence $\varphi_1 \varphi_2 x \equiv \varphi_1 \varphi_3 x$ neither $\varphi_1 x \equiv \varphi_1 x$ nor $\varphi_1 \varphi_2 x \equiv \varphi_1 x$ nor $\varphi_1 x \equiv \varphi_1 \varphi_3 x$ hold true.

Conclusion.

In this way we are not able to prove the decidability of the equivalence of acceptable rewriting systems.

---

*) The systems Σ, Σ' of the latter counterexample are ~~not~~ !? equivalent as can be proved by use of the kleene sequences.

For Σ the ~~Kleene's~~ sequence for $\varphi_1 x$ is $\langle \bar{f}^{(n)} \rangle$ where

$$\begin{cases} \begin{cases} f^{(0)} = S^0 \varphi_1 x = \varphi_1 x \\ h^{(0)} = \quad = \varphi_2 x \end{cases} \\ \begin{cases} f^{(n+1)} = S f^{(n)} = f \cdot f^{(n)}(h^{(n)}/x) \cdot x \\ h^{(n+1)} = S h^{(n)} = f \cdot f^{(n)} \cdot h^{(n)} \end{cases} \end{cases} \qquad \begin{cases} \begin{cases} \bar{f}^{(0)} = W f^{(0)} = \Omega \\ \bar{h}^{(0)} = W h^{(0)} = \Omega \end{cases} \\ \begin{cases} \bar{f}^{(n+1)} = W f^{(n+1)} = f \cdot \bar{f}^{(n)}(\bar{h}^{(n)}/x) \cdot x \\ \bar{h}^{(n+1)} = W h^{(n+1)} = f \cdot \bar{f}^{(n)} \cdot \bar{h}^{(n)} \end{cases} \end{cases}$$

and for Σ' we get similarly

$$\begin{cases} \begin{cases} f^{(0)} = S'^0 \varphi_1 x = \varphi_1 x \\ h^{(0)} = \quad = \varphi_2 x \end{cases} \\ \begin{cases} f^{(n+1)} = S' f^{(n)} = f \cdot f^{(n)}(h^{(n)}/x) \cdot x \\ h^{(n+1)} = S' h^{(n)} = f \cdot h^{(n)}(h^{(n)}/x) \cdot (fx)^n \cdot \varphi x \end{cases} \end{cases} \qquad \begin{cases} \begin{cases} \bar{f}^{(0)} = \Omega \\ \bar{h}^{(0)} = \Omega \end{cases} \\ \begin{cases} \bar{f}^{(n+1)} = f \cdot \bar{f}^{(n)}(\bar{h}^{(n)}/x) \cdot x \\ \bar{h}^{(n+1)} = f \cdot \bar{h}^{(n)}(\bar{h}^{(n)}/x) \cdot (fx)^n \cdot \Omega \end{cases} \end{cases}$$

For instance, the first terms of the seq. for Σ are

$\bar{f}^{(0)} = \Omega$

$\bar{f}^{(1)} = f\Omega$

$f^{(2)} = f f \Omega f \Omega \Omega x$

$f^{(3)} = f f f \Omega f \Omega \Omega f f \Omega x f \Omega \Omega x$

$f^{(4)} = f f f f \Omega f \Omega \Omega f f \Omega f f f \Omega f \Omega \Omega x f f \Omega x f \Omega \Omega x f \Omega \Omega f f f \Omega f$
$\Omega \Omega x f f \Omega x f \Omega \Omega$

(may 1974.)

Remark to chapter 4, i.p. to 4.3.

In a discussion with Bruno Courcelle after finisching this manuscript (i.e. the first 33 pages) it appeared that the reason of failure was the point that the ~~ter~~ two systems Σ and Σ' had to be merged into one system Σ'' over $\bar{\Phi}'' = \bar{\Phi} + \bar{\Phi}'$. When this is done, the formulation of the crucial lemma should be like the first formulation in (4.3.1). The equivalence relation $\equiv$ should take both expressions of <u>one</u> system Σ'' in stead of an expression of Σ at the left hand side and an expression of Σ' at the right hand side.

References:

"Semantics and axiomatics of a simple recursive language" by B. Courcelle and J. Vuillemin SIGACT 1974
(lemma 7 gives the crucial lemma).