SET RECURSION AND Π^1_2-LOGIC

by

Jean-Yves Girard
Université Paris VII

Dag Normann
University of Oslo

1. Introduction

1.1. The recent transformation of recursion theory into _generalized_ recursion theory essentially consists in

_i_ Retaining the formal aspect: One gives reasonable generalizations of concepts and results familiar from traditional recursion theory.

_ii_ Giving up the finitary aspect: The generalized computations are infinite processes.

If we think that the aim of recursion theory is to analyze infinite "lawlike" sets by means of finitary methods then something essential has been lost here: the infinite is no longer "analyzed" since it is taken as part of the primitive data. We have replaced _potential_ infinity by _actual_ infinity, and though the formal aspect of the original theory is successfully kept, its spirit is partly lost!

This evolution towards generalized recursion which started with Gödel's constructible (i.e. lawlike) sets, has ultimately led to _set recursion_ introduced in Normann[9]. What has been achieved is the transplantation of the general concepts of _lawlikeness_ from its original soil (integers, arithmetic) to the more general one of sets.

1.2. The situation with proof theory is quite similar: many of the improvements and generalizations which have taken place since the time of Gentzen have essentially retained the formal aspect of the theory: typically completeness, cut-elimination theorems... . But an essential difference is that the finitary aspect has not at all disappeared. This is due to the fact that the objects of proof theory must be graspable, i.e. at least theoretically they must be mechanically accessible. Recall that ω-proofs are always recursive in proof-theoretic applications.

However, the theory succeeds in (of course in a very slow process) its progressive transplantation from arithmetic to set-theory. This suggests the idea that the actual contents of generalized recursion theory is not so different from the one of recent proof-theory. But, if this is true, which part of proof-theory can match the increase of power that has been gained by allowing infinitary methods in recursion theory? The obvious answer is: the logical complexity of the new proof-theoretic concepts. As individuals the objects of proof-theory remain graspable (i.e. finitary, recursive in the familiar sense) while it is by no means mechanically checkable if such an object is a generalized proof, or "a generalized integer". ω-proofs, recursive ordinals are Π^1_1-complete concepts whereas β-proofs, recursive dilators (= denotation system) are Π^1_2-complete concepts. These Π^1_2-concepts (essential denotation systems) originally introduced in Girard [2] will in this paper be compared with notions coming from set-recursion.

1.3. Of course this opposition between set-recursion which was actual infinite and Π^1_2-logic which uses potential infinite (the actual infinite being relegated in the concept itself) is a bit artificial and rhetorical. Since the relation $\{e\}(a) \simeq b$ is essentially Π^1_1 there will be trees $T_e(a,b)$ uniformly recursive in sets of integers encoding a and b such that

$$T(a,b) \text{ is well-founded iff } \{e\}(a) \simeq b.$$

Hence set-recursion can also be expressed by means of traditional recursion together with a use of logical complexity in the concept. This renders the possibility of a link between set-recursion and Π^1_2-logic extremely probable, but this also casts a doubt as to the genuine interest of such a link!

When we write the relation $\{e|\}(a) \simeq b$ on a Π^1_2-normal form we hide the interesting well-founded objects, the computation trees, and it is within the computation trees that we find the structure of computations. Thus a link between set-recursion and dilators based on the computation trees would be less obvious but more interesting. Dilators are in many respects simpler than general algorithms and when the set of total set-recursive functions can be characterized using dilators we have gained information about these functions. In fact denotation systems (or dilators) have a very simple, regular structure and it seems that among the many ways of constructing total effective ordinal functions without loosing "the essential", dilators (denotation systems) are the ultimate simplification.

(To give a close example: It is well-known since Spector [13] that Bar-recursion of type 2 is a very powerful tool; this principle expresses the well-foundedness of a tree of finite sequences of type 2 objects given by means of a type 3 functional. Despite its theoretical importance no significant use of this principle has ever been made because of the rather anarchic, messy structure of the involved trees. <u>Induction on dilators</u> (see 3.12) is a principle which is presumably equivalent to Bar-recursion of type 2, but the well-founded structures have sufficiently been simplified and induction on dilators already has a lot of applications!)

Finally we arrive to the conclusion that the actual interest of a relation between set-recursion and Π^1_2-logic lies in a <u>significant simplification of the class of algorithms</u> leading to total functions.

1.4. The first significant applications of Π^1_2-logic to generalized recursion was done for successor admissibles [1]. Under reasonable conditions on the ordinal α [See the article of Ressayre [16] for optimum conditions] every Σ_1 function f over L_{α^+} is bounded by a recursive dilator for arguments $\geq \alpha$. The value of this reduction essentially lies in the very simple algebraic structure of denotation systems (= dilators) which make the computation of $F(x)$ effective in the argument x. More precisely the computation of a function of the kind $f(x)$ can be done as follows:

Starting with an oracle expressing $x \geq \alpha$ as a direct limit of integers $\underrightarrow{\lim}(x_i, f_{ij})$ we can

1. Compute the linear order $R = \underrightarrow{\lim}(F(x_i), F(f_{ij}))$
2. Introduce the ordinal $\|R\|$
3. Compute $f(x)$ by means of $\|R\|$-bounded quantifiers.

Of course only step 1. is effective whereas 2. and 3. are noneffective, but a great simplification has undoubtly been achieved.

Of course functions of the form $x \rightsquigarrow \|F(x)\|$ can be accepted as recursive functions in every possible acception of the word "recursive", because $F(x)$ can really be computed recursively in x! The result states that up to inessential things (bounded quantifiers needed to keep the <u>formal aspect</u> of recursion theory) there are no other recursive total functions.

1.5. Van de Wiele's theorem [14] proves that if f is uniformly Σ_1 over all admissibles, then for a well-chosen recursive dilator F we have

$$f(x) \leqslant F(x)$$

for all $x \in On$.

It is quite remarkable that this result

1 yields a similar majoration for set-recursive functions

2 proves that uniformly Σ_1 = set recursive for total set-functions.

The result 2 was unknown before Van de Wiele; this illustrates the simplifying power of Π^1_2-logic: These two notions of recursion were reduced to the same "skeleton", dilators.

Of course specialists were soon afterwards able to give direct proofs of 2 [12]. The direct proof is not so difficult which precisely enhances the fact that we want to stress: Π^1_2-logic increases our basic understanding of general recursion, it gives us a more graspable class of generalized recursive functions.

1.6. The main result of this paper will be a relativisation of Van de Wiele's argument to a given Δ_0 funciton h. The result is as follows: We construct by induction on dilators a hierarchy Φ_F of set-recursive functions relative to h

$$\Phi_F(x) = I(F,x,h)$$

and we prove that if g is set-recursive in h then $g(x) \leqslant \Phi_F(x)$ for a certain recursive dilator F and for all $x \in On$.

The hierarchy $\Phi_F(x)$ is obviously effective in the data F,x,h, hence this is a genuine generalization of Van de Wiele's theorem to relative recursion. (The case of Σ^g_1 functions is sketched in 5.10.)

1.7. As an application, if α is admissible and smaller than the first recursively Mahlo then it will be possible to express all functions which are Σ_1 over L_α by a hierarchy as in 1.6, with g a Δ_0-function such that $\alpha = \omega_1^g$. But our result has no corollary for the first recursively Mahlo.

1.8. Since this result was proved (May 1982) a new proof of 1.7. in the case of the first recursively inaccessible has been obtained; the method which is quite general makes use of inductive definitions and would obviously give the same results as in 1.7. Furthermore an analysis of the first recursively Mahlo by the same kind of hierarchies is given by the same inductive definitions-method and this new result is a proper extension of the main results of this paper; see [5].

2. DENOTATION SYSTEMS

2.1 Some examples

Denotation systems are general Cantor-Normal-Form-type of representations. Before giving the definition we will consider three examples.

The first will be

$$F_1(x) = 2^x$$

where x is an ordinal.

If $y < 2^x$ there will be a unique ascending sequence $x_0 < x_1 < \ldots < x_{n-1} < x$ such that

$$y = 2^{x_{n-1}} + 2^{x_{n-2}} + \ldots + 2^{x_0}$$

Since the sequence $(x_0,\ldots,x_{n-1})$ describes the number y and all numbers less than 2^x can be described this way we may call

$$(x_0,\ldots,x_{n-1}) \rightsquigarrow 2^{x_{n-1}} + \ldots + 2^{x_0}$$

a denotation system for F_1. This will, however, not be completely according to our formalism.

Our second example will be

$$F_2(x) = x^2.$$

If $y < F_2(x)$ we can write y uniquely in the form

$$y = x \cdot u_1 + u_2$$

where $u_1, u_2 < x$.

If we list the coefficients in increasing order there are three ways of denoting ordinals $<x^2$:

i $\quad y = x \cdot x_0 + x_1 \quad (x_0 < x_1 < x)$

ii $\quad y = x \cdot x_1 + x_0 \quad (x_0 < x_1 < x)$

iii $\quad y = x \cdot x_0 + x_0 \quad (x_0 < x)$

If we use codes for these three ways of organizing the coefficients we again obtain a way of denoting all ordinals $y < F_2(x)$ using ordinals less than x.
Formaly we write

$$(1;x_0,x_1;x) = x\cdot x_0 + x_1$$
$$(2;x_0,x_1;x) = x\cdot x_1 + x_0$$
$$(0;x_0;x) = x\cdot x_0 + x_0$$

It is not essential how we choose the codes or indices 1,2,0. In this example we have followed a standard strategy: Take a canonical prototype of the form we want to code and use the value as the index

<u>i</u> $x\cdot x_0 + x_1$: Prototype $x_0 = 0$, $x_1 = 1$, $x = 2$, value $x\cdot x_0 + x_1 = 1$

<u>ii</u> $x\cdot x_1 + x_0$: Prototype $2\cdot 1 + 0 = 2$

<u>iii</u> $x\cdot x_0 + x_0$: Prototype $x_0 = 0$, $x = 1$ gives value 0.

Our third example will be

$$F_3(x) = (1+x)^x$$

If $y < F_3(x)$ then there are unique numbers $u_1 > \ldots > u_{k-1}$, $v_1,\ldots,v_{k-1}$, all $<x$, such that

$$y = (1+x)^{u_1}(1+v_1) + \ldots + (1+x)^{u_{k-1}}(1+v_{k-1})$$

Again any number $y < (1+x)^x$ can be uniquely denoted by x, an increasing sequence

$$x_0 < \ldots < x_{n-1} < x$$

and an index coding how the numbers $x_0,\ldots,x_{n-1}$ are distributed as coefficients and exponents. We will regard one example

$$y = \omega^{17}\cdot 18 + \omega\cdot 17 + 13,$$

or written on our form

$$y = (1+\omega)^{17}(1+17) + (1+\omega)^1(1+16) + (1+\omega)^0(1+12).$$

The "coefficients" are $0,1,12,16,17$, and the canonical prototype is $(x = 5)$

$$y_0 = (1+5)^4(1+4) + (1+5)^1(1+3) + (1+5)^0(1+2) = 6506.$$

Thus the denotation for $y < F_3(\omega)$ will be

$$(6506;0,1,12,16,17;\omega).$$

We consider $(1+x)^x$ instead of x^x because it will be impossible to find <u>unique</u> denotations for all ordinals $< x^x$.

By our convention the index and the length of the sequence of coefficients will determine the algebraic form that we have in mind when the general normal form is given. It will in general not be possible to recapture the full denotation system from such pairs (c,n).

2.2. Denotation Systems

We will now give a set of axioms for denotation systems. It is easily checked that our examples from 2.1 satisfy these axioms.

Definition 2.1

Let $F:On \to On$.

A denotation-system D for F is a class of ordinal denotations

$$y = D((c;x_0,\ldots,x_{n-1};x))$$

for all ordinals $y < F(x)$ such that

I $\quad x_0 < \ldots < x_{n-1} < x$

II $\quad$ If $y < F(x)$ then y has a unique denotation $(c;x_0,\ldots,x_{n-1};x)$

III $\quad$ If $(c;x_0,\ldots,x_{n-1};x)$ is a denotation and $y_0 <\ldots< y_{n-1} < y$ then $(c;y_0,\ldots,y_{n-1};y)$ is a denotation.

IV If $D((c_1;x_0,\ldots,x_{n-1};x)) \leq D((c_2;x'_0,\ldots,x'_{m-1};x))$,

if

$$y_0 < \ldots < y_{n-1} < y,\ y'_0 < \ldots < y'_{m-1} < y$$

and if

$x_i \leq x'_j \iff y_i \leq y'_j$ and $x_i \geq x'_j \iff y_j \geq y'_j$

for $i < n$, $j < m$

then

$D((c_1;y_0,\ldots,y_{n-1};y)) \leq D((c_2;y'_0,\ldots,y'_{m-1};y))$.

Remark 2.2

a In a denotation $(c;x_0,\ldots,x_{n-1};x)$ we will call c the <u>index</u> and $x_0,\ldots,x_{n-1}$ the <u>coefficients</u> of the denotation.

b Normally $(c;x_0,\ldots,x_{n-1};x)$ will be used both for the denotation and for the denoted ordinal, i.e. we drop the D.

c The index c represents some "algebraic" way of describing y in terms of $x_0,\ldots,x_{n-1},x$. The unicity II assumes that we have some "canonical form", III means that this "form" always gives a meaning and IV states that in order to decide the relative value of

$$z_1 = (c_1;x_0,\ldots,x_{n-1};x)$$

and

$$z_2 = (c_2;x'_0,\ldots,x'_{m-1};x)$$

we only have to consider c_1,c_2 and the relations

$$\{(i,j);x_i \leq x'_j\} \text{ and } \{(i,j);x_i \geq x'_j\}$$

d The axioms I-IV say nothing about which objects the indices may be, and there will be many equivalent denotatin systems. At some places we will make use of this freedom to gain notational simplicity. On the other hand any system may be represented in standard form as described below:

Definition 2.3

a A denotation system D is in standard form if whenever $(c;x_0,\ldots,x_{n-1};x)$ is a denotation then

$$c = D((c;0,\ldots,n-1;n))$$

b The trace $Tr(D)$ of the denotation system D is the set

$$\{(c,n);\ (c;0,\ldots,n-1;n) \text{ is a denotation}\}$$

when D is in standard form.

In our examples F_2 and F_3 we gave the denotation-systems in standard form.

The axioms I-IV are quite powerfull, as we will see later. First we will show that the denotations will be monotone in the coefficients.

Lemma 2.4

Let D be a denotation system and let $(c;x_0,\ldots,x_{n-1};x)$ be a denotation where $x_i + 1 < x_{i+1}$. Then $(c;x_0,\ldots,x_i,\ldots,x_{n-1},;x)$ $< (c;x_0,\ldots,x_i+1,\ldots,x_{n-1};x)$.

Proof

Assume not. By II we have

$$\star \qquad (c;x_0,\ldots,x_i+1,\ldots,x_{n-1},;x) < (c;x_0,\ldots,x_i,\ldots,x_{n-1};x).$$

By III $(c;\omega\cdot x_0,\ldots,\omega\cdot x_i+m,\ldots,\omega\cdot x_{n-1};\omega\cdot x)$ are denotations for each m, and by $\star$ and IV we have

$$(c;\omega\cdot x_0,\ldots,\omega\cdot x_i+m+1,\ldots,\omega\cdot x_{n-1};\omega\cdot x)$$
$$< (c;\omega\cdot x_0,\ldots,\omega\cdot x_i+m,\ldots,\omega\cdot x_{n-1};\omega\cdot x)$$

for each m. We will then get an infinite descending sequence of ordinals, which is absurd.

Another important consequence is

Theorem 2.5

Any denotation-system is uniquely determined by its restriction to the natural numbers.

Proof

Let D be a denotation-system for F and let x be given. Let

$$D_x = \{(c;x_0,\dots,x_{n-1};x);\ x_0 <\dots< x_{n-1} < x \ \&\ (c,n) \in Tr(D)\}.$$

Here we regard $(c;x_0,\dots,x_{n-1};x)$ just as a formal expression, since we want to recapture its value.

We give D_x the following ordering:
Let $(c_1;x_0,\dots,x_{n-1};x)$ and $(c_2;x'_0,\dots,x'_{m-1};x)$ be elements of D_x. Let

$$\{z_0 <\dots< z_{t-1}\} = \{x_0,\dots,x_{n-1}\} \cup \{x'_0,\dots,x'_{m-1}\}.$$

Let σ,τ be such that

$$x_i = z_{\sigma(i)} \ (i < n) \quad \text{and} \quad x'_j = z_{\tau(j)} \ (j < m)$$

We let

$$(c_1;x_1,\dots,x_{n-1};x) \leqslant_{D_x} (c_2;x'_1,\dots,x'_{m-1};x)$$

if and only if

$$D((c_1;\sigma(0),\dots,\sigma(n-1);t)) \leqslant D((c_2;\tau(0),\dots,\tau(m-1);t))$$

By axiom IV the ordering $\leqslant_{D_x}$ is the same as the ordering between the denoted ordinals. Since $\leqslant_{D_x}$ is definable from x and $D\restriction\mathbb{N}$ we have proved the theorem.

Remark 2.6

a A system defined on $\mathbb{N}$, satisfying I-IV restricted to $\mathbb{N}$ and satisfying monotonicity in the coefficients is called a pre-denotation-system. Given a pre-denotation-system we may try to construct a denotation-system like in the proof of Theorem 2.5. The problem is that $\leqslant_{D_x}$ may not be a well-ordering. However, if $\leqslant_{D_{\aleph_1}}$ is a well-ordering then all $\leqslant_{D_x}$ will be well-orderings and we are dealing with a denotation-system.

b If $F(n) \in \mathbb{N}$ whenever $n \in \mathbb{N}$ we call the system weakly finite. Weakly finite systems are called recursive etc. when their restrictions to $\mathbb{N}$ are so.

c The proof of Theorem 2.5 shows that denotation-systems represent a finitary approach to their functions. Thus functions permitting a denotation-system have a kind of continuity-property.

d There is a close connection between denotation-systems and certain functors on the ordinals commuting with pull-backs and direct limits. These functors are called Dilators and are treated in full detail in Girard [2]. Dilators are in fact isomorphic to denotation-systems; the two notions are different presentations of the same basic material. For that reason it will be possible to avoid the use of dilators in this paper. For a deeper understanding, however, we find that dilators are as important here as linear operators are to linear algebra.

2.3. The sum of denotation-systems

Let us once more consider our examples from section 2.1, $F_2(x) = x^2$ and $F_3(x) = (1+x)^x$. Let $F_4(x) = x^2 + (1+x)^x$ and let $y < F_4(x)$. Then either $y < x^2$ or $y = x^2 + y'$ for some $y' < (1+x)^x$. In the first case we use the denotation-system for x^2 to denote y. In the other case we take the $(1+x)^x$-denotatin for y'. If we code into the index which system we use, this gives us a denotation-system $D_4 = D_2 + D_3$ for F_4.

The method used here is general and can be used for any well-ordered sequence of denotation-systems.

Definition 2.7

Let $\{D_i\}_{i<\beta}$ be denotation-systems for $\{F_i\}_{i<\beta}$. We let

$$D = \sum_{i<\beta} D_i$$

be defined as follows:

If $(c,n) \in Tr(D_i)$ then we let $\langle c,i\rangle$ be an index for D and

$$D((\langle c,i\rangle; x_0,\dots,x_{n-1}; x)) = \sum_{j<i} F_j(x) + D_i((c; x_0,\dots,x_{n-1}; x)).$$

Remark 2.8

If each D_i are in standard form then we get D in standard form if we use $\sum_{j<i} F_j(n) + c$ instead of $\langle c,i\rangle$ in defining D.

Definition 2.9

a The denotation-system $\underline{0}$ is the empty system for the constant 0 function.

b The denotation-system $\underline{1}$ is the system with one denotation $(0;;x) = 0$.

c A non-zero denotation-sytem D is called connected if D is not the sum of two systems $\neq \underline{0}$.

d If D, D' are denotation-systems in standard from and D' is a subfunction of D (i.e. $\mathrm{graph}(D') \subseteq \mathrm{graph}(D)$) we say $D' \leqslant D$.

e If D is a denotation-system then let

$$I_D = \{D'; D' \leqslant D\}.$$

Remark 2.10

a Connected systems correspond to perfect dilators in Girard [2].

b $\underline{1}$ is connected and $I_{\underline{1}} = \{\underline{0}, \underline{1}\}$.

Lemma 2.11

a $D' \in I_D$ if and only if for some D'' we have that $D = D' + D''$.

b If $D_1, D_2 \in I_D$ are systems for F_1, F_2 resp., then

$$D_1 \leqslant D_2 \iff F_1(\omega) \leqslant F_2(\omega)$$

Proof

a If we assume that D, D' and D'' are in standard form this is trivial.

b $\Rightarrow$ is trivial so assume that $F_1(\omega) \leqslant F_2(\omega)$. Let $(c; x_0, \ldots, x_{n-1}; x)$ be a D_1-denotation. Since $D_1 \leqslant D$ it is also a D-denotation with the same value. Assume that it is not a D_2-denotation. Then $(c; k_0, \ldots, k_{n-1}; \omega)$ is never a D_2-denotation. Since $D_2 \leqslant D$ there cannot be any other D_2-denotation $(c_1; k_0', \ldots, k_{m-1}'; \omega)$ such that

$$(c_1; k_0', \ldots, k_{m-1}'; \omega) \geqslant (c; k_0, \ldots, k_{n-1}; \omega)$$

since then D_2 must either fail to be a subfunction of D or fail to be a denotation-system, by not being onto $F_2(\omega)$. Consequently

$$F_2(\omega) \leqslant (c; k_0, \ldots, k_{n-1}; \omega) < F_1(\omega)$$

which contradicts the assumption.

An important consequence is the first decomposition theorem:

Theorem 2.12

Let D be a denotation-system. Then D can uniquely be given as the sum $\sum_{i<\beta} D_i$ of connected denotation-systems.

Proof

By Lemma 2.11.b the set I_D is well-ordered by the ordering $\leqslant$; let $\{F_i\}_{i \leqslant \beta}$ be I_D ordered by $F_i \leqslant F_j \iff i \leqslant j$. For $j < \beta$ let $D_i = F_{i+1} - F_i$ (by Lemma 2.11a this makes sense). Each D_i is clearly connected and $F_i = \sum_{j<i} D_j$ for all $i \leqslant \beta$. Moreover, if $D = \sum_{j<\alpha} D'_j$ then each $\sum_{j'<j} D'_{j'} = F_i$ for some F_i so the alternative decomposition will be coarser than the one we defined.

Remark 2.13

We call this decomposition of a system decomposition into sums.

2.4. Connected systems of denotations

When we decompose a disconnected system into sums we see that the trace $Tr(D)$ may be stratified into layers according to which component the element comes from. If (c_1,n) and (c_2,m) comes from D_i, D_j resp. with $i < j$ then

$$(c_1;x_0,\ldots,x_{n-1};x) < (c_1;x'_0,\ldots,x'_{m-1};x)$$

for all choices of $x_0,\ldots,x_{n-1},x'_0,\ldots,x'_{m-1},x$.

For a connected system the situation is different, there the values for the different indices will be interwoven. This is in fact the reason why they are called connected.

Lemma 2.14

Let $D \neq \underline{1}$ be a connected system for F. Let x be a limit ordinal and let $(c,n) \in Tr(D)$. Then

$$\{(c;x_0,\ldots,x_{n-1};x);\ x_0 <\ldots< x_{n-1} < x\}$$

is cofinal in $F(x)$.

Proof

Let

$$X = \{(c',m) \in Tr(D);\ \exists y_0 < \ldots < y_{m-1} < x,\ x_0 < \ldots < x_{n-1} < x$$
$$((c';y_0,\ldots,y_{m-1};x) \leq (c;x_0,\ldots,x_{n-1};x))\}$$

Since the value of a denotation is monotone in the coefficients we may without loss of generality assume that $x_0 > y_{m-1}$ in defining X. By axiom IV this means that X is independent of the limit ordinal x.

Claim

For any ordinal y

$$\{(c';y_0,\ldots,y_{m-1};y);\ y_0 < \ldots < y_{m-1} < y \ \&\ (c',m) \in X\}$$

is an initial segment of $F(y)$.

Proof of claim:

If y is a limit ordinal this holds by the definition of X, since we may use y instead of x in defining X.

If y is a successor ordinal, $(c',m) \in X$ and $(d;x_0,\ldots,x_{t-1};y) < (c';y_0,\ldots,y_{m-1};y)$ then

$$(d;x_0,\ldots,x_{t-1};y+\omega) < (c';y_0,\ldots,y_{m-1};y+\omega)$$

so $(d,t) \in X$ for X defined from $y + \omega$. This proves the claim.

By the claim $D \restriction X \leq D$. Since D is connected we must have $D \restriction X = D$ and $X = Tr(D)$. The lemma then follows from the definition of X.

If we have two denotations for ordinals $u,v < (1+x)^x$ we can decide the relative order of u and v by looking at the coefficients and exponents. The one with the largest exponent is largest.

If they are the same we regard the corresponding coefficients. If they also are the same look at the next exponents etc.

Suitably modified this strategy can be used for all connected denotation-systems. We will neither prove nor need the full result, but the general idea will be clear from what we do, and details can be found in Girard [2].

<u>Definition 2.15</u>

Let $D \neq \underline{1}$ be a connected denotation-system for F and let $(c,n) \in Tr(D)$. Then $(c;0,2,\ldots,2n-2;2n)$ is a denotation. Let $i \leq n-1$ and let

$$a_i = (c;0,2,\ldots,2i/2i+1,\ldots,2n-2;2n)$$

i.e. we replace $2i$ with $2i+1$.

If $a_i > a_j$ we say that i is <u>more important</u> than j. Let $i_{c,n}$ be the <u>most important index</u>. In a denotation

$$(c;x_0,\ldots,x_{n-1};x)$$

we will call $x_{i_{c,n}}$ the <u>most important coefficient</u>.

<u>Remark 2.16</u>

<u>a</u> When $D \neq \underline{1}$ is connected and $(c,n) \in Tr(D)$ then $n > 0$.

<u>b</u> If $u = (c;u_0,\ldots,u_{n-1};x)$ and

$$v = (c;v_0,\ldots,v_{n-1};x)$$

and if we have

$$u_i < v_i,\ u_j > v_j \quad \text{and} \quad t \neq i,j \Rightarrow u_t = v_t$$

then

$$u < v \iff j \text{ is more important than } i.$$

The ordering "more important than" is a strict ordering of $\{0,\ldots,n-1\}$ i.e. it defines a permutation of $\{0,\ldots,n-1\}$.

We can decide the relative order of two denotations by looking at the relative order of the coefficients with falling importance. A fragment of this result is the following:

Lemma 2.17

Let $D \neq \underline{1}$ be a connected denotation-system and use the notation of Definition 2.15. Let

$$u = (c; u_1, \ldots, u_{n-1}; x)$$
$$v = (c'; v_0, \ldots, v_{m-1}; x)$$

be two denotations, let $p = i_{c,n}$ and $q = i_{c',m}$. If $u \leq v$ then $u_p \leq v_q$.

Proof

In order to obtain a contradiction assume that $u \leq v$ but $u_p > v_q$. Without loss of generality we may assume that x is a limit ordinal, $u_p \geq v_q + \omega$ and $v_{j+1} \geq v_j + \omega$ (If we multiply the coefficients and x by ω ($\omega \cdot x$ etc.) the relative order will not be altered).

If we reduce some of the coefficients in a denotation we reduce its value, so

$$(c; u_0, \ldots, u_p, u_p+1, \ldots, u_p+(n-1-p); x) \leq u.$$

If we increase the value of the most important coefficient and decrease any of the other coefficients then we will increase the value. If we let

$$s = (c'; v_0, \ldots, v_{q-1}, v_q+k, v_q+k+1, \ldots, v_q+2k; x)$$

where $k = m - q - 1$, then we have $s \geq v$. Since $v_q + 2k < u_p$ axiom IV gives that whenever $u_p \leq z_p < \ldots < z_{n-1} < x$ we have

$$(c; u_0, \ldots, u_{p-1}, z_p, \ldots, z_{n-1}; x) < s.$$

But since $u_0, \ldots, u_{p-1}$ are of small importance compared to z_p we see that

$$(c;z_0,\ldots,z_{n-1};x) < s$$

for all $z_0 <\ldots< z_{n-1} < x$.

This contradicts Lemma 2.14 and this lemma is proved.

If we fix the value of the most important coefficient to $y < x$ then the set

$$\{(c;x_0,\ldots,x_{p-1},y,x_{p+1},\ldots,x_{n-1};x) \ ; \ (c,n) \in Tr(D)$$
$$\& \ p \ \text{is the most important index} \ i_{c,n}\}$$

forms an interval of ordinals $< F(x)$. This is a consequence of Lemma 2.17. $F(x)$ is the union of these intervals and we may think of the denotations leading to ordinals in each interval as components of the system for $F(x)$. We get the components of D by fixing y and let $x > y$ vary. This leads us to the following concepts:

<u>Definition 2.18</u>

Let $D \neq \underline{1}$ be a connected denotation-system. Let y,x be ordinals.

<u>a</u> Let

$$X_{y,x} = \{(c;u_0,\ldots,u_{p-1},y,y+1+x_{p+1},\ldots,y+1+x_{n-1};y+1+x); \\ (c,n) \in Tr(D),\ p = i_{c,n},\ u_0 <\ldots< u_{p-1} < y \\ \text{and} \ x_{p+1} <\ldots< x_{n-1} < x\}.$$

<u>b</u> Let $\Pi_{y,x}$ map the interval $X_{y,x}$ order-preservingly onto the ordinal

$$F^y(x) = \text{Ordertype of } X_{y,x}.$$

<u>c</u> Let D^y be the denotation-system for F^y defined by

$$(\langle c,u_0,\ldots,u_{p-1}\rangle_y;x_{p+1},\ldots,x_{n-1};x)$$

denotes

$$\Pi_{y,x}((c;u_0,\ldots,u_{p-1},y,y+1+x_{p+1},\ldots,y+1+x_{n-1};y+1+x))$$

we call D^y the <u>y'th component</u> of D.

<u>Remark 2.19</u>

<u>a</u> It is easily verified that D^y is a denotation-system for F^y.

<u>b</u> The decomposition in Girard [2] corresponds to

$$E^y = \sum_{y'<y} D^{y'}$$

<u>c</u> We have not described D^y in standard form. The index for the standard form of

$$(\langle c, u_0, \ldots, u_{p-1}\rangle_y; x_{p+1}, \ldots, x_{n-1}; x)$$

will be

$$c' = \Pi_{y,n-p}((c; u_0, \ldots, u_{p-1}, y, y+1, \ldots, y+n-p-1; y+n-p))$$

which is the value of

$$(\langle c, u_0, \ldots, u_{p-1}\rangle_y; 0, 1, \ldots, n-p-2; n-p-1)$$

<u>Definition 2.20</u>

<u>a</u> If $D = \sum_{i<\beta} D_i$ is the decomposition of a non-connected denotation-system D into sums, then $D_i \prec D$ for all $i < \beta$.

<u>b</u> If $D \neq \underline{1}$ is connected then $D^y \prec D$ for each $y \in On$.

<u>c</u> $\prec$ is the minimal transitive ordering satisfying <u>a</u> and <u>b</u>.

The second decompostition theorem states that $\prec$ is well-founded. In a sense this means that any denotation-system can be constructed from $\underline{1}$ by sum and a special kind of diagonalisation at cofinality On. We will not explore this aspect further here.

<u>Lemma 2.21</u>

Let $D \neq \underline{1}$ be a connected system, F^y as in Definition 2.18. Let y, a be two ordinals such that $a > 0$ and $y < \omega^a$. Then

$$F^y(\omega^a) < F(\omega^a)$$

Proof

If $y < \omega^a$ then $y + 1 + \omega^a = \omega^a$, so

$$X_{y,\omega^a} \subseteq F(\omega^a)$$

Moreover, any denotation for an ordinal $< F(y+1+\omega^a) = F(\omega^a)$ where the most important coefficient is $> y$ will dominate X_{y,ω^a}. There are clearly such denotations, so X_{y,ω^a} is bounded in $F(\omega^a)$. Thus $F^y(\omega^a) = \text{Order-type } (X_{y,\omega^a}) < F(\omega^a)$.

Theorem 2.22

The ordering $\prec$ of Definition 2.20 is well-founded.

Proof

Assume not. Then there is a descending sequence $\{D_i\}_{i\in N}$ where $D_{i+1} \prec D_i$ by a or b of 2.20.

If $D_i = \sum_{j<\beta} (D_i)_j$ then $D_{i+1} = (D_i)_{j_i}$ for some $j_i < \beta$. If D_i is connected then $D_{i+1} = D_i^{y_i}$ for some $y_i \in On$. Let a dominate all the y_i's in question. Then $\{F_i(\omega^a)\}_{i\in N}$ will be a descending sequence of ordinals, where F_i is the function associated with D_i.

Remark 2.23

In Girard [2] the decomposition is formulated different, and the predecessor ordering will be linear. His predecessors correspond to the Kleene-Brouwer order of $\prec$ in a certain sense. We will discuss this ordering at the end of paragraph 3.

3. INTERPRETATION OF TREES

3.1 A representation of Π^1_2-sets

Π^1_2-logic is a collection of concepts of complexity Π^1_2 like β-proofs, dilators, homogeneous trees etc. and the mathematics thereof. Denotation systems together with the equivalent concept Dilators is one of the possible paths to Π^1_2-logic. We will not treat all the concepts of Π^1_2-logic, only establish the link between Π^1_2-sets and denotation systems.

Let $A \subseteq \mathbb{N}$ be a Π^1_2-set and let B be Σ^1_1 such that

$$n \in A \iff \forall g \in \mathbb{N}^{\mathbb{N}} (n,g) \in B$$

Then there is a recursive map $n,g \rightsquigarrow T_{n,g}$ where $T_{n,g}$ is a tree on $\mathbb{N}$ such that

$$(n,g) \in B \iff T_{n,g} \text{ is not well founded.}$$

We use the letter f for elements of $\mathrm{On}^{\mathbb{N}}$. We then have

$$(n,g) \in B \iff \forall f \, \exists \sigma \in T_{n,g} \, \exists t < \mathrm{lh}(\sigma)-1$$
$$(f(\bar{\sigma}(t)) \leq f(\bar{\sigma}(t+1)))$$

where $\bar{\sigma}(t)$ is the sequence $(\sigma(0),\ldots,\sigma(t-1))$ identified with its sequence number.

From $f: \mathbb{N} \to \mathrm{On}$ we may define $g_f: \mathbb{N} \to \mathbb{N}$ as follows

$$g_f(0) = \mu j \, (f(j+1) \geq f(j)), \quad h_f(0) = g_f(0)+1$$

$$g_f(n+1) = \mu j \, (f(h_f(n)+j+1) \geq f(h_f(n)+j)$$

$$h_f(n+1) = h_f(n) + g_f(n+1)+1 .$$

(An explanation is in order: $g_f(0)$ is the number of steps f is decreasing. $g_f(1)$ is the number of steps from then on that f is decreasing etc.)

It is easily seen that $f \rightsquigarrow g_f$ is a projection of $\mathrm{On}^{\mathbb{N}}$ onto $\mathbb{N}^{\mathbb{N}}$.

We then have

$$n \in A \iff \forall f_1 \ \forall f_2 \ \exists \sigma \in T_{n,g_{f_2}} \ \exists t < lh(\sigma)-1$$

$$(f_1(\bar{\sigma}(t)) \leqslant f_1(\bar{\sigma}(t+1)))$$

Any sequence f may be split into

$$f_1(i) = f(2i), \quad f_2(i) = f(2i+1)$$

Moreover, when we for each σ decide if $\sigma \in T_{n,g_{f_2}}$ or if $\exists t < lh(\sigma)-1 \ (f_1(\bar{\sigma}(t)) \leqslant f_1(\bar{\sigma}(t+1)))$, we use only the relation

$$\{(i,j);\ f(i) \leqslant f(j)\},$$

not the actual values f take.

This gives us the following result:

<u>Theorem 3.1</u>

Let A be Π^1_2. Then there are trees $\{S_n\}_{n \in N}$ of finite sequences of ordinals such that

<u>i</u> $\forall n \ \forall \sigma, \tau \in On^N \ (lh(\sigma) = lh(\tau) \wedge \forall i,j < lh(\sigma)$
$(\sigma(i) \leqslant \sigma(j) \iff \tau(i) \leqslant \tau(j)) \Rightarrow (\sigma \in S_n \iff \tau \in S_n))$

<u>ii</u> $n \in A \iff \forall f \ \exists t \ \neg S_n(\bar{f}(t))$

<u>iii</u> $S_n \restriction \mathbb{N}^{\underset{\smile}{\mathbb{N}}}$ is primitive recursive uniformly in n.

<u>Remark 3.2</u>

<u>a</u> $On^{\underset{\smile}{\mathbb{N}}}$ and $N^{\underset{\smile}{\mathbb{N}}}$ are standard notations for the sets of finite sequences from On and $\mathbb{N}$.

<u>b</u> The property <u>i</u> says that if σ and τ are order isomorphic then $\sigma \in S_n \iff \tau \in S_n$.

A tree satsifying this property is called <u>order invariant</u>.

<u>c</u> Without loss of generality we may assume that the tree $S_n \restriction m^{\underset{\smile}{\mathbb{N}}}$ is well-founded, i.e. finite. In the next section we will show how order-invariant trees with this extra property

correspond to weakly finite predenotation-systems, and they correspond to denotation-systems if and only if the tree is well-founded.

d Jervell [6] introduces a class of trees called $[\alpha,\beta[$-homogeneous trees, $\alpha < \beta \leq \infty$, and they correspond to a subclass of the denotation system. Our well-founded order invariant trees will be $[0,\infty[$-homogeneous. An $[\alpha,\beta[$-homogeneous tree is a tree of sequences of ordinals where each koordinate is bounded by β and which is order invariant for koordinates $\geq \alpha$. All the subtrees that we study in decomposing the order-invariant trees will be $[\alpha,\infty[$-homogeneous for some α.

e The tree S_n is a variant of the tree constructed by Shoenfield [11] and it can be used to prove the Shoenfield absoluteness theorem.

3.2 Order-invariant trees

Definition 3.3

Let S be a non-empty order-invariant tree.

a If x is an ordinal, let

$$S_x = \{\sigma \in S;\ \forall i < lh(\sigma)(\sigma(i) < x)\}$$

b Let $<_x$ be the Kleene-Brouwer ordering on S_x. If S_x is well-founded, let $\|\ \|_x$ be the ordinal norm for $<_x$.

c If S_x is well-founded, $\sigma \in S_n$ and

$$\{x_0 < \cdots < x_{n-1}\} = \{\sigma(i);\ i < lh(\sigma)\}$$

then we let

$$D_S((c;x_0,\ldots,x_{n-1};x)) = \|\sigma\|_x$$

where c is obtained as follows: Let $\tau: lh(\sigma) \to n$ such that $\sigma(i) = x_{\tau(i)}$. Then $c = \|\tau\|_n$.

Lemma 3.4

a If each S_n is well-founded then D_S is a pre-denotation-system.

b D_S is a denotation system if and only if S is well-founded if and only if each S_x is well-founded.

Proof

If $y \leq x$ then $S_y \subseteq S_x$ so

S_x is well-founded $\Rightarrow$ S_y is well-founded.

Both a and b then follow from

Claim

D satisfies the axioms for a denotation system for the ordinals x such that S_x is well-founded.

Proof of claim

I and II are trivial.

III follows from the fact that S is order invariant.

IV follows from the fact that the Kleene-Brouwer ordering is invariant under order-preserving transformations on subsets of On.

Remark 3.5

a The constructions in Theorem 3.1 and Lemma 3.4 are effective so we have reduced any Π^1_2-relation on ω to the set of recursive denotation systems. This set is itself Π^1_2 and thus complete Π^1_2.

b The reduction of well-founded order-invariant trees to denotation-systems cannot be reversed; there are denotation systems that do not correspond to such trees. x^2 and 2^x correspond to such trees but x^2+2^x does not.

$(x^2 \simeq \{\sigma;\ \mathrm{lh}(\sigma) \leq 2\})$

$(2^x \simeq \sigma;\ \sigma$ is decreasing$\})$

3.3 The decomposition of D_S

In this section we will describe the decomposition of D_S. This description will be used in later paragraphs where we will study the connection between denotation-systems and set-recursive functions.

Definition 3.6

Let S be a well-founded order-invariant tree on On. Let $\sigma \in S$.

a Let $S_\sigma = \{\tau; \sigma^* \tau \in S\}$ (where * is concatenation).

b Let $m(\sigma) = \max\{\sigma(0), \ldots, \sigma(n-1)\}+1$ where $n = lh(\sigma)$.

c Let $S_{\sigma,x} = \{\tau \in S_\sigma; \forall i < lh(\tau)(\tau(i) < m(\sigma)+x\}$

d Let $\| \ \|_{\sigma,x}$ be the Kleene-Brouwer norm on $S_{\sigma,x}$.

e Let $\|\sigma, x\|$ be the order-type of $S_{\sigma,x}$ under the K.-B. ordering.

We will construct denotation-systems D_σ corresponding to the norms $\| \ \|_{\sigma,x}$.

If $\tau \in S_\sigma$ we separate τ into two parts:

$$\tau_1(i) = \begin{cases} \tau(i) & \text{if } \tau(i) < m(\sigma) \\ \text{undefined otherwise.} \end{cases}$$

$$\tau_2(i) = \begin{cases} x & \text{if } \tau(i) = m(\sigma)+x \text{ for some } x \\ \text{undefined otherwise.} \end{cases}$$

S_σ will be order-invariant with respect to τ_2 but not with respect to τ_1. Thus when we construct a denotation for $\|\tau\|_{\sigma,x}$ we will code τ_1 into the index and τ_2 into the coefficients. This leads us to the following:

Definition 3.7

Let S be a well-founded order-invariant tree. Let $\sigma \in S$, $x \in$ On and $\tau \in S_{\sigma,x}$.

Let

$$\{x_0 < \cdots < x_{n-1}\} = \{\ \tau_2(i),\ \tau_2(i) \text{ is defined}\}$$

and let $s\colon \operatorname{dom}(\tau_2) \to n$ be such that

$$\tau_2(i) = x_{s(i)}$$

Then

$$D_\sigma((c;\ x_0,\ldots,x_{n-1};x)) = \|\tau\|_{\sigma,x}$$

where c is defined as follows:

Let

$$\tau'(i) = \begin{cases} \tau(i) & \text{if } i \in \operatorname{dom}(\tau_1) \\ m(\sigma) + s(i) & \text{if } i \in \operatorname{dom}(\tau_2) \end{cases}$$

Then $c = \|\tau'\|_{\sigma,n}$.

<u>Lemma 3.8</u>

Each D_σ is a denotation-system.

The proof is simple and is left for the reader.

<u>Lemma 3.9</u>

$\underline{a}$ $\quad D_{\langle\ \rangle} = D_S$

$\underline{b}$ $\quad$ If $\sigma \in S$ but σ has no extension in S then

$$D_\sigma = \underline{1}$$

$\underline{c}$ $\quad$ If $\sigma \in S$ and σ has an extension in S then

$$D_\sigma = \sum_{j<m(\sigma)} D_{\sigma+j} + \langle D_{\sigma+(m(\sigma)+y)}\rangle_{y\in On} + \underline{1}.$$

<u>Proof</u>

$\underline{a}$ and $\underline{b}$ are trivial.

The final $\underline{1}$ in $\underline{c}$ comes from the denotation corresponding to $\langle\ \rangle$.

If $j < m(\sigma)$ then $m(\sigma^\star j) = m(\sigma)$. Moreover if $\sigma^\star\tau \in S$ and $\tau(0)=j < m(\sigma)$ then the norm $\|\tau\|_{\sigma,x}$ of τ in $S_{\sigma,x}$ is exactly

$$\sum_{j'<j} \|\sigma^\star j', x\| + \|\tau^-\|_{\sigma^\star j,x}$$

where $\tau^-(i) = \tau(i+1)$.

Let $S_\sigma^{m(\sigma)} = \{\tau;\ \sigma^*\tau \in S \ \&\ \tau(0) \geq m(\sigma)\}$ and let $D_{\sigma,m(\sigma)}$ be the denotation-system corresponding to the Kleene-Brouwer ordering on $S_\sigma^{m(\sigma)}$. Again it is easy to show that $D_{\sigma,m(\sigma)}$ really is a denotation-system. Moreover

$$D_\sigma = \sum_{j<m(\sigma)} D_{\sigma^*j} + D_{\sigma,m(\sigma)} + \underline{1},$$

since the value of $\tau(0)$ is most important in order to decide the order of τ in the Kleene-Brouwer ordering.

It remains to show that if $D_{\sigma,m(\sigma)} \neq \underline{0}$ then it is connected and $\neq \underline{1}$ and to show that

$$D^y_{\sigma,m(\sigma)} = D_{\sigma^*(m(\sigma)+y)} .$$

If $\sigma^*m(\sigma) \in S$ then $\sigma^*(m(\sigma)+y) \in S$ for all $y \in On$, by order invariance. Since the value of $\tau(0)$ separates the values of the corresponding denotations we have

1. If c_1 and c_2 are $D_{\sigma,m(\sigma)}$-indices we cannot have that all c_1-denotations from a limit ordinal x dominates all c_2-denotations from x. Thus $D_{\sigma,m(\sigma)}$ is connected.

2. For a fixed $D_{\sigma,m(\sigma)}$-denotation

$$(c;x_0,\ldots,x_{n-1};x)$$

the coefficient corresponding to $\tau(0)$ will be the most important one.

Now fix y,z and let $X_{y,z}$ be as in Definition 2.18 <u>a</u> for $D_{\sigma,m(\sigma)}$. The ordinals in $X_{y,x}$ will give the location of $S_{\sigma^*(m(\sigma)+y),x}$ in $S^{m(\sigma)}_{\sigma,x}$ where

$$S^{m(\sigma)}_{\sigma,x} = \{\tau \in S^{m(\sigma)}_\sigma;\ \forall i < lh(\tau)\ (\tau(i) < m(\sigma)+x)\}$$

By this correspondance we see that the standard denotation-systems of $D^y_{\sigma,m(\sigma)}$ and $D_{\sigma^*(m(\sigma)+y)}$ will be the same.

This ends the proof of Lemma 3.9.

Remark 3.10

Lemma 3.9 shows that there is a connection between the sub-components of a denotation-system D ($\{D';D'\prec D\}$, see Definition 2.20).

For a more systematic discussion of the connection between denotation-systems and ordinal trees, see Girard [2], Jervell [6] or Girard-Jervell [4].

3.4 Linear decomposition and imbeddings

We have been discussing the decomposition of a denotation-system into the subcomponents. There is an alternative way which corresponds to the Kleene-Brouwer ordering on the decomposition tree.

Definition 3.11

a Let D be a denotation-system, $D = D_1+D_2$ where $D_1, D_2 \neq \underline{0}$. Then D_1 is a predecessor of D.

b If $D = D_1+D_2$ and $D_2 \neq \underline{1}$ is connected, then each $D_1 + \sum_{j<y} D_2^j$ is a predecessor of D.

c The predecessor relation is the minimal transitive relation satisfying a and b.

Lemma 3.12

Let D be a denotation-system. The predecessors of D are well-ordered by the predecessor relation.

Proof

By Theorem 2.22 and general facts about Kleene-Brouwer orderings we see that it is well-founded.

Linearity follows by a simple induction on the hight in this new ordering. Notice that the predecessor relation itself is not well-ordered.

Remark 3.13

Let $D = D_S$ be a denotation-system obtained from an order-invariant well-founded tree. The predecessors are all defined from initial segments of the Kleene-Brouwer ordering in the following way; let I be an initial segment.

i If I has a maximal element σ, let

$$I' = \{\tau;\ \tau < \sigma \text{ lexiographically}\}$$

Then $D_I = D_{I'} + D_\sigma$.

ii I has no maximal element, σ is minimal outside I and σ has extensions in S. Let I' be as above and let

$D_I = D_{I'} + (D_\sigma - 1)$.

iii I has no maximal element, σ is minimal outside I and σ has no extension in S. Let

$$X = \{\tau;\ \tau < \sigma \text{ lexicographically, } \tau \text{ is not the extension of any element in } I\}.$$

Then $D_I = \Sigma_{\tau \in X} D_\tau$, where X is ordered lexicographically.

So far we have defined two "less than" orderings, subcomponents and predecessors. There is also a third natural one:

Definition 3.14

Let E, D be two denotation systems. We say that T is an imbedding of E into D if

i $T: Tr(E) \xrightarrow{1-1} Tr(D)$

ii If $(c,n) \in Tr(E)$ then $T(c,n) = (c',n)$ for some c'.

iii T induces an order-preserving map on denoted ordinals.

Remark 3.15

Let E, D be denotation-systems for G, F resp. Then an imbedding $T: E \to D$ will induce imbeddings $T_x: G(x) \to F(x)$ as follows.

If $y = (c;x_0,\dots,x_{n-1};x)_E$ and $T(c,n) = (c',n)$ then $T_x(y) = (c';x_0,\dots,x_{n-1};x)_D$.

If E and D are in standard form we will have $T(c,n) = (T_n(c),n)$. Imbeddings between denotation-systems correspond to natural transformations between Dilators, see Girard [2].

<u>Lemma 3.16</u>

<u>a</u> If $T: E \to D$ and D is connected, $E \neq \underline{0}$ then E is connected. Moreover there is a canonical decomposition of T into $T^y: E^y \to D^y$.

<u>b</u> If $T: E \to D$, $D = \Sigma_{i<\beta} D_i$, $E = \Sigma_{j<\alpha} E_j$ where D_i, E_j all are connected, then there is a unique $\rho: \alpha \to \beta$ and a canonical decomposition of T into $T_j: E_j \to D_{\rho(j)}$.

<u>Proofs</u>

The details are left for the reader. For <u>a</u> notice that the most important coefficient for (c,n) will be the same as that for $(c',n) = T(c,n)$. To see <u>b</u> notice that T induces an imbedding of each E_i into D. The image must by <u>a</u> be in one of the connected parts, D_j. Let $\rho(i) = j$.

<u>Remark 3.17</u>

A property of ordinals is that any ordinal is the direct limit of numbers and finite morphisms. This is used to develope Π^1_2-logic in a functorial way.

Using imbeddings between denotation systems we can show that any denotation-system is the limit of a directed system of systems with finite traces. This can be used to define higher type versions of denotation systems, see Girard [2] for details.

In general a subcomponent cannot be imbedded into a denotation system. The following observations will moretheless be useful:

1. If $D = \sum_{i<\beta} D_i$ then each D_i is imbeddable into D.

2. If $D \neq \underline{1}$ is connected then D^y is imbeddable in D^z when $y \leq z$.

4. SET RECURSION

4.1 The recursion theory

For the sake of completeness we here give the definition of set-recursion and state the main results that we need. We do not give any proofs since they are covered by a vast litterature on the subject.

The set-recursive functions are defined by six schemes, each having an index, and the definition is really an inductive definition of the relation

$$\{e\}(\vec{x}) \simeq y$$

which has the following interpretation

> algorithm no. e applied to the sequence of sets $\vec{x}$ halts and takes the value y.

Definition 4.1

Set-recursion is defined by the following schemes:

i $e = \langle 1, n, i \rangle$

$\{e\}(x_1, \ldots, x_n) = x_i$.

ii $e = \langle 2, n, i, j \rangle$

$\{e\}(x_1, \ldots, x_n) = x_i - x_j$.

iii $e = \langle 3, n, i, j \rangle$

$\{e\}(x_1, \ldots, x_n) = \{x_i, x_j\}$.

iv $e = \langle 4, n, e' \rangle$

$\{e\}(x_1, \ldots, x_n) \simeq \bigcup_{y \in x_1} \{e'\}(y, x_2, \ldots, x_n)$.

v $e = \langle 5, n, m, e', e_1, \ldots, e_m \rangle$

$\{e\}(x_1, \ldots, x_n) \simeq \{e'\}(\{e_1\}(x_1, \ldots, x_n), \ldots, \{e_m\}(x_1, \ldots, x_n))$

vi $e = \langle 6, n, m \rangle$

$\{e\}(e_1, x_1, \ldots, x_n, y_1, \ldots, y_m) \simeq \{e\}(x_1, \ldots, x_n)$

In most expositions set-recursion is relativized to relations, but here we will relativize it to set-functions. This will complicate the theory at the advanced level but not for the results that we are interested in.

If $g: V \to V$ is a function we relativize set-recursion to g by adding the scheme

<u>vii</u> $e = \langle 7,n,i \rangle$

$\{e\}^g(x_1,\ldots,x_n) = g(x_i)$

<u>Remark 4.2</u>

<u>a</u> In the Union-scheme <u>iv</u> the computation halts if the computations $\{e'\}(y,x_2,\ldots,x_n)$ halt for all $y \in x_1$.

<u>b</u> In the composition scheme $\{e\}(x_1,\ldots,x_n)$ halts if and only if $\{e_i\}(x_1,\ldots,x_n)$ halts for $i = 1,\ldots,m$ and $\{e'\}(y_1,\ldots,y_m)$ halts, where g_i is the i'th value above.

An important aspect of set-recursion is the computation-tree and the subcomputation relation:

<u>Definition 4.3</u> (Essentially Y.N. Moschovakis [8])

<u>a</u> A computation tuple is any sequence $\langle e,x_1,\ldots,x_n \rangle$ where $e \in \mathbb{N}$ and each $x_i \in V$.

<u>b</u> If e does not indicate that it accepts a sequence of length n, we let $\langle e,x_1,\ldots,x_n \rangle$ be a subcomputation of itself.

<u>c</u> Computations from <u>i</u>-<u>iii</u> and <u>vii</u> are called <u>initial</u> and have no subcomputations:

<u>d</u> If $e = \langle 4,n,i,j \rangle$ then $\langle e,x_1,\ldots,x_n \rangle$ has the following subcomputations:

$$\{\langle e',y,x_2,\ldots,x_n \rangle ; y \in x_1\}.$$

<u>e</u> If $e = \langle 5,n,m,e',e_1,\ldots,e_m \rangle$ then $\langle e_i,x_1,\ldots,x_n \rangle$ are subcomputations of $\langle e,x_1,\ldots,x_n \rangle$ for $i = 1,\ldots,n$.

Moreover, if there are $y_1,\dots,y_m$ such that

$$\{e_i\}(x_1,\dots,x_n) = y_i$$

for all $i = 1,\dots,n$ then $\langle e', y_1,\dots,y_n\rangle$ is also a subcomputation.

The subcomputation-relation is the minimal transitive relation satisfying a-d.

Remark 4.4

a We do not ask if a computation halts when we define the subcomputations.

b In all cases except for composition the set of subcomputations is primitive recursive in the given computation-tuple. The first subcomputations are simple, but if they halt the values may be more complex and so will the last subcomputation. This is one of the phenomena that makes set-recursion difficult but also interesting.

c A computation will halt if and only if the sub-computation relation below it is well-founded. We call this relation for the computation tree. The computation tree is recursive in the input if the computation halts, but not in general.

We will mainly work with functions $g\colon On \to On$. If we let $g'(x) = g(\mathrm{rank}(x))$ we have an immediate extension to all sets.

In an application of the main result we will use the following observation:

Lemma 4.5

Let $g\colon On \to On$, $x \in V$ and let $E^g(x)$ be the least transitive set that contains x as an element and is closed under $\{e\}^g$ for all $e \in N$. The relation "$\{e\}^g(x)$ halts" is uniformly Σ_1^g over $E^g(x)$ by a formula that is absolute with respect to V.

Remark 4.6

$E^g(x)$ will be a subset of $A^g(x)$; the 'next admissible' relative to g. Thus the relation '$\{e\}^g(\vec{x})$ halts' is uniformly Σ_1 over all g-admissibles.

4.2 The denotation-system of an algorithm

Definition 4.7

We call a function g Δ_0 if the function

$$g'(x) = L_{g(x)}$$

has a Δ_0 graph.

Examples of Δ_0-functions are

$\alpha \rightsquigarrow \alpha^+$

$\alpha \rightsquigarrow$ the first recursively inaccessible above α

$\alpha \rightsquigarrow$ the first recursively Mahlo above α

From now on in this section we will let g be a fixed increasing Δ_0-function on On and we will assume that

$$\forall \alpha \in \mathrm{On}\ \{e\}^g(\alpha) \text{ halts.}$$

We will construct a first order theory T stating that for some α $\{e\}^g(\alpha)$ does not halt. T may well be consistent but will not have any well-founded models. This will be used to construct a denotation system that controlles the computations $\{e\}^g(\alpha)$.

Definition 4.8

Let T be the first order theory defined as follows:

a Language L

i The language of set theory, $=$, $\in$, and two special symbols for On and Rank.

ii A constant $\underline{\alpha}$

iii Two lists of constants

$$\underline{c}_0, \underline{c}_1, \ldots$$

$$\underline{d}_0, \underline{d}_1, \ldots$$

iv Extend the above language to L such that there are Henkin-constants for all quantifiers in Δ_0-formulas.

b Axioms

ST (Set Theory): Relevant Δ_0-facts about set theory like extensionality and axioms describing On and Rank.

$\underline{A}_0$ $\underline{\alpha} \in \underline{On} \wedge \underline{c}_0 = \langle e, \underline{\alpha} \rangle$

$\underline{A}_{i+1}$ is a conjunction of axioms describing the relation between $\underline{c}_i$, $\underline{d}_i$ and $\underline{c}_{i+1}$. The point is to say in a Δ_0-way that $\underline{c}_{i+1}$ is a subcomputation of $\underline{c}_i$. The axioms are as follows:

- If $\underline{c}_i$ is a computation by i-iii, vii, then $\underline{d}_i = \emptyset$ and $\underline{c}_{i+1} \neq \underline{c}_{i+1}$.
- If $\underline{c}_i$ is a computation by iv or vi then $\underline{d}_i = \emptyset$ and c_{i+1} is a subcomputation.
- If $\underline{c}_i = \langle\langle 5, n, m, e_0, e_1, \ldots, e_m \rangle, x_1, \ldots, x_n \rangle$ then either $\underline{d}_i = \emptyset$ and $\underline{c}_{i+1} = \langle e_j, x_1, \ldots, x_n \rangle$ for some $j \leq m$, or $\underline{d}_i$ is a transitive set containing well-founded computation trees for $\{e_j\}(x_1, \ldots, x_n) \simeq y_j$; $j = 1, \ldots, n$ and $\underline{c}_{i+1} = \langle e_0, y_1, \ldots, y_m \rangle$.
- If $\underline{c}_i$ is not a computation-tuple then $\underline{d}_i = \emptyset$.

$\underline{H}_e$ We add Henkin-axioms for all our Henkin constants $\underline{e}$, i.e. the Δ_0-part of the theory will be a Henkin-theory.

Remark 4.9

All the axioms are Δ_0, so a term-model for a completion of T will satisfy all the axioms. We use that g is Δ_0 when we define computation-trees in a Δ_0-way.

Lemma 4.10

T has no well-founded model.

Proof

In such a model we would interprete $\underline{\alpha}$ as an ordinal α and $\{\underline{c}_i\}_{i\in N}$ as a descending path in the computation-tree of $\{e\}(\alpha)$.

Let T_i be the part of T where $\underline{c}_j$, $\underline{d}_j$ do not occur for $j > i$, i.e. T_i is the Δ_0-Henkin extension for the axioms ST and $A_0,\ldots,A_i$.

Let $\underline{e}_0, \underline{e}_1,\ldots$ be a recursive enumeration of the constants of the theory T such that $\underline{c}_i$, $\underline{d}_i$ are enumerated before any constant in $T-T_i$. Let $f\colon \mathbb{N} \to On$ and let

$$T^f = T \cup \{\mathrm{Rank}(\underline{e}_i) \leq \mathrm{Rank}(\underline{e}_j);\ f(i) \leq f(j)\}.$$

Lemma 4.11

T^f is inconsistent.

Proof

If T^f is consistent let T^* be a completion. The term-model will be well-founded by the rank-function f, which contradicts Lemma 4.10.

If σ is a finite sequence of ordinals, let

$$T^\sigma = T_j \cup \{\mathrm{Rank}(\underline{e}_i) \leq \mathrm{Rank}(\underline{e}_j);\ i,j < \mathrm{lh}(\sigma) \wedge \sigma(i) \leq \sigma(j)\}.$$

where j is maximal such that $\underline{c}_j = \underline{e}_i$ for some $i < \mathrm{lh}(\sigma)$.
Let $S = \{\sigma;\ T^\sigma \text{ is consistent}\}$.

Lemma 4.12

S is a well-founded order-invariant tree on On.

Proof

Immediate.

Remark 4.13

We will work with the tree S which is Π^0_1. There is no problem in extending S to a tree S' which is still well-founded and order invariant but also primitive recursive. Any statement we prove about S will also be true for S'.

Remark 4.14

If g is the identity-function we can show that

$\forall\alpha\ \{e\}(\alpha) < \|S_{\alpha+1}\|$.

For general g this will not hold, but we will dominate $\{e\}(\alpha)$ via primitive recursion over the decomposition of the denotation-system corresponding to S. This will be the theme of the next paragraph.

4.3 The domination of a computation

In this section we will let g, e, L, T and S be as in section 4.2.

The tree S_x cannot be expected to dominate $\{e\}^g(x)$ in any sense because

$$x \rightsquigarrow \|S_x\|$$

is outright set-recursive while g may not be.

If we let D be the corresponding denotation-system we will show that we can dominate

$$\lambda x\ \{e\}^g(x)$$

by a function obtained from g and a simple uniform primitive recursion on the decomposition of D.

Definition 4.15

Let h be a function, x an ordinal and E a denotation-system. By induction on the linear decomposition of E (see section 3.4) we define i $I(\underline{0},x,h) = h(x)$

ii $I(E+1,x,h) = I(E,x,h)+1$

iii If $E = \sum_{i<\beta} E_i$ where β is a limit ordinal and each E_i is connected, let

$$I(E,x,h) = \sup\{I(\sum_{j<i} E_j,x,h);\ i < \beta\}$$

iv If $E = E_1+E_2$ where $E_2 \neq \underline{1}$ is connected, let

$$I(E,x,h) = I(E_1+\sum_{y<I(E_1,x,h)} E_2^y,\ I(E_1,x,h),h).$$

Lemma 4.16 (Monotonicity)

Assume that h is increasing.

a If $x \leq y$ then $I(E,x,h) \leq I(E,y,h)$ for each denotation system E.

b If $T: E' \to E$ is an imbedding (section 3.4) then $I(E',x,h) \leq I(E,x,h)$ for each ordinal x.

Both a and b are proved by induction on the decomposition-tree for E. Observe Lemma 3.16 for b.

Our aim is to show that if we let $D = D_S$ then

$$\{e\}(x) \leq I(D,x,g)$$

for all x.

To this end we let $x \in On$ be fixed and we let $c_0,c_1,\ldots,c_k$ be a sequence of computation tuples starting with $\langle e,x\rangle = c_0$ and such that each c_{i+1} is an immediate subcomputation of c_i; $i = 0,\ldots,k-1$.

Let t be maximal such that

$$\{\underline{e}_0,\ldots,\underline{e}_t\} \subseteq T_k$$

(i.e. $\underline{e}_0,\ldots,\underline{e}_t$ is the maximal segment of our listing of the constants such that $\underline{c}_{k+1}$, $\underline{d}_{k+1}$ is not used.)

Choose interpretations $e_0,\ldots,e_t$ of $\{\underline{e}_0,\ldots,\underline{e}_t\}$ resp. consistent with $c_0,\ldots,c_k$ and x. Let σ be the sequence

$$\sigma(i) = \mathrm{rank}(e_i), \quad i \leq t.$$

Then $\sigma \in S$ since the universe is a model for T^σ, with e_i as the interpretation of $\underline{e}_i$.

Lemma 4.17

Let $\delta = m(\sigma) = \max\{\sigma(i);\ i < \mathrm{lh}(\sigma)\}+1$. Let $\beta = I(D_\sigma,\delta,g)$ (see Definition 3.7 for D_σ). Then the computation-tree of c_k is in L_β.

Proof

We use induction on the height of c_k in the computation-tree of $\{e\}^g(x)$.

i If c_k is an initial computation we have $D_\sigma \geq \underline{1}$ so

$$\beta = I(D_\sigma,\delta,g) \geq g(\delta)+1.$$

Then the computation-tree of c_k will be in L_β.

ii If c_k is an application of the union scheme

$$\{d\}(y,\vec{y}) = \bigcup_{z\in y}\{d'\}(z,y,\vec{y})$$

then

$$I(D_\sigma,\delta,g) \geq \sup\{I(D_{\sigma^*\gamma},\delta,g);\ \gamma < \mathrm{rank}(y)\}+1,$$

since we have an imbedding $T\colon \Sigma_{\gamma<\mathrm{rank}(y)} D_{\sigma^*\gamma} + 1 \rightarrow D_\sigma$.

Let $\beta' = \sup\{I(D_{\sigma^*\gamma},\delta,g);\ \gamma < \mathrm{rank}(y)\}$. By the induction hypothesis all computation-trees for the subcomputations of c_k will be in $L_{\beta'}$. Since $\beta \geq \beta'+1$ the computation-tree of c_k will be in L_β.

iii The enumeration scheme S 6 is treated in a similar way.

iv c_k is an application of composition

$$\{d\}(\{d_1\}(\vec{x}),\ldots,\{d_k\}(\vec{x}))$$

Let $\beta' = I(\Sigma_{\gamma<\delta} D_{\sigma^*\gamma}, \delta, g)$

Since the rank of each $\{d_i\}(\vec{x})$, as a computation-tuple, does not exeed the rank of c_k it follows from the induction-hypothesis that the computation-tree for each $\{d_i\}(\vec{x})$ is in $L_{\beta'}$.

Let $y_i = \{d_i\}(\vec{x})$, $i = 1,\ldots,k$.

If $\operatorname{rank}(y_i) < \delta$ for each $i = 1,\ldots,k$ then the computation-tree of $\{d\}(y_1,\ldots,y_k)$ is also in $L_{\beta'}$. $\beta' < \beta$ so the tree of c_k will be in L_β.

If $\operatorname{rank}(y_i) \geq \delta$ for some i, then

$$D_\sigma = \sum_{\gamma<\delta} D_{\sigma^*\gamma} + \langle D_{\sigma^*(\delta+\beta)}\rangle_{\beta\in On} + \underline{1}$$

where each $D_{\sigma^*(\delta+\beta)} \neq \underline{0}$. This follows from the order-invariance by letting $c_{k+1} = \{d\}(y_1,\ldots,y_k)$. Then $\operatorname{rank}(c_{k+1}) \geq \delta$.

Again let $\beta' = I(\Sigma_{\gamma<\delta} D_{\sigma^*\gamma}, \delta, g)$. Let

$$D' = \sum_{\gamma<\delta} D_{\sigma^*\gamma} + \sum_{y\leq\beta'} D_{\sigma^*(\delta+y)} + \underline{1}$$

Then

$$\beta = I(D_\sigma, \delta, g) = I(D', \beta', g)$$

and

$$\beta' < \beta.$$

By the induction-hypothesis, $\operatorname{rank}(y_i) < \beta'$. Let δ' be the rank of the computation-tuple $\{d\}(y_1,\ldots,y_k)$. Then $\delta' \leq \beta'$. By the induction-hypothesis the computation-tree of $\{d\}(y_1,\ldots,y_k)$ will be in $L_{\beta''}$ where

$$\beta'' = I(D_{\sigma^*\delta'}, \delta'+1, g).$$

But $D_{\sigma^*\delta'}$ can be imbedded in D'. It follows that the computation-tree of c_k will be in L_β.

This ends the proof of Lemma 4.17.

Theorem 4.18

Assume that for all $x \in On$

$$\{e\}^g(x) \text{ halts}$$

where g is Δ_0 and increasing. Then there is a denotation-system D and a uniform primitive recursive operator $I(D,x,g)$ such that

$$\forall x \in On \quad \{e\}^g(x) \leqslant I(D,x,g)$$

Proof

Immediate from Lemma 4.17 and the constructions leading up to it.

Remark 4.19

Since any Δ_0-function g can be dominated by a Δ_0-function h primitive recursive in g such that h is increasing, that assumption is mainly technical.

5. RECURSION ON DENOTATION SYSTEMS

5.1 General primitive recursion

In paragraph 4 we defined the operator I by means of a certain primitive recursion over the linear decomposition of a denotation system. In this section we will give a general definition of such primitive recursion. We have not worked out any detailed properties of this notion, and it might not be the richest possible. On the other hand it is clear from our results and their proofs that any reasonable notion of primitive recursion on denotation-systems will share the properties we are interested in.

Definition 5.1

Let α, β, γ denote ordinals, D,E denote denotation-systems with corresponding functions F_D, F_E respectively, and let f,g denote ordinal functions.

Let $\vec{x}$ denote a sequence of α's, D's and f's. We define the set of primitive recursive operators with arguments $\vec{x}$ and values in On by schemes as follows:

A Schemes for primitive recursion on On:

$$I_0(\alpha,\vec{x}) = \alpha$$

$$I_1(\alpha,\vec{x}) = \alpha+1$$

$$I_2(\alpha,\vec{x}) = \begin{cases} J_1(\vec{x}) & \text{if } \alpha = 0 \\ J_2(\lambda\beta < \alpha\ I_2(\beta,\vec{x}),\alpha,\vec{x}) & \end{cases}$$

where J_1 and J_2 are primitive recursive operators, and

$$(\lambda\beta<\alpha\ g(\beta))(\gamma) = \begin{cases} g(\gamma) & \text{if } \gamma < \alpha \\ 0 & \text{if } \gamma \geq \alpha \end{cases}$$

B Schemes of application:

$$I_3(\alpha,D,\vec{x}) = F_D(\alpha)$$

$$I_4(\vec{\alpha},g,\vec{x}) = g(\vec{\alpha})$$

$$I_5(\alpha,g,\vec{x}) = \sup\{g(\beta);\ \beta<\alpha\}$$

<u>C</u> Schemes of generation:

$$I_6(\vec{x}) = J_1(J_2(\vec{x}),\vec{x})$$

where J_1 and J_2 are primitive recursive operators.

$$I_7(\vec{x}) = J(\tau(\vec{x}))$$

where J is primitive recursive and τ is a permutation of the variables.

<u>D</u> A scheme for recursion over denotation systems:

Let J_1, J_2, J_3 be primitive recursive operators. Then I is primitive recursive where I is defined by

<u>i</u> $I(\underline{0},\vec{\alpha},\vec{f},\vec{D}) = J_1(\vec{\alpha},\vec{f},\vec{D})$

<u>ii</u> $I(D+\underline{1},\vec{\alpha},\vec{f},\vec{D}) = J_2(\lambda\vec{\beta}I(D,\vec{\beta},\vec{f},\vec{D}),\vec{\alpha},\vec{f},\vec{D}))$

<u>iii</u> If $D = \Sigma_{i<\alpha} D_i$ where α is a imit ordinal and each D_i is connected, then

$$I(D,\vec{\alpha},\vec{f},\vec{D}) = J_2(\lambda\vec{\beta} \sup_{i<\alpha} I(\sum_{j<i} D_j,\vec{\beta},\vec{f},\vec{D}),\vec{\alpha},\vec{f},\vec{D}))$$

<u>iv</u> If $D = D'+E$ where $\underline{E} \neq 1$ is connected we let

$$I(D,\vec{\alpha},\vec{f},\vec{D}) = J_3(\lambda(y,\vec{\beta})I(D'+\sum_{z<y} E^z,\vec{\beta},\vec{f},\vec{D}),\vec{\alpha},\vec{f},\vec{D}).$$

<u>Remark 5.2</u>

<u>a</u> Clearly the operator I of paragraph 4 is primitive recursive by this definition.

<u>b</u> By the decomposition-theorems clearly all primitive recursive operators are total.

<u>c</u> If we add a scheme of enumeration in analogy with Kleene's S9 ([7]) we get a notion of full recursion on denotation-systems. This notion is however of no particular interest in this paper.

d Another possible extension is to add an 'oracle-scheme' in analogy with Kleene's S8:

If J_1, J_2 are recursive and

$D = \lambda(c; x_0, \ldots, x_{n-1}; x) J_1(\langle c, x_0, \ldots, x_{n-1}, x\rangle, \vec{x})$

is a denotation-system, then

$I(\vec{x}) = J_2(D, \vec{x})$

is recursive.

This scheme will introduce partial functions. It turns out that the total ordinal functions of this theory is exactly the total set-recursive functions. This can be relativized to functions g with Δ_0-graph.

Our first task now is to reduce primitive recursion to set-recursion and to this end we will represent denotation-systems by sets. By Theorem 2.5 a system D is determined by $D\restriction\mathbb{N}$ which is a set. For simplicity we will write D but we will always mean $D\restriction\mathbb{N}$ when we use D as an argument for an algorithm.

<u>Lemma 5.3</u>

a The function $F_D(\alpha)$ is uniformly set-recursive in D, α.

b Uniformly set-recursive in D we can decide if D is connected and if $D = \underline{1}$ or $\underline{0}$.

c If $D = \Sigma_{i<\alpha} D_i$ where each D_i is connected then α and each D_i are uniformly set-recursive in D.

d If $\underline{D} \neq \underline{0}, \underline{1}$ is connected then D^y is uniformly set-recursive in D, y.

The proofs are implicit in the discussion of the decomposition and in the constructions of the subcomponents. Notice that $D\restriction\mathbb{N}$ an infinite object so ω will be set-recursive in $D\restriction\mathbb{N}$.

Lemma 5.3 and the recursion-theorem for set-recursion gives us

Theorem 5.4

Each primitive recursive operator is uniformly set-recursive in an index for the scheme defining it.

Remark 5.5

In general we cannot set-recursively decide if a pre-denotion-system really is a denotation-system. Thus the algorithm of Theorem 5.4 may work in cases where the input is not a denotation-system.

Theorem 5.4 can be relativized to any function g without further effort.

5.2 General domination of total Σ_1-functions

Theorem 4.8 was proved for total set-recursive functions relative to Δ_0-functions g. There are deep problems in relativizing the result to arbitrary sets, since the construction of the countable theory T is essential to the proof. In a forthcomming note we intend to indicate how a more general relativization still can be partly achieved.

The proof of Theorem 4.8 can easily be relativized to enumerated transitive sets. This gives us the following application:

Theorem 5.6

Let α be an admissible ordinal such that

$$L_\alpha \models \text{All sets are countable.}$$

Then the following are equivalent:

i α is recursively Mahlo

ii For all total α-recursive $g: \alpha \to \alpha$ there is a total α-recursive f such that f is not dominated by any function primitive recursive in g and a denotation-system in L_α.

Proof

i => ii. Assume that α is recursively Mahlo and let g be given. For each $x \in L_\alpha$ we have that $E^g(x)$, the set-recursive closure of x relative to g, is an element of L_α. Thus the relation "$\{e\}^g(x)$ halts" is Δ_1 over L_α, see Remark 4.6. By Theorem 5.4 we can α-enumerate all functions primitive recursive in g and a denotation-system in L_α in a Δ_1-way (we will necessarily include a few more functions in the enumeration since we cannot decide when a pre-system is a system in a Δ_1-way, but this do not hurt our argument). By a diagonal construction we find a Δ_1-function f that is not dominated by any function in the enumeration.

ii => i. Now assume that α is not recursively Mahlo. Then there is an α-recursive h such that α is the least h-admissible ordinal. Let $x \in L_\alpha$ be such that h is Σ_1^x. Let

$$h(\gamma) = \beta \iff \exists y\ \phi(\gamma,\beta,x,y)$$

Let $g(\gamma) = \mu\beta\ \exists y \in L_\beta\ \exists\beta'<\beta\ \phi(\gamma,\beta',x,y)$. Then g is Δ_0 and g dominates h. Moreover $L_\alpha = E^g(x)$ (see Remark 5.7).
Let f be α-recursive. Then f is set-recursive in g and some parameter y. By a relativized version of Theorem 4.18 we can find a denotation-system D primitive recursive in $\mathbb{N}$-codes for x,y such that

$$f(\gamma) < I(D,\gamma,g)$$

(If $\gamma \geq \alpha$ we let $f(\gamma) = \gamma$.)
Then f is dominated as required by the theorem.

Remark 5.7

The set-recursive closure of an enumerated set will be the next admissible. This holds even when relativized to a Δ_0-function g. Essential in the argument for this is that when α can be enumerated then $\beta = g(\alpha)$ can be enumerated by a Skolem-Löwenheim argument. Thus it is more out of convenience than out of mathematical necessity that we use set-recursion in proving these results.

We may use a similar trick to prove a relativized version of Van de Wiele's theorem.

Theorem 5.8

Let $g\colon On \to On$ be Δ_0. Let f be uniformly Σ_1^g-definable over all g-admissible structures L_α. Then f is set-recursive in g.

Indication of proofs

One alternative is to employ a method deviced by T. Slaman [12] which is purely set-recursive. Alternatively one may show that f is dominated by a primitive recursion in g and some primitive recursive denotation-system . To this end we need a notation-system for the next admissible after α relative to g, and to describe this system inside α. Here it is essential that the cardinality of $g(\beta)$ is that of β and that this is effective in $g(\beta)$. We omit the details.

6. FUNCTORIALITY

We have so far used constructions involving ordinals and dilators such as I of 4.15 in a generalized recursion spirit. Of course, a treatment of these concepts more in the spirit of Π^1_2-logic is possible; let us first question the interest of such a treatment! We will from now on have to assume a certain familiarity with the general notions of Π^1_2-logic.

6.1 Interest

When we define, say, a function $\Phi(x,D)$ mapping ordinals and denotation systems into ordinals then to be in agreement with the spirit of Π^1_2-logic we should try to make it functorial. This means that we have to define Φ also on morphisms of the corresponding categories. We must define $\Phi(f,T)$ where f is an increasing function from one ordinal to another, and T is an imbedding of one denotation system into another in such a way that Φ is a functor preserving direct limits and pullbacks, i.e. Φ is a ptyx, see Girard [3] Ch.XII. If such a thing can be done (and essentially it can be done) then we gain something since we are now able to do our computations by means of direct limits: for instance we can express D as a direct limit of finite dimentional denotation systems etc. Hence functoriality is an additional step in the direction of the simplification of the class of algorithms.

6.2 Example

Assume that h is a given function from On to On and that h is normal, i.e. strictly increasing and continuous. Then we can define hierarchy of functions as follows:

i) $\Lambda(\underline{D},x,h) = x$

ii) $\Lambda(D+\underline{1},x,h) = \Lambda(D,h(x),h)$

iii) $\Lambda(\sum_{\gamma<\alpha} D_\gamma, x, h)$ = the x^{th} point in the intersection of the classes

$$rg(\lambda y\ \Lambda(\sum_{i<\gamma} D_i, y, h))$$

when α is a limit ordinal.

iv) $\Lambda(D_1+D_2, x, h) = \Lambda(D_1+\sum_{y<x} D_2^y, 0, h)$

when D_2 is connected and $\neq \underline{1}$.

It is not very difficult to show that given a recursive F one can find a recursive D such that

(1) $\quad I(F,x,h) \leqslant \Lambda(D,x,h)$ for all $x \in On$.

Moreover, with a rather slight modification we can turn Λ into a functor. Let us be a bit more precise.

1. We will assume that h is such that

$$h(x+1) = h(x)+1 + H_1(x)$$

for a certain denotation system H_1. Then it is easy to see that h itself is of the form $h(x) = H(x)$ for a certain denotation system H (such a denotation system is called a nice flower).

2. If $f \in I(x,x_1)$, T is an imbedding from D to D_1 and V is an imbedding from H to H_1 of the form

$$V(x+1) = V(x)+E_1+V'(x)$$

then it is possible to define

$$\Lambda(T,f,V) \in I(\Lambda(D,x,H), \Lambda(D_1,x_1,H_1))$$

This extension makes Λ a functor of the 3 arguments preserving direct limits and pull-backs.

Let us take an example inside our example: It is possible to choose H (not at all recursive) such that $H(x) = \omega_x^{CK}$ for all x

in a very large initial segment s_0 of the first stable σ_0. The majoration (1) (or the result of Girard-Vauzeilles, directly in terms of Λ) yields

$$I_0^{CK} = \sup\{\Lambda(D,0,H);\ D \text{ is a recursive denotation system}\}$$

and in particular, every ordinal $< I_0^{CK}$ can be (non-uniquely) written as

$$x = \Lambda(D,0,H)$$

for a certain recursive D.

The fact that the construction is functorial enables us to "compute x by menas of a direct system (H_i, V_{ij}) of finite-dimentional " denotation systems.

6.3 Other possibilities

Not any function $\Phi(x,D)$ can be extended into a ptyx; in particular the primitive recursive schemes of §5 are not, strictly speaking, definable by pytxes. But the essential part of the schemes can be reformulated in a functorial way. Let us give an example:

Consider for instance

$$\Phi(\underline{0},D') = \phi_0(D')$$

$$\Phi(D+\underline{1},D') = \Phi(D,D')+\phi_1(D,D',\phi(D,D'))$$

$$\Phi(\sum_{\gamma<\alpha} D_\gamma, D') = \sup_{\gamma<\alpha} \Phi(\sum_{i<\gamma} D_i, D')$$

$$\Phi(D_1+D_2,D') = \Phi(D_1,D')+\phi_2(D_1,D',\lambda x\Phi(D_1+\sum_{x'<x} D_2^{x'},D'))$$

is in fact functorial (provided of course ϕ_0, ϕ_1 and ϕ_2 are already functorial).

There is no trouble in defining $\Phi(T,T')$ (similar equations). This clearly indicates that the primitive recursion of §5 can be handled functorially. This Φ is indeed one of the many variants of the functor Λ of [2], Ch. 5.