View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

INFORMATION AND COMPUTATION **99**, 123–153 (1992)

Ordinal Complexity of Recursive Definitions

M. V. H. FAIRTLOUGH* AND S. S. WAINER[†]

Department of Pure Mathematics, Leeds University,
Leeds LS2 9JT, U.K.

We classify and compare recursive and iterative definitions of total computable functions according to the complexity of the ordinal structures needed to verify termination. An old theorem of Tait is generalised to show that the cost of transforming recursive programs into while-loops is at most an exponential increase in the length of their termination orderings. © 1992 Academic Press, Inc.

1. INTRODUCTION

We are concerned here with the classification and comparison of ordinary recursive functions, according to the "complexity" of their recursive or iterative definitions. Of course the theory of computation must take account of many different data-structures other than just the natural numbers ℕ. However, it seems that this classical case already exhibits many (most?) of the fundamental characteristics of computability over more general data-structures, and furthermore there are still deep questions to be resolved even over ℕ!

Recursion is the definition of a (number-theoretic) function f as the least fixed point of a (monotone, continuous) operator Φ thus:

$$f(x) = \Phi(f)(x). \qquad (1)$$

Typically, the "body" $\Phi(f)(x)$ will be given as a term built out of the variable (or variables) x by applying given functions and f itself. Possibly there will be nested occurrences of f.

If f is totally defined then evaluation of f on any given number $x = x_0$ will call for the prior evaluation of f on certain other arguments $x_1, ...$ determined by Φ. Each of these may call for further evaluations of f on arguments $x_2, ...$. Thus the computation of f by Φ can be described in terms of a tree of finite sequences $\langle x_0, x_1, ..., x_k \rangle$ where for each $i < k$,

* Research supported by the U.K. Science and Engineering Research Council, studentship 87000845.

[†] Supported by SERC Research Grant GR/D 90345 and SERC Logic for IT Senior Fellowship B/ITF/216.

0890-5401/92 $5.00
Copyright © 1992 by Academic Press, Inc.
All rights of reproduction in any form reserved.

$f(x_{i+1})$ is one of the calls made by Φ in evaluating $f(x_i)$. Imagine the tree being developed downwards so that $\langle x_0 \rangle$ lies above $\langle x_0, x_1 \rangle$ which lies above $\langle x_0, x_1, x_2 \rangle$, etc. Denote the tree $T(\Phi)$.

Since we are assuming $f(x_0)$ to be defined, each branch $\langle x_0 \rangle$, $\langle x_0, x_1 \rangle$, $\langle x_0, x_1, x_2 \rangle$, ... must terminate at a point $\langle x_0, x_1, x_2, ..., x_m \rangle$ where $f(x_m)$ is evaluated outright by Φ, without any further calls on f being made. *Thus the totality of f corresponds to the well-foundedness of its computation tree $T(\Phi)$.*

The well-foundedness of $T(\Phi)$ means that we can make definitions and proofs by induction upwards through its branches. In particular we can recast the original recursion in a way which makes explicit the inductive nature of its intended computation. For, given any point $\sigma = \langle x_0, ..., x_k \rangle \in T(\Phi)$ and any $z \in \mathbb{N}$, let $P_z(\sigma)$, the "z-predecessor" of σ, be obtained by tagging z on the end of σ thus:

$$P_z(\sigma) = \langle x_0, ..., x_k, z \rangle.$$

Now define a new function f: $T(\Phi) \times \mathbb{N} \to \mathbb{N}$ by induction over $T(\Phi)$,

$$f(\sigma, x) = \Phi(f_{P(\sigma)})(x), \tag{2}$$

where $f_{P(\sigma)}$: $\mathbb{N} \to \mathbb{N}$ is given by

$$f_{P(\sigma)}(z) = f(P_z(\sigma), z).$$

Then by induction up the tree $T(\Phi)$ we have *if $\sigma \in T(\Phi)$ and x is its final component then*

$$f(\sigma, x) = \textit{the original } f(x)$$

and so for each $x \in \mathbb{N}$, putting $\sigma = \langle x \rangle$,

$$f(\langle x \rangle, x) = \textit{the original } f(x).$$

The point of all this is that the structure of the sub-tree of $T(\Phi)$ below $\langle x \rangle$ reflects the complexity of the induction needed to evaluate $f(x)$. Thus $\langle x \rangle$ measures the "stage" in $T(\Phi)$ at which $f(x)$ becomes defined.

We can go further, and well-order $T(\Phi)$ by the Kleene–Brouwer ordering, which places $\langle x_0, ..., x_k \rangle$ below $\langle y_0, ..., y_l \rangle$ if either $\langle x_0, ..., x_k \rangle$ extends $\langle y_0, ..., y_l \rangle$ or there is a j such that $x_0 = y_0$, $x_1 = y_1, ..., x_{j-1} = y_{j-1}$ and $x_j < y_j$. *The structure and length of this well-ordering is then a measure of the "induction-complexity" needed to verify that f is indeed totally defined.* It can be used to classify and compare various kinds of recursion, and it is this classification, and its relationship with more usual notions of computational complexity, which we study here.

In particular we are interested in the comparative definitional powers of recursions as against while-loops. A condition B determines a class of while-programs:

$$\textbf{while } B(x) \textbf{ do } S(x) \textbf{ od}.$$

It also determines a tree of "secured sequences":

$$T(B) = \{\langle x_0, ..., x_k\rangle : \forall i < k . B(x_i)\}.$$

If S computes a number-theoretic function s then evaluation of the while-loop, starting with $x = x_0$, is described by the following s-path down through $T(B)$:

$$\langle x_0\rangle, \langle x_0, s(x_0)\rangle, \langle x_0, s(x_0), s^2(x_0)\rangle, ...$$

Then to say that the loop terminates after k iterations is to say that the s-path ends with $\langle x_0, s(x_0), s^2(x_0), ..., s^k(x_0)\rangle$ where $B(s^k(x_0))$ fails.

Again we see that for any class C of programs S, computing number-theoretic functions s, *the while-loop will always terminate if and only if its tree $T(B)$ is well-founded with respect to all C-paths.* As before, the complexity of the well-ordered part of $T(B)$ provides a measure of complexity of the while-loop, and we can recast the loop as a tail-recursion over $T(B)$ as follows: define h: $T(B) \times \mathbb{N} \to \mathbb{N}$ by

$$h(\sigma, x) = \textbf{if } \sigma \text{ is terminal } \textbf{then } x \textbf{ else } h(P_{s(x)}(\sigma), s(x)). \tag{3}$$

Then provided the while-loop terminates, we can prove as before, using induction up along the s-path in $T(B)$, that $h(\langle x\rangle, x)$ is its final value.

What we have seen is that, in order to simulate the computation of a while-loop by a recursion as in (3), a new recursion variable σ ranging over $T(B)$ is introduced. This recursion can itself be rewritten as a while-loop *controlled by* $T(B)$,

$$\textbf{while } \sigma \in T(B) \textbf{ do } S(x);\ \sigma := P_x(\sigma) \textbf{ od}, \tag{4}$$

where the original condition $B(x)$ has now been replaced by $\sigma \in T(B)$.

The introduction of trees $T(B)$, $T(\Phi)$ as parameters provides a unifying format for while-loops and recursions. This new model is introduced not primarily as a practical tool, but as a convenient theoretical one which should enable the definitional powers of recursions and while-loops to be compared and classified, in such a way as to give some mathematical analysis of the costs of their respective executions.

We shall take the forms (2) and (4) as our basic formulations of recursion and while, but with arbitrary well-ordered trees as *parameters* (replacing the fixed $T(\Phi)$, $T(B)$). Thus we require a uniform way of presenting the type of well-ordered trees, and to this end, our first definitions in Section 2 will supply the set Ω^s of so-called structured tree ordinals α, β, ..., ω, ...

Before stating the main results, we describe informally the main classes of programs involved; precise definitions are given in Sections 2 and 3.

Given a set $A \subseteq \Omega^s$, T-WHILE(A) is the set of (functions defined by) programs built from sequencing, conditionals, and while-loops of the form (4), wherein the tree parameter σ lies in A. REC(A) is the set of (functions defined by) composition, conditionals, and recursions of the form (2), wherein the tree parameter σ lies in A. $H(A)$ is the set of (functions defined by) tail recursions of the form (3), wherein the tree parameter σ lies in A. These $H(A)$ functions are essentially the so-called "Hardy functions" of subrecursive hierarchy theory (see for example Buchholz and Wainer, 1987) and they allow us to connect the results here with various classes of recursive functions arising naturally out of proof theory.

In the following, we assume that A satisfies suitable conditions such as "Turing machine representability" and closure under addition and multiplication. We also use $\equiv$ between classes of programs to denote the fact that there are effective each-way compilations between them. These compilations can be read off from the proofs in Sections 4 and 5.

THEOREM II.

$$T\text{-WHILE}(A) \equiv \text{ELEM}(H(A)) \equiv \text{SPACE}(H(A)).$$

THEOREM III.

$$\text{REC}(\omega \cdot A) \equiv T\text{-WHILE}(\omega^A).$$

Theorem II is a generalization of an old theorem of Tait (1961) showing that arbitrary nested recursions over a given well-ordering can be compiled to unnested ones but at the cost of an exponential increase in the size of the well-ordering. In subrecursive terms, this corresponds closely to the simple fact that the "fast-growing" Grzegorczyk hierarchy $\{F_\alpha\}$ is reducible to the Hardy hierarchy $\{H_\alpha\}$, again at the expense of an exponential increase: $F_\alpha = H_{\omega^\alpha}$. Informally, then, the cost of transforming recursive into while-programs is, in the worst case, an exponential increase in induction complexity. Furthermore, the hidden complexity of recursions over $\omega \cdot A$ is measured by $H(\omega^A)$, equivalently $F(A)$.

As an example, with $A = \mathbb{N}$:

$$\begin{aligned} \text{REC}(\omega \cdot \mathbb{N}) &\equiv T\text{-WHILE}(\omega^{\mathbb{N}}) \\ &\equiv \text{PRIMITIVE RECURSIVE} \\ &\equiv \text{PROVABLY RECURSIVE IN } \Sigma_1\text{-INDUCTION}. \end{aligned}$$

Thus the Ackermann function ($=H_{\omega^\omega}$) is not computable by any T-WHILE($\omega^{\mathbb{N}}$) program, but is computable by a while-program from the parameter ω^ω.

2. Preliminary Definitions and Results

Definition 2.1. The set Ω of so-called "tree-ordinals" is defined inductively by the rules

$$\begin{array}{lrl} (1) & & 0\in\Omega \\ (2) & \alpha\in\Omega\Rightarrow & \alpha+1\in\Omega \\ (3) & \lambda \text{ is a function from } \mathbb{N} \text{ to } \Omega\Rightarrow & \lambda\in\Omega, \end{array}$$

where $\alpha+1:=\alpha\cup\{\alpha\}$ and if λ is a function from $\mathbb{N}$ to Ω, we shall often henceforth write λ_x for the value $\lambda(x)$ and $\sup_{x\in\mathbb{N}}(\lambda_x)$ for λ.

The "set-theoretic" ordinal rank of $\alpha\in\Omega$ is rank(α), where

$$\text{rank}(0):=0, \qquad \text{rank}(\alpha+1):=\text{rank}(\alpha)+1,$$

and

$$\text{rank}(\sup(\lambda_x)):=\bigcup_{x\in\mathbb{N}}\text{rank}(\lambda_x).$$

Conventions used throughout this paper:

$\alpha, \beta, \gamma, \delta, \ldots$	members of Ω
$A, B, C, \ldots$	subsets of Ω
$a, \ldots, z$	natural numbers
$\mathbf{a}, \ldots, \mathbf{z}$	finite sequences of natural numbers
$\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{U}$,	sets of functions from $\mathbb{N}$ to $\mathbb{N}$.

Definition 2.2. $\prec$ is a binary relation on Ω which is the transitive closure of

$$\begin{array}{ll} (1) & \alpha\prec\alpha+1 \\ (2) & \lambda_n\prec\sup\limits_{x\in\mathbb{N}}(\lambda_x) \qquad \text{for all} \quad n\in\mathbb{N}. \end{array}$$

If A is a subset of Ω then say A is *downward-closed* iff

$$\alpha\prec\beta\in A\Rightarrow\alpha\in A.$$

DEFINITION 2.3. For $x \in \mathbb{N}$, $\prec_x$ is a binary relation on Ω which is the transitive closure of

$$(1) \quad \alpha \prec_x \alpha + 1$$

$$(2) \quad \lambda_x \prec_x \sup_{x \in \mathbb{N}}(\lambda_x).$$

DEFINITIONS 2.4 (Addition, Multiplication, and Exponentiation in Ω).
Addition:

$$(1) \quad \alpha + 0 := \alpha$$

$$(2) \quad \alpha + (\beta + 1) := (\alpha + \beta) + 1$$

$$(3) \quad \alpha + \sup_{x \in \mathbb{N}}(\lambda_x) := \sup_{x \in \mathbb{N}}(\alpha + \lambda_x).$$

Multiplication:

$$(1) \quad \alpha \cdot 0 := 0$$

$$(2) \quad \alpha \cdot (\beta + 1) := (\alpha \cdot \beta) + \alpha$$

$$(3) \quad \alpha \cdot \sup_{x \in \mathbb{N}}(\lambda_x) := \sup_{x \in \mathbb{N}}(\alpha \cdot \lambda_x).$$

Exponentiation:

$$(1) \quad \alpha^0 := 1$$

$$(2) \quad \alpha^{(\beta + 1)} := (\alpha^\beta) \cdot \alpha$$

$$(3) \quad \alpha^{\sup(\lambda_x)} := \sup_{x \in \mathbb{N}} (\alpha^{\lambda_x}).$$

Note 2.5. Addition, multiplication, exponentiation, and the ordering $\prec$ are preserved under the rank function. Also the associative and distributive laws holding for set-theoretic ordinals also hold in Ω, e.g., $\gamma \cdot (\alpha + \beta) = \gamma \cdot \alpha + \gamma \cdot \beta$ for any α, β, and $\gamma \in \Omega$, but not $(\alpha + \beta) \cdot \gamma = \alpha \cdot \gamma + \beta \cdot \gamma$.

DEFINITIONS 2.6 (The Predecessor Functions P_x on Ω). For $\alpha \in \Omega$ and $x \in \mathbb{N}$, $P_x(\alpha)$ and $\alpha[x]$ are defined by

$$\text{(i)} \quad \begin{aligned} &(1) \quad P_x(0) := 0 \\ &(2) \quad P_x(\alpha + 1) := \alpha \\ &(3) \quad P_x(\sup(\lambda_x)) := P_x(\lambda_x) \end{aligned}$$

$$\text{(ii)} \quad \alpha[x] := \{P_x^{n+1}(\alpha) : n < G_\alpha(x)\},$$

where $G_\alpha(x) :=$ the least k such that $P_x^k(\alpha) = 0$. (See Cichon, 1983).

Note 2.7. Alternatively, $\alpha[x] = \{\beta : \beta + 1 \preccurlyeq_x \alpha\}$. $G_\alpha(x)$ is the cardinality of $\alpha[x]$. G is called the "Slow-Growing Hierarchy."

DEFINITION 2.8 (Structuredness). Ω^s, the set of *structured tree ordinals* (*ordinal structures*), consists of the following subset of Ω: $\alpha \in \Omega^s$ if $\alpha \in \Omega$ and for all limits $\lambda \preccurlyeq \alpha$, $\lambda_0 \neq 0$ and for all $x, y \in \mathbb{N}$

$$x < y \Rightarrow \lambda_x \in \lambda[y].$$

(See Dennis-Jones and Wainer (1984), where the term "nice" is used instead of "structured.")

DEFINITION 2.9. Let $\omega = \sup_{x \in \mathbb{N}}(1 + x)$. Then for example ω, $\omega \cdot 2 + 2$, ω^2, ω^{ω^2} are all structured.

Facts 2.10. Suppose $\alpha, \beta \in \Omega^s$ and $x < y \in \mathbb{N}$. Then

$$\begin{aligned} &\text{(i)} && \alpha \prec_x \beta \Rightarrow \alpha \prec_y \beta \\ &\text{(ii)} && \alpha \preccurlyeq_x \beta \Rightarrow P_x(\alpha) \preccurlyeq_y P_y(\beta) \\ &\text{(iii)} && \alpha[x] \subseteq \alpha[y] \\ &\text{(iv)} && \alpha \prec \beta \Rightarrow \exists z \cdot \alpha \in \beta[z]. \end{aligned}$$

If $\alpha \in \Omega^s$ then the set $\{\beta : \beta \prec \alpha\}$ is linearly ordered by $\prec$. Ω^s is downward-closed and closed under addition. Ω^s is closed under multiplication and exponentiation in the following sense: for all $\beta \in \Omega^s$, if $0 \prec \alpha$ then $\alpha \cdot \beta \in \Omega^s$ and if $1 \prec \alpha$ then $\alpha^\beta \in \Omega^s$.

Subrecursion, Complexity

DEFINITIONS 2.11. (1) If $\mathscr{C}$ is a class of number-theoretic functions, we denote by ELEM($\mathscr{C}$) the closure of $\mathscr{C}$ under the operations of explicit definitions and limited primitive recursion.

(2) $\mathscr{E}^3$ is the set of (Kalmar) Elementary functions $= \text{ELEM}(\{0, +, \times, \exp\})$. (See Rose, 1984).

DEFINITIONS 2.12. We will be concerned with bounds on Turing Machine (TM) computations. We will use the following scheme for encoding sequences of natural numbers on TM tape. Let $\bullet$, # be tape symbols. Then for each sequence $\mathbf{x} = \langle x_1, ..., x_n \rangle$, define the tape-word for $\mathbf{x}$ by

$$[\mathbf{x}] := \underbrace{\bullet \cdots \bullet}_{x_1 + 1} \# \underbrace{\bullet \cdots \bullet}_{x_2 + 1} \# \cdots \# \underbrace{\bullet \cdots \bullet}_{x_n + 1} \#.$$

A number will be encoded as a sequence of length 1. If w is a tape-word then $|w|$ will denote its length.

Let $h\colon \mathbb{N} \to \mathbb{N}$. We denote by SPACE($h$) the set of all functions $f\colon \mathbb{N}^k \to \mathbb{N}^l$ which are Turing Machine computable in space bounded by h composed with an elementary function r, that is, $f \in \text{SPACE}(h)$ iff for all $\mathbf{a} \in \mathbb{N}^k$, the computation of $[f(\mathbf{a})]$ from $[\mathbf{a}]$ requires no more than $h(r(|\mathbf{a}|))$ tape squares (including the initial input $[\mathbf{a}]$ and the final output $[f(\mathbf{a})]$ replacing $[\mathbf{a}]$). If $f \in \text{SPACE}(h)$ we say also "f is space-bounded by h." If $\mathscr{C}$ is a set of number-theoretic functions then we define $\text{SPACE}(\mathscr{C}) := \bigcup_{f \in \mathscr{C}} \text{SPACE}(f)$. Finally we denote by $S_f(\mathbf{a})$ the space required to compute $[f(\mathbf{a})]$ from $[\mathbf{a}]$. (We could restrict r above to being linear without affecting the results, since we are concerned here with "large" rather than "small" complexity classes. Note that with r elementary we have $\text{SPACE}(h) = \text{TIME}(h)$.)

Definition 2.13. Let f, $g\colon \mathbb{N} \to \mathbb{N}$ and $x \in \mathbb{N}$. If for every $z \geqslant x$, $f(z) \leqslant g(z)$ we write $f \leqslant_x g$. Similarly define $f <_x g$ etc.

Definitions 2.14 (Subrecursive hierarchies). Let $f\colon \mathbb{N} \to \mathbb{N}$ and let α be a variable ranging over Ω. The Hardy and Fast- and Slow-Growing Hierarchies are defined by:

(1) The Hardy Hierarchy

$$H_\alpha(f)(x) := \begin{cases} \text{if } \alpha = 0 & \text{then } x \\ & \text{else } H_{P_x(\alpha)}(f)(f(x)) \end{cases}$$

(2) The Fast-Growing Hierarchy

$$F_\alpha(f)(x) := \begin{cases} \text{if } \alpha = 0 & \text{then } f(x) \\ & \text{else } F_{P_x(\alpha)}(f)^{x+1}(x) \end{cases}$$

(3) Extension of the Slow-Growing Hierarchy

$$G_\alpha(f)(x) := \begin{cases} \text{if } \alpha = 0 & \text{then } 0 \\ & \text{else } f(G_{P_x(\alpha)}(f)(x)) \end{cases}$$

(4) If K is one of H, F, or G and $A \subseteq \Omega^s$ then $K(A)$ denotes $\{K_\alpha(\text{succ})\colon \alpha \in A\}$.

Where f is understood, it will be omitted, and unless indicated otherwise, f will be taken to be the successor function.

Note 2.15. (i) If $\alpha \neq 0$ then define

$$h_\alpha(f)(x) = 1 + (\text{the least } n \text{ such that } P_{f^n(x)} P_{f^{n-1}(x)} \cdots P_{f(x)} P_x(\alpha) = 0).$$

Then for $\alpha \neq 0$, $H_\alpha(f)(x) = f^{h_\alpha(f)(x)}(x)$.

(ii) For each $\alpha \in \Omega^s$, the assertion that H_α (G_α) is a total functional is equivalent to the assertion that the ordering $\{\beta : \beta \prec \alpha\}$ is well-founded, i.e., the existence of H_α (G_α) as total functionals witnesses the fact that the ordering on α is well-founded.

(iii) The Hardy Hierarchy has the following combinatorial property, due originally to Ketonen and Solovay (1981):
Let $f\colon \mathbb{N} \to \mathbb{N}$. Let an interval $[n, m] \subseteq \mathbb{N}$ be

(1)	0-large	if $[n, m] \neq \varnothing$
(2)	$(\alpha+1)$-large	if there exists $k \in [n, m]$ with $k \geqslant f(n)$ and $[k, m]$ α-large
(3)	λ-large	if $[n, m]$ is λ_n-large.

Then $H_\alpha(f)(n)$ is the least m such that $[n, m]$ is α-large.

This leads us to the "*α-bounding principle*," which is one of the fundamental ideas of this paper:

The α-Bounding Principle. For $\alpha \in \Omega^s$, the number-theoretic and combinatorial properties associated with α should be bounded by H_α, i.e., be "*α-small*." Thus SPACE(H_α) is the set of α-small computations. For a subset A of Ω^s to be representable, we essentially require that for each $\alpha \in A$, the function $\langle x, \alpha \rangle \mapsto \langle x, P_x(\alpha) \rangle$ should be computable in α-small space.

There is a third characterisation of the Hardy and Slow- and Fast-Growing Hierarchies which emerges when they are considered as hierarchies of *functionals*. First we define for a type two functional C and $\alpha \in \Omega$, the αth iterate $C^{(\alpha)}$ of C:

DEFINITION 2.16.

$$(1) \quad C^{(0)}(f) := f$$

$$(2) \quad C^{(\alpha+1)}(f) := C(C^{(\alpha)}(f))$$

$$(3) \quad C^{(\sup(\lambda_x))}(f) := \text{the function } y \mapsto C^{(\lambda_y)}(f)(y).$$

LEMMA 2.17. *Let $f\colon \mathbb{N} \to \mathbb{N}$. Let the functional* "Apply to f" *be defined by* (Apply to f)$(g) := g \circ f$ *and let the functional* "Apply f" *be defined by*

(Apply f)$(g) := f \circ g$ for all g: $\mathbb{N} \to \mathbb{N}$. Let It *be the functional defined by* $\mathrm{It}(g)(x) := g^{x+1}(x)$. *Then*

$$\text{(i)} \quad H_\alpha(f) = (\text{Apply to } f)^{(\alpha)} \text{ (identity)}$$

$$\text{(ii)} \quad F_\alpha(f) = (\mathrm{It})^{(\alpha)}(f)$$

$$\text{(iii)} \quad G_\alpha(f) = (\text{Apply } f)^{(\alpha)} \text{ (zero)}.$$

Facts 2.18. For f: $\mathbb{N} \to \mathbb{N}$ and $\alpha \in \Omega$

$$\text{(i)} \quad H_{\alpha+\beta}(f) = H_\alpha(f) \circ H_\beta(f)$$

$$\text{(ii)} \quad H_{\alpha \cdot \beta}(f) = H_\beta(H_\alpha(f))$$

$$\text{(iii)} \quad (H_\alpha)^{(\beta)}(f) = H_{\alpha^\beta}(f)$$

$$\text{(iv)} \quad H_{\omega^\alpha}(f) = F_\alpha(f).$$

(For the relationship between G and F, see Wainer (1989).)

Definition 2.19. $\mathrm{SI}(\mathbb{N}^\mathbb{N})$ denote the set of strictly increasing (s.i. for short) functions from $\mathbb{N}$ to $\mathbb{N}$. Suppose that C: $\mathrm{SI}(\mathbb{N}^\mathbb{N}) \to \mathrm{SI}(\mathbb{N}^\mathbb{N})$ is a type-2 functional such that

(1) For each $x \in \mathbb{N}$, C preserves $\leqslant_x$; i.e., for all $f, g \in \mathrm{SI}(\mathbb{N}^\mathbb{N})$, $f \leqslant_x g \Rightarrow C(f) \leqslant_x C(g)$

(2) For all $f \in \mathrm{SI}(\mathbb{N}^\mathbb{N})$, $f \leqslant_0 C(f)$.

Then we say C is *positive increasing*. Note that if f is strictly increasing and "positive" in the sense that $f(x) > 0$, then so is $C(f)$. We abbreviate "strictly increasing and positive" as s.i.p.

Lemma 2.20. *Let* C: $\mathrm{SI}(\mathbb{N}^\mathbb{N}) \to \mathrm{SI}(\mathbb{N}^\mathbb{N})$ *be a positive increasing type-2 functional and let* $\alpha, \beta \in \Omega^s$. *Then*

(i) $C^{(\alpha)}$: $\mathrm{SI}(\mathbb{N}^\mathbb{N}) \to \mathrm{SI}(\mathbb{N}^\mathbb{N})$.

(ii) $C^{(\alpha)}$ *is positive increasing.*

(iii) $\beta \prec_x \alpha \Rightarrow C^{(\beta)}(f) \leqslant_x C^{(\alpha)}(f)$.

Proof. By induction on $\alpha \in \Omega^s$.

$\alpha = 0$

$C^{(0)}(f) = f$ so $C^{(0)}$ clearly satisfies (i) to (iii).

$\alpha = \alpha' + 1$

(i) $C^{(\alpha)}(f) = C(C^{(\alpha')}(f))$ which is s.i. by induction hypothesis (i) and positive if C preserves positive functions and f is positive.

(ii) If $f \leqslant_x g$ then

$$\begin{aligned} C^{(\alpha)}(f) &= C(C^{(\alpha')}(f)) \\ &\leqslant_x C(C^{(\alpha')}(g)) \qquad \text{[by induction hypothesis (ii) and Property 1)]} \\ &= C^{(\alpha)}(g). \end{aligned}$$

For any s.i. f,

$$\begin{aligned} f &\leqslant_0 C^{(\alpha')}(f)) \qquad \text{[by induction hypothesis (ii)]} \\ &\leqslant_0 C(C^{(\alpha')}(f)) \qquad \text{[by Property 2]} \\ &= C^{(\alpha)}(f). \end{aligned}$$

(iii) If $\beta \prec_x \alpha$ then $\beta \preccurlyeq_x \alpha'$. Thus

$$\begin{aligned} C^{(\beta)}(f) &\leqslant_x C^{(\alpha')}(f) \qquad \text{[by induction hypothesis (iii)]} \\ &\leqslant_0 C(C^{(\alpha')}(f)) \qquad \text{[by Property 2]} \\ &= C^{(\alpha)}(f). \end{aligned}$$

$\lambda = \sup(\lambda_x)$

(i) Let $y < z$. Then

$$\begin{aligned} C^{(\lambda)}(f)(y) &= C^{(\lambda_y)}(f)(y) \\ &< C^{(\lambda_y)}(f)(z) \\ &\leqslant C^{(\lambda_z)}(f)(z) \qquad \text{[by i.h. (iii), since } \lambda_y \preccurlyeq_z \lambda_z\text{]} \\ &= C^{(\lambda)}(f)(z). \end{aligned}$$

(ii) Let $f \leqslant_x g$ and $x \leqslant z$. Then

$$\begin{aligned} C^{(\lambda)}(f)(z) &= C^{(\lambda_z)}(f)(z) \\ &\leqslant C^{(\lambda_z)}(g)(z) \qquad \text{[by induction hypothesis (ii) for } \lambda_z\text{]} \\ &= C^{(\lambda)}(g)(z). \end{aligned}$$

By induction hypothesis (ii), $f(y) \leqslant C^{(\lambda_y)}(f)(y)$, thus $f \leqslant_0 C^{(\lambda)}(f)$.

(iii) If $\beta \prec_x \lambda$ then $\beta \preccurlyeq_x \lambda_x$, so $\beta \preccurlyeq_z \lambda_z$ for $z \geqslant x$ also. Thus $C^{(\beta)}(f)(z) \leqslant C^{(\lambda)}(f)(z)$ by induction hypothesis (iii) for λ_z. ∎

LEMMA 2.21 (J, K Lemma). *If $J, K: \mathrm{SI}(\mathbb{N}^{\mathbb{N}}) \to \mathrm{SI}(\mathbb{N}^{\mathbb{N}})$ are positive increasing, and also for some $x \in \mathbb{N}$, $J(f) \leqslant_x K(f)$ for all $f \in \mathrm{SI}(\mathbb{N}^{\mathbb{N}})$, then whenever $\alpha \in \Omega^s$, $J^{(\alpha)}(f) \leqslant_x K^{(\alpha)}(f)$ for all $f \in \mathrm{SI}(\mathbb{N}^{\mathbb{N}})$.*

Proof. By simple induction over Ω^s. ▌

Note 2.22. The functionals It and "Apply to f" are positive increasing. We may deduce the following:

LEMMA 2.23 (Majorisation properties). *Let $\alpha \preccurlyeq_x \beta \in \Omega^s$ and let $g, f: \mathbb{N} \to \mathbb{N}$ be s.i. (s.i.p.) with $g \leqslant_x f$. Then:*

(i) *$H_\alpha(f)$ is s.i. (s.i.p. if $0 \prec \alpha$); $F_\alpha(f)$ is s.i. (s.i.p.)*

(ii) *$H_\alpha(f) \leqslant_x H_\beta(f)$; $F_\alpha(f) \leqslant_x F_\beta(f)$ (if $x > 0$)*

(iii) *$H_\alpha(g) \leqslant_x H_\alpha(f)$; $F_\alpha(g) \leqslant_x F_\alpha(f)$*

(iv) *$H_\alpha(f) \leqslant_1 F_\alpha(f)$.*

Proof. (i) and (ii) are immediate from Lemma 2.20.

The second part of (iii) is immediate from Lemma 2.20 and the first is immediate from Lemma 2.20 and the J, K Lemma.

(iv) is easily proved by induction on α. ▌

DEFINITION 2.24 (Representability of subsets of Ω^s). Let A be a subset of Ω^s which is downward-closed. A is Turing Machine (TM)-representable if the following conditions hold for every $\alpha \in A$:

(1) There is a uniform way of representing every $\beta \preccurlyeq \alpha$ on Turing Machine tape by a tape word $[\beta]_\alpha$. We denote the length of this word by $|\beta|$ when α is understood.

(2) There is an elementary function r_α such that $[P_x(\beta)]_\alpha$ is computable from $[x]$ and $[\beta]_\alpha$ within space bounded by $H_\beta(r_\alpha)(|x| + |\beta|)$. We will always assume that r_α is s.i.p. We denote by space $(P_x(\beta))$ the space required to compute $[P_x(\beta)]_\alpha$ from $[\beta]_\alpha$ and $[x]$.

DEFINITION 2.25. Let $A \subseteq \Omega$ and $\beta \in \Omega$ with $1 \prec \beta$.

(1) $\beta \cdot A :=$ the closure of $\{\beta \cdot \alpha : \alpha \in A\} \cup \{1\}$ under $+$.

(2) $\beta^A :=$ the closure of $\{\beta^\alpha : \alpha \in A\} \cup \{0\}$ under $+$.

LEMMA 2.26. *If $A \subseteq \Omega^s$ is downward-closed, then*

(1) *$\omega \cdot A \subseteq \Omega^s$ and is downward-closed.*

(2) *$\omega^A \subseteq \Omega^s$ and is downward-closed.*

LEMMA 2.27 (Representability Lemma). *If $A \subseteq \Omega^s$ is representable then*

(i) *$\omega \cdot A$ is representable.*

(ii) *ω^A is representable.*

Proof of (ii). Suppose A is representable. By Lemma 2.26, ω^A is downward-closed. Let $\gamma \in \omega^A$. Then if $\gamma = 0$ both the conditions of representability are clearly satisfied. If $\gamma \neq 0$ then $\gamma = \sum_{j=1}^{k} \omega^{\alpha_j}$ for $\alpha_1, ..., \alpha_k \in A$. We write α_k as α. Choose a coding scheme $[\_]_\gamma$ for $\delta \leqslant \gamma$ as follows:

Let $*$ be a new tape symbol. Then $[0]_\gamma$ is $*$ and

$$[\gamma]_\gamma \text{ is } ** [\alpha_1]_{\alpha_1} * \cdots * [\alpha]_\alpha.$$

The coding schemes $[\_]_{\alpha_j}$ exist by virtue of the representability of A. If $0 \prec \delta \prec \gamma$ then $\delta = \sum_{j=1}^{l} \omega^{\alpha_j} + \sum_{j=1}^{m} \omega^{\beta_j}$ where $k > l \geqslant 0$, $m \geqslant 0$ and $\beta_1, ..., \beta_m \leqslant \alpha_{l+1}$. Then

$$[\delta]_\gamma \text{ is } ** [\alpha_1]_{\alpha_l} * \cdots * [\alpha_l]_{\alpha_l} * [\beta_1]_{\alpha_{l+1}} * \cdots * [\beta_m]_{\alpha_{l+1}}.$$

Thus condition (1) is established for ω^A. To establish (2) we consider only the case of computing predecessors-at-x of γ, the proof being almost identical for $\delta \prec \gamma$.

If γ is non-limit then $[\gamma]_\gamma = s * [0]_\alpha$ and $[P_x(\gamma)]_\gamma = s$ for some string s, so condition (2) is obviously satisfied. So suppose γ is a limit. Now

(i) $P_x(\alpha + \beta) = \alpha + P_x(\beta)$ if $0 \prec \beta$

(ii) $P_x(\alpha \cdot \beta) = \alpha \cdot P_x(\beta) + P_x(\alpha)$ if $0 \prec \beta$

(iii) $P_x(\alpha^\beta) = \sum_{k=1}^{G_\beta(x)} \alpha^{P_x^k(\beta)} \cdot P_x(\alpha)$ if $0 \prec \beta$ and $1 \prec \alpha$;

therefore

$$P_x(\gamma) = \sum_{j=1}^{k-1} \omega^{\alpha_j} + \omega^{P_x(\alpha)} \cdot x + \omega^{P_x^2(\alpha)} \cdot x + \cdots + \omega^{P_x^n(\alpha)} \cdot x,$$

where $n = G_\alpha(x)$, and therefore $\omega^{P_x^n(\alpha)} \cdot x = x$. Hence

$$* [P_x(\gamma)]_\gamma = \left[\sum_{j=1}^{k-1} \omega^{\alpha_j}\right]_\gamma * \underbrace{[P_x(\alpha)]_\alpha * \cdots * [P_x(\alpha)]_\alpha}_{x}$$

$$* \underbrace{[P_x^2(\alpha)]_\alpha * \cdots * [P_x^2(\alpha)]_\alpha}_{x} * \cdots * \underbrace{[0]_\alpha * \cdots * [0]_\alpha}_{x}.$$

We consider first the case where $x \geqslant 5$. This ensures $n \geqslant 5$ as $\gamma \notin \mathbb{N}$. From now on in the proof, we write H_α for $H_\alpha(r_\alpha)$, where space$(P_x(\beta)) \leqslant$

$H_\beta(r_\alpha)(|x|+|\beta|)$ for every $\beta \preccurlyeq \alpha$. From the expression for $[P_x(\gamma)]_\gamma$, it is clear that

$$\operatorname{space}(P_x(\gamma)) \leqslant \left|\sum_{j=1}^{k-1} \omega^{\alpha_j}\right| + x \cdot \sum_{j=1}^{n} \operatorname{space}(P_x^j(\alpha)).$$

Since A is representable,

$$\operatorname{space}(P_x(P_x^{j-1}(\alpha))) \leqslant H_{P_x^{j-1}(\alpha)}(|x| + |P_x^{j-1}(\alpha)|) \qquad [0 < j \leqslant n].$$

Now $|P_x^{j-1}(\alpha)| \leqslant \operatorname{space}(P_x^{j-1}(\alpha))$, so from $(*)$,

$$\begin{aligned}
&\operatorname{space}(P_x(\gamma))\\
&\quad \leqslant |\gamma| + x \cdot H_\alpha(|x|+|\gamma|) + x \cdot H_{P_x(\alpha)}(|x| + H_\alpha(|x|+|\gamma|))\\
&\qquad \vdots\\
&\qquad + x \cdot H_{P_x^{n-1}(\alpha)}(|x| + H_{P_x^{n-2}(\alpha)}(|x| + \cdots + H_{P_x(\alpha)}(|x| + H_\alpha(|x|+|\gamma|))\cdots))\\
&\quad \leqslant |\gamma| + x \cdot H_{P_x^{n-1}(\alpha)}(4 \cdot H_{P_x^{n-2}(\alpha)}(\cdots(4 \cdot H_{P_x(\alpha)}(4 \cdot H_\alpha(|x|+|\gamma|)))\cdots)).
\end{aligned}$$

Now $H_\beta \leqslant_1 F_\beta$ for $\beta \in \Omega^s$ and, if we assume that $\lambda z \cdot 4z \leqslant_0 r_\alpha$, we have $\lambda z \cdot 4z \leqslant_0 F_{P_x^j(\alpha)}$ for $1 \leqslant j < n$; thus

$$\begin{aligned}
\operatorname{space}(P_x(\gamma)) &\leqslant |\gamma| + x \cdot F_{P_x^{n-1}(\alpha)} F^3_{P_x^{n-2}(\alpha)} \cdots F^2_{P_x(\alpha)} H_\alpha(y)\\
&\qquad\qquad [\text{where } y := |x| + |\gamma|]\\
&\leqslant |\gamma| + x \cdot F_{P_x^{n-1}(\alpha)} F^3_{P_x^{n-2}(\alpha)} \cdots F^2_{P_x(\alpha)} F^2_{P_y(\alpha)}(y)\\
&\leqslant |\gamma| + F^x_{P_x^{n-1}(\alpha)} F_{P_x^{n-1}(\alpha)} F^3_{P_x^{n-2}(\alpha)} \cdots F^2_{P_x(\alpha)} F^2_{P_y(\alpha)}(y)\\
&\leqslant |\gamma| + F_{P_x^{n-2}(\alpha)} F^3_{P_x^{n-2}(\alpha)} \cdots F^2_{P_x(\alpha)} F^2_{P_y(\alpha)}(y)\\
&\leqslant |\gamma| + F_{P_x^{n-3}(\alpha)} \cdots F^2_{P_x(\alpha)} F^2_{P_y(\alpha)}(y)\\
&\ \ \vdots\\
&\leqslant |\gamma| + F^3_{P_x(\alpha)} F^2_{P_y(\alpha)}(y)\\
&\leqslant |\gamma| + F^5_{P_y(\alpha)}(y)\\
&\leqslant F^6_{P_y(\alpha)}(y)\\
&\leqslant F_\alpha(y).
\end{aligned}$$

But $F_\alpha(y) = H_{\omega^\alpha}(|x|+|\gamma|) \leqslant H_\gamma(|x|+|\gamma|)$ since $\gamma = (\sum_{j=1}^{k-1} \omega^{\alpha_j}) + \omega^\alpha$. So we have established the result for $x \geqslant 5$. If we set $r := \lambda y \cdot [\operatorname{space}(P_5(\gamma)) + r_\alpha(y)]$ we now have that for each $x \in \mathbb{N}$,

$$\operatorname{space}(P_x(\gamma)) \leqslant H_\gamma(r)(|x|+|\gamma|). \quad ∎$$

The proof of (i) is similar but easier.

LEMMA 2.28 (Exponent Lemma). *Let $f\colon \mathbb{N} \to \mathbb{N}$ be s.i.p. Then using Facts* 2.18,

(i) *If $2 \preccurlyeq_x \alpha$ and $2 \preccurlyeq \beta$ for $\alpha, \beta \in \Omega^s$, then*

$$H_{\alpha+\beta}(f) \leqslant_x H_{\alpha\cdot\beta}(f).$$

(ii) *If $A \subseteq \Omega^s$ is downward-closed and closed under $+$, and $\alpha \in \omega^A$, then*

$$H_\alpha(f) \leqslant_1 H_{\omega^\gamma}(f)$$

for some $\gamma \in A$ with $2 \preccurlyeq_1 \omega^\gamma$.

(iii) *If $1 \in A \subseteq \Omega^s$ and A is downward-closed and closed under $+$, then $H(\omega^A)$ satisfies the following closure condition for multiplication: For every α, $\beta \in \omega^A$ with $0 \prec \alpha$, there is some $\gamma \in \omega^A$ such that, for any s.i.p. $f\colon \mathbb{N} \to \mathbb{N}$,*

$$H_{\alpha\cdot\beta}(f) \leqslant_0 H_\gamma(f).$$

LEMMA 2.29 (Honesty Lemma). *The content of this lemma is essentially that the functional H_α is honest, i.e., that whenever $\alpha \in A \subseteq \Omega^s$ and A is representable, then the space required to compute $H_\alpha(f)$ is of the order $H_\alpha(f)$. The equation $S_{H_\alpha(f)} = H_\alpha(f)$ cannot hold strictly, however, for we must take account of the computation of the predecessors of α as well as of the computation of f, which will not in general be bounded by f itself. The actual result is as follows:*

Suppose $f\colon \mathbb{N}^k \to \mathbb{N}^k$ with $f \in \mathrm{SPACE}(h)$. Then for any $\mathbf{a} \in \mathbb{N}^k$ and $\alpha \in A$,

$$S_{H_\alpha(f)}(\mathbf{a}) \leqslant H_{\alpha'}(h')(|\alpha| + |\mathbf{a}|),$$

where $\alpha' := (1+\alpha)\cdot\alpha$ and $h' := h \circ r$ for some elementary r (in general dependent on α and f).

Proof. We first fix some $\gamma \in A$ and then prove by induction on $\alpha \preccurlyeq \gamma$ that

$$S_{H_\alpha(f)}(\mathbf{a}) \leqslant H_\alpha(H_{1+\alpha}(h \circ r))(|\alpha| + |\mathbf{a}|)$$

for some elementary r.

Let r_1 be the elementary s.i.p. function r_γ from condition (2) of the definition of representability. Let r_2 be an elementary s.i.p. function for which $S_f(\mathbf{a}) \leqslant h(r_2(|\mathbf{a}|))$. Now let r be elementary and s.i.p. such that $r_1 \leqslant_0 r$, $r_2 \leqslant_0 r$. Clearly, $\mathrm{space}(P_x(\alpha)) \leqslant H_\alpha(r)(|\alpha| + |x|)$ for all $\alpha \preccurlyeq \gamma$ and $x \in \mathbb{N}$. Now let $\mathbf{a} = \langle a_1, ..., a_k \rangle \in \mathbb{N}^k$.

If $\alpha = 0$ then $S_{H_\alpha(f)}(\mathbf{a}) \leqslant |0| + |\mathbf{a}| = H_0(H_1(h \circ r))(|0| + |\mathbf{a}|)$.

Suppose $\alpha \succ 0$. Let $\eta = P_{a_1}(\alpha)$ and let Q be the space required to compute $\langle [\eta]_\gamma, [f(\mathbf{a})] \rangle$ from $\langle [\alpha]_\gamma, [\mathbf{a}] \rangle$. Let space$(P_{\mathbf{a}}(\alpha))$ denote the space required to compute $\langle [\eta]_\gamma, [\mathbf{a}] \rangle$ from $\langle [\alpha]_\gamma, [\mathbf{a}] \rangle$. Then

$$\begin{aligned} \text{space}(P_{\mathbf{a}}(\alpha)) &\leqslant \text{space}(P_{a_1}(\alpha)) + |\langle a_2, ..., a_k \rangle| \\ &\leqslant H_\alpha(r)(|\alpha| + |a_1|) + |\langle a_2, ..., a_k \rangle| \\ &\leqslant H_\alpha(r)(|\alpha| + |\mathbf{a}|) \qquad [\text{as } H_\alpha(r) \text{ is s.i.}] \\ &\leqslant H_\alpha(h \circ r)(|\alpha| + |\mathbf{a}|). \end{aligned}$$

Thus

$$\begin{aligned} Q &\leqslant \max\{\text{space}(P_{\mathbf{a}}(\alpha)), |\eta| + S_f(\mathbf{a})\} \\ &\leqslant \max\{\text{space}(P_{\mathbf{a}}(\alpha)), |\eta| + h(r(|\mathbf{a}|))\} \\ &\leqslant \max\{\text{space}(P_{\mathbf{a}}(\alpha)), h(r(|\eta| + |\mathbf{a}|))\} \\ &\leqslant h(r(\text{space}(P_{\mathbf{a}}(\alpha)))) \qquad [\text{as } |\eta| + |\mathbf{a}| \leqslant \text{space}(P_{\mathbf{a}}(\alpha))] \\ &\leqslant ((h \circ r) \circ H_\alpha(h \circ r))(|\alpha| + |\mathbf{a}|) \\ &= H_{1+\alpha}(h \circ r)(|\alpha| + |\mathbf{a}|). \end{aligned}$$

By the induction hypothesis,

$$\begin{aligned} S_{H_\eta(f)}(f(\mathbf{a})) &\leqslant H_\eta(H_{1+\eta}(h \circ r))(|\eta| + |\mathbf{a}|) \\ &\leqslant H_\eta(H_{1+\eta}(h \circ r))(Q) \\ &\leqslant H_\eta(H_{1+\alpha}(h \circ r))(H_{1+\alpha}(h \circ r)(|\alpha| + |\mathbf{a}|)) \\ &\qquad [\text{as } 1 + \eta \leqslant_Q 1 + \alpha] \\ &= H_\alpha(H_{1+\alpha}(h \circ r))(|\alpha| + |\mathbf{a}|). \end{aligned}$$

Now $H_\alpha(H_{1+\alpha}(h \circ r)) = H_{(1+\alpha)\cdot\alpha}(h \circ r)$ and this completes the proof of the Honesty Lemma. ▌

3. Terminating Recursive and WHILE-Program Schemes—Main Results

Definition 3.1. Let $A \subseteq \Omega^s$ be downward-closed and let $\mathscr{U}$ be a set of number-theoretic functions. We say that $f\colon A \times \mathbb{N}^k \to \mathbb{N}$ is definable from $\mathscr{U}$ by A-recursion if f satisfies the scheme

$$f(\beta, \mathbf{x}) := \begin{cases} \text{if } \beta = 0 & \text{then } v(\mathbf{x}) \\ & \text{else } T(\mathbf{x}, \mathscr{U}, f_{P(\beta)}), \end{cases}$$

where $v \in U$ and $T(\mathbf{x}, \mathscr{U}, f_{P(\beta)})$ is a term built up by explicit definitions from the number-theoretic functions $\mathscr{U}$ and $f_{P(\beta)}$, where

$$f_{P(\beta)}(s, t_1, ..., t_k) := f(P_s(\beta), t_1, ..., t_k).$$

DEFINITION 3.2. If $f\colon A \times \mathbb{N}^l \to \mathbb{N}$ is definable from $\mathscr{U}$ by A-recursion, then f is a total function, as the recursion is well-founded. A *number-theoretic* function g is definable by A-recursion if $g = f(\beta) := \lambda \mathbf{x} \cdot f(\beta, \mathbf{x})$ for some $\beta \in A$, where $f\colon A \times \mathbb{N}^l \to \mathbb{N}$ is definable by A-recursion as in the preceding definition.

DEFINITION 3.3 (Recursive Programs over Subsets of Ω^s). For $A \subseteq \Omega^s$, we denote by $\mathrm{REC}(A, \mathscr{U})$ the least class of number-theoretic functions containing $\mathscr{U}$ and closed under explicit definitions and A-recursion. We write $\mathrm{REC}(A)$ for $\mathrm{REC}(A, \mathscr{E}^3)$.

DEFINITION 3.4 (WHILE-Programs over Subsets of Ω^s). Let $A \subseteq \Omega^s$. The set TWPR of terminating WHILE-programs is a restricted subclass of WHILE-programs defined as follows:

Terms are of two kinds. Terms of type $\mathbb{N}$ are built up from 0 and a set V of variables v intended to range over natural numbers, by applying succ (successor) and pred (predecessor). Terms of type Ω^s are built up from 0 and a set U of variables u intended to range over Ω^s, by applying P_v.

The TWPR-programs S are generated by:

(i) "**skip**" $\in$ TWPR

(ii) Assignments. If t is a term of type $\mathbb{N}$ then for $v \in V$, '$v := t$' $\in$ TWPR.

(iii) Conditionals. If t_1 and t_2 are terms of type $\mathbb{N}$ and $S_0, S_1 \in$ TWPR then

"**if** $t_1 = t_2$ **then** S_0 **else** S_1 **fi**" $\in$ TWPR.

(iv) Sequencing. If S_0 and $S_1 \in$ TWPR then '$S_0; S_1$' $\in$ TWPR.

(v) WHILE-loops. If $S_0 \in$ TWPR and $v \in V$ and $u \in U$ then

"**while** $u \neq 0$ **do** $u := P_v(u)$; S_0 **od**" $\in$ TWPR

provided that the variable u does not occur in S_0.

DEFINITION 3.5 (Semantics of Terminating WHILE-Programs). Each TWPR-program S defines a function $[\![S]\!]\colon \Omega^U \times \mathbb{N}^V \to \Omega^U \times \mathbb{N}^V$ as follows (where $[\![t]\!](\tau)$ denotes the standard numerical value of t under assignment τ):

(i) $[\![\mathbf{skip}]\!](\sigma, \tau) := (\sigma, \tau)$.

(ii) Assignments.

$$[\![v := t]\!](\sigma, \tau) := (\sigma, \tau[[\![t]\!](\tau)/v]).$$

(iii) Conditionals. If S is the program "**if** $t_1 = t_2$ **then** S_0 **else** S_1 **fi**" then

$$[\![S]\!](\sigma, \tau) := \begin{cases} [\![S_0]\!](\sigma, \tau), & \text{if } [\![t_1]\!](\tau) = [\![t_2]\!](\tau); \\ [[\![S_1]\!](\sigma, \tau), & \text{otherwise.} \end{cases}$$

(iv) Sequencing.

$$[\![S_0; S_1]\!](\sigma, \tau) := [\![S_1]\!]([\![S_0]\!](\sigma, \tau)).$$

(v) WHILE-loops. If S is the program

"**while** $u \neq 0$ **do** $u := P_v(u)$; S_0 **od**"

then, in accord with usual WHILE-program semantics,

$$[\![S]\!](\sigma, \tau) := \begin{cases} (\sigma, \tau) & \text{if } \sigma(u) = 0; \\ [\![S_0; S]\!](\sigma[P_{\tau(v)}(\sigma(u))/u], \tau) & \text{otherwise.} \end{cases}$$

DEFINITION 3.6 (T-WHILE(A) for subsets A of Ω^s). Given $A \subseteq \Omega^s$ we say that a function h: $\mathbb{N}^m \to \mathbb{N}^n$ is definable from parameters $\alpha_1, ..., \alpha_k \in A$ by a TWPR-program S with respect to input variables $\mathbf{u} = u_1, ..., u_k$ and $\mathbf{v} = v_1, ..., v_m$ and output variables $\mathbf{w} = w_1, ..., w_n$ if for all $x_1, ..., x_m$,

$$h(x_1, ..., x_m) = \langle [\![S]\!](\sigma, \tau)(w_j) \rangle_{j=1,...,n},$$

where $\sigma(u) = \alpha_i$ if $u = u_i$ and $\sigma(u) = 0$ otherwise; $\tau(v) = x_i$ if $v = v_i$ and $\tau(v) = 0$ otherwise.

T-WHILE(A) is the class of all such (vector-valued) functions definable from parameters in A by terminating WHILE-programs as above. Normally, however, we will only be concerned with functions taking values in $\mathbb{N}$, i.e., where $n = 1$.

DEFINITION 3.7 (HARDY(A) for subsets A of Ω^s). Let $A \subseteq \Omega^s$. The set HARDY(A) is defined to be the closure of $\mathcal{E}^3$ under (vectorised) explicit definitions, primitive recursion, and Hardy functionals $\{H_\alpha(\_)$: $\alpha \in A\}$ given as follows: Let f: $\mathbb{N}^k \to \mathbb{N}^k$. Then for a fixed i between 1 and k and all $\mathbf{a} \in \mathbb{N}^k$ with ith component b,

$$H_\alpha(f)(\mathbf{a}) := \begin{cases} \mathbf{a} & \text{if } \alpha = 0; \\ H_{P_b(\alpha)}(f)(f(\mathbf{a})) & \text{otherwise.} \end{cases}$$

With $i = k = 1$ we get the original Hardy functionals.

Again, we will normally be concerned with the non-vectorised version, but note that the vectorised versions of these two definitions are important, as we could not establish the connection between T-WHILE(A) and HARDY(A) without them.

We are now in a position to state the main theorems of this paper.

DEFINITION 3.8 (Closure conditions). If $A \subseteq \Omega^s$ satisfies

(i) $1 \in A$

(ii) A is downward-closed and closed under $+$

(iii) A is representable

then we say A satisfies the first closure condition. If in addition A satisfies

(iv) for every $\alpha, \beta \in A$ there exists $\gamma \in A$ such that $H_{\alpha \cdot \beta} \leqslant_0 H_\gamma$, and

(v) $\omega \in A$.

then we say A satisfies the second closure condition.

Note 3.9. By the corollary to the Exponent Lemma, and the Representability Lemma, if A satisfies the first closure condition, then ω^A satisfies the second closure condition.

THEOREM I. *If $\omega \in A \subseteq \Omega^s$, then*

$$T\text{-WHILE}(A) \equiv \text{HARDY}(A).$$

THEOREM II. *If $A \subseteq \Omega^s$ satisfies the second closure condition, then*

$$T\text{-WHILE}(A) \equiv \text{HARDY}(A) \equiv \text{ELEM}(H(A)) \equiv \text{SPACE}(H(A)).$$

THEOREM III. *If $A \subseteq \Omega^s$ satisfies the first closure condition, then*

$$\text{REC}(\omega \cdot A) \equiv T\text{-WHILE}(\omega^A).$$

4. PROOF OF THEOREMS I AND II

We first introduce some notation to be used only for the lemmas in this section. We denote by $[\![\langle \mathbf{v}, \mathbf{w}, S \rangle]\!]$ the function defined from fixed parameters in A by the program S with respect to input numerical variables $\mathbf{v}$ and output variables $\mathbf{w}$. Two easy lemmas give us the proof of Theorem I.

LEMMA 4.1. WHILE(A) $\subseteq$ HARDY(A).

Proof. Since HARDY(A) is closed under explicit definitions and vectorization, it is enough to show that whenever $S \in \mathrm{TWPR}$ and all numerical variables of S occur in $\mathbf{v}$ then $[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!] \in \mathrm{HARDY}(A)$. This can readily be shown by induction on the structure of TWPR, as per Definition 3.4.

(i) id: $\mathbb{N}^k \to \mathbb{N}^k = \langle \Pi_1^k, ..., \Pi_k^k\rangle \in \mathrm{HARDY}(A)$.

(ii) $[\![t]\!] = f \circ \Pi_j^k$, where f is one of the functions

(1) $\lambda x \cdot [x+m]$

(2) $\lambda x \cdot [x \dot{-} m]$

(3) $\lambda x \cdot [m]$,

each of which is in HARDY(A). Thus

$$[\![\langle \mathbf{v}, \mathbf{v}, u_i := t\rangle]\!] = \langle \Pi_1^k, ..., \Pi_{i-1}^k, f \circ \Pi_j^k, \Pi_{i+1}^k, ..., \Pi_k^k\rangle \in \mathrm{HARDY}(A).$$

(iii) The function

$$h(\mathbf{x}) := \begin{cases} f_1(\mathbf{x}) & \text{if } g_1(\mathbf{x}) = g_2(\mathbf{x}); \\ f_2(\mathbf{x}) & \text{otherwise} \end{cases}$$

is definable using primitive recursion from f_1, f_2, g_1, and g_2, and thus if $[\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!] \in \mathrm{HARDY}(A)$, $[\![\langle \mathbf{v}, \mathbf{v}, S_1\rangle]\!] \in \mathrm{HARDY}(A)$, and S is the program "**if** $t_1 = t_2$ **then** S_0 **else** S_1 **fi**," then $[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!] \in \mathrm{HARDY}(A)$.

(iv) Clear from the fact that HARDY(A) is closed under composition.

(v) Suppose S is the program "**while** $u \neq 0$ **do** $u := P_{v_i}(u)$; S_0 **od**" and suppose S_0 computes the function f. Then if u is assigned the value $\alpha \in A$, it is clear from the definition of $[\![S]\!]$ that an induction on α gives

$$[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!] = H_\alpha(f),$$

since for all $\mathbf{a} \in \mathbb{N}^m$ and non-zero α, if b is the ith component of $\mathbf{a}$,

$$\begin{aligned} [\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!](\alpha, \mathbf{a}) &= [\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!](P_b(\alpha)), [\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!](\mathbf{a})) \\ &= [\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!](P_b(\alpha)), f(\mathbf{a})) \\ &= H_{P_b(\alpha)}(f(\mathbf{a})) \\ &\qquad \text{[by induction hypothesis]} \\ &= H_\alpha(\mathbf{a}). \quad \blacksquare \end{aligned}$$

LEMMA 4.2. *If* $\omega \in A$ *then* HARDY(A) $\subseteq$ WHILE(A).

Proof. It is easy to construct function programs which compute the successor, constant, and projection functions. Vectorisation and explicit definitions can be handled by a suitable combination of sequencing and the use of auxiliary variables and initializing variables to zero.

To deal with the Hardy functionals, suppose that $f\colon \mathbb{N}^k \to \mathbb{N}^k \in$ HARDY(A) and $\mathbf{v}$, $\mathbf{w}$ are disjoint lists of variables such that $f = [\![\langle \mathbf{v}, \mathbf{w}, S\rangle]\!]$ for some program S. Let $\mathbf{x}$ be the list of variables of S not occurring in $\mathbf{v}$. Let $\mathbf{w} := \mathbf{v}$, $\mathbf{x} := 0$ stand for the programs "$w_1 := v_1$; $w_2 := v_2$; ...; $w_m := v_m$" and "$x_1 := 0$; ...; $x_n := 0$," respectively. Let T be the program

"$\mathbf{w} := \mathbf{v}$; **while** $u \neq 0$ **do** $u := P_{v_1}(u)$; $\mathbf{x} := 0$; S; $\mathbf{v} := \mathbf{w}$ **od**."

Then $[\![\langle \mathbf{v}, \mathbf{w}, T\rangle]\!](\alpha) = H_\alpha(f)$.

Finally, provided $\omega \in A$, we can handle primitive recursion by using the construction "**do** y **times** S **od**," which can be defined by the following loop where the Ω variable u is assigned value ω:

```
"if $y = 0$
 then skip
 else $y := \mathrm{pred}(y)$;
    while $u \neq 0$ do $u := P_y(u)$; $S$ od;
    $y := \mathrm{succ}(y)$
 fi."
```

For if $\mathbf{v}' := \langle y, v_1, ..., v_m\rangle$ and $g := [\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!]$ then

$$[\![\langle \mathbf{v}', \mathbf{v}, \textbf{do } y \textbf{ times } S \textbf{ od}\rangle]\!] = \lambda z, \mathbf{x} \cdot [g^{(z)}(x)];$$

once we can iterate functions in this way we can do primitive recursion. ∎

LEMMA 4.3. *Provided A satisfies the second closure condition* (*see Definition* 3.8), SPACE($H(A)$) $\subseteq$ ELEM($H(A)$) $\subseteq$ HARDY(A) = WHILE(A).

Proof. The first containment follows from the well-known result that if $h \in$ SPACE(f), then h is elementary in f. The second containment follows from the facts that $\mathscr{E}^3 \subseteq$ HARDY(A) and HARDY(A) is closed under explicit definitions and primitive recursion (and thus under *limited* recursion). ∎

LEMMA 4.4. *Provided A satisfies the second closure condition,* WHILE(A) $\subseteq$ SPACE($H(A)$).

Proof. It is enough to show that for every $S \in \text{TWPR}$ and $\mathbf{v}$ containing the variables of S,

$$[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!] \in \text{SPACE}(H_\alpha) \qquad \text{for some} \quad \alpha \in A.$$

This we do by induction on TWPR.

(i) $[\![\langle \mathbf{v}, \mathbf{v}, \mathbf{skip}\rangle]\!](\mathbf{a}) = \mathbf{a}$, the computation of which is bounded by $H_0(|\mathbf{a}|)$, $0 \in A$.

(ii) The computation

$$[\mathbf{a}] \to [[\![\langle \mathbf{v}, \mathbf{v}, v_i := t\rangle]\!](\mathbf{a})]$$

is clearly bounded by $|\mathbf{a}| + m$ for some finite $m \in A$.

(iii) Suppose S is the program

$$\text{“}\mathbf{if}\ t_0 = t_1\ \mathbf{then}\ S_0\ \mathbf{else}\ S_1\ \mathbf{fi}.\text{”}$$

We may suppose inductively that the computations $[\mathbf{a}] \to [[\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!](\mathbf{a})]$ and $[\mathbf{a}] \to [[\![\langle \mathbf{v}, \mathbf{v}, S_1)]\!](\mathbf{a})]$ are bounded by $H_{\alpha_0}(|\mathbf{a}|)$ and $H_{\alpha_1}(|\mathbf{a}|)$, respectively. Let $f_0 := [\![t_0]\!]$ and $f_1 := [\![t_1]\!]$. $[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!](\mathbf{a})$ is computed by the sequence

$$\begin{aligned}
[\mathbf{a}] &\to \langle [\mathbf{a}], [\langle a_1, ..., a_{i-1}, f_0(\mathbf{a}), a_{i+1}, ..., a_n\rangle]\rangle \\
&\to \langle [\mathbf{a}], [\langle a_1, ..., a_{i-1}, f_0(\mathbf{a}), a_{i+1}, ..., a_n\rangle], \\
&\qquad [\langle a_1, ..., a_{j-1}, f_1(\mathbf{a}), ..., a_n\rangle]\rangle \\
&\to \begin{cases} \langle [\mathbf{a}], [[\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!](\mathbf{a})\rangle \text{ or} \\ \langle [\mathbf{a}], [[\![\mathbf{v}, \mathbf{v}, S_1\rangle]\!](\mathbf{a})]\rangle. \end{cases}
\end{aligned}$$

By case (ii), the computations $[\mathbf{a}] \to \langle [\mathbf{a}], [a_1, ..., a_{i-1}, f_0(\mathbf{a}), a_{i+1}, ..., a_n\rangle]\rangle$ and $[\mathbf{a}] \to \langle [\mathbf{a}], [\langle a_1, ..., a_{i-1}, f_1(\mathbf{a}), a_{i+1}, ..., a_n\rangle]\rangle$ are both bounded by $H_m(|\mathbf{a}|)$ for some $m \in \mathbb{N}$. Thus the computation $[\mathbf{a}] \to [[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!](\mathbf{a})]$ is bounded by

$$\begin{aligned}
&2 \cdot H_m(|\mathbf{a}|) + H_{\alpha_0}(|\mathbf{a}|) + H_{\alpha_1}(|\mathbf{a}|) \\
&\quad \leqslant H_m^2(H_{\alpha_0}(H_{\alpha_1}(4|\mathbf{a}|))) \\
&\quad \leqslant H_m^2(H_{\alpha_0}(H_{\alpha_1}(H_\omega^2(|\mathbf{a}|)))) \\
&\quad = H_{m \cdot 2 + \alpha_0 + \alpha_1 + \omega \cdot 2}(|\mathbf{a}|) \qquad [\text{where } m \cdot 2 + \alpha_0 + \alpha_1 + \omega \cdot 2 \in A].
\end{aligned}$$

(iv) Suppose S is the program S_0; S_1.

Now $[\![\langle \mathbf{v}, \mathbf{v}, S_0; S_1\rangle]\!] = [\![\langle \mathbf{v}, \mathbf{v}, S_1\rangle]\!] \circ [\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!]$, so if $[\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!] \in \text{SPACE}(H_{\alpha_0})$ for $\alpha_0 \in A$ and $[\![\langle \mathbf{v}, \mathbf{v}, S_1\rangle]\!] \in \text{SPACE}(H_{\alpha_1})$ for $\alpha_1 \in A$ then $[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!] \in \text{SPACE}(H_{\alpha_1} \circ H_{\alpha_0}) = \text{SPACE}(H_{\alpha_1 + \alpha_0})$, where $\alpha_1 + \alpha_0 \in A$.

(v) Suppose S is the program

$$\text{“}\mathbf{while}\ u \neq 0\ \mathbf{do}\ u := P_{v_i}(u);\ S_0\ \mathbf{od}\text{”}$$

computing $[\![\langle \mathbf{v}, \mathbf{v}. S\rangle]\!]$ with u assigned the ordinal parameter $\beta \in A$. Assume inductively that $f = [\![\langle \mathbf{v}, \mathbf{v}, S_0\rangle]\!] \in \mathrm{SPACE}(H_\alpha)$ for some $\alpha \in A$. We may assume $i = 1$. Then $[\![\langle \mathbf{v}, \mathbf{v}, S\rangle]\!] = H_\beta(f)$. By the Honesty Lemma, for any $\mathbf{a} \in \mathbb{N}^k$,

$$S_{H_\beta(f)}(\mathbf{a}) \leqslant H_{\beta'}(f')(|\beta| + |\mathbf{a}|),$$

where $\beta' = (1+\beta)\cdot\beta$ and $f' = H_\alpha \circ r$ for some s.i.p. $r \in \mathscr{E}^3$. Now by the closure properties of A, r is dominated by $H_{\omega^3} \leqslant H_\delta$ for some $\delta \in A$. Thus $f' \leqslant_0 H_{\alpha+\delta}$ and we have

$$\begin{aligned} S_{H_\beta(f)}(\mathbf{a}) &\leqslant H_{(1+\beta)\cdot\beta}(H_{\alpha+\delta})(|\beta| + |\mathbf{a}|) \\ &\leqslant H_\gamma(H_{\alpha+\delta})(|\beta| + |\mathbf{a}|) \qquad [\text{for } \gamma \in A, \text{ by part (iv) of the second closure condition}] \\ &= H_{\gamma\cdot(\alpha+\delta)}(|\beta| + |\mathbf{a}|) \\ &\leqslant H_{\gamma'}(|\beta| + |\mathbf{a}|) \qquad [\text{for } \gamma' \in A, \text{ again by part (iv)}] \\ &= H_{\gamma'+|\beta|}(|\mathbf{a}|). \end{aligned}$$

This completes the proof. ∎

Theorem I, Lemma 4.3, and Lemma 4.4 together give us the proof of Theorem II.

5. Proof of Theorem III

Lemma 5.1 (Space-bounding Lemma). *Suppose $A \subseteq \Omega^s$ satisfies the first closure condition and $\beta \in A$. Suppose $f(\beta)$ is defined from $\mathscr{U}$ by A-recursion, and for each u: $\mathbb{N}^k \to \mathbb{N} \in \mathscr{U}$ there are $m \in \mathbb{N}$, v elementary and s.i.p., and $\sigma \in A$ such that for each $\mathbf{a} \in \mathbb{N}^k$,*

$$S_u(\mathbf{a}) \leqslant H_{m^\sigma}(v)(|\mathbf{a}|).$$

Then

$$S_{f(\beta)}(\mathbf{a}) \leqslant [H_{l^\alpha+\beta}(r)]^l (|\beta| + |\mathbf{a}| + l)$$

for some $l \in \mathbb{N}$, r elementary and s.i.p., and $\alpha \in A$.

Proof. First fix some $\gamma \in A$ and a coding scheme $[\_]_\gamma$ for $\{\beta : \beta \preccurlyeq \gamma\}$. Thus for $x \in \mathbb{N}$,

$$\mathrm{space}(P_x(\beta)) \leqslant H_\beta(r_\gamma)(|x| + |\beta|)$$

for some s.i.p. $r_\gamma \in \mathscr{E}^3$. We may assume that $\mathscr{U}$ is finite. Let

$$S_u(\mathbf{a}) \leqslant H_{m_u^{\sigma_u}}(v_u)(|\mathbf{a}|)$$

for each $\mathbf{a} \in \mathbb{N}^k$. Let $l := \max_{u \in U}(m_u)$ and $\alpha := \sum_{u \in U} \sigma_u$ and let r be defined by $r(x) := r_\gamma(x) + \sum_{u \in U} v_u(x)$. We assume that $\lambda y \cdot 2y \leqslant_0 r$. Clearly, r is s.i.p. and elementary and $\alpha \in A$. It is easy to check that

(i) $H_{m_u^{\sigma_u}}(v_u) \leqslant_0 H_{m_u^{\sigma_u}}(r)$ for each $u \in U$,

(ii) $H_{m_u^{\sigma_u}}(r) \leqslant_0 H_{l^{\sigma_u}}(r)$ for each $u \in U$, and

(iii) $H_{l^{\sigma_u}}(r) \leqslant_0 H_{l^\alpha}(r)$ for each $u \in U$.

Thus for each $\mathbf{a} \in \mathbb{N}^k$ and $u \in U$,

$$S_u(\mathbf{a}) \leqslant H_{l^\alpha}(r)(|\mathbf{a}|). \tag{$*$}$$

From this point in the proof onwards we adopt the convention that H_β denotes the Hardy function $H_\beta(r)$, r being henceforth fixed as above.

Suppose f is defined by

$$f(\beta, \mathbf{x}) := \begin{cases} \text{if } \beta = 0 & \text{then } v(\mathbf{x}) \\ & \text{else } T(\mathbf{x}, \mathscr{U}, f_{P(\beta)}). \end{cases}$$

We define for each subterm t of T the length $L(t)$ of t by

$$L(x_i) = L(\text{constant}) := l + 1 \tag{1}$$

$$L(u(t_1, ..., t_k)) := 1 + \sum_{i=1}^{k} L(t_i) \tag{2}$$

$$L(f(P_s(\beta), t_1, ..., t_k)) := 4 + L(s) + \sum_{i=1}^{k} L(t_i). \tag{3}$$

Our aim will be to show that for each $\mathbf{a} \in \mathbb{N}^k$,

$$S_{f(\beta)}(\mathbf{a}) \leqslant H^{L(T)}_{L(T)^{\alpha+\beta}}(|\mathbf{a}| + |\beta| + L(T)). \tag{†}$$

The proof of (†) proceeds by induction on $\beta \preccurlyeq \gamma$.

If $\beta = 0$ then

$$\begin{aligned} S_{f(\beta)}(\mathbf{a}) &= S_u(\mathbf{a}) \\ &\leqslant H_{l^\alpha}(|\mathbf{a}|) \\ &\leqslant H^{L(T)}_{L(T)^\alpha}(|\mathbf{a}| + |0| + L(T)). \end{aligned}$$

Suppose $\beta \neq 0$. We show by a sub-induction on the structure of T that for each $\mathbf{a} \in \mathbb{N}^k$ and subterm t of T,

$$S_t(\mathbf{a}) \leqslant H^{L(t)}_{L^{\alpha+\beta}}(|\mathbf{a}| + |\beta| + L),$$

where L is $L(T)$ and we are ambiguously using t to mean either a subterm or the function that it denotes. (†) then follows by putting $t = T$.
Let t be a subterm of T. There are 3 cases to consider:

Case 1. t is x_i.
Then clearly $S_{x_i}(\mathbf{a}) \leqslant |\mathbf{a}| \leqslant H^{L(t)}_{L^\alpha+\beta}(|\mathbf{a}| + |\beta| + L)$.

Case 2. t is $u(t_1, ..., t_k)$ for subterms $t_1, ..., t_k$.
Let $\mathbf{a} \in \mathbb{N}^k$ and let $\mathbf{t}(\mathbf{a}) := \langle t_1(\mathbf{a}), ..., t_k(\mathbf{a})\rangle$. Let $z := |\mathbf{a}| + |\beta| + L$. Now $|t_i(\mathbf{a})| \leqslant S_{t_i}(\mathbf{a})$ so

$$\begin{aligned} |\mathbf{t}(\mathbf{a})| &\leqslant \sum_{i=1}^{k} S_{t_i}(\mathbf{a}) \\ &\leqslant \sum_{i=1}^{k} H^{L(t_i)}_{L^\alpha+\beta}(z) \qquad \text{[by induction hypothesis for } t_i \ (1 \leqslant i \leqslant k)]. \end{aligned}$$

Next note that if $\alpha \neq 0$ and $\lambda y \cdot 2y \leqslant_0 f$ then $H^m_\alpha(f) + H^n_\alpha(f) \leqslant_0 H^{m+n}_\alpha(f)$.
We now have

$$\begin{aligned} S_t(\mathbf{a}) &\leqslant \max\left\{\sum_{i=1}^{k} S_{t_i}(\mathbf{a}), S_u(\mathbf{t}(\mathbf{a}))\right\} \\ &\leqslant \max\left\{\sum_{i=1}^{k} S_{t_i}(\mathbf{a}), H_{L^\alpha+\beta}\left(\sum_{i=1}^{k} S_{t_i}(\mathbf{a})\right)\right\} && [\text{by } (*)] \\ &= H_{L^\alpha+\beta}\left(\sum_{i=1}^{k} S_{t_i}(\mathbf{a})\right) \\ &\leqslant H_{L^\alpha+\beta}\left(\sum_{i=1}^{k} H^{L(t_i)}_{L^\alpha+\beta}(z)\right) && [\text{by } (*)] \\ &\leqslant H^{(1+\sum_{i=1}^{k} L(t_i))}_{L^\alpha+\beta}(z) && [\text{by note above}] \\ &= H^{L(t)}_{L^\alpha+\beta}(|\mathbf{a}| + |\beta| + L). \end{aligned}$$

Case 3. t is $f(P_s(\beta), t_1, ..., t_k)$.
Let $\mathbf{a} \in \mathbb{N}^k$ and let $\mathbf{t}(\mathbf{a}) := \langle t_1(\mathbf{a}), ..., t_k(\mathbf{a})\rangle$. Let $z := |\mathbf{a}| + |\beta| + L$ and let $\eta := P_{s(\mathbf{a})}(\beta)$. Then

$$S_t(\mathbf{a}) \leqslant \max\left\{S_s(\mathbf{a}) + \sum_{i=1}^{k} S_{t_i}(\mathbf{a}), S_{f(\eta)}(\mathbf{t}(\mathbf{a}\ )), \text{space}(P_s(\beta))\right\}.$$

By the induction hypothesis for $\eta \prec \beta$,

$$S_{f(\eta)}(\mathbf{t}(\mathbf{a})) \leqslant H^{L}_{L^\alpha+\eta}(|\mathbf{t}(\mathbf{a})| + |\eta| + L).$$

By the induction hypothesis for t_i $(1 \leqslant i \leqslant k)$,

$$\begin{aligned}
|\mathbf{t}(\mathbf{a})| &\leqslant \sum_{i=1}^{k} |t_i(\mathbf{a})| \\
&\leqslant \sum_{i=1}^{k} S_{t_i}(\mathbf{a}) \\
&\leqslant \sum_{i=1}^{k} H_{L^\alpha+\beta}(z) \\
&\leqslant H_{L^\alpha+\beta}^{\sum_{i=1}^{k} L(t_i)}(z).
\end{aligned}$$

By the induction hypothesis for s,

$$|s(\mathbf{a})| \leqslant S_s(\mathbf{a}) \leqslant H_{L^\alpha+\beta}^{L(s)}(z).$$

Thus

$$\begin{aligned}
|\eta| &\leqslant \operatorname{space}(P_{s(\mathbf{a})}(\beta)) \\
&\leqslant H_\beta(|s(\mathbf{a})| + |\beta|) \\
&\leqslant H_\beta(H_{L^\alpha+\beta}^{L(s)}(z) + H_{L^\alpha+\beta}(z)) \\
&\leqslant H_{L^\alpha+\beta}(H_{L^\alpha+\beta}^{L(s)+1}(z)) \\
&= H_{L^\alpha+\beta}^{L(s)+2}(z).
\end{aligned}$$

Note that $\operatorname{space}(P_s(\beta))$ is also less than or equal to $H_{L^\alpha+\beta}^{L(s)+2}(z)$.
Hence

$$\begin{aligned}
|\mathbf{t}(\mathbf{a})| + |\eta| + L &\leqslant H_{L^\alpha+\beta}^{\sum_{i=1}^{k} L(t_i)}(z) + H_{L^\alpha+\beta}^{L(s)}(z) + H_{L^\alpha+\beta}(z) \\
&\leqslant H_{L^\alpha+\beta}^{(3+L(s)+\sum_{i=1}^{k} L(t_i))}(z) \\
&=: Q, \qquad \text{say.}
\end{aligned}$$

Now $s(\mathbf{a}) \leqslant Q$, so $\alpha + \eta = P_{s(\mathbf{a})}(\alpha+\beta) \leqslant_Q P_Q(\alpha+\beta) = \alpha + P_Q(\beta)$. Thus $H_{L^\alpha+\eta} \leqslant_Q H_{L^\alpha+P_Q(\beta)}$ on applying Lemma 2.20 to $H_L(\_)$. Hence by the induction hypothesis for η,

$$\begin{aligned}
S_{f(\beta)}(\mathbf{t}(\mathbf{a})) &\leqslant H_{L^\alpha+\eta}^{L}(Q) \\
&\leqslant H_{L^\alpha+P_Q(\beta)}^{L}(Q) \\
&= H_{L^\alpha+\beta}(Q) \\
&= H_{L^\alpha+\beta}(H_{L^\alpha+\beta}^{(3+L(s)+\sum_{i=1}^{k} L(t_i))}(z)). \\
&= H_{L^\alpha+\beta}^{L(t)}(|\mathbf{a}| + |\beta| + L).
\end{aligned}$$

Since both $S_s(\mathbf{a})+\sum_{i=1}^{k} S_{t_i}(\mathbf{a})$ and space$(P_s(\beta))$ are both less than or equal to $H^{L(t)}_{L^\alpha+\beta}(|\mathbf{a}|+|\beta|+L)$, we have $S_t(\mathbf{a}) \leqslant H^{L(t)}_{L^\alpha+\beta}(|\mathbf{a}|+|\beta|+L)$, as required. ▌

Definition 5.2. Let $A \subseteq \Omega^s$ and let $h \in \mathrm{REC}(A)$ be a s.i.p. function from $\mathbb{N}$ to $\mathbb{N}$. Define $B(h)\colon A \times \mathbb{N} \to \mathbb{N}$ by A-recursion as follows:

$$B(h)(\alpha, x) := \begin{cases} \text{if } \alpha = 0 & \text{then } h(x) \\ & \text{else } B(h)(P_x(\alpha), B(h)(P_x(\alpha), x)). \end{cases}$$

Note that $B(h)(\alpha, x) = H_{2^\alpha}(h)(x)$ for all $x \in \mathbb{N}$.

Lemma 5.3. ($B(h)$-Bounding Lemma). *For $\alpha \in \Omega^s$, h s.i.p.,*

$$F_\alpha(h) \leqslant_0 B(h)(\omega \cdot \alpha, \_).$$

Proof. Note that for h s.i., $H_\omega(h) \leqslant_0 H_{2^\omega}(h)$. Thus

$$\begin{aligned} F_\alpha(h) &= H_{\omega^\alpha}(h) \\ &= H_\omega^{(\alpha)}(h) \\ &\leqslant_0 H_{2^\omega}^{(\alpha)}(h) \qquad [\text{by the } J, K \text{ Lemma}] \\ &= H_{2^{\omega \cdot \alpha}}(h) \\ &= B(h)(\omega \cdot \alpha, \_). \quad ▌ \end{aligned}$$

Lemma 5.4. *Provided A satisfies the first closure condition,*

$$\bigcup_{l \in \mathbb{N} \setminus \{0,1\}} \mathrm{SPACE}(H(l^{\omega \cdot A})) = \mathrm{SPACE}(H(\omega^A)).$$

Proof. Let $f \in \mathrm{SPACE}(H(\omega^A))$. Then $S_f \leqslant_0 H_\alpha \circ r$ for some $\alpha \in \omega^A$ and some s.i.p. $r \in \mathscr{E}^3$. Now

$$\begin{aligned} H_\alpha &\leqslant_1 H_{\omega^\gamma} \qquad [\text{for } \gamma \in A \text{ (Exponent Lemma)}] \\ &\leqslant_1 H_{2^{\omega \cdot \gamma}} \qquad [B(h)\text{-bounding Lemma}]. \end{aligned}$$

Thus for all $x \in \mathbb{N}$, $S_f(x) \leqslant H_{2^{\omega \cdot \gamma}}(r(x)+1)$, so $f \in \mathrm{SPACE}(H(2^{\omega \cdot A}))$. Conversely, let $f \in \mathrm{SPACE}(H(l^{\omega \cdot A}))$ for some $l \geqslant 2$. Then $S_f \leqslant_0 H_{\alpha_1} \circ \cdots \circ H_{\alpha_k} \circ r$, where $\alpha_i = \prod_j l^{\omega \cdot \beta_{ij}}$ for $\beta_{ij} \in A$ $(1 \leqslant i \leqslant k)$. (Elements of $l^{\omega \cdot A}$ are all of the form $\sum_{i=1}^{n} \alpha_i$ where α_i has the form above.)

If we can show that for some fixed $y \in \mathbb{N}$,

$$\forall \beta \in A \ \exists \gamma \in \omega^A \qquad \text{such that} \quad H_{l^{\omega \cdot \beta}} \leqslant_y H_\gamma, \tag{$*$}$$

then if we let $\gamma_{ij} \in \omega^A$ be such that $H_{l^\omega \cdot \beta_{ij}} \leqslant_y H_{\gamma_{ij}}$ and let $\delta_i \in \omega^A$ be such that $H_{\Pi_j \gamma_{ij}} \leqslant_0 H_{\delta_i}$, we know that $H_{\alpha_i} \leqslant_y H_{\delta_i}$ for $1 \leqslant i \leqslant k$, so

$$\begin{aligned} S_f(x) &\leqslant H_{\delta_1} \circ \cdots \circ H_{\delta_k}(r(x) + y) \\ &= H_{\sum_j \delta_j}(r(x) + y). \end{aligned}$$

Now $\lambda x[r(x) + y] \in \mathscr{E}^3$ and $\sum_j \delta_j \in \omega^A$, so $f \in \mathrm{SPACE}(H(\omega^A))$.

To prove $(*)$, let $r(x) = l^{x+2}$ and let $z \in \mathbb{N}$ be such that $r \leqslant_z F_3$. (This is possible since F_3 dominates all elementary functions). Let $y = \max(2, z)$. We show by induction on $\alpha \in A$ that

$$H_{l^\omega \cdot \alpha} \leqslant_y H_{\omega^{3+\alpha}} \circ r. \tag{†}$$

But $H_{\omega^{3+\alpha}} \circ r \leqslant_y H_{\omega^{3+\alpha} + \omega^3}$ so we may take γ in $(*)$ to be $\omega^{3+\alpha} + \omega^3 \in \omega^A$.

Proof of (†). (†) is obvious if $\alpha = 0$. Suppose $\alpha = \alpha' + 1$ and $x \geqslant y$. Then

$$\begin{aligned} H_{l^\omega \cdot \alpha}(x) &= H_{l^\omega \cdot \alpha' \cdot l^{x+1}}(x) \\ &= \underbrace{H_{l^\omega \cdot \alpha'} \circ \cdots \circ H_{l^\omega \cdot \alpha'}}_{l^{x+1}}(x) \\ &\leqslant (H_{\omega^{3+\alpha'}} \circ r)^{l^{x+1}}(x) \qquad \text{[by the induction hypothesis]} \\ &\leqslant ((H_{\omega^{3+\alpha'}})^l)^{l^{x+1}}(x) \\ &= H^{r(x)}_{\omega^{3+\alpha'}}(x) \\ &< H^{r(x)+1}_{\omega^{3+\alpha'}}(x) \\ &= H_{\omega^{3+\alpha}}(r(x)). \end{aligned}$$

If λ is a limit then for $x \geqslant y$,

$$\begin{aligned} H_{l^\omega \cdot \lambda}(x) &= H_{l^\omega \cdot \lambda_x}(x) \\ &\leqslant H_{\omega^{3+\lambda_x}}(r(x)) \qquad \text{[by the induction hypothesis]} \\ &\leqslant H_{\omega^{3+\lambda_{r(x)}}}(r(x)) \\ &= H_{\omega^{3+\lambda}}(r(x)). \quad \blacksquare \end{aligned}$$

This concludes the proof of Lemma 5.4. ▮

Proof of Theorem III. Suppose $A \subseteq \Omega^s$ satisfies the first closure condition. We first show that $\mathrm{REC}(\omega \cdot A) \subseteq \mathrm{WHILE}(\omega^A)$. The proof is by induction on $\mathrm{REC}(\omega \cdot A)$. Now $\mathscr{E}^3 \subseteq \mathrm{WHILE}(\omega^A)$ and, by Theorem I, $\mathrm{WHILE}(\omega^A)$ is closed under explicit definitions.

For closure under recursion, suppose f is defined from $\mathcal{U} \subseteq \mathrm{REC}(\omega \cdot A)$ by A-recursion and $\beta \in A$. Assume inductively that $\mathcal{U} \subseteq \mathrm{WHILE}(\omega^A) = \bigcup_{l \in \mathbb{N} \setminus \{0,1\}} \mathrm{SPACE}(H(l^{\omega \cdot A}))$. Then for each $u \in \mathcal{U}$,

$$S_u(\mathbf{a}) \leqslant H_{m^\sigma}(v)(|\mathbf{a}|)$$

for $m \in \mathbb{N}$, $\sigma \in \omega \cdot A$, and v elementary and s.i.p.

Thus by the space-bounding lemma,

$$\begin{aligned}
S_{f(\beta)}(\mathbf{a}) &\leqslant H_{l^{(\alpha+\beta+1)}}(r)(|\beta| + |\mathbf{a}| + l) \\
&\qquad [\text{for } \alpha \in \omega \cdot A,\ 2 \leqslant l \in \mathbb{N} \text{ and } r \in \mathcal{E}^3 \text{ s.i.p.}] \\
&\leqslant H_{l^{(\alpha+\beta+1)}}(H_{l^\delta})(|\beta| + |\mathbf{a}| + l) \\
&\qquad [\text{for } \delta \in \omega \cdot A, \text{ as } \mathcal{E}^3 \subseteq \mathrm{SPACE}(H(l^{\omega \cdot A}))] \\
&= H_{l^{(\alpha+\beta+1+\delta)}}(|\beta| + |\mathbf{a}| + l) \\
&\qquad [\text{where } \alpha + \beta + 1 + \delta \in \omega \cdot A] \\
&\leqslant H_\gamma(|\beta| + |\mathbf{a}| + l) \\
&\qquad [\text{for some } \gamma \in \omega^A] \\
&= H_{\gamma + |\beta| + l}(|\mathbf{a}|) \\
&\qquad [\text{where } \gamma + |\beta| + l \in \omega^A].
\end{aligned}$$

So $f(\beta) \in \mathrm{SPACE}(H(\omega^A)) = \mathrm{WHILE}(\omega^A)$, and hence $\mathrm{REC}(\omega \cdot A) \subseteq \mathrm{WHILE}(\omega^A)$.

To prove the reverse inclusion, we show first that, for any $\alpha \in A$, F_α is elementary in $B(\mathrm{succ})(\omega \cdot \gamma, \_)$ for some $\gamma \in A$, and thus $F_\alpha \in \mathrm{REC}(\omega \cdot A)$. So let $\alpha \in A$. Then

$$\begin{aligned}
S_{F_\alpha}(x) &\leqslant H_{(1+\omega^\alpha) \cdot \omega^\alpha}(r)(|\omega^\alpha| + |x|) && [\text{for } r \in \mathcal{E}^3 \text{ (Honesty Lemma)}] \\
&\leqslant H_{(1+\omega^\alpha) \cdot \omega^\alpha}(H_\delta)(|\omega^\alpha| + |x|) && [\text{for } \delta \in \omega^A] \\
&= H_{\delta \cdot (1+\omega^\alpha) \cdot \omega^\alpha + |\omega^\alpha|}(|x|) && [\text{Exponent Lemma}] \\
&\leqslant H_{\omega^\gamma}(|x|) && [\text{for some } \gamma \in A] \\
&\leqslant B(\mathrm{succ})(\omega \cdot \gamma, |x|).
\end{aligned}$$

Thus F_α is elementary in $B(\mathrm{succ})(\omega \cdot \gamma, \_)$.

Now if $f \in \mathrm{WHILE}(\omega^A)$ then f is elementary in H_γ for some $\gamma \in \omega^A$. Now $H_\gamma = F_{\alpha_1} \circ \cdots \circ F_{\alpha_k}$ where $\alpha_1, \ldots, \alpha_k \in A$. As $F_{\alpha_j} \in \mathrm{REC}(\omega \cdot A)$ for each j, it follows that $H_\gamma \in \mathrm{REC}(\omega \cdot A)$, and so $f \in \mathrm{REC}(\omega \cdot A)$. This completes the proof of Theorem III. ∎

6. Examples

Definitions 6.1. We define, for $n \in \mathbb{N}$, the sets $M_n \subseteq \Omega^s$ and $E_n \subseteq \Omega^s$ by

(i) $M_0 := \mathbb{N}$

(ii) $M_{n+1} := \omega \cdot M_n$

(iii) $E_0 := \mathbb{N}$

(iv) $E_{n+1} := \omega^{E_n}$.

We set $M := \bigcup_{n \in \mathbb{N}} M_n$ and $E := \bigcup_{n \in \mathbb{N}} E_n$. Clearly, if $m < n$ then $M_m \subseteq M_n \subseteq M$ and $E_m \subseteq E_n \subseteq E$. Also $\omega \cdot M = M$ and $\omega^E = E$. We have the following examples:

(1) $\mathrm{REC}(\omega \cdot E_n) = \mathrm{WHILE}(E_{n+1}) = \mathrm{SPACE}(E_{n+1})$
$= \text{PROVABLY RECURSIVE IN } \Sigma_{n+1}\text{-Ind}$
for each $n \in \mathbb{N}$

(2) $\mathrm{REC}(\omega \cdot \mathbb{N}) = \mathrm{WHILE}(\omega^{\mathbb{N}})$
$= \text{PRIMITIVE RECURSIVE}$

(3) $\mathrm{REC}(\omega^{\mathbb{N}}) = \mathrm{WHILE}(\omega^{\omega^{\mathbb{N}}})$
$= \bigcup_{k \in \mathbb{N}} k\text{-RECURSIVE}$

(4) $\mathrm{REC}(M_{k+1}) = (k+1)\text{-RECURSIVE for each } k \in \mathbb{N}$

(5) $\mathrm{REC}(E) = \mathrm{WHILE}(E) = \mathrm{SPACE}(E)$
$= \text{PROVABLY RECURSIVE IN PA}.$

Here PA means Peano Arithmetic and Σ_n-Ind means Peano Arithmetic but with the induction rule restricted to Σ_n formulae. For further details of the proof-theoretical aspects, see Buchholz and Wainer (1987) and Wainer (1990).

Received November 17, 1989; final manuscript received October 9, 1990

References

Buchholz, W., and Wainer, S. (1987), Provably computable functions and the fast growing hierarchy, *in* "Logic and Combinatorics" (S. G. Simpson, Ed.), pp. 179–198, Contemporary Mathematics, Vol. 65, Amer. Math. Soc., Providence, RI.

Cichon, E. A. (1983), A short proof of two recently discovered independence proofs using recursion theoretic methods, *Proc. A.M.S.* **87**, No. 4, 704–706.

DENNIS-JONES, E. C., AND WAINER, S. S. (1984), Subrecursive hierarchies via direct limits, *in* "Computation and Proof Theory" (M. Richter, E. Börger, W. Oberschelp, B. Schinzel and W. Thomas, Eds.), pp. 117–128, Lecture Notes in Mathematics, Vol. 1104, Springer-Verlag, Berlin/New York.

KETONEN, J., AND SOLOVAY, R. (1981), Rapidly growing Ramsey functions, *Ann. of Math.* **113**, 267–314.

ROSE, H. E. (1984), "Subrecursion: Functions and Hierarchies," Oxford Logic Guides, Vol. 9, Clarendon, Oxford.

TAIT, W. W. (1961), Nested recursion, *Math. Ann.* **143**, 236–250.

WAINER, S. S. (1989), Slow growing versus fast growing, *J. Symbolic Logic* **54**, No. 2, 608.

WAINER, S. S. (1991), Computability—Logical and recursive complexity, *in* "Summer School on Logic, Algebra and Computation, Marktoberdorf, Germany, 1989" (F. Bauer, Ed.), NATO ASI Series F, pp. 237–264, Springer-Verlag, Berlin/New York.