SUBRECURSIVE HIERARCHIES

JOEL W. ROBBIN

Unpublished Theses submitted for the Doctor's, Master's and Bachelor's degrees in Princeton University and deposited in the University Library are open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but passages must not be copied without permission of the authors and without proper credit being given in subsequent written or published work.

This Thesis by Joel Robbin .....has been borrowed by the following persons, whose signatures attest their acceptance of the above restrictions.

NAME AND ADDRESS	DATE
R. M. Kay Fine Hall	8/24/66
D. Tsichritzis	4/23/68
W. Burkhard	6/11/70
Tang A.	24/11/70
F. W. KROON	3/1/71

SUBRECURSIVE HIERARCHIES

By

Joel W. Robbin

A DISSERTATION

Presented To The

Faculty of Princeton University

in Candidacy for the Degree

of Doctor of Philosophy

Recommended for Acceptance by the

Department of

Mathematics

April, 1965

ABSTRACT

The classification problem for recursive functions is the problem of assigning ordinals to recursive functions as a measure of their complexity. In this paper we consider three approaches to this problem: the ordinal recursion hierarchy, the extended Grzegorczyk hierarchy, and the Kleene subrecursive hierarchy. We obtain characterizations of the nested n-fold recursive functions in terms of each of these hierarchies. In the last section of the paper we show some of the problems that arise when we try to generalize these hierarchies. A characterization of the nested n-fold recursive functions in terms of computational complexity on a Turing machine is also given in the paper.

ACKNOWLEDGEMENT

I would like to thank my thesis advisor, Professor Alonzo Church, for his assistance and kind encouragement during the preparation of this thesis. Thanks also go to Professor W. W. Boone for reasons which are not dissimilar. Thanks to Professor J. Guard for many stimulating conversations and for permission to use some of his unpublished results in my own work. Finally, I would like to thank the National Science Foundation for its generous support.

TABLE OF CONTENTS

INTRODUCTION

We are here concerned with what has been called "the classification problem for recursive functions"; the ultimate objective is to find a way of assigning ordinal numbers to recursive functions as a measure of their complexity. Three different approaches to this problem are considered in this paper. They are:

I. The ordinal recursion hierarchy. Let R be a well ordering of the natural numbers with order type $|R|$. Let $U(R)$ be the set of unnested R-recursive functions. To each recursive function f we may assign the smallest ordinal $|R|$ such that $f \in U(R)$.

II. The extended Grzegorczyk hierarchy. For each Church-Kleene ordinal notation a define (inductively) the singulary function W_a by

(i) if $a = 1$, $W_a(x) = 2^x$;

(ii) if $a = 2^b$, $W_a(x) = W_b^x(1)$, i.e., $W_a(0) = 1$ and

$W_a(x+1) = W_b(W_a(x))$;

(iii) if $a = 3 \cdot 5^{\underline{q}}$ where $\underline{q}$ is a godel number for q, then

$W_a(x) = W_{q(x)}(x)$.

In other words, we start with 2^x, iterate the preceding function at successor ordinal notations, and diagonalize at limit ordinal notations.

Let $E_a = E(W_a)$, the set of functions elementary recursive in W_a. Let $|a|$ be the ordinal for which a is a notation. To each recursive function f we may assign the smallest ordinal $|a|$ such that $f \in E_a$.

III. The Kleene subrecursive hierarchy. Fix an indexing (i.e., godel numbering) for $P(V)$, the set of functions primitive recursive in the binary function V. For each ordinal notation a define the binary function V_a by

(i) if $a = 1$, $V_a(x,y) = x + y$;

(ii) if $a = 2^b$, $V_a(x,\underline{f}) = f(x_1,\ldots,x_r)$ where $\underline{f}$ is an index for the function $f \in P(V_b)$ and x represents the r-tuple $(x_1,\ldots,x_r)$ via pairing functions;

(iii) if $a = 3\cdot 5^{\underline{q}}$ where $\underline{q}$ is a godel number for q, then $V_a(x,w) = V_{q(u)}(x,v)$ where w represents the pair (u,v) via pairing functions.

Let $P_a = P(V_a)$, the set of functions primitive recursive in V_a. Roughly speaking, this hierarchy is formed by starting with the primitive recursive functions; at successor ordinal notations we add a universal function for the previous class and close under primitive recursive operations, and at limit ordinal notations we add the ternary function which takes the "union" of the functions defined along the fundamental sequence and close under primitive

recursive operations. Now, to each recursive function f we may assign the smallest ordinal $|a|$ such that $f \in P_a$.

Unfortunately, all three hierarchies collapse at ω.

Theorem 1. For every recursive function f there is an (elementary recursive) well ordering R such that

(1) $|R| = \omega$

(2) $f \in U(R)$.

Theorem 2. For every recursive function f there is an ordinal notation a such that

(1) $|a| = \omega$

(2) $f \in E_a$.

Theorem 3. For every recursive function f there is an ordinal notation a such that

(1) $|a| = \omega$

(2) $f \in P_a$.

These three theorems are proved in the text in the section entitled "Pathology". Kleene sought to avoid theorem 3 by restricting attention to ordinal notations formed using only primitive recursive fundamental sequences, but Feferman showed that, even with this restriction, the hierarchy collapses at ω^2.

The answer seems to be to restrict attention to "standard" well orderings and "standard" ordinal notations. Good definitions of these concepts are not known, but for small ordinals we can give well orderings and notations which seem like they ought to be standard. In the text this is done and the following characterizations of N_n, the class of nested n-fold recursive functions, are given:

Theorem 4. For a standard well ordering Q_n of order type ω^{ω^n}, $N_{n+1} = U(Q_n)$.

Theorem 5. Where a and b range over standard notations for the ordinals $<\omega^\omega$,

(1) if $|a| \leq |b|$, then $E_a \subseteq E_b$.

(2) if $|a| < |b|$, then $E_a \neq E_b$.

(3) $N_n = \bigcup E_a \quad (|a| < \omega^n)$.

Theorem 6. Where a and b range over standard notations for the ordinals $<\omega^\omega$,

(1) if $|a| \leq |b|$, then $P_a \subset P_b$.

(2) if $|a| < |b|$, then $P_a \neq P_b$.

(3) $N_n = \bigcup P_a \quad (|a| < \omega^{n-1})$.

It is hoped that an adequate definition for the concept of standardness can be found so that theorems like the following can be proved (G denotes the set of general recursive functions):

A. Where R and R' range over the standard well orderings,

(1) if $|R| \leq |R'|$, then $U(R) \subseteq U(R')$.

(2) if $|R'|$ is sufficiently larger than $|R|$, $U(R) \neq U(R')$.

(3) $G = \bigcup U(R)$.

B. Where a and b range over standard ordinal notations,

(1) if $|a| \leq |b|$, then $E_a \subseteq E_b$.

(2) if $|a| < |b|$, then $E_a \neq E_b$.

(3) $G = \bigcup E_a$.

C. Where a and b range over standard ordinal notations,

(1) if $|a| \leq |b|$, then $P_a \subseteq P_b$.

(2) if $|a| < |b|$, then $P_a \neq P_b$.

(3) $G = \bigcup P_a$.

THE ORDINAL RECURSION HIERARCHY

The variables R, R', R_1, R_2... will range over primitive recursive well-orderings of the natural numbers (non-negative integers). Without loss of generality assume that 0 is always taken to be the least element in such a well-ordering. $x\ R\ y$ will be written instead of $R(x,y)$ and will mean that x is (strictly) smaller than y in the well-ordering R. $|R|$ will denote the order type of R.

$\langle x,y\rangle$, c_1w, c_2w, will be the usual pairing functions; i.e.

$$\langle x,y\rangle = \frac{(x+y)(x+y+1)}{2} + x \quad ,$$

$$c_1w = w - g(w) \quad ,$$

$$c_2w = g(w) - c_1w \quad ,$$

where $g(w)$ = largest m such that $\frac{m(m+1)}{2} = w$. Then

$$c_1\langle x,y\rangle = x \quad ,$$

$$c_2\langle x,y\rangle = y \quad ,$$

$$\langle c_1w, c_2w\rangle = w \quad .$$

The letter c is used because c_1w is the first coordinate of w. The pairing functions give us a one-one onto correspondence between pairs of natural numbers and natural numbers. In general, a correspondence between

n-tuples of natural numbers and natural numbers is established by iteration:

$$\langle x_1 \rangle = x_1 \quad ,$$

$$\langle x_1,\ldots,x_n,x_{n+1} \rangle = \langle \langle x_1,\ldots,x_n \rangle, x_{n+1} \rangle \quad ,$$

$$c(0,i,w) = 0 \quad ,$$

$$c(1,1,w) = w \quad ,$$

$$c(1,i,w) = w \quad \text{for} \quad i \neq 1 \quad ,$$

$$c(n+1,i,w) = c(n,i,c_1 w) \quad \text{for} \quad i = 1,\ldots,n \quad ,$$

$$c(n+1,n+1,w) = c_2 w \quad ,$$

$$c(n+1,i,w) = 0 \quad \text{for} \quad i = 0,n+1,n+2,\ldots \quad .$$

Thus for $n = 1,2,\ldots$ and $i = 1,\ldots,n$,

$$c(n,i,\langle x_1,\ldots,x_n \rangle) = x_i$$

and

$$\langle c(n,1,w),\ c(n,2,w),\ldots,c(n,n,w) \rangle = w \quad .$$

For well-orderings R_1 and R_2 the well-ordering $R_1 \times R_2$ is defined as follows: $u\ (R_1 \times R_2)\ v$ if and only if either $c_2 u\ R_2\ c_2 v$ or $c_2 u = c_2 v$ and $c_1 u\ R_1\ c_1 v$. Hence

$$\langle x_1,x_2 \rangle\ (R_1 \times R_2)\ \langle y_1,y_2 \rangle$$

holds if and only if either $x_2 \; R_2 \; y_2$ or both $x_2 = y_2$ and $x_1 \; R_1 \; y_1$. Clearly then, $R_1 \times R_2$ is primitive recursive in R_1 and R_2 and

$$|R_1 \times R_2| = |R_1||R_2| \; .$$

Now we will give a natural definition of R^* so that $|R^*| = \omega^{|R|}$. Note that the ordinal numbers less than $\omega^{|R|}$ are precisely the ordinal numbers of the form

$$\omega^{\alpha_a} + \ldots + \omega^{\beta_b} + \ldots$$

where

$$0 \leq \ldots \beta < \ldots < \alpha < R$$

and

$$0 \leq a < \omega, \ldots, \; 0 \leq b < \omega, \ldots \; .$$

For such a polynomial let the natural numbers $\underline{a}, \ldots, \underline{b}, \ldots$ correspond to the ordinals $\alpha, \ldots, \beta, \ldots$ respectively in the well-ordering R. Let p_i denote the i^{th} prime $(p_0 = 2)$. Then the product

$$(p_{\underline{a}}^{a} \ldots p_{\underline{b}}^{b} \ldots) - 1$$

corresponds to the polynomial in the well ordering R^*. The -1 appears so that 0 will be the first element in R^*. R^* may be defined as follows:

$$u \; R^* \; v$$

holds if and only if there exists an $i \leq u+v+2$ such that for all $j \leq u+v+2$ both

$$j \; R \; i \text{ implies } (u+1)_j = (v+1)_j$$

and

$$(u+1)_i \leq (v+1)_i \quad .$$

Here $(x)_i$ denotes the exponent of p_i in the prime power decomposition of x. The bounds on the quantifiers are sufficiently large: if p_i divides either $u+1$ or $v+1$, then $i \leq u+v+2$. R^* is defined from R via primitive recursive functions and bounded quantification and hence is primitive recursive in R. R^* clearly has the desired order type.

Take $<$ to be the usual ordering of the integers. Now define $<_n$ for $n = 1,2,3,\ldots$ by

$$<_1 \; = \; <$$

$$<_{n+1} \; = \; < \times <_n$$

and Q_n for $n = 0,1,2,\ldots$ by

$$Q_0 \; = \; <$$

$$Q_{n+1} \; = \; <_n^* \quad .$$

Clearly then, $|<_n| = \omega^n$ for $n = 1,2,3\ldots$ and $|Q_n| = \omega^{\omega^n}$ for $n = 0,1,2,\ldots$.

$<_{r\text{-}lex}$ is the <u>reverse lexicographical ordering</u> on n-tuples; i.e. $(x_1,\ldots,x_n) <_{r\text{-}lex} (y_1,\ldots,y_n)$ holds if and only if $x_n < y_n$ or

both $x_n = y_n$ and $x_{n-1} < y_{n-1}$ or ... or $x_n = y_n$, $x_{n-1} = y_{n-1}$,..., $x_2 = y_2$, and $x_1 < y_1$. Thus $(x_1,\ldots,x_n) <_{r\text{-}lex} (y_1,\ldots,y_n)$ if and only if $\langle x_1,\ldots,x_n\rangle <_n \langle y_1,\ldots,y_n\rangle$. In the reverse lexicographical ordering, the n+1-tuple $(x_0,x_1,\ldots,x_n)$ should be thought of as representing the ordinal $\omega^n x_n+\ldots+\omega x_1+x_0$.

Finally define

$$[R:x,y] = x \text{ if } x\ R\ y\ ;$$
$$= 0 \text{ otherwise .}$$

Note that $[R:x,y]$ is primitive recursive in R.

I. The following scheme is the schema of <u>composition</u> ($m = 0,1,2,\ldots$; $n = 1,2,\ldots$):

$$f(x_1,\ldots,x_n,y_1,\ldots,y_m) = h(x_1,\ldots,x_n,g(y_1,\ldots,y_m)) \ .$$

If f, g, and h satisfy this equation, we say that f is obtained from g and h by composition.

II. Let $x_1,\ldots,x_n$ be a non-empty list of distinct variables and let $\xi_1,\ldots,\xi_m$ be a non-empty list such that each ξ_i is either an x_j or a constant. Then the n+m-tuple $x_1,\ldots,x_n:\xi_1,\ldots,\xi_m$ is called an explicit transformation. For each explicit transformation, the following schema is a schema of <u>explicit transformation</u>:

$$f(x_1,\ldots,x_n) = g(\xi_1,\ldots,\xi_m) \ .$$

$$f(x_1,\ldots,x_n) = g(\xi_1,\ldots,\xi_m) \quad .$$

If f and g satisfy this equation, we say that f is obtained from g by an explicit transformation.

III. Let A be a term built up from previously given functions $g_1,\ldots,g_m$, variables $x_1,\ldots,x_n$, and constants. The following schema is an explicit definition:

$$f(x_1,\ldots,x_n) = A \quad .$$

If $f,g_1,\ldots,g_m$ satisfy this equation, we say that f is obtained from $g_1,\ldots,g_m$ by explicit definition.

IV. The following schema is the schema of primitive recursion:

$$f(x,0) = g(x)$$

$$f(x,y+1) = h(x,f(x,y)) \quad .$$

If f, g, and h satisfy these equations, we say that f is obtained from g and h by primitive recursion.

V. The following schema is the schema of limited recursion:

$$f(x,0) = g(x)$$

$$f(x,y+1) = h(x,y,f(x,y)) \quad .$$

$$f(x,y) \leq j(x,y)$$

If f, g, h, and j satisfy these conditions, we say that f is obtained from g, h, and j by limited recursion. The above sehema is non-effective in the sense that there is no effective procedure whereby, given g, h, and j, we can decide if there is an f satisfying the above conditions. However, we may replace the above schema by the following:

$$F(x,0) = G(x)$$

$$F(x,y+1) = H(x,y,F(x,y))$$

where $G(x) = g(x)(1\dot{-}(g(x)\dot{-}j(x,0)))$

and $H(x,y,z) = h(x,y,z)(1\dot{-}(h(x,y,z)\dot{-}j(x,y+1)))$.

Then $F(x,y) \leq j(x,y)$, and if the function f obtained from g and h by primitive recursion satisfies $f(x,y) \leq j(x,y)$, then $f = F$. Here $\dot{-}$ is proper subtraction: $x\dot{-}y = x-y$ if $y \leq x$; $x\dot{-}y = 0$ otherwise.

VI. The following schema is the schema of <u>unnested</u> R-<u>recursion</u>:

$$f(x,0) = g(x)$$

$$f(x,y) = h(x,y,f(x,t(x,y))) \quad \text{if} \quad y \neq 0 ;$$

$$t(x,0) = 0$$

$$t(x,y) \; R \; y \quad \text{if} \quad y \neq 0 .$$

If f, g, h, and t satisfy these conditions, we say that f is obtained from g, h, and t by unnested R-recursion. As before the

schema is non-effective. It may be replaced by the following effective schema:

$$f(x,0) = g(x)$$
$$f(x,y) = h(x,y,f(x,[R:t(x,y),y])) \quad \text{if} \quad y \neq 0.$$

VII. The following schema is the schema of R-annihilation:

$$m(0) = 0$$
$$m(y) = 1+m(t(y)) \quad \text{for} \quad y \neq 0 ;$$
$$t(0) = 0$$
$$t(y) \; R \; y \quad \text{for} \quad y \neq 0 \; .$$

If m and t satisfy these conditions, we say m is obtained from t by R-annihilation. This is a special case of unnested R-recursion; as before, the schema may be replaced by an effective schema. If $t^n(y)$ is the n^{th} iteration of $t(y)$, then $m(y)$ is the smallest n such that $t^n(y) = 0$. Thus $m(y)$ is a number large enough to "annihilate $t(y)$ by iteration"; hence the terminology.

VIII. The following schema is the schema of R-annihilation with a parameter:

$$m(x,0) = 0$$
$$m(x,y) = 1+m(x,t)x,y)) \quad \text{for} \quad y \neq 0;$$
$$t(x,0) = 0$$
$$t(x,y) \; R \; y \quad \text{for} \quad y \neq 0.$$

If m and t satisfy these conditions, we say that m is obtained from t by R-annihilation with a parameter. This schema is also a special case of unnested R-recursion and may be replaced by an effective schema.

IX. The following schema is a schema of nested R-recursion:

$$f(x,0) = g(x)$$
$$f(x,y) = A \quad \text{for} \quad y \neq 0 \ .$$

Here A is a term built up from the variables x and y, constants, given functions $g_1,\ldots,g_r$, and f. Furthermore, every occurrence of f in A is of the form f(B,[R:C,y]). This schema is effective; the corresponding non-effective schema is the following:

$$f(x,0) = g(x)$$
$$f(x,y) = A' \quad \text{for} \quad y \neq 0;$$
$$C \ R \ y \quad \text{for} \quad y \neq 0.$$

Here A' is a term built up from the variables x and y, constant, given functions $g_1,\ldots,g_r$, and f. C ranges over all terms such that f(B,C) appears in A'. As usual, we say that f is obtained from $g,g_1,\ldots,g_r$ by nested R-recursion.

X. The following schema is a schema of nested n-fold recursion:

$$f(x,0,\ldots,0) = g(x)$$

$$f(x,y_1,\ldots,y_n) = A \quad \text{for } (y_1,\ldots,y_n) \neq (0,\ldots,0).$$

$$(C_1,\ldots,C_n) <_{r\text{-lex}} (y_1,\ldots,y_n)$$

$$\text{for } (y_1,\ldots,y_n) \neq (0,\ldots,0).$$

Here A is a term built up from the variables $x,y_1,\ldots,y_n$ constants, previously given functions $g_1,\ldots,g_t$, and f. $(C_1,\ldots,C_n)$ ranges over all n-tuples such that $f(B,C_1,\ldots,C_n)$ appears in A. This is a non-effective schema; as before, it may be replaced by an effective schema. As usual, we say that f is obtained from g, $g_1,\ldots,g_r$ by nested n-fold recursion.

Note that the schema of explicit definition is equivalent to the two schemas of composition and explicit transformation in the sense that a class of functions is closed under explicit definitions if and only if it is closed under composition and explicit transformations.

E, the class of elementary functions, is the smallest class of functions containing addition and exponentiation as initial functions and closed under explicit definitions and limited recursion. (This definition of E is used by Grzegorczyk on page 18 of [4].)

P, the class of primitive recursive functions, is the smallest class containing addition as an initial function and closed under explicit definitions and primitive recursion.

$U(R)$, the class of unnested R-recursive functions, is the smallest class containing addition as an initial function and closed under explicit definitions, primitive recursion, and unnested R-recursion.

Remark: The functions in $U(R)$ are all recursive, even if R is not a recursive relation. To see this, simply consider the Godel-Kleene-Herbrand definition of the class of recursive functions (via equations); the equations in the schema of unnested R-recursion determine f uniquely and hence do not lead outside of the class of recursive functions.

$A(R)$ is the smallest class containing addition as an initial function and closed under explicit definitions, primitive-recursion, and R-annihilation.

$A_p(R)$ is the smallest class containing addition as an initial function and closed under explicit definitions, primitive recursion, and R-annihilation with a parameter.

$N(R)$, the class of nested R-recursive functions, is the smallest class containing addition and closed under explicit definitions, primitive recursion, and nested R-recursion.

N_n, the class of nested n-fold recursive functions, is the smallest class containing addition and predecessor as initial functions

and closed under explicit definitions and nested n-fold recursion. (Predecessor is included as an initial function so that primitive recursion will be a special case of nested n-fold recursion.)

The relativized notions, $E(f_1,\ldots,f_r)$, $P(f_1,\ldots,f_r)$, $U(R{:}f_1,\ldots,f_r)$, $A(R{:}f_1,\ldots,f_r)$, $A_p(R{:}f_1,\ldots,f_r)$, $N(R{:}f_1,\ldots,f_r)$, and $N_n(f_1,\ldots,f_r)$ may be defined by simply adding $f_1,\ldots,f_r$ as initial functions in the above definitions.

Definition: R' is <u>embeddable</u> in R if there is a primitive recursive R'-R order isomorphism of the natural numbers into the natural numbers; i.e., a primitive recursive function e such that for all y_1' and y_2', if $y_1'\ R'\ y_2'$, then $e(y_k')\ R\ e(y_2')$. We also assume without loss of generality that $e(0) = 0$. Note that if R' is embeddable in R, then $|R'| \leq |R|$. We will use variables y, y_1, y_2,... to range over the natural numbers viewed as elements of a set well ordered by R, variables y', y_1', y_2',... to range over the natural numbers viewed as elements of a set well ordered by R'.

<u>Theorem 1</u>. If R' is embeddable in R, then $N(R') \subseteq N(R)$ and $U(R') \leq U(R)$.

Proof: Special Case: Assume there is a function e' in $U(R)$ such that $e'(e(y')) = y'$ for all y' and $e(e'(y)) = y'$ if $e'(y) \neq 0$. We will prove that $N(R') \leq N(R)$; the proof that $U(R') \leq U(R)$ is a

special case of the former argument. We must show that $N(R)$ is closed under nested R'-recursion. Hence let f' be obtained from functions $g, g_1,\ldots,g_r$ of $N(R)$ by nested R'-recursion; i.e.

$$f'(x,0) = g(x)$$

$$f'(x,y') = A \quad \text{for } y' \neq 0;$$

where $C\ R'\ y'$ if $y' \neq 0$ and C appears in a term $f'(B,C)$ which appears in A. The idea of the proof is to use e and e' to effect a change of variables, perform the recursion in R, and then change variables back. Hence define f by

$$f(x,y) = f'(x,e'(y)) \ .$$

Then

$$f'(x,y') = f(x,e(y')) \ .$$

Let A_1 result from A by replacing y' by $e'(y)$. Then

$$f(x,y) = f'(x,e'(y)) = A_1 \quad \text{if } e'(y) \neq 0;$$

$$= g(x) \text{ if } e'(y) = 0.$$

Let A_2 result from A_1 by replacing each term of form $f'(B,C)$ by $f(B,e(C)(1\dot{-}(1\dot{-}e'(y))))$. Then

$$f(x,y) = A_2 \quad \text{if } e'(y) \neq 0;$$

$$= g(x) \quad \text{if } e'(y) = 0.$$

Since $C\ R'\ \dot{e}'(y)$ implies $e(C)\ R\ y$, this is a nested R-recursion. Hence $f \in N(R)$. Hence $f' \in N(R)$.

Now let us consider the general case. Let L be the set of all numbers of form $\langle e(y'),y' \rangle$. L is a primitive recursive set; i.e. L has a primitive recursive characteristic function. Now define the well ordering R'' as follows:

1) The elements of L precede the elements not in L.

2) The element $\langle e(y_1'),y_1' \rangle$ precedes the element $\langle e(y_2'),y_2' \rangle$ if $y_1'\ R'\ y_2'$ (or equivalently, if $e(y_1')\ R\ e(y_2')$).

3) The elements not in L are ordered among themselves by $<$.

Then R'' is a primitive recursive well ordering of the natural numbers with order type $|R'| + \omega$.

By the special case, $N(R') \subseteq N(R'')$. We must show that $N(R'') \subseteq N(R)$; i.e., we must show that $N(R)$ is closed under nested R''-recursion. To this end choose $g, g_1,\ldots,g_r$ from $N(R)$ and let f be obtained from them by nested R''-recursion; i.e.

$$f(x,0) = g(x)$$
$$f(x,y'') = A \quad \text{for } y'' \neq 0;$$

and $C\ R''\ y''$ if $y'' \neq 0$ and C appears in a term $f(B,C)$ which appears in A. Define f_1 by

$$f_1(x,y'') = f(x,y'') \quad \text{if } y'' \in L;$$
$$= 0 \quad \text{if } y'' \notin L.$$

Let A_1 result from A by replacing y'' by y and each term $f(B,C)$ by $h(B,[R:c_1(C),y])$. Here c_1 is the pairing function: i.e., $c_1(\langle a,b \rangle) = a$. Then

$$h(x,0) = g(x)$$
$$h(x,y) = A_1 \quad \text{for } y \neq 0$$

is a schema of nested R-recursion. Furthermore, by induction on y'',

$$f(x,y'') = h(x,c_1(y'')) \quad \text{for } y'' \in L.$$

Hence $f_1 \in U(R)$.

Now let $L(x)$ be the characteristic function of L; i.e.

$$L(x) = 1 \quad \text{for } x \in L$$
$$= 0 \quad \text{for } x \notin L.$$

Construct A_2 from A by replacing each term $f(B,C)$ by

$$f(B,C\cdot(1\dot{-}L(C)))\cdot(1\dot{-}L(C)) + f_1(C)\cdot L(C) \ .$$

Then

$$f(x,0) = g(x)$$
$$f(x,y'') = f_1(x,y'') \quad \text{for } y'' \in L$$
$$= A_2 \quad \text{for } y'' \notin L$$

is a schema of nested $<$-recursion. By the corollary to theorem 7 below, $N(R)$ is closed under nested $<$-recursion. Hence $f \in N(R)$ as was to be shown. The proof that $U(R) \subseteq U(R)$ is a special case of the above argument. This completes the proof of theorem 1.

Corollary: $N(R_i) \subseteq N(R_1 \times R_2)$ and $U(R_i) \subseteq U(R_1 \times R_2)$ for $i = 1,2$. $N(R) \subseteq N(R^*)$ and $U(R) \subseteq U(R^*)$.

Theorem 2. $N_n = N(<_n)$.

Proof: This follows from the fact that the n-place pairing functions give an order isomorphism between n-tuples of natural numbers under the reverse lexicographical ordering and the natural numbers under the ordering $<_n$. Hence a nested n-fold recursion can be converted into a $<_n$-recursion and conversely.

Theorem 3. $A_p(R) = U(R)$.

Proof: Must show that $A_p(R)$ is closed under unnested R-recursion. Let f be obtained from functions g, h, and t of $A_p(R)$ by unnested R-recursion; i.e.,

$$\begin{aligned} f(x,0) &= g(x) \\ f(x,y') &= h(x,y,f(x,t(x,y))) \quad \text{for } y \neq 0; \\ t(x,0) &= 0, \quad t(x,y)\ R\ y \quad \text{for } y \neq 0. \end{aligned}$$

We must show that $f \in A_p(R)$. Obtain m from t by R-annihilation with a parameter. Then

$$m(x,0) = 0$$
$$m(x,y) = 1 + m(x,t(x,y)) \quad \text{for } y \neq 0.$$

Then $m \in A_p(R)$. Define F by

$$F(x,y,0) = g(x)$$
$$F(x,y,n+1) = h(x,y,F(x,t(x,y),n)) \ .$$

Then F is obtained by primitive recursion and so $F \in A_p(R)$. But by induction on y (in well ordering R),

$$F(x,y) = F(x,y,m(x,y)) \ .$$

Hence $f \in A_p(R)$ as was to be shown. This completes the proof of theorem 3.

Remark: The importance of theorem 3 is that it shows that the power of well orderings lies in how slowly previously defined descending sequences descend.

Theorem 4. $U(<_n) = P$ for $n = 1,2,3,\ldots$.

Proof: By theorem 3 it is enough to show that $A_p(<_n) = P$. Hence it suffices to show that P is closed under $<_n$-annihilation with

a parameter. To this end, choose $t \in P$ and let m be obtained from t by $<_n$-annihilation with a parameter; i.e.,

$$\begin{aligned} m(x,0) &= 0 \\ m(x,y) &= 1 + m(x,t(x,y)) \qquad \text{for } y \neq 0; \\ t(x,0) &= 0, \quad t(x,y) <_n y \qquad \text{for } y \neq 0. \end{aligned}$$

We must show that $m \in P$.

Any ordinal $\alpha < \omega^n$ may be written as a polynomial

$$\alpha = \omega^{n-1}a_{n-1} + \ldots + \omega a_1 + a_0 \ .$$

For $i = 0,1,\ldots,n$, define the i-tail of α to be the ordinal given by the polynomial

$$\omega^{n-1}a_{n-1} + \ldots + \omega^i a_i \ .$$

The concept of i-tail may be mirrored in the natural numbers; i.e., in the well ordering $<_n$ the ordinal α is represented by the natural number $y = \langle a_0, a_1, \ldots, a_{n-1} \rangle$. We may define the i-tail of y to be the natural number $\langle 0,0,\ldots,0,a_i,\ldots,a_{n-1} \rangle$.

Now define the k^{th} iteration of t:

$$\begin{aligned} t(0;x,y) &= y \\ t(k+1;x,y) &= t(x,t(k;x,y)) \ . \end{aligned}$$

Then

$$m(x,y) = \mu k(t(k;x,y) = 0) \ .$$

The sequence

$$t(0;x,y),\ t(1;x,y),\ t(2;x,y),\ldots,\ t(m(x,y);x,y)$$

is a descending sequence in the well ordering $<_n$; the sequence formed by taking i-tails of the members of this sequence is non-ascending.

For each $i = 0,1,\ldots,n$ we define a function $M_i = M_i(j,x,y)$ having the following property: if $M_i(j,x,y) = k$ then either there are at least $j+1$ distinct i-tails in the sequence

$$t(0;x,y),\ t(1;x,y),\ t(2;x,y),\ldots,\ t(k;x,y)$$

or else $t(k;x,y) = 0$. This is done as follows:

$$M_0(j,x,y) = j$$

$$M_{i+1}(0,x,y) = 0$$

$$M_{i+1}(j+1,x,y) = M_i(c(n,i,y)+1,x,y) + M_{i+1}(j,x,t(M_i(c(n,i,y)+1;x,y)))\ .$$

Recall that $c(n,i,\langle y_1,\ldots,y_n\rangle) = y_i$. Then the first term on the right side of the last equation insures that the beginning of the sequence has at least $c(n,i,y)+1$ distinct i-tails and hence at least 2 distinct i+1-tails. The second term insures that the rest of the sequence is long enough for $j+1$ distinct i+1-tails. Hence the whole sequence has at least $j+2 = (j+1)+1$ distinct i+1-tails.

Clearly each M_i is primitive recursive. Now

$$m(x,y) = \mu k \leq M_n(x,y,1)(t(k;x,y) = 0) .$$

Hence $m \in P$. This completes the proof of theorem 4.

Theorem 5. If $< \times R$ is embeddable in R, then $A(R) = A_p(R) = U(R)$.

Proof. $A_p(R) = U(R)$ is theorem 3. $A(R) \subseteq A_p(R)$ is obvious. Hence, all that needs be shown is that $A(R)$ is closed under R-annihilation with a parameter. Choose $t \in A(R)$ and let m be obtained from t by R-annihilation with a parameter; i.e.,

$$\begin{aligned} m(x,0) &= 0 \\ m(x,y) &= 1+m(x,t(x,y)) \quad \text{for } y \neq 0; \\ t(x,0) &= 0, \quad t(x,y)\ R\ y \quad \text{for } y \neq 0. \end{aligned}$$

We must show that $m \in A(R)$. We do this by tucking in the parameter. Define t' by

$$\begin{aligned} t'(\langle x,0 \rangle) &= \langle 0,0 \rangle = 0 \\ t'(\langle x,y \rangle) &= \langle x,t(x,y) \rangle . \end{aligned}$$

Then $t'(w) <\times R\ w$ if $w \neq 0$. Define m' from t' by $<\times R$-annihilation; i.e.,

$$\begin{aligned} m'(0) &= 0 \\ m'(w) &= 1 + m'(t'(w)) \quad \text{if } w \neq 0. \end{aligned}$$

Then

$$m'(\langle 0,y \rangle) = m(0,y)$$

$$m'(\langle x,y \rangle) = m(x,y) + 1 \quad \text{if} \quad x \neq 0.$$

Hence $m \in A(<\times R)$.

We now show that $A(< \times R) \subseteq A(R)$. Since $< \times R$ is embeddable in R, it follows by the argument of theorem 1 that $< \times R$-annihilation can be replaced by unnested R-recursion without a parameter in t; ie., a schema of form

$$f(x,0) = g(x)$$

$$f(x,y) = h(x,y,f(x,t(y)) \quad \text{for} \quad y \neq 0;$$

$$t(0) = 0, \quad t(y) \; R \; y \quad \text{for} \quad y \neq 0.$$

By the argument of theorem 3 this can be replaced by R-annihilation. Hence $A(< \times R) \subseteq A(R)$.

Hence $m \in A(R)$ as was to be shown. This completes the proof of theorem 5.

Corollary. $A(Q_n) = U(Q_n)$.

Proof. For $n = 0$, $A(Q_n) = U(Q_n) = P$. Hence assume $n > 0$. We must show that $< \times Q_n$ is embeddable in Q_n. $|Q_n| = \omega^{\omega^n}$. Consider the map $\underline{e}$ defined by

$$\underline{e}(k,\alpha) = \omega\alpha + k$$

where $k < \omega$ and $\alpha < \omega^{\omega^n}$. Then if the pairs (k,α) are given the reverse lexicographical ordering this gives an order isomorphism from the pairs (k,α) where $k < \omega$ and $\alpha < \omega^{\omega^n}$ onto the initial segment determined by ω^{ω^n}. We mirror this map in the natural numbers; i.e., let $\upsilon(x)$ be the ordinal corresponding to x in the well ordering Q_n. Define e by

$$\upsilon(e(\langle k,x \rangle)) = \underline{e}(k,\upsilon(x)) \quad .$$

Then e is a primitive recursive $< \times Q_n - Q_n$ order isomorphism. Hence $< \times Q_n$ is embeddable in Q_n. This completes the proof of the corollary.

Lemma. $U(R)$ is closed under the schema

$$\begin{aligned} f(x,0) &= g(x) \\ f(x,y) &= h(x,y,f(r(x,y),t(x,y))) \quad \text{for } y \neq 0; \\ t(x,0) &= 0, \quad t(x,y)\ R\ y \quad \text{for } y \neq 0. \end{aligned}$$

Proof. Define F by

$$\begin{aligned} F(x,y,0) &= g(x) \\ F(x,t,n+1) &= h(x,y,f(r(x,y),t(x,y),n)) \quad . \end{aligned}$$

This is a primitive recursive schema. Define m from t by R-annihilation with a parameter. Then by induction on y in the well ordering R

$$f(x,y) = F(x,y,m(x,y)).$$

This completes the proof of the lemma.

Theorem 6. $N(R) \subseteq U(R^*)$.

Proof. We must show that $U(R^*)$ is closed under nested R-recursion. To this end choose $g, g_1,\ldots,g_r$ from $U(R^*)$ and let f be obtained from them by nested R-recursion; i.e.,

$$f(x,0) = g(x)$$
$$f(x,y) = A \qquad \text{for } y \neq 0$$

where every f-term (i.e., term whose initial symbol is f) of A is of the form $f(B,R{:}C,y)$. We must show that $f \in U(R^*)$.

Let us consider the computation of $f(a,b)$ for particular numbers a and b where $b \neq 0$. We first construct a variable free term A_1 from A by replacing x and y by a and b respectively. We construct A_{n+1} from A_n as follows: we examine the right-most f-term of A_n. Let it be $f(B,[R{:}C,d])$ where d is a constant. B and C contain no occurrences of f and therefore may be evaluated. If $[R{:}C,d] = 0$, we construct A_{n+1} from A_n by replacing $f(B,[R{:}C,d])$ by $g(B)$; if $[R{:}C,d] \neq 0$, we construct A_{n+1} from A_n by replacing $f(B,[R{:}C,d])$ by the result of substituting in A the values of B and $[R{:}C,d]$ for x and y respectively. We get a sequence of terms satisfying $A_1 = A_2 = \ldots$. Some A_m will contain no occurrence of f;

the value of that A_m is $f(a,b)$. The idea of the proof is to assign an ordinal $<|R^*|$ to each term A_i in such a way that the associated sequence of ordinals is decreasing. We arithmetize the computation via godel numbers, mirror the assignment of ordinals in R^*, and are thus able to define f by an unnested R^*-recursion.

We call a term a variable free term if it contains no numerical variables, an evaluable term if it is variable free and contains no occurrences of f, a reducible term if it is variable free and not evaluable (i.e. has at least one occurrence of f). For each reducible term D, let D^+ be the term which results from D by applying the reduction described in the previous paragraph; i.e., A_n^+ is A_{n+1}.

For every natural number k, let $\upsilon(k)$ be the ordinal $<|R|$ corresponding to k in the well ordering R. Let q be the number of occurrences of f in A. To each variable free f-term $f(B,[R:C,b])$ where b is a constant assign a non-zero ordinal $\alpha < \omega^{|R|}$ as follows:

$$\alpha = \omega^{\upsilon} q \text{ where } \upsilon = \upsilon([R:C,b]) \quad \text{if } C \text{ is evaluable;}$$

$$= \omega^{\upsilon(b)} \quad \text{if } C \text{ is reducible.}$$

To each variable free term D assign an ord(D) as follows: Let $F_1,\ldots,F_s$ be all the occurrences of f-terms in D in order from left to right; i.e., F_s is the rightmost f-term of D. (If the same f-term appears twice in A it appears twice in the list.) Let $\alpha_1,\ldots,\alpha_s$ be the assigned ordinals. Then

$$\underline{ord}(D) = \alpha_1 \# \ldots \# \alpha_s \ .$$

Here $\#$ denotes coordinate wise addition; i.e.,

$$(\ldots + \omega^{\beta} d_1 + \ldots) \# (\ldots + \omega^{\beta} d_2 + \ldots) = \ldots + \omega^{\beta}(d_1 + d_2) + \ldots \ .$$

Claim: If D is reducible, then $\underline{ord}(D^+) < \underline{ord}(D)$.

Let $\ldots, F_{ji}, \ldots$ be the f-terms of D^+ in order; here $j = 1, \ldots, s$ and $i = 1, \ldots, t_j$. (The terms F_{ji} for $i = 1, \ldots, t_j$ of D^+ are those which arise from the term F_j of D.) Let C_{jk} be the ordinal assigned to F_{ji}. Recall that D^+ results from D by replacing F_s which is of the form $f(B,[R:C,b])$ by a term U, where U is $g(B)$ if $[R:C,b] = 0$ and U results from A by replacing x and y by B and $[R:C,b]$ respectively if $[R:C,b] \neq 0$.

Case 1. If F_j does not contain F_s, then $t_j = 1$, F_{j1} is F_j, and $\alpha_{j1} = \alpha_j$.

Case 2. If F_j properly contains F_s, then $t_j = 1$, and F_{j1} results from F_j by replacing F_s by U. F_j is of the form $f(B_1,[R:C_1,d])$; F_{j1} is of the form $f(B_2,[R:C_2,d])$. If C_1 is evaluable, then C_2 is C_1 and $\alpha_{j1} = \alpha_j$. If C_1 is reducible, then $\alpha_j = \omega^{v(d)} \geq \alpha_{j1}$.

Case 3. The remaining case is $j = s$ so that F_j is F_s. Either $[R:C,b] \neq 0$, in which case $t_j = 0$, or else $[R:C,b] \neq 0$, in which case $t_j = q$ and $F_{s1}, \ldots, F_{sq}$ are the f-terms of the term resulting from A by replacing x and y by B and $[R:C,b]$ respectively.

In this latter case each F_{si} is of the form $f(B_i,[R:C_i,[R:C,b]])$. If C_i is evaluable,

$$\alpha_{si} = \omega^{\lambda}q < \omega^{\upsilon}$$

where $\lambda = \upsilon([R:C_i[R:C,b]])$ and $\upsilon = \upsilon([R:C,b])$. If C_i is reducible, then $\alpha_{si} = \omega^{\upsilon}$. Hence, since at least one F_{si} is evaluable, we have $\alpha_{s1}\#\ldots\#\alpha_{sq} < \omega^{\upsilon}q = \alpha_s$. Hence, $\underline{ord}(D^+) = \alpha_{11}\#\ldots\#\alpha_{sq} < \alpha_1\#\ldots\#\alpha_s = \underline{ord}(D)$. This completes the proof of the claim.

Now for each term D, let "D" be its godel number. Let red be the primitive recursive function satisfying

$$\begin{aligned} red("D") &= "D^+" && \text{if } D \text{ is reducible;} \\ &= "D" && \text{otherwise.} \end{aligned}$$

Let ord be the primitive recursive function which mirrors $\underline{ord}$; i.e.

$$\upsilon^*(ord("D")) = \underline{ord}("D")$$

where $\upsilon^*(k)$ is the ordinal corresponding to k in the well ordering R*. Let sub be the primitive recursive function such that sub(a,b) is the godel number of the term which results from A by replacing x and y by a and b respectively. Let val be the function, primitive recursive in $g, g_1,\ldots,g_r$, such that

$$\begin{aligned} val("D") &= D && \text{if } D \text{ is evaluable;} \\ &= 0 && \text{otherwise.} \end{aligned}$$

Define m so that

$$m("D",y) = 0 \quad \text{if } D \text{ is evaluable;}$$
$$= 1 + m(red("D"),[R^*:ord(red("D")),y]) \quad \text{if } D \text{ is reducible.}$$

By the lemma immediately preceding this theorem, $m \in U(R^*)$. Let $m(w) = m(w,ord(W))$. Then $m("D")$ is the number of reductions required to make D evaluable. Let $red(n;w)$ be the n^{th} iteration of $red(w)$; i.e.,

$$red(0;w) = w$$
$$red(n+1;w) = red(red(n;w)) \ .$$

Then

$$f(x,0) = g(x)$$
$$f(x,y) = val(red(m(sub(x,y));sub(x,y)))) \quad \text{if } y \neq 0.$$

This completes the proof of theorem 6.

Theorem 7. $N(<_{n+1}) \subseteq U(Q_n)$ for $n = 0,1,2,\ldots$.

Proof. The idea is the same as in theorem 6; only the method of assigning ordinals is changed.

Let f be obtained by nested $<_{n+1}$-recursion from $g, g_1,\ldots,g_r$; i.e.,

$$f(x,0) = g(x)$$
$$f(x,y) = A \quad \text{for } y \neq 0;$$

where every f-term of A is of the form $f(B,[<_{n+1}:C,y])$. As before let q be the number of occurrences of f in A. Let h be a primitive recursive function such that

$$0 < q^2 h(i) < h(j) \qquad \text{for } i < j \ .$$

Such an h may be given by $h(0) = 1$, $h(k+1) = q^2 h(x) + 1$.

Define functions φ_1 and φ_2 mapping the initial segment determined by ω^{n+1} into the initial segment determined by ω^{ω^n} as follows: For $\alpha < \omega^{n+1}$ write $\alpha = \omega\beta + b$ where $\beta < \omega^n$ and $b < \omega$. Then

$$\varphi_1(\alpha) = \omega^{\beta} qh(b)$$

$$\varphi_2(\alpha) = \omega^{\beta} h(b) \ .$$

The important facts are that

($) $$\varphi_1(\gamma) \leq \varphi_2(\alpha) \qquad \text{and}$$

($$) $$\varphi_2(\alpha)(q-1) + \varphi_1(\gamma)q < \varphi_1(\alpha)$$

for $\gamma < \alpha < \omega^{n+1}$.

For each natural number k let $\upsilon(k)$ be the ordinal number $<\omega^{n+1}$ corresponding to k in the well ordering $<_{n+1}$.

To each f-term $f(B,[<_{n+1}:C,b])$ where b is a constant assign a non-zero ordinal α as follows:

$$\alpha = \varphi_1(v([<_{n+1}:C,b])) \quad \text{if } C \text{ is evaluable;}$$

$$= \varphi_2(v(b)) \quad \text{if } C \text{ is reducible.}$$

To each variable free term D, assign an ordinal $\underline{ord}_1(D) < \omega^{\omega^n}$ as follows: let $F_1,\ldots,F_s$ be the f-terms of D in order. Let $\alpha_1,\ldots,\alpha_s$ be the assigned ordinals. Then

$$\underline{ord}_1(D) = \alpha_1\#\ldots\#\alpha_s \ .$$

Claim: If D is reducible, $\underline{ord}_1(D^+) < \underline{ord}_1(D)$.

Let $\ldots,F_{ji},\ldots$ be the f-terms of D^+ $(j = 1,\ldots,s;\ i = 1,\ldots,t_j)$ and let α_{ji} be the ordinal assigned to F_{ji}. (The terms F_{ji} for $i = 1,\ldots,t_j$ of D^+ are those which arise from the term F_j of D.) Recall that D^+ results from D by replacing F_s which is of form $f(B,[<_{n+1}:C,b])$ where C is evaluable by U where U is $g(B)$ if $[<_{n+1}:C,b] = 0$ and U results from A by replacing x and y by B and $[<_{n+1}:C,b]$ respectively if $[<_{n+1}:C,b] \neq 0$.

Case 1. If F_j does not contain F_s, then $t_j = 1$, F_{j1} is F_j, and $\alpha_{ji} = \alpha_j$.

Case 2. If F_j properly contains F_s, then $t_j = 1$ and F_{j1} results from F_j by replacing F_s by U. F_j is of the form $f(B_1,[<_{n+1}:C_1,d])$; F_{j1} is of the form $f(B_2,[<_{n+1}:C_2,d])$. If C_1 is evaluable, then C_2 is C_1 and $\alpha_j = \alpha_{j1}$. If C_1 is reducible, then $\alpha_j = \varphi_2(v(d)) \geq \alpha_{j1}$ by ($).

Case 3. The remaining case is $j = s$ so that F_j is F_s. Either $[<_{n+1}:C,b] = 0$ in which case $t_s = 0$ or else $[<_{n+1}:C,b] \neq 0$ in which case $t_s = q$ and $F_s,\ldots,F_{sq}$ are the f-terms of the term resulting from A by replacing x and y by B and $[<_{n+1}:C,b]$ respectively. In this latter case, each F_{si} is of the form $f(B_i,[<_{n+1}:C_i[<_{n+1}:C,b]])$. If C_i is evaluable,

$$\alpha_{si} = \varphi_1([<_{n+1}:C_i[<_{n+1}:C,b]]) .$$

If C_i is reducible,

$$\alpha_{si} = \varphi_2([<_{n+1}:C,b]) .$$

At most q of the C_i are evaluable, and since at least one C_i is evaluable, at most q-1 of the C_i are reducible. Hence, by ($$)

$$\alpha_{si}\#\ldots\#\alpha_{sq} < \alpha_s .$$

Hence

$$\underline{ord}_1(D^+) = \alpha_{1i}\#\ldots\#\alpha_{sq} < \alpha_1\#\ldots\#\alpha_s = \underline{ord}_1(D) .$$

This completes the proof of the claim.

The rest of the proof proceeds exactly as in theorem 6.

Corollary: N(R) and U(R) are closed under nested 1-fold recursion (i.e. nested $<$-recursion).

Proof. Let f be obtained from functions $g,g_1,\ldots,g_r$ by nested $<$-recursion. Then $f \in N(<;g,g_1,\ldots,g_r)$. By relativizing the previous theorem, $f \in U(<;g,g_1,\ldots,g_r)$. By relativizing theorem 4, $f \in P(g,g_1,\ldots,g_r)$. But N(R) and U(R) are closed under the

primitive recursive operations. This completes the proof of the corollary.

Definition. Let

$$f(x,0) = g(x)$$
$$f(x,y) = A \qquad \text{for } y \neq 0$$

be a schema of nested R-recursion. The schema is called <u>essentially unnested</u> if every f-term of A is of the form $f(B,[R:C,y])$ where neither B nor C contains an occurrence of f.

<u>Theorem 8</u>. $U(R)$ is closed under essentially unnested R-recursion.

Proof. Let f be obtained from functions $g,g_1,\ldots,g_r$ of

$$f(x,0) = g(x)$$
$$f(x,y) = A \qquad \text{for } y \neq 0.$$

We must show that $f \in U(R)$.

We say that a term D is <u>essentially unnested</u> if no occurrence of f in D is within the scope of some other occurrence of f; for example, A is essentially unnested. For any variable free, reducible, essentially unnested term D, define $D^{\times}$ as follows: let $F_1,\ldots,F_s$ be the f-terms of D. Each F_i has the form $f(B_i,[R:C_i,b_i])$ where b_i is a constant. Let U_i be $g(B_i)$ if $[R:C_i,b_i] = 0$; let U_i result from A by replacing x and y by B_i and $[R:C_i,b_i]$ respectively if $[R:C_i,b_i] \neq 0$. Then $D^{\times}$ results from D by simultaneously replacing each F_i by U_i.

As before define $\upsilon(k)$ to be the ordinal corresponding to the natural number k in the well ordering R.

Now for every variable free, essentially unnested term D, assign an ordinal $\underline{ord}_2(D) < |R|$ by

$$\underline{ord}_2(D) = \max(\upsilon(R{:}C_1,b_1)+1,\ldots,\upsilon(R{:}C_s,b_s)+1) \ .$$

The $+1$ is added so that $\underline{ord}_2(D) = 0$ if D is evaluable and $\underline{ord}_2(D) \neq 0$ if D is reducible.

Claim: If D is reducible and essentially unnested, then $D^{\times}$ is essentially unnested and $\underline{ord}_2(D^{\times}) < ord_2(D)$.

The claim follows immediately from the definitions. The rest of the proof proceeds as in theorem 6.

Theorem 9. $U(R^*) \subseteq N(<_R)$.

Proof. It suffices to show that $A_p(*) \subseteq N(<_R)$; i.e., that $N(<_R)$ is closed under R^*-annihilation with a parameter. Choose $t \in N(<_R)$ and suppose m is obtained from t by R^*-annihilation with a parameter; i.e.

$$\begin{aligned} m(x,0) &= 0 \\ M(x,y) &= 1 + m(x,t(x,y)) \quad \text{for } y \neq 0; \\ t(x,0) &= 0, \quad t(x,y)\ R^*\ y \quad \text{for } y \neq 0. \end{aligned}$$

We must show that $m \in N(<_R)$.

Let $t(k;x,y)$ be the k^{th} iteration of $t(x,y)$; i.e.

$$\begin{aligned} t(0;x,y) &= y \\ t(k+1;x,y) &= t(x,t(k;x,y)). \end{aligned}$$

Then $m(x,y) = \mu k(t(k;x,y) = 0)$. Hence, our problem is to find a function of $N(<\!\!\times R)$ which bounds $m(x,y)$.

Recall that $|R^*| = \omega^{|R|}$. Thus an ordinal $\alpha < |R^*|$ may be written as a polynomial:

$$\alpha = \ldots + \omega^{\beta}b + \ldots$$

where $|R| > \ldots > \beta > \ldots$ and $0 \leq b < \omega$. For ordinals $\alpha < |R^*|$ and $\gamma < |R|$ we may write

$$(*) \qquad \alpha = \ldots + \omega^{\beta}b + \omega^{\gamma}c + \omega^{\delta}d + \ldots$$

where $R > \ldots \beta < \gamma < \delta \ldots$ and $0 < b,d < \omega$ and $0 \leq c < \omega$. The γ-tail of α is the ordinal given by the polynomial

$$\ldots + \omega^{\beta}b + \omega^{\gamma}c \quad .$$

Further define

$$\underline{B}(\alpha,\gamma) = \delta$$
$$\underline{C}(\alpha,\gamma) = d + 1$$

Let $\upsilon^*(y)$ be the ordinal corresponding to the natural number y in the well ordering R^*. Let $\upsilon(u)$ be the ordinal corresponding to the natural number u in the well ordering R. We let $\underline{t}$ mirror t in the ordinals; i.e.

$$\underline{t}(k;x,\upsilon^*(y)) = \upsilon^*(t(k;x,y)) \quad .$$

Let S be the following sequence or ordinals:

$$\underline{t}(0;x,\alpha),\ \underline{t}(1;x,\alpha),\ \underline{t}(2;x,\alpha),\ldots,\ \underline{t}(k;x,\alpha).$$

Let $S(\gamma)$ be the sequence which results from S by taking γ-tails. Then S is strictly descending (so long as it is non-zero) and $S(\gamma)$ is non-ascending.

We will define a function $\underline{M}(x,\alpha,j,\gamma)$ such that if $\underline{M}(x,\alpha,j,\gamma) = k = 1 +$ the length of S then <u>either</u> (1) there are $j+1$ distinct elements of $S(\gamma)$; i.e., in the non-ascending sequence $S(\gamma)$ strict inequality occurs j times; <u>or else</u> (2) $\underline{t}(k;x,\alpha) = 0$.

$\underline{M}$ can be given by

$$\underline{M}(x,\alpha,j,0) = j$$

$$\underline{M}(x,\alpha,0,\gamma) = 0$$

$$\underline{M}(x,\alpha,j+1,\gamma) = \underline{M}\{x,\alpha,\underline{C}(\alpha,\gamma),\ \underline{B}(\alpha,\gamma)\} + \underline{M}\{x,\ t(\underline{M}\{x,\alpha,\underline{C}(\alpha,\gamma),\ \underline{B}(\alpha,\gamma)\};x,\alpha),j,\gamma\}\ .$$

Since the 0-tail of α is α itself, the sequence $S(0)$ is the same as S; hence strictly descending as long as it is non-zero. Therefore the first equation above is sufficient. To produce $0+1 = 1$ distinct γ-tail, one need only take the sequence S to be of length $0+1 = 1$. Hence the second equation above suffices. The last equation says the following: to get $j+1$ distinct γ-tails we first get $d+2$ distinct δ-tails. (See (*).) For each change in the δ-tail, either the coefficient of ω^{δ} decreases or the δ-tail changes. But the coefficient of ω^{δ} in α is $d = \underline{C}(\alpha,\gamma) - 1$; d can be forced at most d times and then the γ-tail must change. Hence the first

term on the right in the last equation insures at least one change in the γ-tail. The second term merely instructs us to produce j more changes in the γ-tail of the result.

Let $M = M(x,y,j,u)$ mirror $\underline{M} = \underline{M}(x,\alpha,j,\gamma)$; i.e.,

$$M(x,y,j,u) = \underline{M}(x,\upsilon^*(y),j,\upsilon(u)) \ .$$

Similarly let B and C mirror $\underline{B}$ and $\underline{C}$:

$$B(y,u) = \underline{B}(\upsilon^*(y), \upsilon(u))$$
$$C(y,u) = \underline{C}(\upsilon^*(y), \upsilon(u)) \ .$$

Clearly B and C are primitive recursive in R (and hence primitive recursive. Furthermore

$$M(x,y,j,0) = j$$
$$M(x,y,0,u) = 0$$
$$M(x,y,j+1,u) = M\{x,y,C(y,u),B(y,u)\} + M\{x,t(M\{x,y,C(y,u),B(y,u)\};j,u\}.$$

If we write $M(x,y,\langle j,u\rangle) = M(x,y,j,u)$ and make appropriate changes in the above equations we have an instance of nested $<\times R$-recursion (the variable j is a recursion variable in $<$; the variable u is a recursion variable in R.). Thus $M \in U(<\times R)$.

Let $\beta = \beta(\alpha)$ be the largest exponent appearing in the polynomial for α. Let $\underline{b} = \underline{b}(\alpha)$ be the coefficient of ω^{β}. Reasoning as before, there can be at most $\underline{b}+1$ distinct β-tails in

the sequence S. Hence if $k = \underline{M}(x,\alpha,\underline{b}+1,\beta)$, (1) must fail; hence (2) must hold, i.e., $\underline{t}(k;x,\alpha) = 0$. Now let $b = b(y)$ mirror $\underline{b} = \underline{b}(\alpha)$; i.e.,

$$\underline{b}(\upsilon^*(y)) = \upsilon(b(y)) \quad .$$

Let $B = B(y)$ mirror $\beta = \beta(\alpha)$; i.e.,

$$\beta(\upsilon^*(y)) = \upsilon(B(y)) \quad .$$

Then

$$m(x,y) = \mu k \leq M(x,y,b(y)+1,B(y))(m(k;x,y) = 0) \quad .$$

Hence $m \in N(<\!\times R)$ as was to be shown. This completes the proof of theorem 9.

Corollary. $U(Q_n) \subseteq N(<_{n+1})$ for $n = 0,1,2,\ldots$.

Theorem 10. If $<\!\times R$ is embeddable in R, then $N(R) = U(R^*)$.

Proof. Theorems 1, 6, and 9.

Theorem 11. $N_{n+1} = N(<_{n+1}) = U(Q_n) = A(Q_n)$ $n = 0,1,2,\ldots$.

Proof. Theorems 2 and 7 and the corollaries to theorems 5 and 9.

Theorem 12. If $<$ is embeddable in R, then $U(R) \neq N(R\times\!<)$.

Proof. We will in fact sketch the construction of a function F in $N(R\ll)$ which enumerates $U(R)$; i.e., for every function $f \in U(R)$ there is a number w such that for all $x_1,\ldots,x_n$ (f is n-ary) $F(w, \langle x_1,\ldots,x_n\rangle) = f(x_1,\ldots,x_n)$. It will then follow by the usual diagonalization argument that $F \notin U(R)$.

Since $<$ is embeddable in R, primitive recursion can be replaced by unnested R-recursion via the techniques of theorem 1. Hence $U(R)$ may be defined as the smallest class of functions containing a certain set of (primitive recursive) functions as initial functions (i.e., addition and those required to make the translations of theorem 1) and closed under unnested R-recursion, composition, and explicit transformations.

Now notice that there is exactly one schema of unnested R-recursion, but there are infinitely many schemata of composition (one for each pair (n,m) where $m \neq 0$; namely the schema which composes an $n+1$-ary function with an m-ary function to get an $m+n$-ary function) and there are infinitely many schemata of explicit transformation (i.e., one for each explicit transformation).

We number these schemata in such a way that the following two conditions are satisfied:

(1) There are primitive recursive functions C_1 and C_2 such that if z is the number of the composition schema which composes an $n+1$-ary function with an m-ary function, then

$$C_1(z, \langle x_1,\ldots,x_n, y_1,\ldots,y_m \rangle, u) = \langle x_1,\ldots,x_n,u\rangle$$

and

$$C_2(z, \langle x_1,\ldots,x_n, y_1,\ldots,y_m \rangle) = \langle y_1,\ldots,y_m\rangle .$$

(2) There is a primitive recursive function E such that if z is the number of the explicit transformation $x_1,\ldots,x_n:\xi_1,\ldots,\xi_m$, then

$$E(z, \langle x_1,\ldots,x_n \rangle) = \langle \xi_1,\ldots,\xi_m\rangle .$$

At this point it is easy to write down the definition of F; we leave the details to the reader.

THE EXTENDED GRZEGORCZYK HIERARCHY

For each limit ordinal ordinal $\alpha<\omega^{\omega}$, we define a function $\alpha[n]$ so that the sequence $\alpha[0], \alpha[1], \alpha[2],\ldots$ is a strictly increasing sequence of ordinals with limit α. To do this write $\alpha = \omega^{k+1}(\beta+1)$ where $k \geq 0$ and $\beta < \omega^{\omega}$. (This can easily be done by writing α as a polynomial in ω and factoring out the highest common power of ω.) Then set $\alpha[n] = \omega^{k+1}\beta + \omega^{k}n$.

The W Functions:

For each $\alpha<\omega^{\omega}$ define a function W_{α} as follows:

$$W_0(x) = 2^x ;$$

$$W_{\alpha+1}(x) = W_{\alpha}^{x}(1), \text{ i.e.}$$

$$W_{\alpha+1}(0) = 1, \quad W_{\alpha+1}(x+1) = W_{\alpha}(W_{\alpha+1}(x));$$

$$W_{\alpha}(x) = W_{\alpha[x]}(x) \quad \text{for } \alpha \text{ a limit ordinal.}$$

In other words, $W_{\alpha+1}$ is obtained from W_{α} by iteration; and, if α is a limit ordinal, W_{α} is obtained by diagonalization. We shall use the notation $W_{\alpha}^{x}(y)$ quite frequently; note, for example, that $W_{\alpha}^{x}(W_{\alpha+1}(y)) = W_{\alpha+1}(y+x)$.

For each $\alpha<\omega^{\omega}$ define E_{α} as the smallest class of functions containing addition and W_{α} as initial functions and closed under

explicit definitions and bounded recursion. Equivalently, $E_\alpha = E(W_\alpha)$, the class of functions which are elementary recursive in W_α.

The Extended Grzegorczyk Theorem:

(1) If $\alpha \leq \beta$, then $E_\alpha \subseteq E_\beta$.

(2) If $\alpha < \beta$, then $E_\alpha \neq E_\beta$.

(3) $N_n = \bigcup E_\alpha(\alpha < \omega^n)$.

The proof consists of ten lemmas. For $\beta < \omega^\omega$ define the maximum coefficient of β, in symbols, maxcoeff(β), as follows: write $\beta = \omega^n b_n + \ldots + \omega b_1 + b_0$; then $\text{maxcoeff}(\beta) = \max\{b_0, b_1, \ldots, b_n\}$.

Lemma 1: Properties of the W Functions

(1) $x < W_\alpha(x)$.

(2) $W_\alpha(x) \leq W_{\alpha+\beta}(x)$, if $\alpha = \omega^n\gamma$, and $\beta < \omega^{n+1}$ $(n = 0,1,2,\ldots)$.

(3) W_α is strictly increasing.

(4) for $x > \text{maxcoeff}(\beta)$ and $\beta \leq \alpha$, $W_\beta(x) \leq W_\alpha(x)$.

Proof of (1). By induction on α.

Case 1. $\alpha = 0$. Clear, since $x < 2^x$.

Case 2. Assume (1) for α; will prove it for $\alpha+1$ by induction on x. If $x = 0$, $W_{\alpha+1}(x) = 1 > 0$. Suppose $x < W_{\alpha+1}(x)$. Then $x+1 \leq W_{\alpha+1}(x)$. By induction hypothesis on α, $W_{\alpha+1}(x) < W_\alpha(W_{\alpha+1}(x))$. Hence, $x+1 \leq W_{\alpha+1}(x) < W_\alpha(W_{\alpha+1}(x)) = W_{\alpha+1}(x+1)$.

Case 3. α is a limit ordinal. Then by induction hypothesis on α, (1) holds for $\alpha[x]$. Hence, $x < W_{\alpha[x]}(x) = W_\alpha(x)$. This completes the proof of (1).

Proof of (2). By induction of β. We assume W_α is increasing.

Case 1. $\beta = 0$. Clear.

Case 2. Assume (2) for β. If $x = 0$, then $W_\alpha(x) = 1 = W_{\alpha+\beta+1}(x)$. If $x \neq 0$, then by (1), $x \leq W_{\alpha+\beta+1}(x-1)$. Hence,

$$W_\alpha(x) \leq W_\alpha(W_{\alpha+\beta+1}(x-1)) \leq W_{\alpha+\beta}(W_{\alpha+\beta+1}(x-1)) = W_{\alpha+\beta+1}(x) .$$

Case 3. β is a limit ordinal. Since $\beta < \omega^{n+1}$, $\alpha + \beta[x] = (\alpha+\beta)[x]$. Hence, by induction hypothesis, $W_\alpha(x) \leq W_{\alpha+\beta[x]}(x) = W_{(\alpha+\beta)[x]}(x) = W_{\alpha+b}(x)$ $= W_{\alpha+\beta}(x)$.

This completes the proof of (2) under the assumption that W_α is increasing.

Proof of (3). By induction on α.

Case 1. $\alpha = 0$. Clear.

Case 2. Will prove (3) for $\alpha+1$. By (1), $W_{\alpha+1}(x) < W_\alpha(W_{\alpha+1}(x)) = W_{\alpha+1}(x+1)$. Hence, (3) follows by iteration.

Case 3. α is a limit ordinal. Write $\alpha = \omega^{n+1}(\alpha'+1)$. Take $x < y$. Then $\alpha[x] = \omega^{n+1}\alpha' + \omega^n x = \omega^n(\omega\alpha' + x)$; $\alpha[y] = \omega^{n+1}\alpha' + \omega^n y = \omega^n(\omega\alpha' + x) + \omega^n(y - x)$. By induction hypothesis, $W_{\alpha[x]}$ is increasing. Hence by (2), $W_\alpha(x) = W_{\alpha[x]}(x) \leq W_{\alpha[y]}(x)$. By induction hypothesis, $W_{\alpha[y]}$ is increasing. Hence, $W_{\alpha[y]}(x) < W_{\alpha[y]}(y) = W_\alpha(y)$. Hence, $W_\alpha(x) < W_\alpha(y)$.

This completes the proof of (3) and also of (2).

Proof of (4). By induction on α.

Case 1. $\beta = \alpha$. Clear.

Case 2. Assume (4) for α; will prove it for $\alpha+1$. Then, $W_\beta(x) \leq W_\alpha(x)$. By (2) (with $n = 0$), $W_\alpha(x) \leq W_{\alpha+1}(x)$. Hence, $W_\beta(x) \leq W_{\alpha+1}(x)$.

Case 3. α is a limit ordinal with $\beta < \alpha$. Write

$$\beta = \omega^n b_n + \ldots + \omega^{k+1} b_{k+1} + \omega^k b_k + \ldots + \omega b_1 + b_0 \quad ;$$

$$\alpha = \omega^n a_n + \ldots + \omega^{k+1} a_{k+1} \qquad (\text{where } a_{k+1} > 0) \quad ;$$

$$\alpha[x] = \omega^n a_n + \ldots + \omega^{k+1}(a_{k+1} - 1) + \omega^k x \quad .$$

Since $\beta < \alpha$, either $\beta \leq \alpha[0] \leq \alpha[x]$ or $\beta = \alpha[0]$, in which case $\beta < \alpha[x]$ since $x > \text{maxcoeff}(\beta) \geq b_k$. Thus, by induction hypothesis, $W_\beta(x) \leq W_{\alpha[x]}(x) = W_\alpha(x)$.

This completes the proof of (4) and hence of lemma 1.

Roughly speaking, $W_\alpha(x)$ is strictly increasing in x for fixed α, and under certain conditions is non-decreasing in α for fixed x. In the sequel we use these properties without explicitly referring to them.

Our next aim is the Bounding Lemma. It says that every function $f \in N_n$ is bounded by some W_α with $\alpha < \omega^n$. (A constant must be thrown in to compensate for large values of f for small arguments.) In order to carry through the proof, we generalize slightly the concepts of term and nested n-fold recursion.

Definition of term (to be used in a definition by nested n-fold recursion): Take the following primitive symbols:

(i) constants (i.e., numerals) 0,1,2,... .

(ii) for each $k = 1,2,3,\ldots$, a list of k-ary function (letters) $\ldots, g^{(k)}, \ldots$.

(iii) variables $x, y_1,\ldots,y_n$ (ranging over the natural numbers).

(iv) the n+1-ary function (letter) f and the n-ary function (letters) $f_0,f_1,f_2,\ldots$.

(v) (notations for) addition, multiplication, and the functions $I_0,I_1,I_2,\ldots$ defined by

$$I_k(x) = 1 \quad \text{if } x = k ;$$
$$= 0 \quad \text{if } x \neq k .$$

For simplicity we will use the same notation for a function letter and a function assigned as a value of that function letter. More generally, we shall not use distinct notations for a term (a linguistic object) and the value of that term (a number). The discriminating reader will keep the distinction in mind.

Terms are defined inductively:

(T1) a constant or a variable is a term.

(T2) if h is an m-ary function letter and $A_1,\ldots,A_m$ are terms, then $h(A_1,\ldots,A_m)$ is a term.

(T3) if B, $C_1,\ldots,C_{n-1},C_n$ are terms, then

$$\sum_{i=0}^{k} f_i(B,C_i,\ldots,C_{n-1})\cdot I_i(C_n)$$

is a term (for each natural number k).

(T4) only those things are terms as required by (T1)-(T3).

Now we define inductively the <u>depth</u> and <u>length</u> of a term.

(DL1) if k is a constant, $\text{depth}(k) = 0$; $\text{length}(k) = k$.

(DL2) if z is a variable, $\text{depth}(z) = \text{length}(z) = 0$.

(DL3) if h is an m-ary function letter and $A_1,\ldots,A_m$ are terms, then

$$\text{depth}(h(A_1,\ldots,A_m)) = 1 + \max\{\text{depth}(A_1),\ldots,\text{depth}(A_m)\} \quad ;$$

$$\text{length}(h(A_1,\ldots,A_m)) = 1 + m + \max\{\text{length}(A_1),\ldots,\text{length}(A_m)\}.$$

(DL4) if $B, C_1,\ldots,C_{n-1},C_n$ are terms, then

$$\text{depth} \sum_{i=0}^{k} f_i(B,C_1,\ldots,C_{n-1})\cdot I_i(C_n) = \text{depth}(f(B,C_1,\ldots,C_{n-1},C_n));$$

$$\text{length} \sum_{i=0}^{k} f_i(B,C_1,\ldots,C_{n-1})\cdot I_i(C_n) = \text{length}(f(B,C_1,\ldots,C_{n-1},C_n)).$$

A schema of <u>nested n-fold recursion</u> $(n = 1,2,3,\ldots)$ is a set of conditions

$$f(x,0,\ldots,0) = A^0 \ ;$$

$$f(x,y_1,\ldots,y_n) = A \quad \text{for } (y_1,\ldots,y_n) \neq (0,\ldots,0);$$

and for every term of form $f(B,C_1,\ldots,C_n)$ which appears in A

$$(C_1,\ldots,C_n) <_{r\text{-}lex} (y_1,\ldots,y_n) \quad \text{for } (y_1,,,.y_n) \neq (0,\ldots,0).$$

Here x is the only variable appearing in A^0 and f does not appear in A^0. Recall that $<_{r\text{-}lex}$ is the reverse lexicographical ordering on n-tuples.

<u>Lemma 2: Explicit Definition Bounding Lemma</u>

Let A be a term. Let $D = \text{depth}(A)$, $L = \text{length}(A)$. Suppose s, a, and α satisfy

$$h(u_1,\ldots,u_m) \leq W_\alpha^s(u_1 + \ldots + u_m + a)$$

for every function h appearing in A. Let $z_1,\ldots,z_r$ be all the variables appearing in A. Then

$$A \leq W_\alpha^{Ds+L}(z_1+\ldots+z_r+a) \quad .$$

Proof. By induction on the structure of A.

Case 1. A is a constant k. Then $D = 0$, $L = k$, and the lemma follows from

$$k \leq W_\alpha^k(0) \quad .$$

This last inequality is an obvious consequence of property 1 by induction on k.

Case 2. A is a variable x. Then $D = L = 0$. The lemma follows from $x \leq x$.

Case 3. A is $h(A_1,\ldots,A_m)$. Let $D_i = \text{depth}(A_i)$, $L_i = \text{length}(A_i)$. Let $K = \max\{D_1s+L_1,\ldots,D_ms+L_m\}$. Let J range over D_is+L_i $(i = 1,\ldots,m)$. Then, by induction hypothesis,

$$\begin{aligned} A &\leq W_\alpha^s(\ldots+W_\alpha^J(z_1+\ldots+z_r+a)+\ldots+a) \\ &\leq W_\alpha^s((m+1)\cdot W_\alpha^K(z_1+\ldots+z_r+a)) \\ &\leq W_\alpha^s(W_0^{m+1}(W_\alpha^K(z_1+\ldots+z_r+a))) \\ &\leq W_\alpha^s(W_\alpha^{m+1}(W_\alpha^K(z_1+\ldots+z_r+a))) \\ &= W_\alpha^{s+m+1+K}(z_1+\ldots+z_r+a) \quad . \end{aligned}$$

(The transition from the second line to third line above follows from $(m+1)\cdot x \leq W_0^{m+1}(x)$. This is easily proved by induction on m and the fact that $W_0(x) = 2^x$.) But

$$K \leq (\max\{D_1,\ldots,D_m\})\cdot s + \max\{L_1,\ldots,L_m\}$$

$$= (D-1)\cdot s + L - (m+1).$$

Therefore

$$s + m + 1 + K \leq Ds + L.$$

Therefore

$$A \leq W_\alpha^{Ds+L}(z_1+\ldots+z_r+a).$$

Case 4. A is $\sum_{i=0}^{k} f_i(B,C_1,\ldots,C_{n-1})\cdot I_i(C_n)$. Then for $i = 1,\ldots,k$

$$D = \text{depth}(A) = \text{depth}(f(B,C_1,\ldots,C_{n-1},C_n))$$

$$\geq \text{depth}(f_i(B,C_1,\ldots,C_{n-1}))$$

and

$$L = \text{length}(A) = \text{length}(f(B,C_1,\ldots,C_{n-1},C_n))$$

$$\geq \text{length}(f_i(B,C_1,\ldots,C_{n-1})) .$$

Thus by induction hypothesis and case 3,

$$f_i(B,C_1,\ldots,C_{n-1}) \leq W_\alpha^{Ds+L}(z_1+\ldots+z_r+a)$$

for $i = 1,\ldots,k$.

But for some $i = 1,\ldots,k$,

$$A \leq f_i(B,C_1,\ldots,C_{n-1}).$$

Hence

$$A \leq W_{\alpha}^{Ds+L}(z_1+\ldots+z_r+a) .$$

This completes the proof of lemma 2.

Lemma 3: 1-fold Recursion Bounding Lemma

Let

$$f(x,0) = A^{O}$$

$$f(x,y+1) = A$$

be a nested 1-fold recursion. Suppose α and a satisfy

$$g(u_1,\ldots,u_m) \leq W_{\alpha}(u_1+\ldots+u_m+a)$$

for every function g, other than f, occurring in A or A^{O}. Then

$$f(x,y) \leq W_{\alpha+2}(x+y+K)$$

where $K = D^{O}+L^{O}+D+L+a+7$; $D^{O} = \text{depth}(A^{O})$, $L^{O} = \text{length}(A^{O})$, $D = \text{depth}(A)$, $L = \text{length}(A)$.

Proof. Define f_k $(k = 0,1,2,\ldots)$ by

$$f_k(x) = f(x,k) .$$

The idea of the proof is this. f_k is obtained from $f_0,\ldots,f_{k-1}$, (and other functions) by explicit definition. We will use lemma 2 and induction on k to prove lemma 3.

For $k = 0,1,2,\ldots$ let A_k result from A by replacing y by k and every term of form $f(B,C)$ occurring in A by

$$\sum_{i=0}^{k} f_i(B)\cdot I_i(C) \quad .$$

Then

$$f_0(x) = A^0$$

$$f_{k+1}(x) = A_k \quad .$$

Let $L_k = \text{length}(A_k)$. Define H by

$$H(0) = D^0 + L^0$$

$$H(k+1) = D\cdot H(k) + L_k \quad .$$

Note that $\text{depth}(A_k) = \text{depth}(A) = D$. Then by lemma 2 and induction on k

$$f_k(x) \leq W_\alpha^{H(k)}(x+a) \quad .$$

Now

$$H(k) = D^k(D^0+L^0) + D^{k-1}L_1 + D^{k-2}L_2+\ldots+DL_{k-1} + L_k \quad .$$

But $L_j \leq L + j \leq (L+2)^{j+1}$. Hence by the binomial theorem,

$$H(k) \leq (D^0+L^0+1)(D+L+2)^{k+1}$$

$$\leq (D^0+L^0+D+L+3)^{k+2} = (K-a-4)^{k+2} \quad .$$

Now note the following:

(i) $\quad bw + c \leq 2^{b+w+c} = W_0(b+w+c)$

(ii) $\quad b^w + c \leq 2^{bw}2^c = W_0(bw+c) \leq W_0^2(b+w+c) \quad .$

Hence

$$x + a + H(y) \leq x + a + (K-a-4)^{y+2} \leq W_0^2(x+y+K-2) .$$

Thus

$$\begin{aligned} f(x,y) &\leq W_\alpha^{H(y)}(x+a) \\ &\leq W_\alpha^{H(y)}(W_{\alpha+1}(x+a)) \\ &= W_{\alpha+1}(x+a+H(y)) \\ &\leq W_{\alpha+1}(W_0^2(x+y+K-2)) \\ &\leq W_{\alpha+1}(W_{\alpha+1}(W_{\alpha+2}(x+y+K-2))) \\ &= W_{\alpha+2}(x+y+K) \quad . \end{aligned}$$

This completes the proof of lemma 3.

Lemma 4: n-fold Recursion Bounding Lemma

Let

$$f(x,0,\ldots,0) = A^\circ$$

$$f(x,y_1,\ldots,y_n) = A$$

be a nested n-fold recursion. Suppose α has form $\omega^{n-1}\beta$ and that α and a satisfy $a > \text{maxcoeff}(\alpha)$ and

$$g(u_1,\ldots,u_m) \leq W_\alpha(u_1+\ldots+u_m+a)$$

for every function g, other than f, occurring in A° or A.

Then

$$f(x,y_1,\ldots,y_n) \leq W_{\alpha+\omega^{n-1}2}(x+y_1+\ldots+y_n+K_n)$$

where $K_n = D^o+L^o+D+L+a+6+n$; $D^o = \text{depth}(A^o)$, $L^o = \text{length}(A^o)$, $D = \text{depth}(A)$, $L = \text{length}(A)$.

Proof. The idea of the proof is this. Define f_k $(k = 0,1,2,\ldots)$ by

$$f_k(x,y_1,\ldots,y_{n-1}) = f(x,y_1,\ldots,y_{n-1},k) \quad .$$

Then f_k is obtained from $f_0,\ldots,f_{k-1}$ (and other functions) by a nested n-1 fold recursion. Thus the proof will be by induction on n.

Case 1. $n = 1$. Lemma 4 is lemma 3.

Case 2. Assume lemma 4 for n-1 $(n = 2,3,4,\ldots)$; will prove it for n.

Let A_k result from A by replacing y_n by k and every term of form $f(B,C_1,\ldots,C_{n-1},C_n)$ occurring in A by

$$\sum_{i=0}^{k} f_i(B,C_1,\ldots,C_{n-1})\cdot I_i(C_n) \quad .$$

Let A_k^o result from A_k by replacing each y_i $(i = 1,\ldots,n-1)$ by 0. Then each of the pairs of equations

$$f_0(x,0,\ldots,0) = A^0$$

$$f_0(x,y_1,\ldots,y_{n-1}) = A_0$$

and

$$f_{k+1}(x,0,\ldots,0) = A_k^0$$

$$f_{k+1}(x,y_1,\ldots,y_{n-1}) = A_{k+1} \qquad (k = 0,1,2,\ldots)$$

is a nested n-1 fold recursion.

Let $L_k = \text{length}(A_k)$ $(= \text{length}(A_k^0))$. Note that $\text{depth}(A_k) = \text{depth}(A_k^0) = \text{depth}(A) = D$. Define H by

$$H(0) = D^0 + L^0 + D + L_0 + a + 6 + (n-1) \quad ;$$

$$H(k+1) = D + L_k + D + L_{k+1} + H(k) + 6 + (n-1) \quad .$$

Claim.

$$f_k(x,y_1,\ldots,y_{n-1}) \leq W_{\alpha+\omega^{n-2}2(k+1)}(x+y_1+\ldots+y_{n-1}+H(k)).$$

We prove the claim by induction on k. For $k = 0$ this is simply the induction hypothesis (on n). Assume the claim for $i \leq k$; will prove it for $k+1$. Since $H(i) \leq H(k)$ for $i = 0,\ldots,k$, we have by induction hypothesis (on k) that

$$f_i(x,y_1,\ldots,y_{n-1}) \leq W_{\alpha+\omega^{n-2}2(k+1)}(x+y_1+\ldots+y_{n-1}+H(k)).$$

for $i = 0,\ldots,k$. Also, since $a \leq H(k)$ and α is of form $\omega^{n-1}\alpha'$

$$g(u_1,\ldots,u_m) \leq W_\alpha(u_1+\ldots+u_m+a)$$

$$W_{\alpha+\omega^{n-2}2(k+1)}(u_1+\ldots+u_m+H(k))$$

for every function g, not an f_i, occurring in A_k^o or A_{k+1}.

Hence, by induction hypothesis (on n),

$$f_{k+1}(x,y_1,\ldots,y_{n-1}) \leq W_{\alpha+\omega^{n-2}2(k+2)}(x+y_1+\ldots+y_{n-1}+H(k+1)).$$

This completes the proof of the claim

Now $L_k \leq L + k$. Hence, by the definition of $H(k+1)$,

$$H(k+1) - H(k) \leq 2(K_{n-1}+k) + 1 \quad .$$

This, together with $H(0) = K_{n-1}$, implies that

$$H(k) \leq (K_{n-1}+k)^2 \quad .$$

Note that for $c \geq 4$, $b + c^2 \leq 2^{b+c} = W_0(b+c)$. Hence,

$$x+y_1+\ldots+y_{n-1} + H(y_n) \leq W_0(x+y_1+\ldots+y_{n-1}+y_n+K_{n-1}) \quad .$$

Hence, by the claim and since $2(y_n+1) \leq W_0(x+y_1+\ldots+y_n+K_{n-1})$,

$$f(x,y_1,\ldots,y_n) \leq W_{\alpha+\omega^{n-1}}(W_0(x+y_1+\ldots+y_n+K_{n-1}))$$

$$\leq W_{\alpha+\omega^{n-1}}(W_{\alpha+\omega^{n-1}+1}(x+y_1+\ldots+y_n+K_{n-1}))$$

$$= W_{\alpha+\omega^{n-1}+1}(x+y_1+\ldots+y_n+K_{n-1}+1)$$

$$\leq W_{\alpha+\omega^{n-1}2}(x+y_1+\ldots+y_n+K_n) \quad .$$

This completes the proof of lemma 4.

Lemma 5: Bounding Lemma

If $f \in N_n$, there exist $\alpha < \omega^n$ and a natural number a such that

$$f(z_1,\ldots,z_r) \leq W_\alpha(z_1+\ldots+z_r+a) \quad .$$

Proof. By induction on the definition of f.

Case 1. f is an initial function. Then either $f(x,y) = x + y$ or $f(x) = x \dot{-} 1$. In either case take $\alpha = a = 0$.

Case 2. f is obtained by an explicit definition; i.e.,

$$f(z_1,\ldots,z_r) = A$$

where A is a term built up from previously obtained functions, constants, and the variables $z_1,\ldots,z_r$. By the induction hypothesis and the properties of the W functions, we may choose an ordinal $\beta<\omega^n$ and a constant b so that

$$g(u_1,\ldots,u_m) \leq W_\beta(u_1+\ldots+u_m+b)$$

for every function g appearing in A. By lemma 2 there is a natural number K such that

$$f(z_1,\ldots,z_r) \leq W^K_\beta(z_1+\ldots+z_r+b)$$

$$\leq W^K_\beta(W_{\beta+1}(z_1+\ldots+z_r+b))$$

$$= W_{\beta+1}(z_1+\ldots+z_r+b+K) \quad .$$

Hence take $\alpha = \beta+1$ and $a = b + K$.

Case 3. f is obtained by a nested n-fold recursion; i.e.,

$$f(x,0,\ldots,0) = g(x) \ ;$$

$$f(x,y_1,\ldots,y_n) = A \quad \text{if} \quad (y_1,\ldots,y_n) \neq (0,\ldots,0).$$

By induction hypothesis and the properties of the W functions, there is an ordinal $\beta<\omega^n$ and a constant b such that

$$h(u_1,\ldots,u_m) \leq W_\beta(u_1+\ldots+u_m+b)$$

for every function h such that either h is g or h is a function occurring in A other than f. Clearly we may also choose β and b so that β is of form $\omega^{n-1}\beta'$. Then, by lemma 4, we have a constant K such that

$$f(x,y_1,\ldots,y_n) \leq W_{\beta+\omega^{n-1}2}(x+y_1+\ldots+y_n+K) \quad .$$

Take $a = K$ and $\alpha = \beta + \omega^{n-1}2$.

This completes the proof of lemma 5.

Lemma 6: Honesty Lemma

For each $\alpha<\omega^{\omega}$ there is an elementary recursive function V_α and a constant B_α such that if $t \geq W_\alpha(x+B_\alpha)$, then

$$W_\alpha(x) = V_\alpha(x,t) .$$

Proof. The motivation of the proof is this. The function W_α is honest in the sense that its complexity is reflected by the magnitude of its value. (The characteristic function of a complicated recursive set would be dishonest, for example.) More precisely, there is a Turing machine Z_α which computes W_α such that the length of the computation of $W_\alpha(x)$ is bounded by some monotonic elementary recursive function f_α of the value $W_\alpha(x+B_\alpha)$. Hence we may define V_α so that $V_\alpha(x,t)$ = the number represented on the tape of Z_α if it is started with input x and stops within $f_\alpha(t)$ steps.

We now proceed to the gory details. Let n satisfy $\alpha < \omega^{n+1}$. Define a $n+2$-ary function W by the condition

$$W(x,y_0,y_1\ldots,y_n) = W_\beta(x)$$

if

$$\beta = \omega^n y_n+\ldots+\omega y_1+y_0 .$$

Then

(1) $W(x,0,0,\ldots,0) = 2^x$

(2) $W(0,y_0+1,y_1,\ldots,y_n) = 1$

(3) $W(x+1,y_0+1,y_1,\ldots,y_n) = W(W(x,y_0+1,y_1,\ldots,y_n),y_0,y_1,\ldots,y_n)$

(4) $W(x,0,\ldots,0,0,y_k+1,y_{k+1},\ldots,y_n) =$

$= W(x,0,\ldots,0,x,y_k,y_{k+1},\ldots,y_n)$ for $k = 1,\ldots,n$.

This, of course, is a nested n+2-fold recursion.

We now construct a Turing machine Z to compute W. For convenience, we give Z a semi-infinite tape to the left. Z has the two tape symbols / and * and other tape symbols to compute 2^x as indicated below. For each constant k, $\underline{k}$ shall denote the tape word consisting of k+1 repetitions of /; i.e., $\underline{0}$ is /, $\underline{1}$ is //, $\underline{2}$ is ///, etc. * will be used for punctuation. Then the word $\underline{x}*\underline{y_0}*\underline{y_1}*\ldots*\underline{y_n}$ represents the n+2-tuple $(x,y_0,y_1,\ldots,y_n)$. To compute $W(x,y_0,y_1,\ldots,y_n)$, give the machine the input $\underline{x}*\underline{y_0}*\underline{y_1}*\ldots*\underline{y_n}$ and start it at the left end of that word. Z performs a complete cycle as follows:

(a1) If Z starts the cycle at the left end of a tape word of form $\underline{x}*\underline{0}*\underline{0}*\ldots*\underline{0}$ (n+1 0's), it replaces that word by $\underline{2^x}$ and goes to the left end of it. Note that a machine which replaces $\underline{x}$ by $\underline{2^x}$ need use no more than 2^x+1 (i.e., the length of $\underline{2^x}$) tape squares. To do this, auxiliary tape symbols are needed.

(a2) If Z starts the cyele at the left end of a tape word of form $\underline{0}*\underline{y_0}+\underline{1}*\underline{y_1}*\ldots*\underline{y_n}$, it replaces that word by $\underline{1}$ and goes to the left end of it.

(a3) If Z starts the cycle at the left end of a tape word of form $\underline{x+1}*\underline{y_0+1}*\underline{y_1}*\ldots*\underline{y_n}$, it replaces that word by $\underline{x}*\underline{y_0+1}*\underline{y_1}*\ldots*\underline{y_n}*\underline{y_0}*\underline{y_1}*\ldots*\underline{y_n}$ and goes to the left end of it.

(a4) If Z starts the cycle at the left end of a tape word of form $\underline{x}*\underline{0}*\ldots\underline{0}*\underline{0}*\underline{y_k+1}*\underline{y_{k+1}}*\ldots*\underline{y_n}$ $(k = 1,\ldots,n)$, it replaces that word by $\underline{x}*\underline{0}*\ldots*\underline{0}*\underline{x}*\underline{y_k}*\underline{y_{k+1}}*\ldots*\underline{y_n}$ and goes to the left end of it.

The machine repeats these cycles until no more occurrences of * are on the tape, at which point it stops.

We now define the function $\overline{Z}$: $\overline{Z}(x,y_0,y_1,\ldots,y_n)$ = the number of tape squares used by Z when it commences its computation at the left end of the input word $\underline{x}*\underline{y_0}*\underline{y_1}*\ldots*\underline{y_n}$; i.e., $\overline{Z}$ is the amount of tape used in the computation of W. Then

(b1) $\overline{Z}(x,0,0,\ldots,0) = \max\{2^x+1,\ x+1+2(n+1)\}$.

(2^x+1 is the length of $\underline{2}^{\underline{x}}$; $x+1+2(n+1)$ is the length of $\underline{x}*\underline{0}*\underline{0}*\ldots*\underline{0}$.)

(b2) $\overline{Z}(0,y_0+1,y_1,\ldots,y_n) = (y_0+1)+y_1+\ldots+y_n+2n+3$

(i.e., the length of $\underline{0}*\underline{y_0+1}*\underline{y_1}*\ldots*\underline{y_n}$).

(b3) $\overline{Z}(x+1,y_0+1,y_1,\ldots,y_n) = \max\{\overline{Z}(x,y_0+1,y_1,\ldots,y_n) +$

$+y_0+y_1+\ldots+y_n+2n+2,\ \overline{Z}(W(x,y_0+1,y_1,\ldots,y_n),y_0,y_1,\ldots,y_n)\}$.

(b4) $\overline{Z}(x,0,\ldots,0,0,y_k+1,y_{k+1},\ldots,y_n) = \overline{Z}(x,0,\ldots,0,x,y_k,y_{k+1},\ldots,y_n)$

$(k = 1,\ldots,n)$.

Now for each $\beta < \omega^{n+1}$ define a Turing machine Z_β to compute W_β as follows: write $\beta = \omega^n b_n + \ldots + \omega b_1 + b_0$. Then Z_β takes an input $\underline{x}$, replaces it by $\underline{x}*\underline{b}_0*\underline{b}_1*\ldots*\underline{b}_n$, and then acts like Z. Let $\overline{Z}_\beta(x)$ be the amount of tape used by Z_β to compute $W_\beta(x)$; i.e.,

$$\overline{Z}_\beta(x) = \overline{Z}(x,b_0,b_1,\ldots,b_n) .$$

Define B_β by

$$B_\beta = b_0+b_1+\ldots+b_n+2n+3 .$$

Then

(c1) $\overline{Z}_0(x) = \max\{2^x+1, x+B_0\}$;

(c2) $\overline{Z}_{\beta+1}(0) = B_{\beta+1}$;

(c3) $\overline{Z}_{\beta+1}(x+1) = \max\{\overline{Z}_{\beta+1}(x)+B_\beta-1,\ \overline{Z}_\beta(W_{\beta+1}(x))\}$;

(c4) $\overline{Z}_\beta(x) = \overline{Z}_{\beta[x]}(x)$ for β a limit ordinal.

Also

$$B_{\beta+1} = B_\beta + 1$$

$$B_{\beta[x]} = B_\beta - 1 + x \qquad \text{for } \beta \text{ a limit ordinal.}$$

Now we prove some inequalities.

(d1) $2^x+1 \leq W_0(x+B_0)$; $x + B_0 \leq W_0(x+B_0)$.

(d2) $B_{\beta+1} \leq W_{\beta+1}(B_{\beta+1})$.

(d3) (i) $W_{\beta+1}(x+B_{\beta+1})+B_\beta-1 \leq W_0(W_{\beta+1}(x+B_{\beta+1}))$

$\leq W_{\beta+1}(x+1+B_{\beta+1})$;

(ii) $W_\beta(W_{\beta+1}(x)+B_\beta) \leq W_\beta(W_{\beta+1}(x+B_{\beta+1}))$

$= W_{\beta+1}(x+1+B_{\beta+1})$.

(d4) $W_{\beta[x]}(x+B_{\beta[x]}) = W_{\beta[x]}(x+B_\beta-1+x)$

$\leq W_{\beta[x]}(W_0(x+B_\beta-1))$

$\leq W_{\beta[x]+1}(x+B_\beta)$

$\leq W_{\beta[x+B_\beta]}(x+B_\beta)$

$= W_\beta(x+B_\beta)$.

Claim. $\overline{Z}_\beta(x) \leq W_\beta(x+B_\beta)$.

Prove the claim by induction on β.

Case 1. $\beta = 0$. Claim follows from (c1) and (d1).

Case 2. Assume claim for β; will prove it for $\beta+1$ by induction on x. Claim for $x = 0$, follows from (c2) and (d2). Claim for $x+1$ follows from claim for x, (c3), and (d3).

Case 3. β a limit ordinal. Claim for β follows from claim for $\beta[x]$, (c4), and (d4).

This completes the proof of the claim.

Now we construct V_α and f_α. Define the function Z^*_α as follows: $Z^*_\alpha(x)$ = the number of operations performed by the machine Z_α (until it stops) if it is started at the left end of the input word $\underline{x}$. Then if p is the number of tape symbols and q is the number of states in Z_α,

$$Z^*_\alpha(x) \leq q \cdot \overline{Z} \cdot p^{\overline{Z}} \qquad \text{where } \overline{Z} = \overline{Z}_\alpha(x) \ .$$

Indeed, the expression on the right in the inequality is simply the number of instantaneous descriptions of Z_α with tape length $\overline{Z}_\alpha(x)$; if one of these were repeated in the course of the computation, the machine would go into a loop and never stop.

Now define f_α by $f_\alpha(t) = q \cdot t \cdot p^t$. Then f_α is elementary recursive and

$$Z^*_\alpha(x) \leq f_\alpha(t) \qquad \text{for } t \geq W_\alpha(x+B_\alpha) \ .$$

Now define U_α as follows: $U_\alpha(x,r)$ = the number (if any) represented on the tape of Z_α after r operations, the machine starting at the left end of the input word $\underline{x}$. Then $U_\alpha(x,r) = W_\alpha(x)$, if $r \geq Z^*(x)$. U_α is elementary recursive and we may define V_α by

$$V_\alpha(x,t) = U_\alpha(x, f_\alpha(t)) \ .$$

This completes the proof of lemma 7.

Lemma 8. $W_{\alpha+1} \notin E_\alpha$; hence, $E_\alpha \neq E_\beta$ for $\alpha < \beta$.

Proof. Godel number the functions of E_α. We will construct an elementary recursive function K such that if $\underline{f}$ is a godel number of $f \in E_\alpha$,

$$f(u_1,\ldots,u_m) \leq W_\alpha^{K(\underline{f})}(u_1+\ldots+u_m)$$

$$\leq W_{\alpha+1}(u_1+\ldots+u_m+K(\underline{f})) \quad .$$

K is constructed by induction on the definition of f and lemma 2.

Case 1. f is an initial function of E_α ; i.e., $f(x,y) = x + y$ or $f(x) = W_\alpha(x)$. In the former case $K(\underline{f}) = 0$; in the latter case $K(\underline{f}) = 1$.

Case 2. f is obtained by a composition:

$$f(u_1,\ldots,u_m) = A.$$

Let s be the maximum of $K(\underline{g})$ where g ranges over the functions appearing in A. Let D = depth(A); L = length(A). Then by lemma 2, may take $k(\underline{f}) = Ds + L$.

Case 3. f is obtained by limited recursion:

$$f(x,0) = g(x)$$

$$f(x,y+1) = h(x,y,f(x,y))$$

$$f(x,y) \leq j(x,y) \quad .$$

Take $K(\underline{f}) = K(\underline{j})$.

Suppose $W_{\alpha+1} \in E_\alpha$. Then the function $F \in E_\alpha$ where

$$F(x) = W_{\alpha+1}(x+K(x)) + 1 \quad .$$

Let $\underline{F}$ be a godel number of F. Then

$$F(x) < W_{\alpha+1}(x+K(\underline{F})) + 1 \quad .$$

Hence,

$$F(\underline{F}) < F(\underline{F}) \quad ,$$

a contradiction. Since $W_{\alpha+1} \in E_\beta$ for $\alpha < \beta$ (by lemma 7), we have $E_\alpha \neq E_\beta$. This completes the proof of lemma 8.

Lemma 9. $W_\alpha \in N_n$ for $\alpha < \omega^n$; hence, $E_\alpha \subseteq N_n$ for $\alpha < \omega^n$.

Proof. For $k = 0,1,2,\ldots$ define a function $\underline{W}_k$ by the condition

$$\underline{W}_k(x,y_0,y_1,\ldots,y_{n-2}) = W_\beta(x)$$

where $\beta = \omega^{n-1}k + \omega^{n-2}y_{n-2}+\ldots+ \omega y_1 + y_0$.

Then each $\underline{W}_k \in N_n$; indeed, for $n = 2,3,3,\ldots$

$$\underline{W}_0(x,0,0,\ldots,0) = 2^x$$

$$\underline{W}_0(0,y_0+1,y_1,\ldots,y_{n-2}) = 1$$

$$\underline{W}_0(x+1,y_0+1,y_1,\ldots,y_{n-2}) =$$

$$= \underline{W}_0(\underline{W}_0(x,y_0+1,y_1,\ldots,y_{n-2}),y_1,\ldots,y_{n-2})$$

$$\underline{W}_0(x,0,\ldots,0,0,y_m+1,y_m,\ldots,y_{n-2}) =$$

$$= \underline{W}_0(x,0,\ldots,0,x,y_m,y_{m+1},\ldots,y_{n-2}) \qquad \text{for } m = 1,\ldots,n-2$$

and

$$\underline{W}_{k+1}(x,0,\ldots,0,0) = \underline{W}_k(x,0,\ldots,0,x)$$

$$\underline{W}_{k+1}(0,y_0+1,y_1,\ldots,y_{n-2}) = 1$$

$$\underline{W}_{k+1}(x+1,y_0+1,y_1,\ldots,y_{n-2}) =$$

$$= \underline{W}_{k+1}(\underline{W}_{k+1}(x,y_0+1,y_1,\ldots,y_{n-2}),y_0,y_1,\ldots,y_{n-2})$$

$$\underline{W}_{k+1}(x,0,\ldots,0,0,y_m+1,y_{m+1},\ldots,y_{n-2}) =$$

$$= \underline{W}_{k+1}(x,0,\ldots,0,x,y_m y_{m+1},\ldots y_{n-2}) \qquad \text{for } m = 1,\ldots,n-2$$

while for $n = 1$

$$\underline{W}_0(x) = 2^x$$

$$\underline{W}_{k+1}(0) = 1$$

$$\underline{W}_{k+1}(x+1) = \underline{W}_k(\underline{W}_{k+1}(x)) \quad .$$

Thus $\underline{W}_0$ is defined by a nested n-fold recursion; $\underline{W}_{k+1}$ is defined from $\underline{W}_k$ by a nested n-fold recursion. Hence, each $\underline{W}_k \in N_n$. Let

$$\alpha = \omega^{n-1}k + \omega^{n-2}a_{n-2} + \ldots + \omega a_1 + a_0 \quad .$$

Then $W_\alpha(x) = \underline{W}_k(x,a_0,a_1,\ldots,a_{n-2})$. Hence, $W_\alpha \in N_n$. This completes the proof of lemma 9.

Lemma 10. If $f \in N_n$, then $f \in E_\alpha$ for some $\alpha > \omega^n$.

Proof. Since $N_n = A(Q_{n-1})$ by a theorem of the previous section, we prove lemma 10 by induction on the definition of f as a function of $A(Q_{n-1})$.

Case 1. f is an initial function; i.e., $f(x,y) = x + y$. Then $f \in E_0$.

Case 2. f is obtained by an explicit definition; i.e.,

$$f(u_1,\ldots,u_m) = A.$$

By induction hypothesis and lemma 7 there is an $\alpha < \omega^n$ such that $g \in E_\alpha$ for every g occurring in A. Thus $f \in E_\alpha$.

Case 3. f is obtained from g and h by primitive recursion; i.e.,

$$f(x,0) = g(x)$$
$$f(x,y+1) = h(x,y,f(x,y)) \quad .$$

By induction hypothesis and lemma 7, $g, h \in E_\alpha$ for some $\alpha < \omega^n$. By the proof of lemma 8, we may select a constant K so that

$$g(x) \leq W_{\alpha+1}(x+K)$$

$$h(x,y,z) = W_{\alpha+1}(x+y+z+K) .$$

But the schema defining f is a nested 1-fold recursion. Hence by lemma 3, there is a constant b such that

$$f(x,y) \leq W_{\alpha+3}(x+y+b) .$$

Then, since $g,h \in E_{\alpha+3}$ by lemma 7 and f is obtained from functions in $E_{\alpha+3}$ by limited recursion, $f \in E_{\alpha+3}$.

Case 4. $f = m$ where m is obtained from t by Q_{n-1}- annihilation; i.e.,

$$m(y) = \mu k[t^k(y) = 0]$$

where $t(y)\ Q_{n-1}\ y$ for $y \neq 0$ and $t(0) = 0$. Here $t^k(y)$ is the k^{th} iteration of $t(y)$; i.e., $t^0(y) = y$ and $t^{k+1}(y) = t(t^k(y))$. Also μ is the (unbounded) least number operator. By induction hypothesis, $t \in E_\alpha$, for some $\alpha < \omega^n$. Hence, by lemma 8, $t \in N_n = A(Q_{n-1})$. Hence, $m \in A(Q_{n-1}) = N_n$. Hence, by the bounding lemma,

$$m(y) \leq W_\beta(y+b)$$

where $\beta < \omega^n$. We may also suppose that $\alpha+3 \leq \beta$. Define

$$\underline{t}(k,y) = t^k(y) .$$

By case 3 and lemma 7, $\underline{t} \in E_\beta$. But

$$m(y) = \mu k \leq W_\beta(y+b)[\underline{t}(k,y) = 0] .$$

Since the bounded least number operator is an elementary recursive operator, $m \in E_\beta$. This completes the proof of lemma 10.

This also completes the proof of the extended Grzegorcsyk theorem: (1) is lemma 7; (2) is lemma 8; (3) follows from lemmas 9 and 10.

Remark. The special case of the extended Grzegorczyk theorem obtained by setting $n = 1$ is essentially theorem 4.13 in [4]. Hence, the title.

APPLICATIONS TO COMPUTATIONAL COMPLEXITY

A rather natural measure of the complexity of a recursive function is the length of the computation of that function. This length of computation is again a function; i.e. if $f = f(x)$ is a recursive function, we may define $L_f = L_f(x)$, the length of the computation of $f(x)$. L_f depends on the method used for calculating x (i.e., Turing machines, equations), the particular definition of f, the definition of the length of computation, and also the notation used for natural numbers. For example, take the Godel Kleene Herbrand definition of the recursive functions by equations (see Davis [2]). The notations for the natural numbers 0, 1, 2,... are 0, s(0), s(s(0)),... (abbreviated $\underline{0}$, $\underline{1}$, $\underline{2}$,...) respectively. For each system of equations E with principal function letter F defining a function f (i.e., $F(\underline{x}) = \underline{y}$ is provable from E if and only if $f(x) = y$), we may define $L_E(x)$ to be the smallest number of lines in any proof of $F(\underline{x}) = \underline{y}$. I believe that the results of this section would go through using this notion of the length of computation; however, I prefer the more straightforward approach via Turing machines.

For Turing machines there are two concepts of the length of a computation: the amount of tape used and the number of machine operations performed. Fix some notation for natural numbers (e.g., unary, binary, decimal, dyadic, etc.); for each natural number x,

let the word $\underline{x}$ (a word with symbols from a finite alphabet A) correspond to x. Then the n-tuple $(x_1,\ldots,x_n)$ of natural numbers can be represented by the word $\underline{x}_1*\ldots*\underline{x}_n$. (The notation * is here used for punctuation; it should not be confused with the notation Z* introduced below.) Then for each Turing machine Z (whose language contains the alphabet A), we may define the n-ary functions Z* and $\overline{Z}$ as follows: $Z^*(x_1,\ldots,x_n)$ = the number of operations performed by Z when it is started with input $\underline{x}_1*\ldots*\underline{x}_n$ until Z halts. $\overline{Z}(x_1,\ldots,x_n)$ = the number of tape squares used by Z when it is started with input $\underline{x}_1*\ldots*\underline{x}_n$; i.e., $\overline{Z}(x_1,\ldots,x_n)$ is the length of the longest tape word appearing in an instantaneous description of the computation by Z beginning with the instantaneous description consisting of the initial state of Z, the tape word $\underline{x}_1*\ldots*\underline{x}_n$, and the initial position of Z at the left end of that word.

For simplicity we assume that all our Turing machines have a semi-infinite tape to the right and that they start and end their computations at the left end of that tape.

In order to prove the theorems below we must place the following restrictions on the notation system:

(NS1) There are machines Z_+ and Z_p computing addition and predecessor respectively such that $\overline{Z}_+$ and $\overline{Z}_p$ are elemtary recursive.

(NS2) length $(\underline{x})$, the length of the word $\underline{x}$, is an elementary recursive function of x; hence, length$(\underline{x}_1*\ldots*\underline{x}_n)$ is an elementary recursive function of $x_1,\ldots,x_n$.

(NS3) length$(\underline{x+1})$ is an elementary recursive function of x.

(NS4) There is a godel numbering of Turing machines such that the function $U(e,x,t)$ defined in lemma 5 below is elementary recursive.

A sufficient condition for conditions (NS1)-(NS4) is that there be Turing machines Z_1 and Z_2 such that Z_1 replaces the unary notation for x by $\underline{x}$ and Z_2 replaces $\underline{x}$ by the unary notation for x and the amounts of tape used by Z_1 and Z_2 respectively are elementary recursive functions of x. Thus the usual notation systems (unary, binary, decimal, dyadic) all satisfy these conditions. The reader who wishes to ignore these subtleties may think of $\underline{x}$ as the unary notation for x. (The unary notation for x consists of $x+1$ vertical strokes.)

Now define for each singulary function T the set $\overline{S}(T)$ of functions $f = f(x_1,\ldots,x_n)$ such that there is a Turing machine Z_f which computes f and satisfies $\overline{Z}_f(x_1,\ldots,x_n) \leq T(x_1+\ldots+x_n)$ for all but finitely many n-tuples $(x_1,\ldots,x_n)$. Define $S^*(T)$ analogously. Recall that N_n is the set of nested n-fold recursive functions; recall the definition of W_α. The following theorem is a corollary of the Extended Gzregorczyk Theorem of the last section.

Theorem I. $N_n = \bigcup \bar{S}(W_\alpha) = \bigcup S^*(W_\alpha)$ $(n = 1,2,\ldots;$ in both unions α ranges over the ordinals $<\omega^n$.)

The proof consists of five lemmas. Recall that $U(R)$ is the set of unnested R-recursive functions.

Lemma 1. For every $f \in U(R)$ there is a Turing machine Z_f which computes f such that $\bar{Z}_f \in U(R)$.

Proof of lemma 1. By induction of the definition of f.

Case 1. f is addition. By (NS1).

Case 2. f is obtained from g by an explicit transformation; i.e.,

$$f(x_1,\ldots,x_n) = g(\xi_1,\ldots,\xi_m) .$$

By the induction hypothesis, there is a machine Z_g which computes g and satisfies $\bar{Z}_g \in U(R)$. Construct Z_f as follows: Z_f takes an input $\underline{x}_1*\ldots*\underline{x}_n$, replaces it with $\underline{\xi}_1*\ldots*\underline{\xi}_m$, and behaves like Z_g. Then Z_f computes f and

$$\bar{Z}_f(x_1,\ldots,x_n) = \max\{\text{length}(\underline{x}_1*\ldots*\underline{x}_n),\ \bar{Z}_g(\xi_1,\ldots,\xi_m)\} .$$

Hence by (NS2) $\bar{Z}_f \in U(R)$.

Case 3. f is obtained from g and h by composition; i.e.,

$$f(x_1,\ldots,y_m) = h(x_1,\ldots,x_n,g(y_1,\ldots,y_m)).$$

By induction hypothesis there are appropriate machines Z_h and Z_g computing h and g respectively. Construct Z_f as follows: Z_f takes the input $\underline{x}_1*\ldots*\underline{x}_n*\underline{y}_1*\ldots*\underline{y}_m$, moves to the left end of the tape and acts like Z_h. Then Z_f computes f and

$$\overline{Z}_f(x_1,\ldots,y_m) = \max\{\text{length}(\underline{x}_1*\ldots*\underline{x}_n) + \overline{Z}_g(y_1,\ldots,y_m), \overline{Z}_h(x_1,\ldots,x_n,g(y_1,\ldots,y_m))\} .$$

Hence $\overline{Z}_f \in U(R)$.

Case 4. f is obtained from g and h by primitive recursion; i.e.,

$$f(x,0) = g(x)$$

$$f(x,y+1) = h(x,y,f(x,y)) .$$

By induction hypothesis there are appropriate machines Z_g and Z_h computing g and h respectively. Construct Z_f as follows: Z_f takes an input $\underline{x}*\underline{y}$ and replaces it by $\underline{x}*\underline{y}*\underline{x}*\underline{y-1}*\underline{x}*\underline{y-2}*\ldots*\underline{x}*\underline{1}*\underline{x}$. It then goes to the rightmost $\underline{x}$ and acts like Z_g. Then it goes to the next $\underline{x}$ from the right and acts like Z_h. This last step is repeated until all *'s have been removed from the tape. Then the machine has computed f(x,y). Recall that Z_p computes the predecessor. Then

$$\overline{Z}_f(x,0) = \max \text{length}(\underline{x}*\underline{0}), \overline{Z}_g(x)$$

$$\overline{Z}_f(x,y+1) = \max\{\text{length}(\underline{x}*\underline{y+1}*\underline{x}*) + \overline{Z}_p(y+1), \text{length}(\underline{x}*\underline{y+1}*) + \overline{Z}_f(x,y), \overline{Z}_h(x,y,f(x,y))\} .$$

Hence $\overline{Z}_f \in U(R)$.

Case 5. f is obtained from g, h, and t by unnested R-recursion; i.e.,

$$f(x,0) = g(x)$$

$$f(x,y) = h(x,y,f(x,t(x,y))) \quad \text{for } y \neq 0;$$

$$t(x,0) = 0, \quad t(x,y) \; R \; y \quad \text{for } y \neq 0.$$

Let $t^n(x,y)$ be the n^{th} iteration of $t(x,y)$; i.e., $t^0(x,y) = y$ and $t^{n+1}(x,y) = t(x,t^n(x,y))$. By induction hypothesis there are appropriate machines Z_g, Z_h, and Z_t computing g, h, and t respectively. Z_f operates as follows: it takes an input $\underline{x}*\underline{y}$ and replaces it by $\underline{x}*\underline{t^0(x,y)}*\underline{x}*\underline{t^1(x,y)}*\underline{x}*\underline{t^2(x,y)}*\ldots*\underline{x}*\underline{t^{m-1}(x,y)}*\underline{x}*\underline{0}$ (here m is the least n such that $t^n(x,y) = 0$). Then it erases the last $\underline{0}$ and replaces the rightmost $\underline{x}$ by $\underline{g(x)}$. Then it goes to the next $\underline{x}$ from the right and acts like Z_h. This last step is repeated. Z_f computes f and

$$\overline{Z}_f(x,0) = \max\{\text{length}(\underline{x}*\underline{0}), \overline{Z}_g(x)\}$$

$$\overline{Z}_f(x,y) = \max\{\text{length}(\underline{x}*\underline{y}*\underline{x}*) + \overline{Z}_t(x,y), \text{length}(\underline{x}*\underline{y}*) + \overline{Z}_f(x,t(x,y)), \overline{Z}_h(x,y,f(x,t(x,y)))\} \quad \text{for } y \neq 0.$$

Hence $\overline{Z}_f \in U(R)$.

This completes the proof of lemma 1.

Lemma 2. For $f \in N_n$ there is $\alpha \leq \omega^n$ such that $f(x_1,\ldots,x_r) \leq W_\alpha(x_1+\ldots+x_r)$ for all but finitely many r-tuples $(x_1,\ldots,x_r)$.

Proof of lemma 2. By the Extended Gzregordzyk theorem $f \in E_\beta$ for some $\beta < \omega^n$. By the proof of lemma 8 in the previous section there is a K such that

$$f(x_1,\ldots,x_r) \leq W_{\beta+1}(x_1+\ldots+x_r+K)$$

for all $x_1,\ldots,x_r$. But for x sufficiently large

$$x+K \leq 2^{x-1} = W_0(x-1) \leq W_{\beta+2}(x-1) .$$

Hence

$$W_{\beta+1}(x+K) \leq W_{\beta+1}(W_{\beta+2}(x-1)) = W_{\beta+2}(x)$$

for x large enough. Hence we may take $\alpha = \beta+2$. This completes the proof of lemma 2.

Lemma 3. If $f \in N_n$, then $f \in \overline{S}(W_\alpha)$ for some $\alpha < \omega^n$.

Proof of lemma 3. Recall that $N_n = U(Q_{n-1})$. Hence by lemma 1 there is a machine Z_f which computes f such that $\overline{Z}_f \in N_n$. Then by lemma 2 $f \in \overline{S}(W_\alpha)$ for some $\alpha < \omega^n$. This completes the proof of lemma 3.

Lemma 4. $\overline{S}(W_\alpha) \subseteq S^*(W_{\alpha+1})$.

Proof of lemma 4. Choose $f \in \overline{S}(W_\alpha)$. Let Z_f compute f and satisfy

$$\overline{Z}_f(x_1,\ldots,x_r) \leq W_\alpha(x_1+\ldots+x_r)$$

for all but finitely many r-tuples $(x_1,\ldots,x_r)$. Let p be the number of tape symbols in Z_f and let q be the number of states. Then qwp^w is the number of instantaneous descriptions of Z_f with tape length w. Thus for all but finitely many r-tuples $(x_1,\ldots,x_r)$,

$$Z_f^*(x_1,\ldots,x_r) \leq qWp^W \quad \text{where} \quad W = W_\alpha(x_1+\ldots+x_r) \ ;$$

for if an instantaneous description were repeated in the course of the computation, Z_f would go into a loop and never stop.

For w large enough,

$$qwp^w \leq W_0^2(w) \leq W_\alpha^2(w) \ .$$

For x large enough

$$x \leq W_0(x-3) \leq W_{\alpha+1}(x-3) \ .$$

Hence for x large enough

$$W_\alpha^2(W_\alpha(x)) = W_\alpha^3(x) \leq W_\alpha^3(W_{\alpha+1}(x-3)) = W_{\alpha+1}(x) \ .$$

Hence for $x_1+\ldots+x_r$ large enough

$$\overline{Z}_f(x_1,\ldots,x_r) \leq W_{\alpha+1}(x_1+\ldots+x_r) \ .$$

Hence $f \in S^*(W_{\alpha+1})$. This completes the proof of lemma 4.

Lemma 5. $S^*(W_\alpha) \subseteq E_\alpha$.

Proof of lemma 5. Godel number the Turing machines and define the elementary recursive function U by setting $U(e,x,t)$ = the number represented on the tape of machine with godel number e when started with input x if it has halted within t or less operations; $U(e,x,t) = 0$ otherwise. This can be done because of (NS4). Then for $f \in S^*(W_\alpha)$ there exist e and k such that

$$f(x) = U(e,x,W_\alpha(x)+k)) \quad .$$

The case where f has more than one argument is no different. This completes the proof of lemma 5.

Lemmas 3, 4, and 5 together with the extended Gzregorczyk theorem imply theorem I.

Corollary 1. As α ranges over the ordinals $<\omega^n$ and e and k range over the natural numbers, $U(e,x,W_\alpha(x)+k)$, as a function of x, ranges over all the singulary functions of N_n.

Corollary 2. N_n is the smallest class containing E and the functions W_α for $\alpha < \omega^n$ and closed under the elementary recursive operations.

Corollary 3. N_{n+1} contains a function which enumerates N_n. (To prove corollary 3, note that the function W defined by $W(x,y_0,\ldots,y_{n-1}) = W_\alpha(x)$ where $\alpha = \omega^{n-1}y_{n-1}+\ldots+y_0$ is in N_{n+1}.)

Theorem I suggests other methods for constructing hierarchies. For example, define functions V_n by

$$V_0(x) = x$$

$$V_{n+1}(x) = 2^V \quad \text{where} \quad V = V_n(x).$$

Theorem II. $E = \bigcup_n \overline{S}(V_n) = \bigcup_n S^*(V_n)$.

The proof consists of the following five lemmas.

Lemma 6. For every $f \in E$ there is a Turing machine Z_f which computes f and such that $\overline{Z}_f \in E$.

Proof of lemma 6. By induction on the definition of f. The only place where the proof differs from the corresponding part of the proof of lemma 1 is the case where f is obtained from g, h, and j by limited recursion; i.e.,

$$f(x,0) = g(x)$$

$$f(x,y+1) = h(x,y,f(x,y))$$

$$f(x,y) \leq j(x,y) .$$

Construct Z_f from Z_g, Z_h, and Z_p as in lemma 1. Then

$$\overline{Z}_f(x,0) = \max\{\text{length}(\underline{x}*\underline{0}),\ \overline{Z}_g(x)\}$$

$$\overline{Z}_f(x,y+1) = \max\{\text{length}(\underline{x}*\underline{y+1}*\underline{x}*) + \overline{Z}_p(y+1),\ \text{length}(\underline{x}*\underline{y+1}*) + \overline{Z}_f(x,y),\ \overline{Z}_h(x,y,f(x,y))\}\ .$$

Recall that E is closed under bounded summation. Define

$$L(x,0) = \text{length}(\underline{x}*\underline{0}) + \overline{Z}_g(x)$$

$$L(x,y+1) = \text{length}(\underline{x}*\underline{y+1}*\underline{x}*) + \overline{Z}_p(y+1) + \sum_{i=0}^{i=j(x,y)} \overline{Z}_h(x,y,i)\ .$$

Then

$$\overline{Z}_f(x,0) \leq L(x,0)$$

$$\overline{Z}_f(x,y+1) \leq L(x,y+1) + \overline{Z}_f(x,y)\ .$$

Hence by induction on y

$$\overline{Z}_f(x,y) \leq \sum_{i=0}^{i=y} L(x,i)\ .$$

Hence $\overline{Z}_f$ is defined by a limited recursion. Hence $\overline{Z}_f \in E$. This completes the proof of lemma 6.

Lemma 7. If $f \in E$, then there is a k such that

$$f(x_1,\ldots,x_n) \leq V_k(x_1+\ldots+x_n)$$

for all but finitely many n-tuples $(x_1,\ldots,x_n)$.

Proof of lemma 7. By induction on the definition of f.

Case 1. f is addition. Take $k = 0$.

Case 2. f is exponentiation. Take $k = 2$.

Case 3. f is obtained from g by an explicit transformation; i.e.,

$$f(x_1,\ldots,x_n) = g(\xi_1,\ldots,\xi_m) \quad .$$

By induction hypothesis

$$g(y_1,\ldots,y_m) \leq W_i(y_1+\ldots+y_m)$$

for all but finitely many m-tuples $(y_1,\ldots,y_m)$. For $x_1+\ldots+x_n$ large enough

$$\xi_1+\ldots+\xi_m \leq V_1(x_1+\ldots+x_n) \quad .$$

Since the former inequality fails at most finitely often, we may take $k = i+1$.

Case 4. f is obtained from g and h by composition; i.e.,

$$f(x_1,\ldots,y_m) = h(x_1,\ldots,x_n, g(y_1,\ldots,y_m)) \quad .$$

Then by induction hypothesis (with finitely many exceptions)

$$h(x_1,\ldots,x_n,z) \leq V_r(x_1+\ldots+x_n+z)$$

and

$$g(y_1,\ldots,y_m) \leq V_s(y_1+\ldots+y_m) \quad .$$

Since $V_r(x+V_s(y)) \leq V_r(V_s(x+y)) = V_{r+s}(x+y)$, we may take $k = r+s$.

Case 5. f is obtained from g, h, and j by limited recursion. Then $f(x,y) \leq j(x,y)$. Hence we may find k by the induction hypothesis on j.

This completes the proof of lemma 7.

Lemma 8. If $f \in E$, then $f \in \overline{S}(V_n)$ for some n.

Lemma 8 follows immediately from lemmas 6 and 7.

Lemma 9. $\overline{S}(V_n) \subseteq S^*(V_{n+2})$.

The proof of lemma 9 is similar to the proof of elmma 4; note that $qvp^v \leq V_2(v)$ for v sufficiently large.

Lemma 10. $S^*(V_n) \subseteq E$.

The proof of lemma 10 is similar to the proof of lemma 5; note that $V_n \in E$.

Theorem II now follows from lemmas 8, 9, and 10.

THE KLEENE SUBRECURSIVE HIERARCHY

In this section we will characterize N_n in terms of the subrecursive hierarchy of Kleene [7]. The idea of the proof is not unlike the idea of the proof of the extended Grzegorczyk theorem; the reader should also compare it with a similar theorem in Axt [1].

As a first step we simplify our definition of N_n. If $f(x) = g^x(1)$; i.e.,

$$f(0) = 1$$
$$f(x+1) = g(f(x)) \ ,$$

then we say that f is obtained from g by 1-fold iteration. Suppose the operation of n-fold iteration is defined for some $n \geq 1$. Let $f_0 = f_0(x,y_0,y_1,\ldots,y_{n-2})$ be obtained from $g \in g(x)$ by n-fold iteration. Suppose $f_k = f_k(x,y_0,y_1,\ldots,y_{n-3},y_{n-2})$ is defined; let $f_k^0 = f_k(x,0,0,\ldots,0,x)$ and let $f_{k+1} = f_{k+1}(x,y_0,y_1,\ldots,y_{n-2})$ be obtained from f_k^0 by n-fold iteration. Then we say that the function f defined by

$$f(x,y_o,y_1,\ldots,y_{n-2},y) = f_y(x,y_0,y_1,\ldots,y_{n-2})$$

is obtained from g by n+1-fold iteration.

For $n \geq 2$ the equations defining f from g by an n-fold iteration are

$$f(x,0,0,\ldots,0) = g(x)$$

$$f(0,y_0+1,y_1,\ldots,y_{n-2}) = 1$$

$$f(x+1,y_0+1,y_1,\ldots,y_{n-2}) = f(f(x,y_0+1,y_1,\ldots,y_{n-2}),y_0,y_1,\ldots,y_{n-2})$$

$$f(x,0,\ldots,0,0,y_j+1,y_{j+1},\ldots,y_{n-2}) =$$

$$= f(x,0,\ldots,0,x,y_j,y_{j+1},\ldots,y_{n-2}) \quad .$$

Define $\underline{W}_k$ by

$$\underline{W}_k(x,y_0,y_1,\ldots,y_{n-2}) = W_\alpha(x)$$

where $\alpha = \omega^{n-1}k + \omega^{n-2}y_{n-2} + \ldots + \omega y_1 + y_0$.

(See lemma 9 of the section on the extended Grzegorczyk theorem.) Then W_0 is obtained from 2^x by n-fold iteration and $\underline{W}_{k+1}$ is obtained from $\underline{W}_k$ by an explicit transformation and n-fold iteration. Hence, n-fold iteration is powerful enough to obtain all the functions W_α for $\alpha < \omega^n$. Hence, by a corollary to theorem I of the last section,

Theorem. N_n is the smallest class containing addition and closed under the elementary recursive operations and n-fold iteration.

In order to define the Kleene subrecursive hierarchy, it is necessary to give an indexing (i.e., a godel numbering) to the functions of $P(V)$ where $P(V)$ is the set of functions primitive recursive in the binary function V. Rather than give a particular indexing, I will list the properties which the indexing must have

in order that the theorem below go through. I will state these properties as I need them.

Now for each $\alpha < \omega^{\omega}$ define the binary function V_{α} as follows:

(1) $V_0(x,y) = x + y$.

(2) $V_{\alpha+1}(\langle x_1,\ldots,x_r\rangle, \underline{f}) = f(x_1,\ldots,x_r)$ if $\underline{f}$ is an index of the r-ary function f of $P(V_{\alpha})$; $V_{\alpha+1}(x,\underline{f}) = 0$ if $\underline{f}$ is not an index of any function in $P(V_{\alpha})$.

(3) $V_{\alpha}(x, \langle n,\underline{f}\rangle) = V_{\alpha[n]}(x,\underline{f})$ for α a limit ordinal.

Hence, $V_{\alpha+1}$ is a universal function for $P(V_{\alpha})$ and V_{α} is a universal function for $\bigcup_n P(V_{\alpha[n]})$ for α a limit ordinal.

Define $P_{\alpha} = P(V_{\alpha})$, the set of functions primitive recursive in V_{α}.

Characterization of N_n via the Kleene Subrecursive Hierarchy:

(1) If $\alpha \leq \beta$, then $P_{\alpha} \subseteq P_{\beta}$.

(2) If $\alpha < \beta$, then $P_{\alpha} \neq P_{\beta}$.

(3) $N_n = \bigcup_{\alpha} P_{\alpha}$, $\alpha < \omega^{n-1}$.

Proof. We prove (1) by induction on β. The case $\beta = \alpha$ is trivial. Assume $\alpha < \beta + 1$ and that (1) holds for β. Must show that $P_\alpha \in P_{\beta+1}$. $P_\alpha \subseteq P_\beta$ by induction hypothesis. $V_\beta \in P_{\beta+1}$ is obvious. Hence $P_\beta \subseteq P_{\beta+1}$. Hence $P_\alpha \subseteq P_{\beta+1}$. Hence (1) holds for $\beta+1$. Finally suppose that $\alpha < \beta$, β is a limit ordinal, and that (1) holds for all $\beta' < \beta$. Then choose n such that $\alpha \leq \beta[n] < \beta$. Clearly, $V_{\beta[n]} \in P(V_\beta)$. Hence $P_{\beta[n]} \subseteq P_\beta$. By induction hypothesis, $P_\alpha \subseteq P_{\beta[n]}$. Hence $P_\alpha \subseteq P_\beta$. This completes the proof of (1).

To prove (2) note that $V_{\alpha+1} \in P_\beta$ by (1), but $V_{\alpha+1} \notin P_\alpha$ by the usual diagonalization argument.

The proof of (3) consists of two lemmas. If $V_\alpha(\langle x_1,\ldots,x_r\rangle, \underline{f}) = f(x_1,\ldots,x_r)$, we say that $\underline{f}$ is an α-index of f.

Lemma 1. For each $n = 2,3,\ldots$ there is a primitive recursive function I_n such that if $\alpha = \omega^{n-2}\beta$, $\underline{g}$ is an $\alpha+1$-index of g, and f is obtained from f by n-fold iteration, then $I_n(\underline{g})$ is a $\alpha + \omega^{n-2}2$ index of f.

Proof of lemma 1. We impose some conditions on the indexing in order to prove the lemma.

Property 1. 0 is an index of V; 1 is an index of addition.

Property 2. Let A be a term built up from functions $g_1,\ldots,g_r$, constants $a_1,\ldots,a_s$, and variables $x_1,\ldots,x_m$. Then there is a primitive recursive function S depending on the structure of A such that if $\underline{g}_1,\ldots,\underline{g}_r$ are indices of $g_1,\ldots,g_r$ respectively, then $\underline{f} = S(\underline{g}_1,\ldots,\underline{g}_r, a_1,\ldots,a_s)$ is an index of the function f defined by

$$f(x_1,\ldots,x_m) = A.$$

Property 3. There is a primitive recursive function I such that if $\underline{g}$ is an index for a function g and if f is obtained from g by 1-fold iteration (i.e., $f(x) = g^x(1)$), then $\underline{f} = I(\underline{g})$ is an index of f.

Property 4. Each primitive recursive function has an index which is an α-index for every $\alpha > 0$; i.e., if $f = f(x_1,\ldots,x_r)$ is primitive recursive, then there is a number $\underline{f}$ such that for all $\alpha > 0$,

$$V_\alpha(\langle x_1,\ldots,x_r\rangle,\underline{f}) = f(x_1,\ldots,x_r) .$$

The proof of lemma 1 now proceeds by induction on n.

Case 1. $n = 2$. Let f_0 be obtained from g by 1-fold iteration and let f_{k+1} be obtained from f_k by 1-fold iteration. Then

$$f(x,y) = f_y(x).$$

Let the primitive recursive function L be defined by

$$L(0,\underline{g}) = I(g)$$
$$L(k+1,\underline{g}) = I(L(k,g)).$$

Then by property 3 and induction on y,

$$f(x,y) = f_y(x) = V_{\alpha+1}(x,L(y,\underline{g})) .$$

This is an explicit definition of f in terms of functions from $P_{\alpha+1}$. By property 3 there is a function $S = S(\underline{V},\underline{L},\underline{g})$ such that if $\underline{V}$ and $\underline{L}$ are indices of $V_{\alpha+1}$ and L respectively, then $S(\underline{V},\underline{L},\underline{g})$ is an index of f. By property 1, $V_{\alpha+1}$ always has $\alpha+2$-index 0 and by property 4 we may choose $\underline{L}$ independent of α. Hence define I_2 by

$$I_2(\underline{g}) = S(0,\underline{L},\underline{g}) .$$

Case 2. We assume the lemma for n and prove it for $n+1$. For any n-ary function $h = h(x,y_0,\ldots,y_{n-3},y_{n-2})$ define h^0 by setting $h^0(x) = h(x,0,\ldots,0,x)$; i.e., the singulary function h^0 is obtained from the n-ary function h by identifying the first and last variables and setting the other variables equal to 0. By property 2 let S be a primitive recursive function which converts an index of h into an index of h^0. Now let $f_0 = f_0(x,y_0,y_1,\ldots,y_{n-2})$ be obtained from g by n-fold iteration and let f_{k+1} be obtained from f_k^0 by n-fold iteration. Then, if f is obtained from g by $n+1$-fold iteration,

$$f(x,y_0,y_1,\ldots,y_{n-2},y) = f_y(x,y_0,y_1,\ldots,y_{n-2}) .$$

Define the primitive recursive function L by

$$L(0,\underline{g}) = I_n(\underline{g})$$

$$L(k+1,\underline{g}) = I_n(S(L(k,\underline{g}))) .$$

Then by induction hypothesis and induction on k, $L(k,g)$ is a $\alpha+\omega^{n-2}2(k+1)$-index of f_k; i.e.,

$$\begin{aligned} f(x,y_0,\ldots,y_{n-2},y) &= f_y(x,y_0,\ldots,y_{n-2}) \\ &= V_{\alpha+\omega^{n-2}2(y+1)}(\langle x,y_0,\ldots,y_{n-2}\rangle, L(y,\underline{g})) \\ &= V_{\alpha+\omega^{n-1}}(\langle x,y_0,\ldots,y_{n-2}\rangle, \langle 2(y+1), L(y,\underline{g})\rangle) . \end{aligned}$$

This is an explicit definition of f in terms of primitive recursive functions and $V_{\alpha+\omega^{n-1}}$. By property 4 the primitive recursive functions have $\alpha+\omega^{n-1}2$-indices which are independent of α. Define $v_0 = 0$ and $v_{k+1} = \langle 1, v_k\rangle$. Then, if γ is divisible by ω^k, v_k is a $\gamma+\omega^k$-index of V_γ. (This is proves from property 1 by induction on k.) By one of the hypotheses of this lemma, $\alpha + \omega^{n-1}$ is divisible by ω^{n-1}. Now by property 3 we may define I_{n+1} in exactly the same way we defined I_2 in case 1.

This completes the proof of lemma 1.

Lemma 2. $V_\alpha \in N_n$ for $\alpha < \omega^{n-1}$.

Proof of lemma 2. We list two further restrictions on the indexing required to prove this lemma.

Property 5. There are singulary primitive recursive functions G and H, a ternary primitive recursive function J, and a binary primitive recursive function K such that for every positive integer $\underline{f}$, $G(f) < \underline{f}$ and $H(\underline{f}) < f$ and such that for every natural number $\underline{f}$ exactly one of the following six cases holds:

Case 0. $\underline{f}$ is not an index of any function.

Case 1. $\underline{f} = 0$; i.e., $\underline{f}$ is an index of V.

Case 2. $\underline{f} = 1$; i.e., $\underline{f}$ is an index of addition.

Case 3. $G(\underline{f}) = \underline{g}$ and $H(\underline{f}) = \underline{h}$ are indices for functions g and h respectively, and $\underline{f}$ is an index of the function f obtained from g and h by composition; i.e.,

$$f(x_1,\ldots,x_n,y_1,\ldots,y_m) = h(x_1,\ldots,x_n,g(y_1,\ldots,y_m)) .$$

In this case,

$$J(\underline{f}, \langle x_1,\ldots,x_n,y_1,\ldots,y_m\rangle ,u) = \langle x_1,\ldots,x_n,u\rangle$$

and

$$K(\underline{f}, \langle x_1,\ldots,x_n,y_1,\ldots,y_m\rangle) = \langle y_1,\ldots,y_m\rangle .$$

Case 4. $G(\underline{f}) = \underline{g}$ is an index for a function g and $\underline{f}$ is an index for a function f obtained from g by an explicit transformation; i.e.,

$$f(x_1,\ldots,x_n) = g(\xi_1,\ldots,\xi_m) .$$

In this case,

$$K(\underline{f}, \langle x_1,\ldots,x_n\rangle) = \langle \xi_1,..,\xi_m\rangle .$$

Case 5. $G(\underline{f}) = \underline{g}$ and $H(\underline{f}) = \underline{h}$ are indices of functions g and h respectively and $\underline{f}$ is an index of a function f which is obtained from g and h by primitive recursion; i.e.,

$$\begin{aligned} f(x,y) &= g(x) && y = 0, \\ f(x,y) &= h(x,y-1,f(x,y-1)) && \text{for } y \neq 0. \end{aligned}$$

Property 6. There is a primitive recursive function whose value for argument $\underline{f}$ is $i = 0,1,\ldots,5$ according as case i above holds.

Lemma 2 is trivial for $n = 1$; hence, suppose $n \geq 2$. For each $k = 0,1,2,\ldots$ define the n-ary function $\underline{V}_k$ by

$$\underline{V}_k(w,\underline{f},y_0,y_1,\ldots,y_{n-3}) = V_\alpha(w,\underline{f})$$

where

$$\alpha = \omega^{n-2}k + \omega^{n-3}y_{n-3} + \ldots + \omega y_1 + y_0 .$$

Let $w = \langle x,y\rangle$ and $\underline{f} = \langle u,v\rangle$.

Then

$$\underline{V}_k(w,\underline{f},0,\ldots,0) = w + \underline{f} \qquad \text{if } k = 0,$$

$$\underline{V}_k(w,\underline{f},0,\ldots,0,0) = \underline{V}_{k-1}(w,v,0,\ldots,0,u) \qquad \text{if } k \neq 0.$$

Abbreviate $y_0,y_1,\ldots,y_{n-3}$ by Y and $y_0+1,y_1,\ldots,y_{n-3}$ by $Y+1$. Then

$$\underline{V}_k(w,\underline{f},Y+1) = 0 \qquad \text{if case 0;}$$

$$\underline{V}_k(w,\underline{f},Y+1) = \underline{V}_k(x,y,Y) \qquad \text{if case 1;}$$

$$\underline{V}_k(w,\underline{f},Y+1) = x + y \qquad \text{if case 2;}$$

$$\underline{V}_k(w,\underline{f},Y+1) = \underline{V}_k(J(\underline{f},w,\underline{V}_k(K(\underline{f},w),G(\underline{f}),Y+1)), H(\underline{f}),Y+1) \qquad \text{if case 3;}$$

$$\underline{V}_k(w,\underline{f},Y+1) = \underline{V}_k(K(\underline{f},w),G(\underline{f}),Y+1) \qquad \text{if case 4;}$$

$$\underline{V}_k(w,\underline{f},Y+1) = \underline{V}_k(x,G(\underline{f}),Y+1) \qquad \text{if case 5 and } y = 0;$$

$$\underline{V}_k(w,\underline{f},Y+1) = \underline{V}_k(\langle x,y-1,\underline{V}_k(\langle x,y-1\rangle,\underline{f},Y+1)\rangle, H(\underline{f}),Y+1) \qquad \text{if case 5 and } y \neq 0.$$

$$\underline{V}_k(w,\underline{f},0,\ldots,0,0,y_j+1,y_{j+1},\ldots,y_{n-3}) =$$

$$= \underline{V}_k(w,v,0,\ldots,0,u,y_j,y_{j+1},\ldots,y_{n-3}) \quad .$$

This is a schema of nested n-fold recursion in the variables y, $\underline{f}$, y_0, $y_1,\ldots,y_{n-3}$. Hence $\underline{V}_0$ is obtained by a nested n-fold recursion and for $k \neq 0$, $\underline{V}_k$ is obtained from $\underline{V}_{k-1}$ by a nested n-fold

recursion. Hence by induction on k, $\underline{V}_k \in N_n$ for each k. Hence $V_\alpha \in N_n$ for $\alpha < \omega^{n-1}$ as was to be shown. This completes the proof of lemma 2.

Now by lemma 1, $\bigcup_\alpha P_\alpha \; \alpha < \omega^{n-1}$ is closed under n-fold iteration and hence contains N_n. By lemma 2 the converse containment holds. This completes the proof of the theorem.

PATHOLOGY

Theorem 1. For every general recursive function f there is an elementary recursive well ordering R such that

(1) $|R| = \omega$.

(2) $f \in U(R)$.

Proof. Let Z by a Turing machine which computes f; let $Z^*(x)$ be the number of operations performed by Z when the input is x; let $U(x,t)$ be the number on the tape of Z if it is started with input x and stops in t or less operations. Then U is elementary recursive and

$$f(x) = U(x,t) \qquad \text{for} \quad t \geq Z^*(x).$$

At this point there are two ways of proceeding. The first is advantageous in that it will enable us to give a definition of f without using a schema of primitive recursion. The second has an interesting corollary.

First Proof. For each natural number x let C_x be the set of all numbers of the form $\langle t,x \rangle + 1$ where $t < Z^*(x)$. Then C_x has exactly $Z^*(x)$ elements. Let C be the union of the C_x. Define the well ordering R as follows:

(1) Every element of C_x precedes every element of C_y if $x < y$.

(2) The element $\langle t_1,x\rangle + 1$ of C_x precedes the element $\langle t_2,x\rangle + 1$ of C_x if $t_2 < t_1$.

(3) An element x not in C precedes all the elements of C_y for $y \geq x$ and follows all the elements of C_y for $y < x$.

(4) The elements not in C are ordered among themselves by $<$.

Then R is an elementary recursive well ordering with 0 as its least element. Also $|R| = \omega$.

Define the function T as follows:

(1) $T(y) = \langle t+1,x\rangle + 1$ if $y = t,x + 1$ and both $\langle t,x\rangle + 1$ and $\langle t+1,x\rangle + 1$ are elements of C;

(2) $T(y) = 0$ otherwise.

Then T is elementary recursive and

$$T(0) = 0$$
$$T(y)\ R\ y \qquad \text{for } y \neq 0.$$

Note that T applied to an element of C_x gives the next smaller element of C_x in the ordering R if such exists. Define M from T by R-annihilation; i.e.,

$$M(0) = 0$$
$$M(y) = 1 + M(T(y)) \qquad \text{for } y \neq 0.$$

Then $M \in U(R)$, and

$$Z^*(x) = M(\langle 0,x \rangle) \quad .$$

Hence $Z^* \in U(R)$. Hence since U is elementary recursive, $f \in U(R)$. This completes the first proof.

Second Proof. Let S_0 be the set of y such that $0 < y \leq Z^*(0)$ and for $x > 0$ let S_x be the set of y such that

$$Z^*(0)+\ldots+Z^*(x-1) < y \leq Z^*(0)+\ldots+Z^*(x) \ ;$$

i.e., S_0 is the first block of $Z^*(0)$ positive integers, S_1 is the next block of $Z^*(1)$ integers, etc. Define the well ordering R as follows:

(1) 0 is the least element in R.

(2) The elements of S_u precede the elements of S_v if $u < v$.

(3) For $y_1, y_2 \in S_x$, $y_1 \, R \, y_2$ if $y_2 < y_1$.

The two place predicate $t = Z^*(x)$ is elementary recursive. (Although, of course, the function Z^* may not be.) Define s by

$$s(0) = 0$$

$$y \in S_{s(y)} \qquad \text{for } y \neq 0.$$

Then $s(y+1) = s(y) + 1$ if there exist numbers $t_0, t_1, \ldots, t_{s(y)}$ such that $t_i \leq y$ and $t_i = Z^*(i)$ for $i = 0,1,\ldots,s(y)$ and such

that $t_0+t_1+\ldots+t_{s(y)} = y$; $s(y+1) = s(y)$ otherwise. Hence s is elementary recursive. Now $y_1 \; R \; y_2$ if and only if $y_1 = 0$ and $y_2 \neq 0$, or $s(y_1) < s(y_2)$, or $s(y_1) = s(y_2)$ and $y_2 < y_1$. Hence R is elementary recursive. Clearly $|R| = \omega$.

Define T as follows:

$$T(0) = 0$$

$$T(y) = 0 \quad \text{if} \quad s(y) \neq s(y+1)$$

$$T(y) = y+1 \quad \text{if} \quad s(y) = s(y+1).$$

Then T is elementary recursive and $T(y) \; R \; y$ for $y \neq 0$. T takes an element of S_x which is not the smallest element of S_x (in the ordering R) into the next smaller element of S_x (in the ordering R). Define M from T by R-annihilation; i.e.,

$$M(0) = 0$$

$$M(y) = 1 + M(T(y)) \qquad \text{for} \quad y \neq 0.$$

Then $M \in U(R)$ and

$$Z^*(0) = M(1)$$

$$Z^*(x+1) = M(1 + \sum_{i=0}^{x} Z^*(i)) .$$

This is a primitive recursive schema. Hence $Z^* \in U(R)$. Hence $f \in U(R)$. This completes the second proof.

Corollary. There is an elementary recursive well ordering R such that

(1) $|R| = \omega$.

(2) $<$ is not embeddable in R.

Proof. Recall that $<$ is embeddable in R if there is a primitive recursive function e such that $e(x)\ R\ e(y)$ whenever $x < y$. Consider the well ordering R defined from the function f in the second proof above. We will show that if $<$ is embeddable in R, then f is primitive recursive. Hence we may construct an R satisfying the corollary if we take f to be any recursive, non-primitive recursive function.

Hence suppose e is a primitive recursive function and $e(x)\ R\ e(y)$ whenever $x < y$. Recall the definitions of U, Z^*, S_x, and s from above. Let $r(0) = 0$ and for $y \neq 0$, let $r(y)$ be the number of elements z such that $z \in S_{s(y)}$ and $y\ R\ z$. Then

$$r(0) = r(1) = 0$$

$$r(y+1) = 1 + r(y) \qquad \text{if } s(y) = s(y+1)$$

$$r(y+1) = 0 \qquad \text{otherwise.}$$

Then r is elementary recursive. Now note that

$$e(x)\ R\ e(x+1)\ R\ e(x+2)\ R \ldots R\ e(x+n) \ .$$

Hence if $e(x) \in S_u$, then $e(x + r(e(x)) + 1) \in S_v$ where $u < v$. Define g by

$$g(0) = 1$$

$$g(x+1) = g(x) \dot{+} r(e(g(x))) \dot{+} 1$$

and define h by

$$h(x) = e(g(x)) \quad .$$

Then h is primitive recursive. Since $g(x) < g(x+1)$, we have $h(x)\ R\ h(y)$ whenever $x < y$. Furthermore, $s(h(x)) < s(h(x+1))$; i.e., if $h(x) \in S_u$, then $h(x+1) \in S_v$ where $u < v$. Hence by the definition of S_x,

$$Z^*(0) \dot{+} \ldots \dot{+} Z^*(x) \leq h(x+1).$$

Thus $Z^*(x) \leq h(x+1)$ and so,

$$f(x) \in U(x, h(x+1)) \quad .$$

Thus f is primitive recursive as was to be shown. This completes the proof of the corollary.

We now generalize the Grzegorczyk and Kleene hierarchies. Recall the definition of the set of Church Kleene ordinal notations (see Davis [2]). If a is an ordinal notation, we let $|a|$ be the ordinal which it denotes. Then

(1) 1 is a notation for 0; i.e., $|1| = 0$.

(2) If a is an ordinal notation, then so is 2^a and $|2^a| = |a| + 1$.

(3) If $3 \cdot 5^{\underline{q}}$ is an ordinal notation, then $\underline{q}$ is a godel number of a recursive function q, and $q(n)$ is an ordinal notation for every n. The sequence of ordinals $|q(0)|, |q(1)|, |q(2)|, \ldots$ is a strictly increasing sequence with limit $|3 \cdot 5^{\underline{q}}|$ (i.e., is a <u>fundamental sequence</u> for $|3 \cdot 5^{\underline{q}}|$).

Remark: The reader should be careful not to confuse the concept of a godel number of a general rcursive function (which might for instance be the godel number of a Turing machine which computes the function) with the primitive recursive indices used to define the Kleene sub-recursive hierarchy.

Following Kleene we let n_o be the (unique) ordinal notation for the finite ordinal n; i.e.,

$$0_o = 1$$

$$(n+1)_o = 2^{n_o}$$

so that

$$|n_o| = n \ .$$

Now for each ordinal notation a define the function W_a as follows:

(1) $W_a(x) = 2^x$ if $a = 1$.

(2) $W_a(x) = W_b^x(1)$ if $a = 2^b$.

(3) $W_a(x) = W_{q(x)}(x)$ if $a = 3 \cdot 5^{\underline{q}}$ where $\underline{q}$ is a godel number of q.

Now define $E_a = E(W_a)$, the set of functions elementary recursive in W_a.

Theorem 2. For every general recursive function f there is an ordinal notation a such that

(1) $|a| = \omega$.

(2) $f \in E_a$.

Proof. As in theorem 1 let Z^* and U satisfy

$$f(x) = U(x,t) \quad \text{for} \quad t \geq Z^*(x) \ ,$$

where Z^* is general recursive and U is primitive recursive. Choose a strictly increasing recursive function r such that

$$Z^*(x) \leq r(x) \ .$$

Define q by

$$q(x) = (r(x))_0 \ ;$$

then

$$|q(x)| = r(x).$$

Let $\underline{q}$ be a godel number of q and let $a = 3 \cdot 5^{\underline{q}}$. Then a is an ordinal notation with $|a| = \omega$.

It is easy to show that

$$W_{n_o}(x) \geq n \qquad \text{for } x \neq 0.$$

Hence

$$W_a(x) = W_{q(x)}(x) \leq r(x) \geq Z^*(x) .$$

Hence

$$f(x) = U(x, W_a(x)).$$

Hence $f \in E_a$. This completes the proof of theorem 2.

Let $P(V)$ be the set of functions which are primitive recursive in the binary function V. Fix an indexing of these functions which satisfies the properties laid down in the last section. For each ordinal notation a define the function A_a as follows:

(1) $V_a(x,y) = x + y$ if $a = 1$.

(2) $V_a(\langle x_1,\ldots,x_r\rangle, \underline{f}) = f(x_1,\ldots,x_r)$ where $\underline{f}$ is an index of the function $f \in U(V_b)$ if $a = 2^b$.

(3) $V_a(x, \langle u, \underline{f}\rangle) = V_{q(u)}(x, \underline{f})$ where q is the function with godel number $\underline{q}$ if $a = 3 \cdot 5^{\underline{q}}$.

Let $P_a = P(V_a)$, the set of functions primitive recursive in V_a. If $\underline{f}$, f, and a satisfy (2), we say that $\underline{f}$ is an a-index of f.

Theorem 3. For every general recursive function f there is an ordinal notation a such that

(1) $|a| = \omega$

(2) $f \in P_a$.

Proof. The proof requires two lemmas. The first lemma is a version of the recursion theorem.

Lemma 1. If $F \in P(V_b)$, there is a natural number e such that

$$V_c(\langle x_1,\ldots,x_r\rangle, e) = F(x_1,\ldots,x_r,e)$$

where $c = 2^b$ and F is r+1-ary.

Proof of lemma 1. By property 2 (see the last section) let S be a primitive recursive function such that if $\underline{h}$ is an index of the r+1-ary function $h = h(x_1,\ldots,x_r,z)$, then $S(\underline{h},k)$ is an index of the r-ary function $h_k = h(x_1,\ldots,x_r,k)$. Let g be defined by

$$g(x_1,\ldots,x_r,z) = F(x_1,\ldots,x_r,S(z,z)).$$

Let $\underline{g}$ be a c-index of g. Let $e = S(\underline{g},\underline{g})$.

$$\begin{aligned} V_c(\langle x_1,\ldots,x_r\rangle, e) &= V_c(\langle x_1,\ldots,x_r\rangle, S(\underline{g},\underline{g})) \\ &= g(x_1,\ldots,x_r,\underline{g}) \\ &= F(x_1,\ldots,x_r,S(\underline{g},\underline{g})) \\ &= F(x_1,\ldots,x_r,e) . \end{aligned}$$

This completes the proof of lemma 1.

Lemma 2. There is a natural number e such that for all n

$$V_{n_o}(1,e) = e + n + 1 .$$

Proof of lemma 2. Consider the function F defined by

$$F(x,z) = V_{n_o}(x,z) + x .$$

The definition of F depends on n, but by properties 1 and 2 of the preceding section a $(n+1)_o$-index of F may be chosen independent of n. (0 is always a $(n+1)_o$-index of V_{n_o}.) Thus by the proof of the last lemma there is a natural number e such that for all n

$$V_{(n+1)_o}(x,e) = V_{n_o}(x,e) + x.$$

Then

$$V_{0_o}(1,e) = 1 + e = e + 0 + 1$$

and if $V_{n_o}(1,e) = e + n + 1$, then

$$V_{(n+1)_o}(1,e) = V_{n_o}(1,e) + 1$$

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

Hence lemma 2 is proved by induction on n.

We now return to the proof of theorem 3. Choose an arbitrary recursive function f. As in theorem 2, choose recursive functions Z^*, r, and q, a primitive recursive function U, and an ordinal notation a such that

(1) $f(x) = U(x,t)$ for $t \geq Z^*(x)$.

(2) $Z^*(x) \leq r(x)$.

(3) $q(0), q(1), q(2), \ldots$ is an increasing sequence of finite ordinal notations with $|q(x)| = r(x)$.

(4) $a = 3 \cdot 5^{\underline{q}}$ where $\underline{q}$ is a godel number of q. Hence $|a| = \omega$.

For the natural number e of the previous lemma

$$V_a(1, \langle x,e \rangle) = V_{q(x)}(1,e) = e + r(x) + 1.$$

Hence, $r \in P_a$. Hence $f \in P_a$ since $f(x) = U(x,r(x))$. This completes the proof of theorem 3.

BIBLIOGRAPHICAL REMARKS

Davis [2] and Ritchie [12] contain good expositions of the theory of Turing machines. [2] also contains an exposition of the theory of Church Kleene ordinal notations. Grzegorczyk [4] contains a good exposition of the theory of elementary recursive functions and Peter [11] contains an exposition of the theory of nested n-fold recursion.

Most of the theorems in the section entitled "The Ordinal Recursion Hierarchy" were proven by Tait in [14]. The special case of theorem 1 was proven there; the general case is due to J. Guard (personal communication). Theorems 6 and 8 appear in [14] with the same proofs I have given. Theorem 7 appears in [14] for the special case $n = 0$; Guard pointed out to me that the general case can be proven with essentially the same argument. Theorem 9 appears in [14]; the simpler proof which I have given is due to Guard.

The case $n = 1$ of the Extended Grzegorczyk Theorem is essentially theorem 4.13 of Grzegorczyk [4]. His hierarchy differs from mine in that the first few classes of his hierarchy are not elementary recursive degrees; in fact, they do not contain all the elementary recursive functions. Also, he uses binary functions to define his hierarchy where I used singulary functions.

The ideas in the section entitled "Applications to Computational Complexity" were inspired by Hartmanis and Stearns [5] and [6] and by Ritchie [12].

The Kleene subrecursive hierarchy was defined in Kleene [7]. The characterization of N_n in terms of that hierarchy is very much like a theorem in Axt [1].

Theorem 1 in the section entitled "Pathology" was first announced by Myhill [10]. A slightly weaker theorem was proved independently by Routledge [13]. Similar theorems appear in Liu [8] and [9]. The corollary is due to Guard. Theorem 3 of that section is a slightly stronger version of a theorem in Axt [1]; he showed that every recursive set is obtained at level ω. A very nice pathology theorem appears in Feferman [3]. He shows that even if ordinal notations are restricted so as to allow only primitive recursive fundamental sequences, the Kleene hierarchy collapses at ω^2.

BIBLIOGRAPHY

1. Paul Axt, On a Subrecursive Hierarchy and Primitive Recursive Degrees, Trans. Amer. Math. Soc., V 92 (1959) pp. 85-105.

2. Martin Davis, Computability and Unsolvability, McGraw-Hill Book Co., 1958.

3. Solomon Feferman, Classifications of Recursive Functions by Means of Hierarchies, Trans. Amer. Math. Soc., V 104 (1962)

4. Andrej Grzegorczyk, Some Classes of Recursive Functions, Rozprany Matematyczne, Warsaw, 1953.

5. J. Hartmanis and R. E. Stearns, Computational Complexity of Recursive Sequences, Switching Circuit Theory and Logical Design, Proceedings of the 5th Annual Symposium, Princeton University, November 11-13, 1964.

6. _____, On the Computational Complexity of Recursive Sequences, Trans. Amer. Math. Soc., (in press).

7. S. C. Kleene, Extension of an Effectively Generated Class of Functions by Enumeration, Colloquim Mathematicum, VI (1958) pp. 67-78.

8. Shih-Chao Liu, A Theorem on General Recursive Functions, Proceedings Amer. Math. Soc., V 11 (1960) pp 184-187.

9. _____, Proof of a Conjecture of Routledge, Proc. Amer. Math. Soc., V 11 (1960) pp. 967-969.

10. J. Myhill, A Stumblingblock in Constructive Mathematics (abstract), J. Symbolic Logic, V 18 (1953) p. 190.

11. Rozsa Péter, Rekursive Funktionen, Akademiai Kiado, Budapest, 1957.

12. R. W. Ritchie, Classes of Predictably Computable Functions, Trans. Amer. Math. Soc., V 106 (1963) pp. 139-173.

13. N. A. Routledge, Ordinal Recursion, Proceedings Cambridge Philos. Soc., V 49 (1953) pp. 175-182.

14. W. W. Tait, Nested Recursion, Math. Annalen, V 143 (1961) pp. 236-250.