ABSTRACT FIRST ORDER COMPUTABILITY. I(1)

BY
YIANNIS N. MOSCHOVAKIS

The recent development of recursion theory has turned in part toward studying notions of computability on domains other than the natural numbers. Without any attempt for completeness, we mention the theories of Takeuti [23] (and others), Machover [15], Kripke [10], Kreisel-Sacks [9] and Platek [20] on classes or sets of ordinals, Kleene's theory of recursive functionals on objects of arbitrary finite type over the integers [6], [7] (and others) and the theories of computability on arbitrary structures of Fraïssé [1], Lacombe [11], [12], Kreisel [8] (and others) and more recently Platek [20], Montague [16] and Lambert [13]. (Levy's development of a hierarchy of set-theoretic predicates in [14] is also relevant.) Some of these theories attempt to abstract the computational (or combinatorial) aspects of recursion theory while others (notably Kreisel's and Montague's) conceive of recursion theory as a branch of definability theory.

Here we study computability theory on abstract (unordered) domains primarily as a tool for hierarchy theory. The term "first order" in the title indicates that we restrict ourselves to computabilities relative to given functions (i.e. objects of type-1) rather than functionals (i.e. objects of higher finite type); however the main contribution of this work is the abstraction of the theory of hyperarithmetic functions to arbitrary domains, which can be viewed as a second order computability theory.

In part I of the paper (§1–§9) we study two computabilities, one (*prime computability*) in general stronger than the other (*search computability*), and we formulate a weak version of Church's thesis: every prime computable function is intuitively effectively computable and every intuitively effectively computable function is search computable. The technical results of this part include the normal form and first recursion theorems and the development of the projective hierarchy on a first order structure, our abstract analog of the arithmetical hierarchy on the integers.

In the second part of the paper, *Abstract first order computability.* II we study the *hyperprojective hierarchy* on a first order structure, our abstract analog of the hyperarithmetical hierarchy on the integers.

We were motivated for this work by two discrepancies in the analogy between the arithmetical hierarchy (of number-theoretic predicates) and the analytic hierarchy (of function-theoretic predicates). We first added the search operator ν (§6) to

Received by the editors October 27, 1966 and, in revised form, June 7, 1968.
(1) The preparation of this paper was sponsored in part by an NSF Grant.

Kleene's schemata in [6] for relative recursiveness for objects of type-2 so that Post's theorem could be proved for the analytic hierarchy. Later, while working on [17], we proved that the class of predicates recursively enumerable in $^3\mathbf{E}$ is not closed under existential function quantification. Since these predicates are related to the hyperanalytic predicates as the Π^1_1 predicates are related to the hyperarithmetic predicates, this was a discrepancy in the analogy hyperarithmetic$\sim$hyperanalytic. It turned out that if we added the search operator to Kleene's schemata for recursion relative to $^3\mathbf{E}$, then the resulting hyperprojective hierarchy did not suffer from this defect. Moreover, all the standard theorems about the hyperarithmetic and hyperanalytic hierarchies extended to this larger class of predicates, even though not trivially. In writing these results up we realized that they did not depend in any essential way on properties of number-theoretic functions and could be extended without much difficulty to arbitrary domains.

We have attempted to keep this paper down to a reasonable size by including only those examples which are directly relevant to the development of the theory. The important special case of computability on the space of number-theoretic functions will be treated in [19]. In [18] we give a summary of the results of both this paper and [19].

Our debt to Stephen C. Kleene and Clifford Spector will be obvious to anyone familiar with [5], [6], [7], [21] and [22]. In the case of Kleene it goes much deeper than our adoption of several specific methods from [6] (which are indicated in the text) and has to do with Kleene's pioneering work in the use and analysis of inductive definitions. From this point of view the most significant contribution of [6] is the casting of the definition of partial recursive functionals in the form of a (transfinite) induction. The main technical contribution of this paper is the introduction and systematic exploitation of existential, nondeterministic clauses in inductive definitions.

(*Added May* 1968: I wish to thank the students in my Recursion Theory Seminar in the Spring of 1967, and in particular Carl Gordon, Tom McCutcheon, Ben Freedman and Perry Smith for discovering a host of minor and not-so-minor errors in the manuscript.)

1. **Preliminaries.** Let B be an arbitrary set, let 0 be some object not in B, let

$$B^0 = B \cup \{0\}. \tag{1.1}$$

We define the set B^* by the inductive clauses

$$\begin{aligned} &\textit{if } x \in B^0, \textit{ then } x \in B^*, \\ &\textit{if } x, y \in B^*, \textit{ then } (x, y) \in B^*. \end{aligned} \tag{1.2}$$

Here (x, y) is the ordered pair of x and y and we assume that we have chosen a particular set-theoretic operation to represent the ordered pair so that no object

in B^0 is an ordered pair. Thus if $z \in B^*$, then either $z \in B^0$ or $z=(x, y)$ with uniquely determined $x, y \in B^*$.

We identify the natural numbers $0, 1, 2, \ldots$ with the objects $0, (0, 0), ((0, 0), 0), \ldots$ of B^*:

$$\begin{aligned} 0 &= 0, \\ n+1 &= (n, 0), \\ \omega &= \{0, 1, 2, \ldots\}. \end{aligned} \tag{1.3}$$

To each $z \in B^*$ we assign its components πz and δz by the induction

$$\begin{aligned} \pi 0 &= \delta 0 = 0, \\ \pi z &= \delta z = 1 \quad \text{if } z \in B, \\ \pi z &= x, \delta z = y \quad \text{if } z = (x, y). \end{aligned} \tag{1.4}$$

(The difference in defining the components of z for the cases $z=0$ and $z \in B$ is for technical reasons; one of its consequences is that the singleton $\{0\}$ will be a "primitive computable" subset of B^* without requiring a special axiom.)

Each element of B^* is constructed from a finite number of elements of B^0 by a finite number of applications of the pairing operation. We assign inductively to each $z \in B^*$ the finite subset D_z of B^0 (excluding 0) on which z depends:

$$\begin{aligned} D_0 &= \varnothing, \\ D_z &= \{z\} \quad \text{if } z \in B, \\ D_z &= D_x \cup D_y \quad \text{if } z = (x, y). \end{aligned} \tag{1.5}$$

For a subset A of B^* we put

$$A^* = \left\{ z \in B^* : D_z \subset \bigcup_{x \in A} D_x \right\}. \tag{1.6}$$

Each A^* then contains the set $\varnothing^* = \{0\}^* = 0^*$ of "pure objects", i.e. elements z of B^* with $D_z = \varnothing$. Clearly $\omega \subset 0^*$.

Throughout this paper we let x, y, z, w, u, v, t, s be arbitrary elements of B^* and k, l, n, m, i, j arbitrary elements of ω. We use boldface italic letters to denote finite sequences of elements of B^*, e.g. $\boldsymbol{u} = u_1, \ldots, u_k$, $\boldsymbol{x} = x_1, \ldots, x_n$, $\boldsymbol{y} = y_1, \ldots, y_m$, $\boldsymbol{u}' = u'_1, \ldots, u'_{k'}, \ldots$ etc.

Our concern is with functions $\mathrm{f}(\boldsymbol{u})$ of k variables from B^* into B^*. Although we are ultimately interested in single-valued totally defined functions of the usual kind it will be convenient at the outset to allow $\mathrm{f}(\boldsymbol{u})$ to be *undefined* or *multiple-valued*. A *partial multiple-valued* (*p.m.v.*) function of k variables is a mapping f which assigns to each k-tuple $\boldsymbol{u}$ of elements of B^* a subset $\mathrm{f}(\boldsymbol{u})$ of B^*, possibly empty,

possibly with more than one element. In connection with p.m.v. functions we use the following notation conventions:

$$
\begin{aligned}
&\mathrm{f}(\boldsymbol{u}) \to z && \text{for } z \in \mathrm{f}(\boldsymbol{u}) \ (\mathrm{f}(\boldsymbol{u}) \textit{ yields } z),\\
&\mathrm{f}(\boldsymbol{u}) \downarrow && \text{for } \mathrm{f}(\boldsymbol{u}) \neq \varnothing \ (\mathrm{f}(\boldsymbol{u}) \textit{ is defined}),\\
&\mathrm{f}(\boldsymbol{u}) \subset \mathrm{g}(\boldsymbol{u}) && \text{for } (z)[\mathrm{f}(\boldsymbol{u}) \to z \Rightarrow \mathrm{g}(\boldsymbol{u}) \to z],\\
&\mathrm{f}(\boldsymbol{u}) = \mathrm{g}(\boldsymbol{u}) && \text{for } \mathrm{f}(\boldsymbol{u}) \subset \mathrm{g}(\boldsymbol{u}) \ \&\ \mathrm{g}(\boldsymbol{u}) \subset \mathrm{f}(\boldsymbol{u}),\\
&\mathrm{f} \subset \mathrm{g} && \text{for } (\boldsymbol{u})[\mathrm{f}(\boldsymbol{u}) \subset \mathrm{g}(\boldsymbol{u})],\\
&\mathrm{f} = \mathrm{g} && \text{for } \mathrm{f} \subset \mathrm{g} \ \&\ \mathrm{g} \subset \mathrm{f},\\
&\mathrm{f}(\boldsymbol{u}) = z && \text{for } \mathrm{f}(\boldsymbol{u}) = \{z\},\\
& && \text{i.e. } \mathrm{f}(\boldsymbol{u}) \to z \ \&\ (w)[\mathrm{f}(\boldsymbol{u}) \to w \Rightarrow w = z].
\end{aligned}
\tag{1.7}
$$

Substitution of p.m.v. functions is defined in the natural way:

$$\mathrm{f}(\boldsymbol{x}, \mathrm{g}(\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{y}) \to z \Leftrightarrow (Eu)[\mathrm{g}(\boldsymbol{x}, \boldsymbol{y}) \to u \ \&\ \mathrm{f}(\boldsymbol{x}, u, \boldsymbol{y}) \to z]. \tag{1.8}$$

We always interpret simultaneous substitution by successive substitution. In particular

$$\mathrm{f}(\mathrm{g}(\boldsymbol{x}), \mathrm{g}(\boldsymbol{x})) \to z \Leftrightarrow (Eu)(Ev)[\mathrm{g}(\boldsymbol{x}) \to u \ \&\ \mathrm{g}(\boldsymbol{x}) \to v \ \&\ \mathrm{f}(u, v) \to z],$$

so that (in effect) a multiple-valued term that occurs more than once in a formula may have different denotations in its several occurrences.

Most of these conventions are self-explanatory, but we list them to liberate the reader from any prejudices he may have against the use of multiple-valued functions.

Put

$$\mathrm{s}_0(x) = (x, 0), \qquad \mathrm{s}_{n+1}(x_0, \ldots, x_{n+1}) = (x_0, \mathrm{s}_n(x_1, \ldots, x_{n+1})), \tag{1.9}$$

$$(x)_0 = \pi x, \qquad (x)_{i+1} = (\delta x)_i. \tag{1.10}$$

One easily sees by induction on n that

$$\textit{if } x = \mathrm{s}_n(x_0, \ldots, x_n), \textit{ then for } i = 0, \ldots, n, (x)_i = x_i. \tag{1.11}$$

Put

$$\mathrm{Seq}\,(0, x) \Leftrightarrow \delta x = 0 \ \&\ x \neq 0, \qquad \mathrm{Seq}\,(n+1, x) \Leftrightarrow \mathrm{Seq}\,(n, \delta x); \tag{1.12}$$

then

$$\mathrm{Seq}\,(n, x) \Leftrightarrow x = \mathrm{s}_n(x_0, \ldots, x_n) \textit{ for suitable } x_0, \ldots, x_n. \tag{1.13}$$

Finally we put

$$\langle x_1, \ldots, x_n \rangle = s_n(n, x_1, \ldots, x_n), \tag{1.14}$$

$$\mathrm{Seq}\,(x) \Leftrightarrow \pi x \in \omega \ \&\ \mathrm{Seq}\,(\pi x, x) \tag{1.15}$$

and observe that

(1.16) $\mathrm{Seq}(x) \Leftrightarrow \pi x \in \omega \;\&\; x = \langle x_1, \ldots, x_{\pi x}\rangle$ *for suitable* $x_1, \ldots, x_{\pi x}$,

(1.17) *if* $x = \langle x_1, \ldots, x_n\rangle$, *then for* $i = 1, \ldots, n$, $(x)_i = x_i$.

We need an analog of the familiar course of values function of ordinary recursion theory. To each p.m.v. function $\mathrm{f}(y, \boldsymbol{x})$ we assign the p.m.v. function $\bar{\mathrm{f}}(y, \boldsymbol{x})$ defined inductively by:

$$\begin{aligned} \bar{\mathrm{f}}(y, \boldsymbol{x}) &= 1 \quad \text{if } y \in B^0, \\ \bar{\mathrm{f}}((s, t), \boldsymbol{x}) &= \langle \mathrm{f}(s, \boldsymbol{x}), \mathrm{f}(t, \boldsymbol{x}), \bar{\mathrm{f}}(s, \boldsymbol{x}), \bar{\mathrm{f}}(t, \boldsymbol{x})\rangle. \end{aligned} \tag{1.18}$$

2. **Primitive computable functions.** Let $\boldsymbol{\varphi} = \varphi_1, \ldots, \varphi_l$ be a finite (possibly empty) list of p.m.v. functions on B^* to B^*, φ_i of n_i arguments. (We sometimes call the sequence $l, n_1, \ldots, n_l$ the *characteristic* of $\boldsymbol{\varphi}$.) In the schemata C0–C10 below $\boldsymbol{x} = x_1, \ldots, x_n$, $y = y_1, \ldots, y_m$ are arbitrary tuples of elements of B^* (empty if $n=0$ or $m=0$), the projections π and δ are those defined by (1.4) and composition of p.m.v. functions is to be understood as in (1.8). The expressions to the right will be explained shortly. In this section we are only interested in schemata C0–C7 but we list the rest for future reference.

C0.	$\mathrm{f}(t_1, \ldots, t_{n_i}, \boldsymbol{x}) = \varphi_i(t_1, \ldots, t_{n_i})$	$\langle 0, n_i+n, i\rangle$
C1.	$\mathrm{f}(\boldsymbol{x}) = y$	$\langle 1, n, y\rangle$
C2.	$\mathrm{f}(y, \boldsymbol{x}) = y$	$\langle 2, n+1\rangle$
C3.	$\mathrm{f}(s, t, \boldsymbol{x}) = (s, t)$	$\langle 3, n+2\rangle$
$\mathrm{C4}_0$.	$\mathrm{f}(y, \boldsymbol{x}) = \pi y$	$\langle 4, n+1, 0\rangle$
$\mathrm{C4}_1$.	$\mathrm{f}(y, \boldsymbol{x}) = \delta y$	$\langle 4, n+1, 1\rangle$
C5.	$\mathrm{f}(\boldsymbol{x}) = \mathrm{g}(\mathrm{h}(\boldsymbol{x}), \boldsymbol{x})$	$\langle 5, n, g, h\rangle$
C6.	$\mathrm{f}(y, \boldsymbol{x}) = \mathrm{g}(y, \boldsymbol{x})$ if $y \in B^0$, $\mathrm{f}((s, t), \boldsymbol{x}) = \mathrm{h}(\mathrm{f}(s, \boldsymbol{x}), \mathrm{f}(t, \boldsymbol{x}), s, t, \boldsymbol{x})$	$\langle 6, n+1, g, h\rangle$
C7.	$\mathrm{f}(\boldsymbol{x}) = \mathrm{g}(x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n)$	$\langle 7, n, j, g\rangle$
C8.	$\mathrm{f}(e, \boldsymbol{x}, y) = \{e\}(\boldsymbol{x})$	$\langle 8, n+m+1, n\rangle$
C9.	$\mathrm{f}(\boldsymbol{x}) = \nu y[\mathrm{g}(y, \boldsymbol{x}) \to 0]$	$\langle 9, n, g\rangle$
C10.	$\mathrm{f}(\boldsymbol{x}) = \mathbf{E}(\lambda y \mathrm{g}(y, \boldsymbol{x}))$	$\langle 10, n, g\rangle$

Schemata C0–C7 are the obvious analogs of the schemata for primitive recursive functions in ordinary recursion theory. Instead of defining the class of functions primitive computable in $\boldsymbol{\varphi}$ by imitating the primitive recursive derivations of the usual sort, we prefer to introduce indices for these functions at this early stage.

The expressions to the right of schemata C0–C7 indicate the clauses for an inductive definition of the set *PRI* of indices of primitive computable functions.

Notice that each index f is a sequence element (i.e. Seq (f)), that $(f)_1$ gives the number of the schema involved and that $(f)_2$ is the number of arguments of the function in question. We use a notation convention to which we shall adhere: if a function has been named f, g, h, etc., then an index for it will be f, g, h, etc.

The definition of the set PRI of indices is by course of values induction on $f \in B^*$ and has the following eight clauses.

Case C0. $\text{Seq}(f) \,\&\, (f)_0 = 3 \,\&\, (f)_1 = 0 \,\&\, 1 \leqq (f)_3 \leqq l$
$\quad \&\, \{[(f)_3 = 1 \,\&\, (f)_2 \geqq n_1] \vee \cdots \vee [(f)_3 = l \,\&\, (f)_2 \geqq n_l]\}.$

Case C1. $\text{Seq}(f) \,\&\, (f)_0 = 3 \,\&\, (f)_1 = 1 \,\&\, (f)_2 \in \omega.$

Case C2. $\text{Seq}(f) \,\&\, (f)_0 = 2 \,\&\, (f)_1 = 2 \,\&\, (f)_2 \geqq 1.$

Case C3. $\text{Seq}(f) \,\&\, (f)_0 = 2 \,\&\, (f)_1 = 3 \,\&\, (f)_2 \geqq 2.$

Case $C4_0$. $\text{Seq}(f) \,\&\, (f)_0 = 3 \,\&\, (f)_1 = 4 \,\&\, (f)_2 \geqq 1 \,\&\, (f)_3 = 0.$

Case $C4_1$. $\text{Seq}(f) \,\&\, (f)_0 = 3 \,\&\, (f)_1 = 4 \,\&\, (f)_2 \geqq 1 \,\&\, (f)_3 = 1.$

Case C5. $\text{Seq}(f) \,\&\, (f)_0 = 4 \,\&\, (f)_1 = 5 \,\&\, (f)_2 \in \omega$
$\quad \&\, (f)_3 \in PRI \,\&\, (f)_4 \in PRI \,\&\, (f)_{4,2} = (f)_2 \,\&\, (f)_{3,2} = (f)_2 + 1.$
(Here $(x)_{i,j} = ((x)_i)_j$.)

Case C6. $\text{Seq}(f) \,\&\, (f)_0 = 4 \,\&\, (f)_1 = 6 \,\&\, (f)_2 \geqq 1$
$\quad \&\, (f)_3 \in PRI \,\&\, (f)_4 \in PRI \,\&\, (f)_{3,2} = (f)_2 \,\&\, (f)_{4,2} = (f)_2 + 3.$

Case C7. $\text{Seq}(f) \,\&\, (f)_0 = 4 \,\&\, (f)_1 = 7 \,\&\, 0 \leqq (f)_3 < (f)_2$
$\quad \&\, (f)_4 \in PRI \,\&\, (f)_{4,2} = (f)_2.$

(Whenever $x \leqq y$ or $x < y$ is a condition in one of the clauses, the conditions $x \in \omega$, $y \in \omega$ are also implied.)

We now assign to each $f \in PRI$, by induction on the definition of PRI and using the schemata C0–C7, a p.m.v. function $\{f\}_{\text{pr}}(u_1, \ldots, u_k)$ of k variables, where $k = (f)_2$. For example, if $f = \langle 0, n_i + n, i \rangle$, we put

$$\{f\}_{\text{pr}}(t_1, \ldots, t_{n_i}, x_1, \ldots, x_n) = \varphi_i(t_1, \ldots, t_{n_i}),$$

if $\langle 5, n, g, h \rangle$, we put $\{f\}_{\text{pr}}(\boldsymbol{x}) = \{g\}_{\text{pr}}(\{h\}_{\text{pr}}(\boldsymbol{x}), \boldsymbol{x})$, etc. We call these p.m.v. functions *primitive computable in* φ and we put

(2.1) $\quad \mathbf{PR}(\varphi) = \textit{set of all functions } \{f\}_{\text{pr}}(\boldsymbol{u}), \textit{ with } f \in PRI \textit{ and } k = (f)_2.$

A predicate $P(\boldsymbol{u})$ is primitive computable in φ, if its characteristic function

$$\begin{aligned} \chi_P(\boldsymbol{u}) &= 0 \quad \text{if } P(\boldsymbol{u}), \\ &= 1 \quad \text{if } \bar{P}(\boldsymbol{u}) \end{aligned} \tag{2.2}$$

(which by definition is single-valued and total) is in $\mathbf{PR}(\varphi)$.

If C, E are subsets of B^*, we call a k-ary p.m.v. function $f: C \Rightarrow E$ primitive computable in φ, if f is the restriction to C of a p.m.v. function (on B^* to B^*) which is primitive computable in φ. Thus our ultimate concern may be with functions from B to B or from B to ω, but the pairing operation makes B^* a suitable

domain for studying these functions. (The same remark will apply to later definitions of larger classes of functions.)

In introducing the constant functions $\mathrm{f}(\boldsymbol{u})=y$ as primitive computable by C1 without discussion we skirted on a philosophical question of some interest. In what sense are the arbitrary constant functions on a set B^* computable? If the set B is uncountable, this procedure yields uncountably many primitive computable functions, a state of affairs to which many people might object. However each function $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ depends in its definition on only finitely many applications of C1, and all the relevant constants y have been coded into the index f. We are thus led to the following *relativised* definition:

Let A be a subset of B^*. A p.m.v. function $\mathrm{f}(\boldsymbol{u})$ is *primitive computable* (with constants derived) *from* A in $\boldsymbol{\varphi}$, $\mathrm{f}(\boldsymbol{u}) \in PR(A, \boldsymbol{\varphi})$, if there exists an $f \in A^* \cap PRI$, $(f)_2=k$, such that $\mathrm{f}(\boldsymbol{u})=\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ (A^* is defined in (1.6)). In particular

$$\mathbf{PR}(\boldsymbol{\varphi}) = PR(B, \boldsymbol{\varphi}). \tag{2.3}$$

We set

$$PR(\boldsymbol{\varphi}) = PR(\varnothing, \boldsymbol{\varphi}) \tag{2.4}$$

and call the functions in $PR(\boldsymbol{\varphi})$ *absolutely primitive computable in* $\boldsymbol{\varphi}$.

The definition of $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ for each $f \in PRI$, $(f)_2=k$ depends on the given sequence $\boldsymbol{\varphi}$ of p.m.v. functions, so we should really be writing $\{f\}_{\mathrm{pr}}(\boldsymbol{\varphi}, \boldsymbol{u})$. In this paper we shall reserve the letter "$\boldsymbol{\varphi}$" for a fixed sequence of functions (which in fact we shall often assume to be single-valued and totally defined) so we shall not make explicit this dependence of $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ on $\boldsymbol{\varphi}$. However, we may apply the definition to a sequence $\boldsymbol{\varphi}'=\mathrm{g}_1, \ldots, \mathrm{g}_m, \boldsymbol{\varphi}$, where we may think of $\mathrm{g}_1, \ldots, \mathrm{g}_m$ as arbitrary functions of $l_1, \ldots, l_m$ arguments respectively. We shall then write $\{f\}_{\mathrm{pr}}(\mathrm{g}_1, \ldots, \mathrm{g}_m, \boldsymbol{u})$ for $\{f\}_{\mathrm{pr}}(\boldsymbol{\varphi}', \boldsymbol{u})$, exhibiting the dependence of $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ on $\mathrm{g}_1, \ldots, \mathrm{g}_m$. Thus each $f \in PRI$ (where PRI is defined for the list $\boldsymbol{\varphi}'$) defines a *functional* $\{f\}_{\mathrm{pr}}(\mathrm{g}_1, \ldots, \mathrm{g}_m, \boldsymbol{u})$ ($k=(f)_2$) which we call *primitive computable from* A *in* $\boldsymbol{\varphi}$, if $f \in A^*$.

The following lemma is very easy to prove by induction on $f \in PRI$.

LEMMA 1. (a) *If each* φ_i *is single-valued, then each* $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ *is single-valued.*

(b) *If each* φ_i *is totally defined, then each* $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ *is totally defined.*

(c) *Suppose that for* $i=1, \ldots, l$, *if* $\boldsymbol{\varphi}_i(t_1, \ldots, t_{n_i}) \to z$, *then* $D_z \subset D_{t_1} \cup \cdots \cup D_{t_{n_i}}$. *Then for each* $f, u_1, \ldots, u_k, z$, *if* $\{f\}_{\mathrm{pr}}(u_1, \ldots, u_k) \to z$, *then* $D_z \subset D_f \cup \cdots \cup D_{u_k}$.

Thus, if $\boldsymbol{\varphi}$ consists of single-valued, total functions, then each function primitive computable in $\boldsymbol{\varphi}$ is single-valued and total.

EXAMPLE 1. The only total function of one variable $\mathrm{f}(x)$ from B to B which is absolutely primitive computable in the empty $\boldsymbol{\varphi}$ is the identity $\mathrm{f}(x)=x$; because by the lemma, if $\mathrm{f}(x)=\{f\}_{\mathrm{pr}}(x)$ with $f \in 0^*$ and $\{f\}_{\mathrm{pr}}(x) \to z$ then we must have $D_z \subset D_x \cup D_f = D_x$ and for $z, x \in B$, this means $z=x$.

EXAMPLE 2. Let $B=$*all number-theoretic functions*, take $\varphi=\varphi_1, \ldots, \varphi_l$ where $n_1=2$,

$$\begin{aligned}\varphi_1(x, i) &= x(i) \quad \text{if } x \in B, i \in \omega,\\ &= 0 \qquad \text{otherwise,}\end{aligned}$$

and $\varphi_2, \ldots, \varphi_l$ are arbitrary. This example has been the initial motivation for much of the work presented here. We shall treat it in detail in [19].

3. **Elementary theory of primitive computable functions.** Let us call a p.m.v. function $f(u)$ *combinatorial* if it is absolutely primitive computable in the empty φ. Such functions can be defined through schemata C2–C7; by Lemma 1 they are single-valued and total and they satisfy

$$f(u_1, \ldots, u_k) \in (D_{u_1} \cup \cdots \cup D_{u_k})^*. \tag{3.1}$$

LEMMA 2. *If* $g(u)$ *is in* $PR(A, \varphi)$, *then the function*

$$f(x, u, y) = g(u) \tag{3.2}$$

is also in $PR(A, \varphi)$.

Proof. We first show by induction on an index g of $g(u)$ that the function

$$f_1(u, y) = g(u) \tag{3.3}$$

is in $PR(A, \varphi)$. The idea here is that the schemata are not affected by the addition of extra variables at the end.

From (3.3) it follows that the function

$$f_2(u, x, y) = g(u) \tag{3.4}$$

is in $PR(A, \varphi)$, and from this $f(x, u, y)$ can be defined by repeated applications of C7.

REMARK 1. Lemma 13 is a stronger version of Lemma 2 and asserts that an index of f can be obtained from an index of g by a combinatorial function. The details of the proof outlined above can be found in the proof of Lemma 13.

REMARK 2. We shall not state explicitly any more results which, like Lemma 2, pertain to explicit definition, e.g. that a primitive computable function remains primitive computable if we interchange the order or identify some of its variables. Such results are immediate from schemata C7 and C5.

LEMMA 3. (a) *If* $y \in 0^*$, *then the constant function* $f(x)=y$ *is combinatorial.*

(b) *Let* $\phi(u)$ *be a function of* k *arguments on* ω *to* ω *which is primitive recursive in the sense of* [4] (*or equivalently* [6]). *There exists a combinatorial* $f(u)$ *on* B^* *to* B^* *whose restriction to* ω *is* $\phi(u)$.

Proof. (a) is immediate from C1, since $\{\langle 1, n, y\rangle\}_{pr}(x)=y$ and if $y \in 0^*$, then $\langle 1, n, y\rangle \in 0^*$.

We prove (b) by induction on a primitive recursive derivation of $\phi(\boldsymbol{u})$ from schemata S1–S6 of [6].

Case S1. $\phi(y, \boldsymbol{x})=y+1$. Set $f(y, \boldsymbol{x})=(y, 0)$ using C2, (a), C3 and C5.

Case S2. $\phi(\boldsymbol{u})=q \in \omega$. Immediate from (a).

Cases S3, S4 and S6 are immediate by C2, C5 and C7.

Case S5. $\phi(0, \boldsymbol{x})=\psi(\boldsymbol{x})$, $\phi(y+1, \boldsymbol{x})=\chi(y, \phi(y, \boldsymbol{x}), \boldsymbol{x})$.

By ind. hyp. there are combinatorial functions $g(\boldsymbol{x})$ and $h(y, u, \boldsymbol{x})$ which agree with $\psi(\boldsymbol{x})$ and $\chi(y, u, \boldsymbol{x})$ respectively when restricted to ω. By Lemma 2 and C7 the functions

$$g'(y, \boldsymbol{x}) = g(\boldsymbol{x}), \quad h'(u, v, s, t, \boldsymbol{x}) = h(s, u, \boldsymbol{x})$$

are combinatorial. We set by C6

$$f(y, \boldsymbol{x}) = g'(y, \boldsymbol{x}) = g(\boldsymbol{x}),$$
$$f((s, t), \boldsymbol{x}) = h'(f(s, \boldsymbol{x}), f(t, \boldsymbol{x}), s, t, \boldsymbol{x}) = h(s, f(s, \boldsymbol{x}), \boldsymbol{x}),$$

and then verify by induction on $i \in \omega$ (using the relation $i+1=(i, 0)$) that $f(i, \boldsymbol{x}) = \phi(i, \boldsymbol{x})$ when $i, \boldsymbol{x}$ are in ω.

REMARK 3. We shall often define a function "by induction on $i \in \omega$" i.e. by the schema:

$$f(0, \boldsymbol{x}) = g(\boldsymbol{x}), \quad f(i+1, \boldsymbol{x}) = h(i, f(i, \boldsymbol{x}), \boldsymbol{x}). \tag{3.5}$$

It will be always assumed in these cases that we treat the definition as we did in Case S5 of the proof, using C6 to define a function $f(y, \boldsymbol{x})$ $(y \in B^*)$ which agrees with $f(i, \boldsymbol{x})$ as defined by (3.5) when $y=i \in \omega$.

LEMMA 4. *The sets* $\{0\}$ *(singleton* 0*),* B, ω, *the function* $\langle x_1, \ldots, x_n\rangle$ *defined by* (1.14) *and for each fixed* i, *the function* $(x)_i$ *defined by* (1.10) *are combinatorial.*

Proof. The characteristic function $\chi_0(z)$ of $\{0\}$ is defined by C4 and C6:

$$\begin{aligned} &\chi_0(y) = \pi y \quad \text{if } y \in B^0, \\ &\chi_0((s, t)) = 1. \end{aligned} \tag{3.6}$$

For $\chi_B(z)$, the characteristic function of B, we put

$$\begin{aligned} &\chi_B(y) = \bar{s}\bar{g}(\pi y) \quad \text{if } y \in B^0, \\ &\chi_B((s, t)) = 1. \end{aligned} \tag{3.7}$$

(Here $\bar{s}\bar{g}(x)$ is the function associated by Lemma 3 to the number-theoretic primitive recursive function $\bar{s}\bar{g}(x)=1 \dot{-} x$.) For $\chi_\omega(z)$, the characteristic function of ω, we put

$$\begin{aligned} &\chi_\omega(y) = \pi y \quad \text{if } y \in B^0 \\ &\chi_\omega((s, t)) = \mathrm{sg}(\chi_\omega(s)+\chi_0(t)). \end{aligned} \tag{3.8}$$

(Again sg and $+$ are the functions associated by Lemma 3 to the number-theoretic functions $\mathrm{sg}(x)=1 \dot{-} (1 \dot{-} x)$ and $x+y$.)

We prove that each function $\langle x_1, \ldots, x_n\rangle$ is combinatorial by first showing by induction on n that each $s_n(x_0, \ldots, x_n)$ of (1.9) is combinatorial and then using the definition (1.14). For each fixed i, $(x)_i$ is combinatorial by an easy induction on i.

REMARK 4. From now on we associate with each primitive recursive number-theoretic function $\phi(u_1, \ldots, u_k)$ the combinatorial function

$$\mathrm{f}(u_1, \ldots, u_k) = \mathrm{f}(\boldsymbol{u}) \cdot \bar{\mathrm{s}}\bar{\mathrm{g}}(\chi_\omega(u_1)) \cdot \cdots \cdot \bar{\mathrm{s}}\bar{\mathrm{g}}(\chi_\omega(u_k)), \tag{3.9}$$

where the $\mathrm{f}(\boldsymbol{u})$ on the right is associated with $\phi(\boldsymbol{u})$ by Lemma 3 and the multiplication is the combinatorial function associated to ordinary multiplication by Lemma 3. This extends each primitive recursive number-theoretic function to a combinatorial function on B^* which is $=0$ if any of the arguments is not a natural number.

LEMMA 5. *Let* φ *be empty, let* $\mathrm{f}(\boldsymbol{u})$ *be a primitive computable function which maps* ω *into* ω. *Then the restriction of* $\mathrm{f}(\boldsymbol{u})$ *to* ω *is primitive recursive in the sense of* [4].

Proof. We prove the lemma for the case $B \neq \varnothing$, the case $B = \varnothing$ being similar and a little easier.

Let $b \in B$ be a fixed element of B. We assign to each element x of $b^* = \{b\}^* = \{z \in B^* : D_z \subset \{b\}\}$ an integer $\mathrm{c}x$ by the induction:

$$\begin{aligned} &\text{(a)} \quad \mathrm{c}0 = 1, \\ &\text{(b)} \quad \mathrm{c}b = 2, \\ &\text{(c)} \quad \mathrm{c}(s, t) = 3^{\mathrm{cs}} \cdot 5^{\mathrm{ct}}. \end{aligned} \tag{3.10}$$

Let P be the set of integers $\mathrm{c}x$ which are assigned to elements x of b^*. To each function $\mathrm{f}(\boldsymbol{u})$ from b^* into b^* we assign the number-theoretic function $\mathrm{cf}(v_1, \ldots, v_k)$ by

$$\begin{aligned} \mathrm{cf}(v_1, \ldots, v_k) &= 0 && \text{if for some } i,\ i = 1, \ldots, k,\ v_i \notin P, \\ &= \mathrm{c}(\mathrm{f}(u_1, \ldots, u_k)) && \text{if } v_1 = \mathrm{c}u_1, \ldots, v_k = \mathrm{c}u_k. \end{aligned} \tag{3.11}$$

Now suppose that $\mathrm{f}(\boldsymbol{u}) = \{f\}_{\mathrm{pr}}(\boldsymbol{u})$ is primitive computable from b, i.e. $f \in b^*$. Lemma 1 implies that f carries b^* into b^*, so the corresponding number-theoretic function cf is well-defined. It is easy to show by induction on $f \in PRI$ using elementary properties of primitive recursive functions, that cf is primitive recursive. On the other hand, if f maps ω into ω, then the restriction of f to ω can be obtained from cf by a simple primitive recursive transformation, hence this restriction is primitive recursive. This concludes the proof of the lemma for functions primitive computable from b.

We assign to each element z of B^* its projection $z^\#$ into b^* by the induction:

$$\begin{aligned} &\text{(a)} \quad 0^\# = 0, \\ &\text{(b)} \quad y^\# = b \quad \text{if } y \in B, \\ &\text{(c)} \quad (s, t)^\# = (s^\#, t^\#). \end{aligned} \tag{3.12}$$

It is easy to show by induction on $z \in B^*$ that

$$\pi(z^{\#}) = (\pi z)^{\#}, \quad \delta(z^{\#}) = (\delta z)^{\#}. \tag{3.13}$$

Using (3.13) then we easily show by induction on $f \in PRI$ that for φ empty,

$$\text{if } \{f\}_{\mathrm{pr}}(\boldsymbol{u}) \to z, \text{ then } \{f^{\#}\}_{\mathrm{pr}}(\boldsymbol{u}^{\#}) \to z^{\#}, \tag{3.14}$$

where $\boldsymbol{u}^{\#} = u_1^{\#}, \ldots, u_k^{\#}$. Let us associate then with each primitive computable function $\mathrm{f}(\boldsymbol{u}) = \{f\}_{\mathrm{pr}}(\boldsymbol{u})$ the function $\mathrm{f}^{\#}(\boldsymbol{u})$, primitive computable from b, given by

$$\mathrm{f}^{\#}(\boldsymbol{u}) = \{f^{\#}\}_{\mathrm{pr}}(\boldsymbol{u}).$$

Since the function $^{\#}$ is the identity on b^*, if $u_1, \ldots, u_k, z \in b^*$, and $\{f\}_{\mathrm{pr}}(\boldsymbol{u}) \to z$, then $\{f^{\#}\}_{\mathrm{pr}}(\boldsymbol{u}) \to z$, so that f *agrees with* $\mathrm{f}^{\#}$ *on those k-tuples from* b^* *which* f *maps into* b^*. Thus if f maps ω into ω, then f agrees with $\mathrm{f}^{\#}$ on ω, so the restriction of f to ω is primitive recursive.

LEMMA 6. (a) *Let* $\varphi = \varphi_1, \ldots, \varphi_l$, $\varphi' = \varphi'_1, \ldots, \varphi'_{l'}$, *assume that for each* $i = 1, \ldots, l$ *there is a* j $(1 \leqq j \leqq l')$ *such that* $\varphi_i = \varphi'_j$. *Then for each* $A \subset B$, $PR(A, \varphi) \subset PR(A, \varphi')$.

(b) *If* $\mathrm{f} \in PR(A, \mathrm{g}_1, \ldots, \mathrm{g}_m, \varphi)$ *and for each* $i = 1, \ldots, m$, $\mathrm{g}_i \in PR(A, \varphi)$, *then* $\mathrm{f} \in PR(A, \varphi)$.

Both of these statements are proved by easy inductions on an index $f \in A^*$ of the function in question which we omit.

LEMMA 7. *There is a functional*

$$\mathrm{f}(\boldsymbol{u}) = \mathrm{f}(\mathrm{g}_1, \ldots, \mathrm{g}_m, \mathrm{h}_1, \ldots, \mathrm{h}_m, \boldsymbol{u}) \tag{3.15}$$

which is absolutely primitive computable and such that if $\mathrm{g}_1, \ldots, \mathrm{g}_m, \mathrm{h}_1, \ldots, \mathrm{h}_m$ *are completely defined and single-valued and if for each* $\boldsymbol{u}$ *there is exactly one* i *such that* $\mathrm{h}_i(\boldsymbol{u}) = 0$, *then*

$$\begin{aligned} \mathrm{f}(\boldsymbol{u}) &= \mathrm{g}_1(\boldsymbol{u}) \quad \text{if } \mathrm{h}_1(\boldsymbol{u}) = 0, \\ &= \mathrm{g}_2(\boldsymbol{u}) \quad \text{if } \mathrm{h}_2(\boldsymbol{u}) = 0, \\ &= \cdots \qquad \cdots \\ &= \mathrm{g}_m(\boldsymbol{u}) \quad \text{if } \mathrm{h}_m(\boldsymbol{u}) = 0. \end{aligned} \tag{3.16}$$

(*Definition by cases.*)

Proof. First set by C6 and Lemma 2

$$\begin{aligned} \mathrm{p}_1(u, y) &= 0 \quad \text{if } y \in B^0, \\ \mathrm{p}_1(u, y) &= u \quad \text{if } y \notin B^0, \end{aligned}$$

and then put $\mathrm{p}(u, y) = \mathrm{p}_1(u, 1 \dot{-} \chi_0(y))$ so that

$$\begin{aligned} \mathrm{p}(u, y) &= u \quad \text{if } y = 0, \\ &= 0 \quad \text{if } y \neq 0. \end{aligned} \tag{3.17}$$

Similarly the function

$$\text{(3.18)}\qquad \begin{aligned} \mathrm{q}(u, v) &= u \quad \text{if } v = 0, \\ &= v \quad \text{if } v \neq 0 \end{aligned}$$

is also absolutely primitive computable. We define by induction on k the functions $\mathrm{r}_k(u_1, \ldots, u_k)$,

$$\text{(3.19)}\qquad \mathrm{r}_1(u_1) = u_1, \quad \mathrm{r}_{k+1}(u_1, \ldots, u_{k+1}) = \mathrm{q}(u_{k+1}, \mathrm{r}_k(u_1, \ldots, u_k)).$$

It is immediate by induction on k that *if* $u_j = 0$, $j = 1, \ldots, \mathrm{k}$, *then* $\mathrm{r}_k(u_1, \ldots, u_k) = 0$ *and if* $u_j = 0, j = 1, \ldots, k, j \neq i$, *and* $u_i \neq 0$, *then* $\mathrm{r}_k(u_1, \ldots, u_k) = u_i$. The lemma follows by setting

$$\text{(3.20)}\qquad \mathrm{f}(\boldsymbol{u}) = \mathrm{r}_m(\mathrm{p}(\mathrm{g}_1(\boldsymbol{u}), \mathrm{h}_1(\boldsymbol{u})), \ldots, \mathrm{p}(\mathrm{g}_m(\boldsymbol{u}), \mathrm{h}_m(\boldsymbol{u}))).$$

REMARK 5. It is immediate that for a fixed sequence $\mathrm{g}_1, \ldots, \mathrm{g}_m, \mathrm{h}_1, \ldots, \mathrm{h}_m$ of functions, the function $\mathrm{f}(\boldsymbol{u})$ defined by cases in (3.14) is in

$$PR(\mathrm{g}_1, \ldots, \mathrm{g}_m, \mathrm{h}_1, \ldots, \mathrm{h}_m).$$

Thus by Lemma 6, if $\mathrm{g}_1, \ldots, \mathrm{g}_m, \mathrm{h}_1, \ldots, \mathrm{h}_m$ are in $PR(A, \varphi)$, then $\mathrm{f}(\boldsymbol{u}) \in PR(A, \varphi)$.

Clearly we can relax somewhat the hypothesis of Lemma 7, for example by allowing some of the g_i to be multiple-valued. However the lemma fails for arbitrary p.m.v. functions and we shall only use it in this form.

LEMMA 8. *There is an absolutely primitive computable functional*

$$\mathrm{f}(y, \boldsymbol{x}) = \mathrm{f}(\mathrm{g}, y, \boldsymbol{x})$$

such that for fully defined single-valued g *and all* $y, \boldsymbol{x}$,

$$\text{(3.21)}\qquad \mathrm{f}(y, \boldsymbol{x}) = \mathrm{g}(\bar{\mathrm{f}}(y, x), y, \boldsymbol{x}).$$

Proof. We first set up an induction for $\bar{\mathrm{f}}(y, \boldsymbol{x})$,

$$\text{(3.22)}\qquad \begin{aligned} &\bar{\mathrm{f}}(y, \boldsymbol{x}) = 1 \quad \text{if } y \in B^0, \\ &\bar{\mathrm{f}}((s, t), \boldsymbol{x}) = \langle \mathrm{g}(\bar{\mathrm{f}}(s, \boldsymbol{x}), s, \boldsymbol{x}), \mathrm{g}(\bar{\mathrm{f}}(t, \boldsymbol{x}), t, \boldsymbol{x}), \bar{\mathrm{f}}(s, \boldsymbol{x}), \bar{\mathrm{f}}(t, \boldsymbol{x}) \rangle \end{aligned}$$

and then put

$$\text{(3.23)}\qquad \mathrm{f}(y, \boldsymbol{x}) = (\bar{\mathrm{f}}((y, 0), \boldsymbol{x}))_1.$$

LEMMA 9. (a) *There is an absolutely primitive computable functional*

$$\mathrm{f}(i, x) = \mathrm{f}(\mathrm{g}, \mathrm{h}, \mathrm{p}, i, x)$$

which for completely defined single-valued g, h, p *satisfies the nested recursion over* ω:

$$\text{(3.24)}\qquad \begin{aligned} \mathrm{f}(i, x) &= 0 && \text{if } i \notin \omega, \\ &= \mathrm{g}(x) && \text{if } i = 0, \\ &= \mathrm{h}(\mathrm{f}(j, \mathrm{p}(j, x)), j, x) && \text{if } i = j+1 \in \omega. \end{aligned}$$

In particular, if g, h, p *are in* $PR(A, \varphi)$, *so is* f.

(b) *The predicate* Seq (x) *defined by* (1.15) *and the function* $(x)_i$ *defined by* (1.10) $(=0$ *if* $i \notin \omega)$ *are combinatorial.*

Proof of (a) is similar to the one for ordinary recursion theory and will be omitted. To prove (b) we notice that (1.12) gives a nested recursion for the characteristic function of Seq (n, x), hence Seq (n, x) and thus Seq (x) are combinatorial. Similarly, (1.10) gives a nested recursion for $(x)_i$.

LEMMA 10. *If the predicates* $P(\boldsymbol{u})$ *and* $Q(\boldsymbol{u})$ *are in* $PR(A, \varphi)$, *then so are the predicates* $-P(\boldsymbol{u})$, $P(\boldsymbol{u})$ & $Q(\boldsymbol{u})$, $P(\boldsymbol{u}) \vee Q(\boldsymbol{u})$, $P(\boldsymbol{u}) \Rightarrow Q(\boldsymbol{u})$. *If* $P(i, \boldsymbol{x})$ *is in* $PR(A, \varphi)$, *then so are the predicates*

$$(i)_{i \leqq j} P(i, \boldsymbol{x}) \Leftrightarrow (i)[\mathrm{i} \in \omega \ \& \ i \leqq j \Rightarrow P(i, \boldsymbol{x})],$$
$$(Ei)_{i \leqq j} P(i, \boldsymbol{x}) \Leftrightarrow (Ei)[i \in \omega \ \& \ i \leqq j \ \& \ \mathrm{P}(i, \boldsymbol{x})],$$

and similarly with "$i<j$", "$j_1 \leqq i \leqq j_2$" *etc. in the place of* "$i \leqq j$."

Proof is easy using Lemmas 3 and 4.

4. **Index constructions.** It is convenient to develop for primitive computability some of the theory of "arithmetization" which in ordinary recursion theory is usually done for partial recursiveness.

LEMMA 11. *The sets PRI and* PRI_k,

$$f \in PRI_k \Leftrightarrow f \in PRI \ \& \ (f)_2 = k, \tag{4.1}$$

are combinatorial.

Proof. One can easily rewrite the clauses for the definition of *PRI* which we gave in §2 to obtain a course of values induction for the characteristic function of *PRI* and then use Lemma 8.

LEMMA 12. *For each* m *there is a combinatorial function* $S^m(f, y_1, \ldots, y_m) = S^m(f, \boldsymbol{y})$ *such that*

$$\{f\}_{\mathrm{pr}}(\boldsymbol{y}, \boldsymbol{x}) = \{S^m(f, \boldsymbol{y})\}_{\mathrm{pr}}(\boldsymbol{x}). \tag{4.2}$$

(*The* S^m *theorem.*)

Proof. We define the functions $S^m(f, \boldsymbol{y})$ by induction on m, using Kleene's method in XIII of [6]. Put

$$S^1(f, y) = \langle 5, (f)_2 \dot{-} 1, f, \langle 1, (f)_2 \dot{-} 1, y \rangle \rangle,$$
$$S^{m+1}(f, y_1, \ldots, y_{m+1}) = S^1(S^m(f, y_1, \ldots, y_m), y_{m+1}).$$

LEMMA 13. (a) *There is a combinatorial function* $\mathrm{ev}(f, k')$ *such that, with* $\boldsymbol{u} = u_1, \ldots, u_k$, $\boldsymbol{u}' = u'_1, \ldots, u'_{k'}$, *if* $f \in PRI$ *and* $(f)_2 = k$, *then*

$$\{\mathrm{ev}(f, k')\}_{\mathrm{pr}}(\boldsymbol{u}, \boldsymbol{u}') = \{f\}_{\mathrm{pr}}(\boldsymbol{u}). \tag{4.3}$$

(b) *There is a combinatorial function* $\mathrm{fv}(f, k')$ *such that, with* $\boldsymbol{u}=u_1, \ldots, u_k$, $\boldsymbol{u}'=u'_1, \ldots, u'_{k'}$, *if* $f \in PRI$ *and* $(f)_2=k$, *then*

$$\{\mathrm{fv}(f, k')\}_{\mathrm{pr}}(\boldsymbol{u}', \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(\boldsymbol{u}). \tag{4.4}$$

Proof. (a) We define $\mathrm{ev}(f, k')$ by a course of values induction of f, using Lemma 8. The defining function g of Lemma 8 is given by cases, using Lemma 7. We give (these cases informally, omitting the details of casting them in a formal definition suitable for an application of Lemma 8.

The case hypotheses are the same C0–C7 which we formulated in the definition of *PRI* in §2. Here we do not repeat these combinatorial conditions on f but simply remind ourselves what the notation is in each case by writing down the defining condition for $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$.

Case C0. $\{f\}(t_1, \ldots, t_{n_i}, \boldsymbol{x})=\varphi_i(t_1, \ldots, t_{n_i})$. Put $\mathrm{ev}(f, k')=\langle 0, n_i+n+k', i\rangle$.

Case C1. $\{f\}(\boldsymbol{x})=y$. Put $\mathrm{ev}(f, k')=\langle 1, n+k', y\rangle$.

Case C2. $\{f\}(y, \boldsymbol{x})=y$. Put $\mathrm{ev}(f, k')=\langle 2, n+1+k'\rangle$.

Case C3. $\{f\}(s, t, \boldsymbol{x})=(s, t)$. Put $\mathrm{ev}(f, k')=\langle 3, n+2+k'\rangle$.

Case $C4_0$. $\{f\}(y, \boldsymbol{x})=\pi y$. Put $\mathrm{ev}(f, k')=\langle 4, n+1+k', 0\rangle$.

Case $C4_1$. $\{f\}(y, \boldsymbol{x})=\delta y$. Put $\mathrm{ev}(f, k')=\langle 4, n+1+k', 1\rangle$.

Case C5. $\{f\}(\boldsymbol{x})=\{g\}(\{h\}(\boldsymbol{x}), \boldsymbol{x})$. Put $\mathrm{ev}(f, k')=\langle 5, n+k', \mathrm{ev}(g, k'), \mathrm{ev}(h, k')\rangle$.

Case C6. $\{f\}(y, \boldsymbol{x})=\{g\}(y, \boldsymbol{x})=$ if $y \in B^0$,
$\{f\}((s, t), \boldsymbol{x})=\{h\}(\{f\}(s, \boldsymbol{x}), \{f\}(t, \boldsymbol{x}), s, t, \boldsymbol{x})$.
Put $\mathrm{ev}(f, k')=\langle 6, n+1+k', \mathrm{ev}(g, k'), \mathrm{ev}(h, k')\rangle$.

Case C7. $\{f\}(x)=\{g\}(x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n)$.
Put $\mathrm{ev}(f, k')=\langle 7, n+k', j, \mathrm{ev}(g, k')\rangle$.

Otherwise. Put $\mathrm{ev}(f, k')=0$.

It is now easy to prove (4.3) by course of values induction on $f \in PRI$.

To prove (b) we first define an auxiliary function $\mathrm{p}(f, i)$ by the induction:

$$\mathrm{p}(f, 0) = f, \quad \mathrm{p}(f, i+1) = \langle 7, (f)_2, (f)_2 \dot{-} 1, \mathrm{p}(f, i)\rangle. \tag{4.5}$$

It is easy to show by induction on i, $0 \leqq i \leqq k$, that if $(f)_2=k+1$ and $f \in PRI$, then

$$\{\mathrm{p}(f, i)\}_{\mathrm{pr}}(y, \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(u_{k-i+1}, u_{k-i+2}, \ldots, u_k, y, u_1, \ldots, u_{k-i}); \tag{4.6}$$

hence

$$\{\mathrm{p}(f, (f)_2 \dot{-} 1)\}_{\mathrm{pr}}(y, \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(\boldsymbol{u}, y), \tag{4.7}$$

if $k=(f)_2 \dot{-} 1$.

We define $\mathrm{fv}(f, k')$ by induction on k' as follows:

$$\mathrm{fv}(f, 0) = f, \quad \mathrm{fv}(f, k'+1) = \mathrm{p}(\mathrm{ev}(\mathrm{fv}(f, k'), 1), (f)_2+k'). \tag{4.8}$$

Now (4.4) is immediate for $k'=0$. Assuming it for k', we have

$$\begin{aligned}\{\mathrm{fv}(f, k'+1)\}_{\mathrm{pr}}(y, \boldsymbol{u}', \boldsymbol{u}) &= \{\mathrm{ev}(\mathrm{fv}(f, k'), 1)\}_{\mathrm{pr}}(\boldsymbol{u}', \boldsymbol{u}, y)\\ &= \{\mathrm{fv}(f, k')\}_{\mathrm{pr}}(\boldsymbol{u}', \boldsymbol{u})\\ &= \{f\}_{\mathrm{pr}}(\boldsymbol{u}).\end{aligned}$$

LEMMA 14. *There is a combinatorial function* $\mathrm{rc}_{\mathrm{pr}}(f)$ *such that if* $m=\mathrm{rc}_{\mathrm{pr}}(f)$, *then*

$$\{f\}_{\mathrm{pr}}(m, \boldsymbol{u}) = \{m\}_{\mathrm{pr}}(\boldsymbol{u}). \tag{4.9}$$

(*The recursion theorem.*)

Proof. Using Lemmas 12, 13 and C7, we can define combinatorial functions $\mathrm{m}_0(f)$ and $\mathrm{m}_1(f)$ such that

$$\{\mathrm{m}_0(f)\}_{\mathrm{pr}}(y, \boldsymbol{u}) = S^1(y, y), \tag{4.10}$$

$$\{\mathrm{m}_1(f)\}_{\mathrm{pr}}(t, y, \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(t, \boldsymbol{u}), \tag{4.11}$$

(pick g so that $\{g\}_{\mathrm{pr}}(y)=S^1(y, y)$, set $\mathrm{m}_0(f)=\mathrm{ev}(g, (f)_2 \dot{-} 1)$, $\mathrm{m}_1(f)=\langle 7, (f)_2+1, 1, \mathrm{fv}(f, 1)\rangle$). Put

$$\mathrm{m}_2(f) = \langle 5, (f)_2, \mathrm{m}_1(f), \mathrm{m}_0(f)\rangle; \tag{4.12}$$

then

$$\begin{aligned}\{\mathrm{m}_2(f)\}_{\mathrm{pr}}(y, \boldsymbol{u}) &= \{\mathrm{m}_1(f)\}_{\mathrm{pr}}(\{\mathrm{m}_0(f)\}_{\mathrm{pr}}(y, \boldsymbol{u}), y, \boldsymbol{u})\\ &= \{\mathrm{m}_1(f)\}_{\mathrm{pr}}(S^1(y, y), y, \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(S^1(y, y), \boldsymbol{u}).\end{aligned}$$

Put

$$\mathrm{rc}_{\mathrm{pr}}(f) = S^1(\mathrm{m}_2(f), \mathrm{m}_2(f)); \tag{4.13}$$

then, if $m=\mathrm{rc}_{\mathrm{pr}}(f)$,

$$\{m\}_{\mathrm{pr}}(\boldsymbol{u}) = \{\mathrm{m}_2(f)\}_{\mathrm{pr}}(\mathrm{m}_2(f), \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(S^1(\mathrm{m}_2(f), \mathrm{m}_2(f)), \boldsymbol{u}) = \{f\}_{\mathrm{pr}}(m, \boldsymbol{u}).$$

5. **Kleene enumeration and prime computable functions.** In 3.2–3.4 of [6] the point is made that in any theory of computability there must be both "a modicum of elementary operations of computation" and "a means for reflecting upon computation procedures already set up." The primitive computable functions here, as the primitive recursive functions in [6] provide the elementary operations. Kleene's idea for a reflection principle in [6] is to enumerate the class of computable (partial) functions by indexing, as we have done here, and then to postulate that the enumerating function itself is computable. The same procedure applied in our general case leads to the class of *prime computable* functions on B^*, the smallest class of functions on B^* on which a reasonable theory of computability can be developed.

Let us add to schemata C0–C7 the new schema

C8 $$\mathrm{f}(e, \boldsymbol{x}, \boldsymbol{y}) = \{e\}(\boldsymbol{x}) \qquad \langle 8, n+m+1, n\rangle.$$

Instead of defining a set of indices for prime computable functions, we shall assign one prime computable function $\{f\}_{\mathrm{p}}(\boldsymbol{u})$ of k variables to each $f \in B^*$. Our method (following Kleene in [6]) is to interpret schemata C0–C8 as the clauses of an inductive definition of a predicate $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z$, as follows: (Again we include clauses C9′–C10′ for future reference. We omit the subscript $_{\mathrm{p}}$ in writing $\{f\}(\boldsymbol{u})$, since choices of other sets of clauses will lead later to other classes of functions which we shall name using curly brackets with other subscripts.)

C0′. *If* $\varphi_i(t_1, \ldots, t_{n_i}) \to z$, *then* $\{\langle 0, n_i+n, i\rangle\}(t_1, \ldots, t_{n_i}, \boldsymbol{x}) \to z$.

C1′. $\{\langle 1, n, y\rangle\}(\boldsymbol{x}) \to y$.

C2′. $\{\langle 2, n+1\rangle\}(y, \boldsymbol{x}) \to y$.

C3′. $\{\langle 3, n+2\rangle\}(s, t, \boldsymbol{x}) \to (s, t)$.

C4′$_0$. $\{\langle 4, n+1, 0\rangle\}(y, \boldsymbol{x}) \to \pi y$.

C4′$_1$. $\{\langle 4, n+1, 1\rangle\}(y, x) \to \delta y$.

C5′. *If there exists a* u *such that* $\{h\}(\boldsymbol{x}) \to u$ *and* $\{g\}(u, \boldsymbol{x}) \to z$, *then* $\{\langle 5, n, g, h\rangle\}(\boldsymbol{x}) \to z$.

C6′. *If* $y \in B^0$ *and* $\{g\}(y, \boldsymbol{x}) \to z$, *then* $\{\langle 6, n+1, g, h\rangle\}(y, \boldsymbol{x}) \to z$. *If there exist* u, v *such that* $\{\langle 6, n+1, g, h\rangle\}(s, \boldsymbol{x}) \to u$, $\{\langle 6, n+1, g, h\rangle\}(t, \boldsymbol{x}) \to v$ *and* $\{h\}(u, v, s, t, \boldsymbol{x}) \to z$, *then* $\{\langle 6, n+1, g, h\rangle\}((s, t), \boldsymbol{x}) \to z$.

C7′. *If* $\{g\}(x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n) \to z$, *then* $\{\langle 7, n, j, g\rangle\}(\boldsymbol{x}) \to z$.

C8′. *If* $\{e\}(\boldsymbol{x}) \to z$, *then* $\{\langle 8, n+m+1, n\rangle\}(e, \boldsymbol{x}, \boldsymbol{y}) \to z$.

C9′. *If* $\{g\}(y, \boldsymbol{x}) \to 0$, *then* $\{\langle 9, n, g\rangle\}(\boldsymbol{x}) \to y$.

C10′. *If* $(y)(Eu)[\{g\}(y, \boldsymbol{x}) \to u]$ & $(Ey)[\{g\}(y, \boldsymbol{x}) \to 0]$, *then* $\{\langle 10, n, g\rangle\}(\boldsymbol{x}) \to 0$. *If* $(y)(Eu)[u \neq 0$ & $\{g\}(y, \boldsymbol{x}) \to u]$, *then* $\{\langle 10, n, g\rangle\}(\boldsymbol{x}) \to 1$.

We define the predicate $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z$ by adding to clauses C0′–C8′ the

Recursion clause for prime computability: $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z$ *only if* $\{f\}(\boldsymbol{u}) \to z$ *follows from* C0′–C8′.

Each $f \in B^*$ now defines for each k a p.m.v. function of k variables

$$\{f\}_{\mathrm{p}}(\boldsymbol{u}) = \{z : \{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z\}.$$

For each $A \subset B^*$ we let $PC(A, \varphi)$ be the class of p.m.v. functions $\{f\}_{\mathrm{p}}(\boldsymbol{u})$, with $f \in A^*$. As before we distinguish the class $PC(\varphi) = PC(\varnothing, \varphi)$ of functions *absolutely prime computable* (*in* φ) and the class $\mathbf{PC}(\varphi) = PC(B, \varphi)$ of functions prime computable (in φ).

Again, each $f \in B^*$ defines for each list $l_1, \ldots, l_m$, k a functional $\{f\}_{\mathrm{p}}(\mathrm{g}_1, \ldots, \mathrm{g}_m, \boldsymbol{u})$, where $\mathrm{g}_1, \ldots, \mathrm{g}_m$ are variables for function variables of $l_1, \ldots, l_m$, arguments respectively. We call this functional prime computable from A (in φ), if $f \in A^*$, absolutely prime computable if $f \in 0^*$, etc.

It seems clear to us that each absolutely prime computable function is intuitively computable, if instructions for computing the functions in φ are given. Similarly, prime computable functions should be accepted as effectively computable by anyone who considers the constant functions on a set as computable. We shall discuss Church's Thesis in some detail in the next section.

In this paper we shall not study prime computability, except incidentally. We prove a few lemmas here which give a general picture of what the classes $PC(A, \varphi)$ look like.

Since the primitive computable functions $\{f\}_{\mathrm{pr}}(\boldsymbol{u})$ were defined by induction on $f \in PRI$, we could prove results about them by induction on $f \in PRI$. For prime computability our standard method of proof will be induction on the form of the definition of the predicate $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z$, which we call "induction on C0′–C8′".

LEMMA 15. *If each function φ_i in φ is single-valued, then each function in $PC(\varphi)$ is single-valued.*

Proof. We show by an easy induction on C0′–C8′ that if $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z$ and $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z'$, then $z = z'$.

Notice that $PC(\varphi)$ always contains partial functions, e.g. those introduced by C8, even if all functions in φ are totally defined.

LEMMA 16. *If $f \in PRI$, then $\{f\}_{\mathrm{pr}}(\boldsymbol{u}) = \{f\}_{\mathrm{p}}(\boldsymbol{u})$. In particular $PR(A, \varphi) \subset PC(A, \varphi)$.*

Proof is by induction on $f \in PRI$.

LEMMA 17. *If $f \in PR(A, \mathrm{g}_1, \ldots, \mathrm{g}_m, \varphi)$ and for each $i = 1, \ldots, m$, $\mathrm{g}_i \in PC(A, \varphi)$, then $\mathrm{f} \in PC(A, \varphi)$.*

Proof is by induction on an index f of f in $PR(A, \mathrm{g}_1, \ldots, \mathrm{g}_m, \varphi)$.

LEMMA 18. *Let $\mathrm{h}_1, \ldots, \mathrm{h}_m$ be totally defined single-valued functions in $PC(A, \varphi)$, let $\mathrm{g}_1, \ldots, \mathrm{g}_m$ be in $PC(A, \varphi)$, assume that for each $\boldsymbol{u}$ there is exactly one i so that $\mathrm{h}_i(\boldsymbol{u}) = 0$. Then the function*

$$\begin{aligned} \mathrm{f}(\boldsymbol{u}) &= \mathrm{g}_1(\boldsymbol{u}) \quad \text{if } \mathrm{h}_1(\boldsymbol{u}) = 0, \\ &= \mathrm{g}_2(\boldsymbol{u}) \quad \text{if } \mathrm{h}_2(\boldsymbol{u}) = 0, \\ &= \cdots \qquad \cdots \\ &= \mathrm{g}_m(\boldsymbol{u}) \quad \text{if } \mathrm{h}_m(\boldsymbol{u}) = 0 \end{aligned}$$

is in $PC(A, \varphi)$. (Definition of partial functions by cases.)

Proof. Let $g_1, \ldots, g_m$ be indices of $\mathrm{g}_1, \ldots, \mathrm{g}_m$, put

$$\begin{aligned} \mathrm{k}(\boldsymbol{u}) &= g_1 \quad \text{if } \mathrm{h}_1(\boldsymbol{u}) = 0, \\ &= g_2 \quad \text{if } \mathrm{h}_2(\boldsymbol{u}) = 0, \\ &= \cdots \qquad \cdots \\ &= g_m \quad \text{if } h_m(\boldsymbol{u}) = 0; \end{aligned}$$

now $\mathrm{k}(\boldsymbol{u})$ is in $PR(A, \varphi)$ by Lemmas 7 and 16. Now put $\mathrm{f}(\boldsymbol{u}) = \{\mathrm{k}(\boldsymbol{u})\}_{\mathrm{p}}(\boldsymbol{u})$ by C8.

REMARK 6. As a consequence of Lemma 17 the classes $PC(A, \varphi)$ are closed under all the primitive computable (from A) operations. For example, if f is defined from g by course of values induction as in Lemma 8 and g is (totally defined, single-valued) in $PC(A, \varphi)$, then f is also in $PC(A, \varphi)$. (We shall see later in Remark 12 that g need not be totally defined or single-valued for this to be true.)

Some of the lemmas about primitive computability were not stated in a "uniform" form in terms of functionals, as the lemmas referred to above, but in the form "if $f_1, \ldots, \in PR(\cdots)$, then $g_1, \ldots, \in PR(\cdots)$". Since the statements are about an arbitrary φ however, the effect is the same. For example suppose $f(x, u, y) = g(u)$ and $g \in PC(A, \varphi)$. Applying Lemma 2 to $\varphi' = g, \varphi$ we have $f \in PR(A, g, \varphi)$, since certainly $g \in PR(A, g, \varphi)$, hence by Lemma 17 again, $f \in PC(A, \varphi)$. Thus $PC(A, \varphi)$ is closed under the operation of introducing new variables, and similarly the class of predicates in $PC(A, \varphi)$ is closed under all the logical operations listed in Lemma 10.

The definition of the predicate $\{f\}_p(u) \to z$ on finite sequences f, u, z of elements of B^* was given by the inductive clauses C0′–C8′, much as the definition of the set O of ordinal notations in [5] was given by the inductive clauses $O1$–$O4$. Such inductive definitions may be interpreted "from the outside" in the manner of Wang's computation of an explicit form of O in [24]. Alternatively, they may be understood as (possibly transfinite) inductions on an initial segment of the ordinals in the following way: the defined set $\mathscr{A}$ is considered as the union $\bigcup_\xi \mathscr{A}_\xi$, where $\mathscr{A}_0$ consists of the objects which are in $\mathscr{A}$ because of the direct clauses in the definition and $\mathscr{A}_\xi$ consists of those objects whose membership in $\mathscr{A}$ follows when we apply the indirect clauses of the definition to elements in $\bigcup_{\eta<\xi} \mathscr{A}_\eta$.

In such an analysis there is associated with each $x \in \mathscr{A}$ an ordinal, namely the least ξ such that $x \in \mathscr{A}_\xi$. Classically, proof by induction over an inductive definition is simply proof by (possibly transfinite) induction on ξ. We prefer to give proofs by induction over a definition rather than by (transfinite) induction on the relevant ordinals, since the first procedure appeals to us as more "constructive", in the case where infinite ordinals are involved. However, for some results that we need we have no such proofs and we are forced to introduce the ordinals.

We assign to each sequence f, u, z such that $\{f\}_p(u) \to z$ the ordinal $|f, u, z|_p$ at which we first recognize that $\{f\}_p(u) \to z$. The definition is cast in the form of an induction on C0′–C8′, by clauses C0″–C8″. As before we omit the subscript $_p$ and include clauses C9″ and C10″ for later use.

C0″–C4″. *If* $\{f\}(u) \to z$ *by virtue of clauses* C0′–C4′, *then* $|f, u, z| = 0$.

C5″. *If there exists a* u *such that* $\{h\}(x) \to u$ *and* $\{g\}(u, x) \to z$, *then*

$$|\langle 5, n, g, h\rangle, x, z| = \textit{infimum}\,\{\textit{max}\,[|h, x, u| + 1, |g, u, x, z| + 1] : \{h\}(x) \to u \;\&\; \{g\}(u, x) \to z\}.$$

C6″. *If* $\{g\}(y, x) \to z$ *and* $y \in B^0$ *then* $|\langle 6, n+1, g, h\rangle, y, x, z| = |g, y, x, z| + 1$.

If there exist u, v *such that* $\{f\}(s, \boldsymbol{x}) \to u$, $\{f\}(t, \boldsymbol{x}) \to v$, $\{h\}(u, v, s, t, \boldsymbol{x}) \to z$, *then*

$$|f, (s, t), \boldsymbol{x}, z| = \textit{infimum}\,\{\textit{max}\,[|f, s, \boldsymbol{x}, u|+1, |f, t, \boldsymbol{x}, v|+1, |h, u, v, s, t, \boldsymbol{x}, z|+1] : \{f\}(s, \boldsymbol{x}) \to u, \{f\}(t, \boldsymbol{x}) \to v \textit{ and } \{h\}(u, v, s, t, \boldsymbol{x}) \to z\},$$

where $f = \langle 6, n+1, g, h\rangle$.

C7″. *If* $\{g\}(x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n) \to z$, *then*

$$|\langle 7, n, j, g\rangle, \boldsymbol{x}, z| = |g, x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n|+1.$$

C8″. *If* $\{e\}(\boldsymbol{x}) \to z$, *then* $|\langle 8, n+m+1, n\rangle, e, \boldsymbol{x}, \boldsymbol{y}, z| = |e, \boldsymbol{x}, z|+1$.

C9″. *If* $\{g\}(y, x) \to 0$, *then* $|\langle 9, n, g\rangle, \boldsymbol{x}, y| = |g, y, \boldsymbol{x}, 0|+1$.

C10″. *If* $(y)(Eu)[\{g\}(y, \boldsymbol{x}) \to u]$ & $(Ey)[\{g\}(y, \boldsymbol{x}) \to 0]$, *then*

$$|\langle 10, n, g\rangle, x, 0| = \textit{infimum}\,\{\textit{supremum}\,[|g, y, \boldsymbol{x}, \alpha(y)|+1 : \textit{all } y] : (y)[\{g\}(y, \boldsymbol{x}) \to \alpha(y)] \,\&\, (Ey)\alpha(y) = 0\};$$

if $(y)(Eu)[\{g\}(y, \boldsymbol{x}) \to u \,\&\, u \neq 0]$, *then*

$$|\langle 10, n, g\rangle, \boldsymbol{x}, 1| = \textit{infimum}\,\{\textit{supremum}\,[|g, y, \boldsymbol{x}, \alpha(y)|+1 : \textit{all } y] : (y)[\{g\}(y, \boldsymbol{x}) \to \alpha(y) \neq 0]\}.$$

(*Here* α *varies over all one-place total functions on* B^* *to* B^*.)

In clause C5″ our analysis is as follows. We may recognize that $\{g\}(\{h\}(\boldsymbol{x}), \boldsymbol{x}) \to z$ because of various possible u's such that $\{h\}(\boldsymbol{x}) \to u$ and $\{g\}(u, \boldsymbol{x}) \to z$. The ordinal at which we recognize that $\{\langle 5, n, g, h\rangle\}(\boldsymbol{x}) \to z$ for the first time is just one step after we have first found a u such that both $\{h\}(\boldsymbol{x}) \to u$ and $\{g\}(u, \boldsymbol{x}) \to z$; this is why the infimum is taken in the definition. The same remark applies to the second part of clause C6″.

REMARK 7. It is our use of multiple-valued functions that requires the infimum operation in the assignment of ordinals when we analyse the definition C0′–C8′, since if all functions were single-valued, then there would be at most one u satisfying the conditions above. Multiple-valued functions are not required for the study of prime computability, because of Lemma 15; however they are necessary in studying search computability in the next section. The use of existential, nondeterministic clauses in inductive definitions and the assignment of ordinals by the infimum operation that they make necessary is the key to the abstract treatment of the hyperprojective hierarchy in §10–§19.

It is immediate from clauses C0″–C8″ that for every f, $\boldsymbol{u}$, z such that $\{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z$, $|f, \boldsymbol{u}, z|_{\mathrm{p}} < \omega$.

LEMMA 19. *The functions* $S^m(f, y_1, \ldots, y_m)$ *defined in Lemma* 12 *satisfy*

$$\{f\}_{\mathrm{p}}(\boldsymbol{y}, \boldsymbol{x}) = \{S^m(f, \boldsymbol{y})\}_{\mathrm{p}}(\boldsymbol{x}). \tag{5.1}$$

Moreover, if $\{f\}_{\mathrm{p}}(\boldsymbol{y}, \boldsymbol{x}) \to z$, *then* $|f, \boldsymbol{y}, \boldsymbol{x}, z|_{\mathrm{p}} < |S^m(f, \boldsymbol{y}), \boldsymbol{x}, z|_{\mathrm{p}}$.

Proof of the first part is immediate, as that of Lemma 12. The assertion about the ordinals $|\ \ |_p$ is immediate about S^1 and follows in general by an easy induction on m.

LEMMA 20. (a) *There is a combinatorial function* $\mathrm{EV}(f, k')$ *such that, with* $\boldsymbol{u}=u_1, \ldots, u_k$, $\boldsymbol{u}'=u'_1, \ldots, u'_{k'}$, *we have*

$$\{\mathrm{EV}(f, k')\}_p(\boldsymbol{u}, \boldsymbol{u}') = \{f\}_p(\boldsymbol{u}). \tag{5.2}$$

Moreover, if $\{f\}_p(\boldsymbol{u}) \to z$, *then* $|f, \boldsymbol{u}, z|_p < |\mathrm{EV}(f, k'), \boldsymbol{u}, \boldsymbol{u}', z|_p$.

(b) *There is a combinatorial function* $\mathrm{FV}(f, k')$ *such that, with* $\boldsymbol{u} = u_1, \ldots, u_k$, $\boldsymbol{u}' = u'_1, \ldots, u'_{k'}$, *we have*

$$\{\mathrm{FV}(f, k')\}_p(\boldsymbol{u}', \boldsymbol{u}) = \{f\}_p(\boldsymbol{u}). \tag{5.3}$$

Moreover if $\{f\}_p(\boldsymbol{u}) \to z$, *then* $|f, \boldsymbol{u}, z|_p < |\mathrm{FV}(f, k'), \boldsymbol{u}', \boldsymbol{u}, z|_p$.

Proof. To prove (a) we notice that

$$\{\langle 8, (f)_2+k'+1, (f)_2\rangle\}_p(e, \boldsymbol{u}, \boldsymbol{u}') = \{e\}_p(\boldsymbol{u}),$$
$$\{\langle 1, (f)_2+k', f\rangle\}_{pr}(\boldsymbol{u}, \boldsymbol{u}') = f,$$

so that it is sufficient to set

$$\mathrm{EV}(f, k') = \langle 5, (f)_2+k', \langle 8, (f)_2+k'+1, (f)_2\rangle, \langle 1, (f)_2+k', f\rangle\rangle. \tag{5.4}$$

The assertion about ordinals is immediate.

To prove (b) we use the function $\mathrm{p}(f, i)$ defined by (4.5) in the proof of Lemma 13. We set

$$\begin{aligned} \mathrm{FV}(f, 0) &= \langle 5, (f)_2, \langle 2, (f)_2+1\rangle, f\rangle, \\ \mathrm{FV}(f, k'+1) &= \mathrm{p}(\mathrm{EV}(\mathrm{FV}(f, k'), 1), (f)_2+k'), \end{aligned} \tag{5.5}$$

and prove (5.3) in the same way that we proved (4.4), noticing that the basic property (4.7) of $\mathrm{p}(f, i)$ holds with the subscript $_p$ rather than $_{pr}$. The assertion about ordinals is again immediate because of our definition of $\mathrm{FV}(f, 0)$.

LEMMA 21. *There is a combinatorial function* $\mathrm{rc}(f)$ *such that for each* f, *if* $m=\mathrm{rc}(f)$, *then*

$$\{f\}_p(m, \boldsymbol{u}) = \{m\}_p(\boldsymbol{u}). \tag{5.6}$$

(*The recursion theorem.*)

Proof is similar to that of Lemma 14, using Lemmas 19, 20 instead of 12, 13.

LEMMA 22. *There is a combinatorial function* $\mathrm{tr}_p(f, c)$ *such that if* $\chi(\boldsymbol{u}')=\{c\}_p(\boldsymbol{u}')$ *and* $\mathrm{f}(\boldsymbol{u})=\{f\}_p(\chi, \boldsymbol{u})$, *then* $\mathrm{f}(\boldsymbol{u})=\{\mathrm{tr}_p(f, c)\}_p(\boldsymbol{u})$. *Thus, if* $\mathrm{f} \in PC(A, \mathrm{g}_1, \ldots, \mathrm{g}_m, \varphi)$ *and* $\mathrm{g}_1, \ldots, \mathrm{g}_m \in PC(A, \varphi)$, *then* $\mathrm{f} \in PC(A, \varphi)$. (*Transitivity lemma.*)

Proof. We first define an absolutely prime computable function $g(p, f, c)$, then choose an element tr_p of 0* such that $g(tr_p, f, c) = \{tr_p\}_p(f, c)$ by Lemma 21 and set

$$\mathrm{tr}_p(f, c) = \{tr_p\}_p(f, c); \tag{5.7}$$

once tr_p is chosen it will be obvious from the definition of $g(p, f, c)$ that $g(tr_p, f, c)$ is in fact combinatorial.

The definition of $g(p, f, c)$ is by ten cases corresponding to the nine forms C0–C8 that f may have as an index of a prime computable function and an Otherwise case by Lemma 18. We identify the cases C0–C8 by writing the form of definition of $\{f\}_p(\chi, \boldsymbol{u})$ according to C0–C8 each time rather than the combinatorial condition that f must satisfy.

In the definition we avoid the notation "$g(p, f, c)$" and the variable "p"; instead we write "$\mathrm{tr}_p(f, c)$" and "tr_p" from the beginning as if the recursion theorem had already been applied. In effect we are defining a function $\mathrm{tr}_p(f, c)$ of the variables tr_p, f, c, and then we fix the variable tr_p to be an index of $\mathrm{tr}_p(f, c)$ by Lemma 21.

Case C0. *Subcase* (a). $\{f\}(\chi, \boldsymbol{u}', \boldsymbol{x}) = \chi(\boldsymbol{u}')$.

Put $\mathrm{tr}_p(f, c) = \mathrm{EV}(c, (f)_2 \dot{-} (c)_2)$.

Case C0. *Subcase* (b). $\{f\}(\chi, t_1, \ldots, t_{n_i}, \boldsymbol{x}) = \varphi_i(t_1, \ldots, t_{n_i})$.

Put $\mathrm{tr}_p(f, c) = \langle 0, n_i + n, i\rangle$.

(REMARK. The case hypothesis for Case C0, *Subcase* (a) is $f = \langle 0, k' + n, 1\rangle$, where $\boldsymbol{x} = x_1, \ldots, x_n$, $\boldsymbol{u}' = u'_1, \ldots, u'_{k'}$, and the case hypothesis for Case C0, *Subcase* (b) is $f = \langle 0, n_i + n, i + 1\rangle$, since φ_i is the $(i+1)$th function in the list $\chi, \boldsymbol{\varphi}$.)

Case C1. $\{f\}(\chi, \boldsymbol{x}) = y$. Put $\mathrm{tr}_p(f, c) = f$.

Case C2. $\{f\}(\chi, y, \boldsymbol{x}) = y$. Put $\mathrm{tr}_p(f, c) = f$.

Case C3. $\{f\}(\chi, s, t, \boldsymbol{x}) = (s, t)$. Put $\mathrm{tr}_p(f, c) = f$.

Case $C4_0$. $\{f\}(\chi, y, \boldsymbol{x}) = \pi y$. Put $\mathrm{tr}_p(f, c) = f$.

Case $C4_1$. $\{f\}(\chi, y, \boldsymbol{x}) = \delta y$. Put $\mathrm{tr}_p(f, c) = f$.

Case C5. $\{f\}(\chi, \boldsymbol{x}) = \{g\}(\chi, \{h\}(\chi, \boldsymbol{x}), \boldsymbol{x})$.
Put $\mathrm{tr}_p(f, c) = \langle 5, n, \mathrm{tr}_p(g, c), \mathrm{tr}_p(h, c)\rangle$.

Case C6. $\{f\}(\chi, y, \boldsymbol{x}) = \{g\}(\chi, y, \boldsymbol{x})$ if $y \in B^0$,
$\{f\}(\chi, (s, t), \boldsymbol{x}) = \{h\}(\chi, \{f\}(\chi, s, \boldsymbol{x}), \{f\}(\chi, t, \boldsymbol{x}), s, t, \boldsymbol{x})$.
Put $\mathrm{tr}_p(f, c) = \langle 6, n+1, \mathrm{tr}_p(g, c), \mathrm{tr}_p(h, c)\rangle$.

Case C7. $\{f\}(\chi, \boldsymbol{x}) = \{g\}(\chi, x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n)$.
Put $\mathrm{tr}_p(f, c) = \langle 7, n, j, \mathrm{tr}_p(g, c)\rangle$.

Case C8. $\{f\}(\chi, e, \boldsymbol{x}, y) = \{e\}(\boldsymbol{x})$. Using Lemmas 19 and 20 it is easy to define a combinatorial function $d(f)$ such that

$$\{d(f)\}_p(tr_p, f, c, e, \boldsymbol{x}, y) = \{\{tr_p\}_p(e, c)\}_p(\boldsymbol{x}) \tag{5.8}$$

and such that, if $\{\{tr_p\}_p(e, c)\}_p(\boldsymbol{x}) \to z$, then

$$|\{tr_p\}_p(e, c), \boldsymbol{x}, z|_p < |d(f), tr_p, f, c, e, \boldsymbol{x}, \boldsymbol{y}, z|_p, \tag{5.9}$$

(take $d(f) = \langle 5, n+m+4, d_1(f), d_2(f)\rangle$ where $d_1(f)$ and $d_2(f)$ are chosen by Lemma 20 and C7 so that $\{d_1(f)\}_p(e', tr_p, f, c, e, \boldsymbol{x}, \boldsymbol{y}) = \{e'\}_p(\boldsymbol{x})$ and

$$\{d_2(f)\}_p(tr_p, f, c, e, \boldsymbol{x}, \boldsymbol{y}) = \{tr_p\}_p(e, c)).$$

Put

$$tr_p(f, c) = S^3(d(f), tr_p, f, c). \tag{5.10}$$

Otherwise. Put $tr_p(f, c) = 0$.

It is clear that the function $tr_p(f, c)$ thus defined is absolutely prime computable, so Lemma 21 applies and we may assume that tr_p is an index of $tr_p(f, c)$. Now once tr_p is chosen, the definition is clearly a course of values induction on f, hence $tr_p(f, c)$ is combinatorial. To prove the lemma it then suffices to show the following two propositions.

$$\textit{If } \{f\}_p(\chi, \boldsymbol{u}) \to z, \textit{ then } \{tr_p(f, c)\}_p(\boldsymbol{u}) \to z, \tag{5.11}$$

$$\textit{if } \{f'\}_p(\boldsymbol{u}) \to z \textit{ and } f' = tr_p(f, c), \textit{ then } \{f\}_p(\chi, \boldsymbol{u}) \to z. \tag{5.12}$$

Now (5.11) follows easily by an induction on C0′–C8′ of the usual sort.

The converse implication (5.12) is proved by induction on the ordinal $|f', \boldsymbol{u}, z|_p$. Here we consider the nine cases C0–C8 by which it is possible to have $f' = tr_p(f, c)$ and $\{f'\}_p(\boldsymbol{u}) \to z$, and in all of them except C8 the result is immediate. In case C8 we have $f' = S^3(d(f), tr_p, f, c)$ and by the form of f, $\{f\}_p(\chi, e, \boldsymbol{x}, \boldsymbol{y}) = \{e\}_p(\chi, \boldsymbol{x})$. Since $\{f'\}_p(e, \boldsymbol{x}, \boldsymbol{y}) \to z$, we know $\{d(f)\}_p(tr_p, f, c, e, \boldsymbol{x}, \boldsymbol{y}) \to z$ and

$$|d(f), tr_p, f, c, e, \boldsymbol{x}, \boldsymbol{y}, z|_p < |f', e, \boldsymbol{x}, \boldsymbol{y}, z|_p$$

by Lemma 19. By (5.9) we have $|\{tr_p\}_p(e, c), \boldsymbol{x}, z|_p < |d(f), tr_p, f, c, e, \boldsymbol{x}, \boldsymbol{y}, z|_p$, hence the ind. hyp. applies to $\{tr_p\}_p(e, c)$, $\boldsymbol{x}$, z and we have $\{e\}_p(\chi, \boldsymbol{x}) \to z$. Hence $\{f\}_p(\chi, e, \boldsymbol{x}, \boldsymbol{y}) \to z$ by C8.

LEMMA 23. *Suppose that* $g(i, \boldsymbol{x})$ *is defined and single-valued, for each* $i \in \omega$, *put*

$$f(\boldsymbol{x}) = \mu i[g(i, \boldsymbol{x}) = 0] = \textit{least } i\,[g(i, \boldsymbol{x}) = 0]. \tag{5.13}$$

If $g(i, \boldsymbol{x})$ *is in* $PC(A, \varphi)$, *then* $f(\boldsymbol{x})$ *is in* $PC(A, \varphi)$.

Hence (by Lemma 3) *if* $\phi(\boldsymbol{u})$ *is a partial number-theoretic function which is partial recursive in the sense of* [4], *then* $\phi(\boldsymbol{u})$ *is the restriction to* ω *of an absolutely prime computable function.*

Proof is after the proof of XVI in [6]. Choose

$$\begin{aligned} h(i, x) &= 0 && \text{if } g(i, \boldsymbol{x}) = 0, \\ &= h(i+1, \boldsymbol{x})+1 && \text{if } g(i, \boldsymbol{x}) \neq 0 \end{aligned}$$

using Lemmas 18 and 21 and set $f(\boldsymbol{x}) = h(0, \boldsymbol{x})$.

To each subset C of B^* we assign the set $[C]_\varphi$ generated from C by the functions in φ and the operations (x, y), πx, δx, by the following inductive definition:

(5.14) (a) If $x \in C$, then $x \in [C]_\varphi$.

(b) If $x, y \in [C]_\varphi$, then πx, δx and $(x, y) \in [C]_\varphi$.

(c) If $t_1, \ldots, t_{n_i} \in [C]_\varphi$ and $\varphi_i(t_1, \ldots, t_{n_i}) \to z$, then $z \in [C]_\varphi$.

LEMMA 24. *If* $\{f\}_p(u_1, \ldots, u_k) \to z$, *then* $z \in [f, u_1, \ldots, u_k]_\varphi$.

Proof is easy by induction on C0′–C8′.

EXAMPLE 3. Suppose B is a group and φ consists of group multiplication $x \cdot y$, inversion x^{-1} and the constant function e (the identity in B). For each $u_1, \ldots, u_k$ in B, it is easy to verify that $[u_1, \ldots, u_k]_\varphi \cap B$ is simply the subgroup of B generated by $u_1, \ldots, u_k$ (prove by induction on (5.13) that if $z \in [u_1, \ldots, u_k]_\varphi$, then D_z is a subset of the subgroup of B generated by $u_1, \ldots, u_k$). Thus if $\mathrm{f}(\boldsymbol{u})$ is single-valued, totally defined from B to B and absolutely prime computable (i.e. the restriction to B of an absolutely prime computable function), then for each $\boldsymbol{u}$, $\mathrm{f}(\boldsymbol{u})$ is in the subgroup of B generated by $\boldsymbol{u}$; in particular $\mathrm{f}(e) = e$ if f is one-place. Thus prime computable functions are related quite intimately to the algebraic structure of the group, and there are always functions on B to B which are not absolutely prime computable, if B has more than one element. If we include in φ the characteristic function of equality $x = y$ on B, an example of a nontrivial absolutely prime computable function on B to ω is $\mathrm{f}(x) = \mu i[x^i = e]$.

EXAMPLE 4. Suppose that φ contains only functions whose range lies in ω, e.g. characteristic functions of predicates. It is easy to verify then that for $u_1, \ldots, u_k \in B$, $[u_1, \ldots, u_k]_\varphi \cap B = \{u_1, \ldots, u_k\}$. Thus by Lemma 24, if f is absolutely prime computable and carries B into B, then f is the identity on B.

6. **The ν-operator.** In the definition of μ-recursive functions on the natural numbers we postulate that if the function $\mathrm{g}(i, \boldsymbol{x})$ is computable, then so is

$$\mu i[\mathrm{g}(i, \boldsymbol{x}) = 0]$$

(as a partial function). On an arbitrary set B and its derived set B^* there is in general no natural well-ordering along which one might perform an indefinite search. The analog of the μ-operator is then an *unordered search operator* ν,

$$\nu y[\mathrm{g}(y, \boldsymbol{x}) \to 0] = \textit{some } y \textit{ such that } \mathrm{g}(y, \boldsymbol{x}) \to 0.$$

This operator may lead to a multiple-valued function even if $\mathrm{g}(y, \boldsymbol{x})$ is single-valued, just as the μ-operator may lead to partial functions even when applied to totally defined functions. The formal definition of ν is

$$\nu y[\mathrm{g}(y, \boldsymbol{x}) \to 0] \to z \Leftrightarrow \mathrm{g}(z, \boldsymbol{x}) \to 0. \tag{6.1}$$

In applying the search operator ν we may imagine that the search is performed by an oracle which, though unable to search in a finite amount of time through all of

B^* and determine whether $(Ey)[g(y, \boldsymbol{x}) \to 0]$, may nevertheless search through arbitrarily large parts of B^* and eventually find all y's such that $g(y, \boldsymbol{x}) \to 0$. The role of that oracle relative to an arbitrary B seems to us not much different from the role of a human computer when he searches for $\mu i[g(i, \boldsymbol{x}) = 0]$. It is natural to expect that for a reasonable computability theory over an arbitrary set B an oracle more powerful than the human computer will be needed.

Those who cannot comprehend an unordered search, may imagine that this oracle is searching through B^* along a well-ordering which he knows, even though we may not know it. Such well-orderings of B^* always exist by the axiom of choice, though they need not be definable in terms of any concepts which the human computer will understand.

We define the class $\mathbf{SC}(\varphi)$ of *search computable* (or *ν-computable*) functions on B^* to B^* by adding to C0–C8 the ν schema,

C9. $f(\boldsymbol{x}) = \nu y[g(y, \boldsymbol{x}) \to 0] \quad \langle 9, n, g\rangle$

and the corresponding clause

C9′. *if* $\{g\}(y, \boldsymbol{x}) \to 0$, *then* $\{\langle 9, n, g\rangle\}(\boldsymbol{x}) \to y$.

Clauses C0′–C9′ then, together with a recursion clause of the usual type define a predicate $\{f\}_\nu(\boldsymbol{u}) \to z$ on finite sequences of elements of B^*. Each $f \in B^*$ defines a search computable function $\{f\}_\nu(\boldsymbol{u}) = \{z; \{f\}_\nu(\boldsymbol{u}) \to z\}$, and again we let $SC(A, \varphi)$ be the class of search computable functions with indices $f \in A^*$, $\mathbf{SC}(\varphi) = SC(B, \varphi)$. The functions in $SC(\varphi) = SC(\varnothing, \varphi)$ we call *absolutely search computable*. For each list $g_1, \ldots, g_m$ of variables over functions of $l_1, \ldots, l_m$ arguments, each $f \in B^*$ defines a search computable functional $\{f\}_\nu(g_1, \ldots, g_m, \boldsymbol{u})$, as before. To each $f, \boldsymbol{u}, z$ such that $\{f\}_\nu(\boldsymbol{u}) \to z$ we assign an ordinal $|f, \boldsymbol{u}, z|_\nu$ by adding to C0″–C8″ clause

C9″. *if* $\{g\}(y, \boldsymbol{x}) \to 0$, *then* $|\langle 9, n, g\rangle, \boldsymbol{x}, y| = |g, y, \boldsymbol{x}, 0| + 1$.

It is easy to verify by induction on C0′–C9′ that, if $\{f\}_\nu(\boldsymbol{u}) \to z$, then $|f, \boldsymbol{u}, z|_\nu < \omega$.

It appears to us that Church's thesis cannot be stated for abstract domains with the precision with which it is stated for the natural numbers. By this we mean that in the absence of more specific information about B and φ, one cannot define a class of functions on B^* which is so intrinsic that one can assert without reservations that it is the class of all functions on B^* intuitively computable from φ. Instead we would suggest that the classes of prime computable and search computable functions form natural bounds for effective computability. We state this precisely as our (weak) abstract version of Church's thesis, for single-valued absolutely computable functions; people who believe in the computability of arbitrary constants on an arbitrary set would wish to omit the word "absolutely", but we have our reservations on this.

Let C, E be subsets of B^*. Every single-valued, totally defined function on C to E which is the restriction to C of an absolutely prime computable function is intuitively effectively computable (from φ). Every single-valued totally defined function on C to E which is intuitively effectively computable (from φ) is the restriction to C of an absolutely search computable function.

The most interesting cases here are of course when $C=B$ and $E=B$ or $E=\omega$.

There are many ways in which classes intermediate between $PC(A, \varphi)$ and $SC(A, \varphi)$ can be introduced, e.g. by restricted search operators

$$\mathrm{f}(x) = \nu y[y \in C \ \& \ \mathrm{g}(y, x) \to 0],$$

where $C \subset B^*$. In this paper however we shall confine attention from now on to studying the classes $SC(A, \varphi)$, leaving aside the more general theory of prime and intermediate computabilities. We shall also restrict attention to functions and predicates on B^*, although we suspect that the more interesting aspects of the theory will develop in studying functions and predicates on B.

EXAMPLE 5. Let $B=\{a, b\}$ consist of exactly two elements, let φ consist of functions with range in ω, say characteristic functions of predicates, including the characteristic function of $x=y$. The function

$$\begin{aligned} \mathrm{f}(x) = \nu y[y \neq x] &= b \quad \text{if } x = a, \\ &= a \quad \text{if } x = b \end{aligned}$$

is absolutely search computable on B, but not absolutely prime computable, because of Example 4.

7. **Elementary theory of search computable functions.** In this section we shall omit the proofs of lemmas which are identical with or parallel to results in §5.

LEMMA 25. *If* $f \in PRI$, *then* $\{f\}_{\mathrm{pr}}(\boldsymbol{u}) = \{f\}_{\nu}(\boldsymbol{u})$. *In particular* $PR(A, \varphi) \subset SC(A, \varphi)$.

LEMMA 26. *If* $f \in PR(A, \mathrm{g}_1, \ldots, \mathrm{g}_m, \varphi)$ *and for* $i=1, \ldots, m$, $\mathrm{g}_i \in SC(A, \varphi)$, *then* $\mathrm{f} \in SC(A, \varphi)$.

LEMMA 27. *The functions* $S^m(f, y_1, \ldots, y_m)$ *of Lemma* 12 *satisfy*

$$\{f\}_{\nu}(\boldsymbol{y}, \boldsymbol{x}) = \{S^m(f, \boldsymbol{y})\}_{\nu}(\boldsymbol{x}). \tag{7.1}$$

Moreover, if $\{f\}_{\nu}(\boldsymbol{y}, \boldsymbol{x}) \to z$, *then* $|f, \boldsymbol{y}, \boldsymbol{x}, z|_{\nu} < |S^m(f, \boldsymbol{y}), \boldsymbol{x}, z|_{\nu}$.

LEMMA 28. (a) *The function* $\mathrm{EV}(f, k')$ *of Lemma* 20 *satisfies*

$$\{\mathrm{EV}(f, k')\}_{\nu}(\boldsymbol{u}, \boldsymbol{u}') = \{f\}_{\nu}(\boldsymbol{u}). \tag{7.2}$$

Moreover, if $\{f\}_{\nu}(\boldsymbol{u}) \to z$, *then* $|f, \boldsymbol{u}, z|_{\nu} < |(\mathrm{EV}f, k'), \boldsymbol{u}, \boldsymbol{u}', z|_{\nu}$.

(b) *The function* $\mathrm{FV}(f, k')$ *of Lemma* 20 *satisfies*

$$\{\mathrm{FV}(f, k')\}_{\nu}(\boldsymbol{u}', \boldsymbol{u}) = \{f\}_{\nu}(\boldsymbol{u}). \tag{7.3}$$

Moreover, if $\{f\}_{\nu}(\boldsymbol{u}) \to z$, *then* $|f, \boldsymbol{u}, z|_{\nu} < |\mathrm{FV}(f, k'), \boldsymbol{u}', \boldsymbol{u}, z|_{\nu}$.

LEMMA 29. *If* $\mathrm{rc}(f)$ *is the function of Lemma* 21 *and* $m=\mathrm{rc}(f)$, *then*

$$\{f\}_{\nu}(m, \boldsymbol{u}) = \{m\}_{\nu}(\boldsymbol{u}). \tag{7.4}$$

(*The recursion theorem.*)

LEMMA 30. *Consider the multiple-valued definition by cases*

$$\begin{aligned} \mathrm{f}(\boldsymbol{u}) &= \mathrm{g}_1(\boldsymbol{u}) \quad \textit{if } \mathrm{h}_1(\boldsymbol{u}) \to 0, \\ &= \cdots \qquad\qquad \cdots \\ &= \mathrm{g}_m(\boldsymbol{u}) \quad \textit{if } \mathrm{h}_m(\boldsymbol{u}) \to 0, \end{aligned} \tag{7.5}$$

i.e.

$$\mathrm{f}(\boldsymbol{u}) \to z \Leftrightarrow \{[\mathrm{h}_1(\boldsymbol{u}) \to 0 \ \&\ \mathrm{g}_1(\boldsymbol{u}) \to z] \vee \cdots \vee [\mathrm{h}_m(\boldsymbol{u}) \to 0 \ \&\ \mathrm{g}_m(\boldsymbol{u}) \to z]\}. \tag{7.6}$$

Then the functional

$$\mathrm{f}(\mathrm{g}_1, \ldots, \mathrm{g}_m, \mathrm{h}_1, \ldots, \mathrm{h}_m, \boldsymbol{u}) = \mathrm{f}(\boldsymbol{u})$$

is absolutely search computable.

Proof. The functional

$$\mathrm{k}(s_1, t_1, \ldots, s_m, t_m) = \nu z[(s_1 = 0 \ \&\ z = t_1) \vee \cdots \vee (s_m = 0 \ \&\ z = t_m)]$$

is clearly absolutely search computable; put

$$\mathrm{f}(\boldsymbol{u}) = \mathrm{k}(\mathrm{h}_1(\boldsymbol{u}), \mathrm{g}_1(\boldsymbol{u}), \ldots, \mathrm{h}_m(\boldsymbol{u}), \mathrm{g}_m(\boldsymbol{u})).$$

For the next two lemmas we need the method of Lemma 22.

LEMMA 31. *There is a combinatorial function* $\mathrm{p}(f)$ *such that for each* f,

$$\{f\}_{\mathrm{p}}(\boldsymbol{u}) = \{\mathrm{p}(f)\}_{\nu}(\boldsymbol{u}). \tag{7.7}$$

In particular, for each $A \subset B$, $PC(A, \varphi) \subset SC(A, \varphi)$.

Proof. We shall define $\mathrm{p}(f)$ as a search computable function from an index p, then apply the recursion theorem, Lemma 29, and choose p to be an index of $\mathrm{p}(f)$. After p is chosen it will be obvious from the form of the definition that $\mathrm{p}(f)$ is combinatorial. The definition is by Cases C0–C8 (on the form of f as an index of a prime computable function) using Lemma 30.

Cases C0–C4. Put $\mathrm{p}(f) = f$.

Case C5. $\{f\}(\boldsymbol{x}) = \{g\}(\{h\}(\boldsymbol{x}), \boldsymbol{x})$. Put $\mathrm{p}(f) = \langle 5, n, \mathrm{p}(g), \mathrm{p}(h)\rangle$.

Case C6. $\{f\}(y, \boldsymbol{x}) = \{g\}(y, \boldsymbol{x})$ if $y \in B^0$,
$\{f\}((s, t), \boldsymbol{x}) = \{h\}(\{f\}(s, x), \{f\}(t, \boldsymbol{x}), s, t, \boldsymbol{x})$.
Put $\mathrm{p}(f) = \langle 6, n+1, \mathrm{p}(g), \mathrm{p}(h)\rangle$.

Case C7. $\{f\}(\boldsymbol{x}) = \{g\}(x_{j+1}, x_1, \ldots, x_j, x_{j+2}, \ldots, x_n)$. Put $\mathrm{p}(f) = \langle 7, n, j, \mathrm{p}(g)\rangle$.

Case C8. $\{f\}(e, \boldsymbol{x}, y) = \{e\}(\boldsymbol{x})$. Using Lemmas 27 and 28, choose a combinatorial $\mathrm{d}(f)$ such that $\{\mathrm{d}(f)\}_{\nu}(p, f, e, \boldsymbol{x}, y) = \{\{p\}_{\nu}(e)\}_{\nu}(\boldsymbol{x})$ and if $\{\{p\}_{\nu}(e)\}_{\nu}(\boldsymbol{x}) \to z$, then

$$|\{p\}_{\nu}(e), \boldsymbol{x}, z|_{\nu} < |\mathrm{d}(f), p, f, e, \boldsymbol{x}, y, z|_{\nu},$$

and put $\mathrm{p}(f) = S^2(\mathrm{d}(f), p, f)$.

Otherwise. Put $\mathrm{p}(f) = 0$.

To prove the lemma we show by induction on C0′–C8′ that

$$\text{if } \{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z, \text{ then } \{\mathrm{p}(f)\}_{\nu}(\boldsymbol{u}) \to z, \tag{7.8}$$

and by induction on $|f', \boldsymbol{u}, z|_{\nu}$ that

$$\text{if } \{f'\}_{\nu}(\boldsymbol{u}) \to z \text{ and } f' = \mathrm{p}(f), \text{ then } \{f\}_{\mathrm{p}}(\boldsymbol{u}) \to z. \tag{7.9}$$

The first implication (7.8) is routine. Under the hypothesis of (7.9) it is impossible that the Otherwise case applies in the definition of $\mathrm{p}(f)$, since $\{0\}_{\nu}(\boldsymbol{u})$ is undefined. Treatment of Cases C0–C7 is routine, and for Case C8 the method is that of Lemma 22: if $\{\mathrm{p}(f)\}_{\nu}(e, \boldsymbol{x}, \boldsymbol{y}) \to z$, then $\{\mathrm{d}(f)\}_{\nu}(p, f, e, \boldsymbol{x}, \boldsymbol{y}) \to z$ and by Lemma 27

$$|\mathrm{d}(f), p, f, e, \boldsymbol{x}, \boldsymbol{y}, z|_{\nu} < |\mathrm{p}(f), e, \boldsymbol{x}, \boldsymbol{y}, z|_{\nu},$$

hence $\{\{p\}_{\nu}(e)\}_{\nu}(\boldsymbol{x}) \to z$ and $|\{p\}_{\nu}(e), \boldsymbol{x}, z|_{\nu} < |\mathrm{d}(f), p, f, e, \boldsymbol{x}, \boldsymbol{y}, z|_{\nu}$, hence by induction hypothesis $\{e\}_{\mathrm{p}}(\boldsymbol{x}) \to z$, hence by C8 $\{f\}_{\mathrm{p}}(e, \boldsymbol{x}, \boldsymbol{y}) \to z$.

LEMMA 32. *There is a combinatorial function* $\mathrm{tr}_{\nu}(f, c)$, *such that if* $\chi(\boldsymbol{u}') = \{c\}_{\nu}(\boldsymbol{u}')$ *and* $\mathrm{f}(\boldsymbol{u}) = \{f\}_{\nu}(\chi, \boldsymbol{u})$, *then* $\mathrm{f}(\boldsymbol{u}) = \{\mathrm{tr}_{\nu}(f, c)\}_{\nu}(\boldsymbol{u})$. *Thus if* $\mathrm{f} \in SC(A, \mathrm{g}_1, \ldots, \mathrm{g}_m, \boldsymbol{\varphi})$ *and* $\mathrm{g}_1, \ldots, \mathrm{g}_m \in SC(A, \boldsymbol{\varphi})$, *then* $\mathrm{f} \in SC(A, \boldsymbol{\varphi})$. (*Transitivity lemma.*)

Proof. The function $\mathrm{tr}_{\nu}(f, c)$ is defined exactly like the function $\mathrm{tr}_{\mathrm{p}}(f, c)$ of Lemma 22, through the recursion theorem (Lemma 29 in this case) from an index tr_{ν} of itself. There are now eleven cases in the definition, Case C9 added to Cases C0–C8 and the Otherwise case. The definition is the same as that of $\mathrm{tr}_{\mathrm{p}}(f, c)$ in Cases C0–C8 and Otherwise.

Case C9. $\{f\}(\chi, \boldsymbol{x}) = \nu y[\{g\}(\chi, y, \boldsymbol{x}) \to 0]$. Put $\mathrm{tr}_{\nu}(f, c) = \langle 9, n, \mathrm{tr}_{\nu}(g, c)\rangle$. Proof of the assertion is again exactly like that of Lemma 22, Case C9 being treated just like Cases C0–C7.

8. **Normal form and recursion theorems.** We seek an abstract analog to the normal form theorem of ordinary recursion theory,

$$\{f\}(\boldsymbol{x}) = U(\mu y T_n(f, \boldsymbol{x}, y))$$

(U and T_n primitive recursive, [4, Theorem XIX]). Here we would expect that the μ-operator is replaced by the ν-operator, but on first thought no other serious difficulty should arise. In setting up an induction for the analog of the T predicate however, we meet clauses of the type "y is a computation for..."; in expressing such clauses formally we need the equality predicate $x = y$, which up till now we have not admitted to be necessarily computable. Of course, we could make the blanket assumption that $\boldsymbol{\varphi}$ contains the representing function of $x = y$; this however would exclude some natural applications (e.g. the case of absolute computability on the continuum, see Example 2 and [19]) and seems unnatural to us on aesthetic grounds. With a little extra work we shall obtain essentially the same theorem with no such assumption on $\boldsymbol{\varphi}$.

Our idea is to notice that whether (our analog for) $T(f, \boldsymbol{x}, y)$ holds or not depends only on the "finitely generated" part $[f, \boldsymbol{x}, y]_{\varphi}$ of B^*; this part can be indexed by elements of ω, on which equality is a combinatorial predicate by Lemma 3.

For simplicity we shall confine ourselves to single-valued totally defined φ's, so that primitive computable functions are single-valued totally defined.

THEOREM 1. *Let each φ_i in φ be single-valued and total. For each k there is a function $U(f, \boldsymbol{u}, y)$ and a predicate $T(f, \boldsymbol{u}, y)$, both absolutely primitive computable in φ, such that*

$$\{f\}_{\nu}(\boldsymbol{u}) = U(f, \boldsymbol{u}, \nu y T(f, \boldsymbol{u}, y)). \tag{8.1}$$

Similarly with function arguments, for fully defined single-valued $\mathrm{g}_1, \ldots, \mathrm{g}_m$,

$$\{f\}_{\nu}(\mathrm{g}_1, \ldots, \mathrm{g}_m, \boldsymbol{u}) = U(\mathrm{g}_1, \ldots, \mathrm{g}_m, f, \boldsymbol{u}, \nu y T(\mathrm{g}_1, \ldots, \mathrm{g}_m, f, \boldsymbol{u}, y)). \tag{8.2}$$

(Normal form theorem.)

Proof. We define the predicate $\{f\}_{\nu}(\boldsymbol{u}) \rightarrow_w z$ on finite sequences $f, \boldsymbol{u}, w, z$ of elements of B^* by restricting the existential quantifiers in clauses C0′–C9′ to $[w]_{\varphi}$ and adding to each clause the condition

$$f, u_1, \ldots, u_k, z \in [w]_{\varphi}. \tag{8.3}$$

Thus C0′–C4′ are unchanged, except for condition (8.3), and C5′, C8′, for example become:

C5′(w). *If there exists a $u \in [w]_{\varphi}$ such that $\{h\}(\boldsymbol{x}) \rightarrow_w u$ and $\{g\}(u, \boldsymbol{x}) \rightarrow_w z$, then $\{f\}(x) \rightarrow_w z$.*

C8′(w). *If $e \in [w]_{\varphi}$ and $\{e\}(\boldsymbol{x}) \rightarrow_w z$, then $\{f\}(e, \boldsymbol{x}, \boldsymbol{y}) \rightarrow_w z$.*

One easily proves by induction on C0′(w)–C9′(w) and C0′–C9′ that:

$$\text{If } \{f\}_{\nu}(\boldsymbol{u}) \rightarrow_w z \text{ and } [w]_{\varphi} \subset [w']_{\varphi}, \text{ then } \{f\}(\boldsymbol{u}) \rightarrow_{w'} z. \tag{8.4}$$

$$\text{If } \{f\}_{\nu}(\boldsymbol{u}) \rightarrow_w z, \text{ then } \{f\}_{\nu}(\boldsymbol{u}) \rightarrow z. \tag{8.5}$$

$$\text{If } \{f\}_{\nu}(\boldsymbol{u}) \rightarrow z, \text{ then there exists a } w \text{ such that } \{f\}_{\nu}(\boldsymbol{u}) \rightarrow_w z. \tag{8.6}$$

For this proof we shall use the notations

$$\langle u'_1, \ldots, u'_k \rangle^{\omega} = p_0^{u'_1}, \ldots, p_{k-1}^{u'_k} \quad (p_0 = 2, p_1 = 3, p_2 = 5, \ldots), \tag{8.7}$$

$$(u')_i^{\omega} = \text{largest } j \text{ such that } p_i^j \text{ divides } u'. \tag{8.8}$$

As indicated in these definitions we shall also use accented italic letters as variables over ω.

The set $I \subset \omega$ of indices of elements in $[w]_{\varphi}$, for an arbitrary w, is defined by the following inductive clauses:

$$\begin{aligned} &\text{(a) } \langle 0, 0 \rangle^{\omega} = 1 \in I, \langle 0, 1 \rangle^{\omega} = 3 \in I, \\ &\text{(b) if } x', y' \in I, \text{ then } \langle 1, x', y' \rangle^{\omega} \in I, \langle 2, x' \rangle^{\omega} \in I \text{ and } \langle 3, x' \rangle^{\omega} \in I, \\ &\text{(c) if } t'_1, \ldots, t'_{n_i} \in I, \text{ then } \langle 4+i, t'_1, \ldots, t'_{n_i} \rangle^{\omega} \in I. \end{aligned} \tag{8.9}$$

It is obvious that I is primitive recursive in the sense of [4], hence combinatorial in our sense. We define a function val (i, w) which assigns to each $i \in I$ an element of $[w]_{\varphi}$ by the following course of values induction on i.

$$
\begin{aligned}
&\text{(a)}\quad \text{val}\,(1, w) = 0,\ \text{val}\,(3, w) = w.\\
&\text{(b)}\quad \text{val}\,(\langle 1, x', y'\rangle^{\omega}, w) = (\text{val}\,(x', w), \text{val}\,(y', w)),\\
&\qquad \text{val}\,(\langle 2, x'\rangle^{\omega}, w) = \pi\ \text{val}\,(x', w),\\
&\qquad \text{val}\,(\langle 3, x'\rangle^{\omega}, w) = \delta\ \text{val}\,(x', w).\\
&\text{(c)}\quad \text{val}\,(\langle 4+i, t'_1, \ldots, t'_{n_i}\rangle, w) = \varphi_i(\text{val}\,(t'_1, w), \ldots, \text{val}\,(t'_{n_i}, w)),\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad i = 1, \ldots, l.\\
&\text{(d)}\quad \text{val}\,(i, w) = 0 \text{ if } i \notin I.
\end{aligned}
\tag{8.10}
$$

It is clear that val (i, w) is absolutely primitive computable in φ.

We define a predicate $P(w, f', \boldsymbol{u}', a, b)$ on finite sequences

$$w, f', \boldsymbol{u}', a, b = w, f', u'_1, \ldots, u'_k, a, b$$

where $w \in B^*$ and $f', u'_1, \ldots, u'_k, a, b \in \omega$, which will mean approximately "a is a computation of length $\leqq b$ which proves that $\{f\}_v(\boldsymbol{u}) \to_w z$", where $f = \text{val}\,(f', w)$, $u_i = \text{val}\,(u'_i, w)$ $(i = 1, \ldots, k)$ and $z = \text{val}\,((a)_0^{\omega}, w)$. The definition is by nested induction on b, and is such that if $P(w, f', \boldsymbol{u}, a, b)$ holds, then $f', u'_1, \ldots, u'_k$, $a < b$, so that $P(w, f', \boldsymbol{u}', a, 0)$ is false. The definition of $P(w, f, \boldsymbol{u}', a, b+1)$ is by eleven cases, namely Cases C0–C9, an Accumulation case which asserts that

$$P(w, f', \boldsymbol{u}', a, b) \Rightarrow P(w, f', \boldsymbol{u}', a, b+1)$$

and an Otherwise case.

In stating the case hypotheses for Cases C0–C9 we follow the practice established in Lemma 12, of writing down the defining condition for $\{f\}(\boldsymbol{u})$ rather than the condition that f must satisfy: here

$$
\begin{aligned}
f &= \text{val}\,(f', w),\\
u_i &= \text{val}\,(u'_i, w), \qquad i = 1, \ldots, k
\end{aligned}
\tag{8.11}
$$

and in general, if the letters c, c' appear in a case, the convention is that

$$c' \in \omega, \quad c = \text{val}\,(c', w). \tag{8.12}$$

To recall the notation conventions established in §2 which help understand the definition in each case, we include in the case hypothesis not only the defining condition for $\{f\}(\boldsymbol{u})$, but also the special notation adopted for $\boldsymbol{u}$ in schemata C0–C8, e.g. in C0, $\boldsymbol{u} = t_1, \ldots, t_{n_i}, \boldsymbol{x}$, $\boldsymbol{u}' = t'_1, \ldots, t'_{n_i}, \boldsymbol{x}'$, where by (8.12)

$$t_1 = \text{val}\,(t'_1, w), \ldots, t_{n_i} = \text{val}\,(t'_{n_i}, w),\ x_1 = \text{val}\,(x'_1, w), \ldots, x_n = \text{val}\,(x'_n, w).$$

The implied conditions on $w, f', \boldsymbol{u}'$, for each case hypothesis are absolutely primitive computable, since by (8.12) $f, \boldsymbol{u}$ are absolutely primitive computable functions of $w, f', \boldsymbol{u}'$.

Accumulation Case. *If* $P(w,f',\boldsymbol{u}',a,b)$, *then* $P(w,f',\boldsymbol{u}',a,b+1)$.

Case C0. $\boldsymbol{u}'=t_1',\ldots,t_{n_i}',\boldsymbol{x}',\{f\}(\boldsymbol{u})=\varphi_i(t_1,\ldots,t_{n_i}),\ i=1,\ldots,l.$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ (a)_0^\omega=\langle 4,i,t_1',\ldots,t_{n_i}'\rangle^\omega.$$

Case C1. $\boldsymbol{u}'=x',\{f\}(\boldsymbol{u})=y.$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ (a)_0^\omega=\langle 2,\langle 3,\langle 3,f'\rangle^\omega\rangle^\omega\rangle^\omega.$$

Case C2. $\boldsymbol{u}'=y',\boldsymbol{x}',\{f\}(\boldsymbol{u})=y.$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ (a)_0^\omega=y'.$$

Case C3. $\boldsymbol{u}'=s',t',\boldsymbol{x}',\{f\}(\boldsymbol{u})=(s,t).$

$$P(w,f',u',a,b+1)\Leftrightarrow f',u',a\leqq b\ \&\ (a)_0^\omega=\langle 1,s',t'\rangle^\omega.$$

Case C4$_0$. $\boldsymbol{u}'=y',\boldsymbol{x}',\{f\}(\boldsymbol{u})=\pi y.$

$$P(w,f',u',a,b+1)\Leftrightarrow f',u',a\leqq b\ \&\ (a)_0^\omega=\langle 2,y'\rangle^\omega.$$

Case C4$_1$. $\boldsymbol{u}'=y',\boldsymbol{x}',\{f\}(\boldsymbol{u})=\delta y.$

$$P(w,f',u',a,b+1)\Leftrightarrow f',u',a\leqq b\ \&\ (a)_0^\omega=\langle 3,y'\rangle^\omega.$$

Case C5. $\boldsymbol{u}'=\boldsymbol{x}',\{f\}(\boldsymbol{u})=\{g\}(\{h\}(\boldsymbol{x}),\boldsymbol{x}).$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f'\boldsymbol{u}',a\leqq b\ \&\ P(w,h^*,\boldsymbol{x}',(a)_1^\omega,b)\ \&\ P(w,g^*,((a)_1^\omega)_0^\omega,\boldsymbol{x}',(a)_2^\omega,b)\ \&\ ((a)_2^\omega)_0^\omega=(a)_0^\omega,$$

where $g^*=\langle 2,\langle 3,\langle 3,f'\rangle^\omega\rangle^\omega\rangle^\omega$, and $h^*=\langle 2,\langle 3,\langle 3,\langle 3,f'\rangle^\omega\rangle^\omega\rangle^\omega\rangle^\omega$.

Case C6. *Subcase* (a). $\boldsymbol{u}'=y',\boldsymbol{x}'$ and $y=\mathrm{val}\,(y',w)\in B^0,\{f\}(\boldsymbol{u})=\{g\}(y,\boldsymbol{x}).$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ P(w,g^*,\boldsymbol{u}',(a)_1^\omega,b)\ \&\ (a)_0^\omega=((a)_1^\omega)_0^\omega.$$

Subcase (b). $\boldsymbol{u}'=y',\boldsymbol{x}'$ and $y=\mathrm{val}\,(y',w)\notin B^0$,
$\{f\}(\boldsymbol{u})=\{h\}(\{f\}(\pi y,\boldsymbol{x}),\{f\}(\delta y,\boldsymbol{x}),\pi y,\delta y,\boldsymbol{x}).$

$$\begin{aligned}P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ & P(w,f,\langle 2,y'\rangle^\omega,\boldsymbol{x}',(a)_1^\omega,b)\\ &\&\ P(w,f,\langle 3,y'\rangle^\omega,\boldsymbol{x}',(a)_2^\omega,b)\\ &\&\ P(w,h^*,((a)_1^\omega)_0^\omega,((a)_2^\omega)_0^\omega,\langle 2,y'\rangle^\omega,\langle 3,y'\rangle^\omega,\\ &\qquad\qquad \boldsymbol{x},(a)_3^\omega,b)\\ &\&\ ((a)_3^\omega)_0^\omega=(a)_0^\omega,\end{aligned}$$

where in both subcases, g^* and h^* are defined as in Case C5.

Case C7. $\boldsymbol{u}'=\boldsymbol{x}',\{f\}(\boldsymbol{u})=\{g\}(x_{j+1},x_1,\ldots,x_j,x_{j+2},\ldots,x_n).$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ P(w,g^{**},x_{j+1}',x_1',\ldots,x_j',x_{j+2}',\ldots,x_n',a,b),$$

where $g^{**}=\langle 2,\langle 3,\langle 3,\langle 3,g\rangle^\omega\rangle^\omega\rangle^\omega\rangle^\omega$.

Case C8. $\boldsymbol{u}'=e',\boldsymbol{x}',y',\{f\}(\boldsymbol{u})=\{e\}(\boldsymbol{x}).$

$$P(w,f',\boldsymbol{u}',a,b+1)\Leftrightarrow f',\boldsymbol{u}',a\leqq b\ \&\ P(w,e',\boldsymbol{x}',a,b).$$

Case C9. $\boldsymbol{u}' = \boldsymbol{x}'$, $\{f\}(\boldsymbol{u}) = \nu y[\{g\}(y, \boldsymbol{x}) \to 0]$.

$$P(w, f', \boldsymbol{u}', a, b+1) \Leftrightarrow f', \boldsymbol{u}', a \leqq b \;\&\; P(w, g^*, (a)_0^\omega, \boldsymbol{x}', (a)_1^\omega, b) \;\&\; \mathrm{val}((a)_1^\omega)_0^\omega, w) = 0,$$

where g^* is as in Case C5.

Otherwise. $P(w, f', \boldsymbol{u}', a, b)$ is false.

It is now tedious but straightforward to prove by induction on b that

$$\begin{aligned} &\textit{if } P(w, f', \boldsymbol{u}', a, b), \textit{ then } \{f\}_v(\boldsymbol{u}) \to_w z, \\ &\textit{where } f = \mathrm{val}(f', w),\ u_i = \mathrm{val}(u_i', w),\ i = 1, \ldots, k,\ z = \mathrm{val}((a)_0^\omega, w), \end{aligned} \tag{8.13}$$

and by induction on the definition of $\{f\}_v(\boldsymbol{u}) \to_w z$ that

$$\begin{aligned} &\textit{if } \{f\}_v(\boldsymbol{u}) \to_w z \textit{ and } f = \mathrm{val}(f', w),\ u_1 = \mathrm{val}(u_1', w), \ldots, u_k = \mathrm{val}(u_k', w), \\ &\textit{then there exist an } a \textit{ and a } b \textit{ such that } P(w, f', \boldsymbol{u}', a, b) \textit{ and } z = \mathrm{val}((a)_0^\omega, b). \end{aligned} \tag{8.14}$$

Now (8.4)–(8.6) imply that

$$\{f\}_v(\boldsymbol{u}) \to z \Leftrightarrow (Ew)[\{f\}_v(\boldsymbol{u}) \to_w z] \Leftrightarrow (Ew)[\{f\}_v(\boldsymbol{u}) \to_{\langle w, f, \boldsymbol{u}\rangle} z] \tag{8.15}$$

(where $\langle w, f, \boldsymbol{u}\rangle = \langle w, f, u_1, \ldots, u_k\rangle$), hence by (8.13) and (8.14) we have

$$\begin{aligned} \{f\}_v(\boldsymbol{u}) \to z \Leftrightarrow (Ew)(Eb)(Ea)[&P(\langle w, f, \boldsymbol{u}\rangle, f^\circ, u_1^\circ, \ldots, u_k^\circ, a, b) \\ &\;\&\; z = \mathrm{val}((a)_0^\omega, \langle w, f, \boldsymbol{u}\rangle)], \end{aligned} \tag{8.16}$$

where $f^\circ, u_1^\circ, \ldots, u_k^\circ$ are fixed integers, depending on k and chosen by (8.9) so that

$$\begin{aligned} &\mathrm{val}(f^\circ, \langle w, f, \boldsymbol{u}\rangle) = f, \\ &\mathrm{val}(u_i^\circ, \langle w, f, \boldsymbol{u}\rangle) = u_i, \qquad i = 1, \ldots, k. \end{aligned} \tag{8.17}$$

Thus relation (8.1) holds if we set

$$T(f, \boldsymbol{u}, y) \Leftrightarrow P(\langle (y)_0, f, \boldsymbol{u}\rangle, f^\circ, u_1^\circ, \ldots, u_k^\circ, (y)_1, (y)_2), \tag{8.18}$$

$$U(f, \boldsymbol{u}, y) = \mathrm{val}(((y)_1)_0^\omega, \langle (y)_0, f, \boldsymbol{u}\rangle). \tag{8.19}$$

To complete the proof we must still show that the predicate $T(f, \boldsymbol{u}, y)$ is absolutely primitive computable in φ, since $U(f, \boldsymbol{u}, y)$ is clearly absolutely primitive computable in φ. For this purpose we condense all the variables in P except b into one,

$$Q(v, b) \Leftrightarrow \mathrm{Seq}(v) \;\&\; (v)_0 \geqq 1 \;\&\; P((v)_1, (v)_2, \ldots, (v)_{(v)_0 \dot{-} 1}, (v)_{(v)_0}, b). \tag{8.20}$$

It is now easy to see that the induction for P becomes a nested induction for the characteristic function of Q of the type covered by Lemma 9, so that Q, hence P and T are absolutely primitive computable in φ.

Relation (8.2) follows immediately, since φ is arbitrary in (8.1).

REMARK 8. Theorem 1 shows that the enumeration schema C8 is superfluous in the definition of search computability. This is not the case in higher order theories, where one studies computability in given objects of higher type over B^*.

REMARK 9. Suppose that φ is empty, $B \neq \varnothing$, $b \in B$, let $\{f\}_v(\boldsymbol{u})$ be single-valued on b^* and mapping b^* into b^*. Using the notation of the proof of Lemma 5 and the result proved there, that if $\mathrm{f}(\boldsymbol{u})$ is primitive computable from b, then $\mathrm{f}(u_1^\#, \ldots, u_k^\#) = (\mathrm{f}(u_1, \ldots, u_k))^\#$, we have

$$(8.21)\quad u_1, \ldots, u_k \in b^* \,\&\, T(f, \boldsymbol{u}, y) \Rightarrow [T(f^\#, \boldsymbol{u}, y^\#) \,\&\, U(f^\#, \boldsymbol{u}, y^\#) = (U(f, \boldsymbol{u}, y))^\#].$$

Conversely,

$$(8.22)\qquad u_1, \ldots, u_k \in b^* \,\&\, T(f^\#, \boldsymbol{u}, y^\#) \Rightarrow T(f, \boldsymbol{u}, y);$$

because if $\overline{T}(f, \boldsymbol{u}, y)$, then $\overline{T}(f^\#, \boldsymbol{u}, y^\#)$. Thus if $\{f\}_v(\boldsymbol{u})$ maps b^* into b^* and is single-valued on b^*, then

$$(8.23)\qquad u_1, \ldots, u_k \in b^* \Rightarrow \{f\}_v(\boldsymbol{u}) = U(f^\#, \boldsymbol{u}, \nu y[y \in b^* \,\&\, T(f^\#, \boldsymbol{u}, y)]).$$

Then, using the technique of the first part of the proof of Lemma 5, it is easy to see that *every function which is search computable in the empty* φ, *which is single-valued on* ω *and maps* ω *into* ω, *coincides on* ω *with a partial recursive function in the sense of* [4]. By Lemma 31 this covers the case of prime computable functions as well. The proof is similar (and a little easier) if $B = \varnothing$.

REMARK 10. Let us define $\{f\}_p(\boldsymbol{u}) \to_w z$ for prime computability by the clauses C0′(w)–C8′(w), just as we defined $\{f\}_v(\boldsymbol{u}) \to_w z$ for search computability. It is easy to show by induction on C0′–C8′ (using Lemma 24) that

$$(8.24)\qquad \{f\}_p(\boldsymbol{u}) \to z \Rightarrow \{f\}_p(\boldsymbol{u}) \to_{\langle f, \boldsymbol{u}\rangle} z.$$

From this and an analysis of the proof of Theorem 1 we see that if φ consists of totally defined single-valued functions, then there exist absolutely primitive computable $U(f, \boldsymbol{u}, y)$ and $T(f, \boldsymbol{u}, y)$ such that

$$\{f\}_p(\boldsymbol{u}) = U(f, \boldsymbol{u}, \nu i \in \omega T(f, \boldsymbol{u}, i)).$$

Since prime computable functions are single-valued, this is equivalent to

$$(8.25)\qquad \{f\}_p(\boldsymbol{u}) = U(f, \boldsymbol{u}, \mu i T(f, \boldsymbol{u}, i)).$$

Thus for prime computability the enumeration schema C8 is superfluous if we allow the μ-operator, exactly as in the case in ordinary recursion theory over ω.

In addition to the analog of (8.1), there are normal forms for partial recursive functionals which make explicit the dependence of each value of the functional on only finitely many values of the argument (see Lemma 34 below). Such normal forms are quite messy here and we shall try to avoid using them. We need one weak result of this type in proving Post's theorem in the next section.

Let α vary over fully defined single-valued one-place functions from B^* to B^* and let us write $\mathrm{val}(\alpha, i, w)$ and $P(\alpha, w, f', \boldsymbol{u}', a, b)$ for the function val and the predicate P of the proof of Theorem 1 relative to the list $\varphi' = \alpha, \varphi$. We define

a new function val (v, i, w) and a new predicate $P(v, w, f', \boldsymbol{u}', a, b)$ by altering clause (c) of (8.9) to

$$\begin{aligned} &\text{val}\,(v, \langle 4+1, t_1'\rangle^{\omega}, w) = (v)_{t_1'+1}, \\ &\text{val}\,(v, \langle 4+i+1, t_1', \ldots, t_{n_i}'\rangle^{\omega}, w) = \varphi_i(t_1', \ldots, t_{n_i}'), \qquad i = 1, \ldots, l \end{aligned} \tag{8.26}$$

and substituting val (v, i, w) for val (α, i, w) in the definition of $P(\alpha, w, f', \boldsymbol{u}', a, b)$ throughout.

Put

$$\tilde{\alpha}(w, i) = \langle \alpha(\text{val}\,(\alpha, 0, w)), \ldots, \alpha(\text{val}\,(\alpha, i \dot{-} 1, w))\rangle. \tag{8.27}$$

It is easy to prove by finite induction on j that

$$j < i \;\&\; v = \tilde{\alpha}(w, i) \Rightarrow [\text{val}\,(\alpha, j, w) = \text{val}\,(v, j, w)]. \tag{8.28}$$

From this it follows by induction on b, using the fact that $P(\alpha, w, f', \boldsymbol{u}', a, b) \Rightarrow f', u_1', \ldots, u_k', a < b$, that

$$b \leqq i \;\&\; v = \tilde{\alpha}(w, i) \Rightarrow [P(\alpha, w, f', \boldsymbol{u}', a, b) \Leftrightarrow P(v, w, f', \boldsymbol{u}', a, b)]. \tag{8.29}$$

Finally we show that

$$v = \tilde{\alpha}(w, i) \Leftrightarrow \text{Seq}\,(v) \;\&\; (v)_0 = i \;\&\; (j)_{j<i}[(v)_{j+1} = \alpha(\text{val}\,(v, j, w))]. \tag{8.30}$$

To prove (8.30) in the direction $\Rightarrow$, simply notice that if $v = \tilde{\alpha}(w, i)$, then for each $j < i$, $(v)_{j+1} = \alpha(\text{val}\,(\alpha, j, w))$, so by (8.28), $(v)_{j+1} = \alpha(\text{val}\,(v, j, w))$. For the converse implication we only need show that for each $j < 1$, $(v)_{j+1} = \alpha(\text{val}\,(\alpha, j, w))$, i.e. that $\text{val}\,(\alpha, j, w) = \text{val}\,(v, j, w)$. We do this by finite induction on j. The only nontrivial case is when $j = \langle 4, t_1'\rangle^{\omega}$, when $\text{val}\,(\alpha, j, w) = \alpha(\text{val}\,(\alpha, t_1', w)) = \alpha(\text{val}\,(v, t_1', w))$ (by induction hypothesis) $= (v)_{t_1'+1}$ (by assumption) $= \text{val}\,(v, t_1', w)$ (by definition). Thus we have the following

LEMMA 33. *Assume* $\boldsymbol{\varphi}$ *consists of totally defined, single-valued functions, let* α *vary over fully defined, single-valued functions. For suitable absolutely primitive computable* val (v, j, w) *and* $P(v, w, f', \boldsymbol{u}', a, b)$, *we have*

$$\begin{aligned} \{f\}_v(\alpha, \boldsymbol{u}) \to z \Leftrightarrow (Ew)(Ev)(Ea)(Eb)[&P(v, \langle w, f, u\rangle, f^{\circ}, u_1^{\circ}, \ldots, u_k^{\circ}, a, b) \\ &\;\&\; z = \text{val}\,(v, (a)_0^{\omega}, \langle w, f, \boldsymbol{u}\rangle) \;\&\; \text{Seq}\,(v) \;\&\; (v)_0 = b \\ &\;\&\; (i)_{i<b}[(v)_{i+1} = \alpha(\text{val}\,(v, i, \langle w, f, \boldsymbol{u}\rangle))]]. \end{aligned} \tag{8.31}$$

For the next lemma we need the notation $\mathrm{g} \subset \mathrm{g}'$ introduced by (1.7).

LEMMA 34. *Let* "g", "g'" *be variables for functions of n variables.*

(a) *If* $\{f\}_v(\mathrm{g}, \boldsymbol{u}) \to z$ *and* $\mathrm{g} \subset \mathrm{g}'$, *then* $\{f\}_v(\mathrm{g}', \boldsymbol{u}) \to z$.

(b) *If* $\{f\}_v(\mathrm{g}, \boldsymbol{u}) \to z$, *then there exists a finite set* $\boldsymbol{x}_1, z_1, \ldots, \boldsymbol{x}_j, z_j$ *of* $n+1$*-tuples such that* $\mathrm{g}(\boldsymbol{x}_1) \to z_1, \ldots, \mathrm{g}(\boldsymbol{x}_j) \to z_j$ *and whenever* $\mathrm{g}'(\boldsymbol{x}_1) \to z_1, \ldots, \mathrm{g}'(\boldsymbol{x}_j) \to z_j$, *then* $\{f\}_v(\mathrm{g}', \boldsymbol{u}) \to z$.

Proofs of both (a) and (b) are easy by induction on C0–C9.

THEOREM 2. *Assume* φ *consists of totally defined, single-valued functions, let* "g" *be a variable over n-place functions, let* $f(g, x)$ *be a p.m.v. functional in* $SC(A, \varphi)$. *There exists a* $g(x)$ *in* $SC(A, \varphi)$ *such that*

$$(x)[f(g, x) = g(x)], \tag{8.32}$$

and

$$\text{if } (x)[f(g', x) = g'(x)], \text{ then } g \subset g'. \tag{8.33}$$

(*The first recursion theorem.*)

Proof is by the usual method of iteration. Set

$$\begin{aligned} &g_0 = \varnothing \text{ (the nowhere defined function)},\\ &g_{j+1}(x) = f(g_j, x)\\ &g(x) = \mathit{limit}_j\, g_j(x), \text{ i.e. } g(x) \to z \Leftrightarrow (Ej)[g_j(x) \to z]. \end{aligned} \tag{8.34}$$

Using Lemma 34 it is easy to prove by induction on j that

$$g_j \subset g_{j+1}, \tag{8.35}$$

$$\text{if } (x)[f(g', x) = g'(x)], \text{ then } g_j \subset g', \text{ hence } g \subset g'. \tag{8.36}$$

To prove (8.32) we use (b) of Lemma 34: If $f(g, x) \to z$, then for some j we must have $f(g_j, x) \to z$, hence $g_{j+1}(x) \to z$, hence $g(x) \to z$. Conversely, if $g(x) \to z$, then for some j, $g_j(x) \to z$, hence by (8.34) $f(g_j, x) = g_{j+1}(x) \to z$, hence by Lemma 34 $f(g, x) \to z$.

To complete the proof it remains to show that g is in $SC(A, \varphi)$. If $f(g, x) = \{f\}_\nu(g, x)$, Lemma 32 implies that

$$g_0(x) = \{0\}_\nu(x), \quad g_{j+1}(x) = \{f\}_\nu(g_j, x) = \{\mathrm{tr}_\nu(f, g_j)\}_\nu(x), \tag{8.37}$$

where g_j is an index of g_j. Hence indices for the functions g_j can be found by the induction

$$g_0 = 0, \quad g_{j+1} = \mathrm{tr}_\nu(f, g_j) \tag{8.38}$$

and using Theorem 1 we can put

$$h(x) = \nu y[T(g_{(y)_0}, x, (y)_1)], \quad g(x) = U((h(x))_0, x, (h(x))_1).$$

REMARK 11. Using Lemma 22 and Remark 10 one can easily see that the first recursion theorem holds also for prime computability.

REMARK 12. The first recursion theorem implies in the usual way that the classes $PC(A, \varphi)$ and $SC(A, \varphi)$ (for totally defined, single-valued φ) are closed under inductive definitions. In particular, they are closed under course of values induction (with the g of (3.22) perhaps partial or multiple-valued) and nested recursion (with the g, h, p of (3.24) perhaps partial or multiple-valued).

9. **The projective hierarchy.** We recall that a predicate $R(u)$ on B^* is search computable if its characteristic function is search computable, and similarly for the

other classes of functions that we have introduced. In this section we extend these classes of predicates by quantification (over B^*) to define what we shall call the *projective hierarchy*. This is the abstract version of the arithmetical and analytical hierarchies of ordinary recursion theory and of the projective hierarchy of descriptive set theory.

For this section we make the blanket assumption that φ consists of single-valued total functions.

Let A be a fixed subset of B. We define the classes of predicates $\sigma_n^0(A, \varphi)$, $\pi_n^0(A, \varphi)$ by the following induction on n:

(9.1)
(a) $P(\boldsymbol{u})$ *is in* $\sigma_1^0(A, \varphi)$, *if there is a predicate* $R(y, \boldsymbol{u})$ *whose characteristic function is in* $SC(A, \varphi)$, *such that* $P(\boldsymbol{u}) \Leftrightarrow (Ey)R(y, \boldsymbol{u})$.
(b) $P(\boldsymbol{u})$ *is in* $\pi_n^0(A, \varphi)$, *if* $\bar{P}(\boldsymbol{u})$ *is in* $\sigma_n^0(A, \varphi)$.
(c) $P(\boldsymbol{u})$ *is in* $\sigma_{n+1}^0(A, \varphi)$ *if there is an* $R(y, \boldsymbol{u})$ *in* $\pi_n^0(A, \varphi)$ *such that* $P(\boldsymbol{u}) \Leftrightarrow (Ey)R(y, \boldsymbol{u})$.

In using these notations we often suppress the dependence on φ and write $\pi_n^0(A)$, $\sigma_n^0(A)$. We also use the printing conventions we have established:

$$\sigma_n^0 = \sigma_n^0(\varnothing), \boldsymbol{\sigma}_n^0 = \sigma_n^0(B), \quad \pi_n^0 = \pi_n^0(\varnothing), \boldsymbol{\pi}_n^0 = \pi_n^0(B). \tag{9.2}$$

We put

$$\delta_n^0(A) = \pi_n^0(A) \cap \sigma_n^0(A), \qquad \delta_n^0 = \delta_n^0(\varnothing), \qquad \boldsymbol{\delta}_n^0 = \delta_n^0(B) \tag{9.3}$$

and we call a predicate in $\bigcup_n \delta_n^0(A)$ *projective from* A, in $\bigcup_n \delta_n^0$ *absolutely projective*, in $\bigcup_n \boldsymbol{\delta}_n^0$ *projective*.

We classify sets in the classes $\pi_n^0(A)$, $\sigma_n^0(A)$, $\delta_n^0(A)$ by their representing predicates.

LEMMA 35. (a) *Each of the classes* $\sigma_n^0(A)$, $\pi_n^0(A)$ *and* $\delta_n^0(A)$ *is closed under the operations of conjunction, disjunction, substitution of functions from* $SC(A)$ *and bounded number quantification.*

(b) $\sigma_n^0(A)$ *is closed under existential quantification over* B^*.

(c) $\pi_n^0(A)$ *is closed under universal quantification over* B^*.

(d) *A predicate* $P(\boldsymbol{u})$ *is in* $\sigma_1^0(A)$ *if and only if there exists an* $f \in A^*$ *such that*

$$P(\boldsymbol{u}) \Leftrightarrow \{f\}_\nu(\boldsymbol{u}) \to 0, \tag{9.4}$$

i.e. if and only if the extension of $P(\boldsymbol{u})$ *coincides with the domain of a p.m.v. function which is search computable from* A.

Proof. Consider the following equivalences:

(i) $(Ey)R(y, \boldsymbol{u}) \,\&\, (Ey)Q(y, \boldsymbol{u}) \Leftrightarrow (Ey)[R((y)_0, \boldsymbol{u}) \,\&\, Q((y)_1, \boldsymbol{u})]$,

(ii) $(Ey)R(y, \boldsymbol{u}) \vee (Ey)Q(y, \boldsymbol{u}) \Leftrightarrow (Ey)[R(y, \boldsymbol{u}) \vee Q(y, \boldsymbol{u})]$,

(iii) $(Ey)R(y, \boldsymbol{u}, \mathrm{f}(\boldsymbol{u})) \Leftrightarrow (Ey)[\chi_R(y, \boldsymbol{u}, \mathrm{f}(\boldsymbol{u})) = 0]$,

(iv) $(Ei)_{i<j}(Ey)R(i, y, \boldsymbol{u}) \Leftrightarrow (Ey)[(y)_0 \in \omega \,\&\, (y)_0 < j \,\&\, R((y)_0, (y)_1, \boldsymbol{u})]$,

(v) $(i)_{i<f}(Ey)R(i, y, \boldsymbol{u}) \Leftrightarrow (Ey)(i)_{i<f}R(i, (y)_i, \boldsymbol{u})$,

(vi) $(Ez)(Ey)R(z, y, \boldsymbol{u}) \Leftrightarrow (Ey)R((y)_0, (y)_1, \boldsymbol{u})$.

Using these equivalences it is easy to show (a)–(c) by induction on n, first for $\sigma_1^0(A)$, then assuming them for $\sigma_n^0(A)$, and proving them for $\pi_n^0(A)$ and $\sigma_{n+1}^0(A)$.

To prove (d) in one direction, notice that if $(Ey)R(y, \boldsymbol{u})$ is in $\sigma_1^0(A)$, the function $f(\boldsymbol{u}) = 0 \cdot \nu y R(y, \boldsymbol{u})$ is in $SC(A)$. For the other direction we use the normal form theorem,

$$f(\boldsymbol{u}) \to 0 \Leftrightarrow (Ey)[T(f, \boldsymbol{u}, y) \ \&\ U(f, \boldsymbol{u}, y) = 0]$$

(the predicate in brackets with $f \in A^*$ fixed is in $SC(A)$, since T and U are absolutely primitive computable and the constant function f is primitive computable from A).

LEMMA 36. *If $R(\boldsymbol{u}')$ is in $\delta_n^0(A)$ and $Q(\boldsymbol{u})$ is search computable from A in $\mathrm{R}(\boldsymbol{u}')$, φ, then $Q(\boldsymbol{u})$ is in $\delta_n^0(A)$.*

Proof. The lemma asserts that if

$$\chi_Q(\boldsymbol{u}) = f(\chi_R, \boldsymbol{u}), \tag{9.5}$$

where χ_Q, χ_R are the characteristic functions of Q and R and the functional $f(\chi, \boldsymbol{u})$ is search computable from A, and if $R(\boldsymbol{u}')$ is in $\delta_n^0(A)$ then $Q(\boldsymbol{u})$ is in $\delta_n^0(A)$.

We may assume that $R(\boldsymbol{u}')$ is a predicate of one variable, since if not, then we can substitute for it $R(u) \Leftrightarrow R((u)_1, \ldots, (u)_{k'})$. If $f \in A^*$ is an index of $f(\chi, \boldsymbol{u})$, using Lemma 33 we have

$$\begin{aligned} Q(\boldsymbol{u}) \Leftrightarrow (Ew)(Ev)(Ea)(Eb)\{P(v, \langle w, f, \boldsymbol{u}\rangle, f^\circ, u_1^\circ, \ldots, u_k^\circ, a, b) \\ \&\ \mathrm{val}\,(v, (a)_0^\omega, \langle w, f, \boldsymbol{u}\rangle) = 0 \ \&\ \mathrm{Seq}\,(v) \ \&\ (v)_0 = b \\ \&\ (i)_{i<b}\{[(v)_{i+1} = 0 \Leftrightarrow R(\mathrm{val}\,(v, i, \langle w, f, \boldsymbol{u}\rangle))] \\ \&\ [(v)_{i+1} = 1 \Leftrightarrow \bar{R}(\mathrm{val}\,(\mathrm{v}, \mathrm{i}, \langle \mathrm{w}, \mathrm{f}, \mathrm{u}\rangle))]\}\}, \end{aligned}$$

which implies that $Q(\boldsymbol{u})$ is in $\sigma_n^0(A)$. We also have

$$\begin{aligned} Q(\boldsymbol{u}) \Leftrightarrow (w)(v)(a)(b)\{\{P(v, \langle w, f, \boldsymbol{u}\rangle, f^\circ, u_1^\circ, \ldots, u_k^\circ, a, b) \\ \&\ \mathrm{Seq}\,(v) \ \&\ (v)_0 = b \\ \&\ (i)_{i<b}\{[(v)_{i+1} = 0 \Leftrightarrow R(\mathrm{val}\,(v, i, \langle w, f, \boldsymbol{u}\rangle))] \\ \&\ [(\mathrm{v})_{i+1} = 1 \Leftrightarrow \bar{R}(\mathrm{val}\,(v, i, \langle\ w, f, \boldsymbol{u}\rangle))]\}\} \\ \Rightarrow \mathrm{val}\,(v, (a)_0^\omega, \langle w, f, \boldsymbol{u}\rangle) = 0\} \end{aligned}$$

which implies that $Q(\boldsymbol{u})$ is in $\pi_n^0(A)$.

THEOREM 3. *Consider the double sequence of predicates*

$$\begin{aligned} &(Ey)T(f, \boldsymbol{u}, y), (Ey_1)(y_2)\bar{T}(f, \boldsymbol{u}, y_1, y_2), (Ey_1)(y_2)(Ey_3)T(f, \boldsymbol{u}, y_1, y_2, y_3), \ldots \\ &(y)\bar{T}(f, \boldsymbol{u}, y), (y_1)(Ey_2)T(f, \boldsymbol{u}, y_1, y_2), (y_1)(Ey_2)(y_3)\bar{T}(f, \boldsymbol{u}, y_1, y_2, y_3), \ldots . \end{aligned} \tag{9.6}$$

Each of these predicates enumerates the class $\sigma_n^0(A)$ or $\pi_n^0(A)$ in which it is, as f varies over A^*. (Enumeration theorem.)

Proof. To prove the theorem for $\delta_1^0(A)$, let $P(\boldsymbol{u}) \Leftrightarrow (Ey)R(y, \boldsymbol{u})$ be given, put $\mathrm{f}(\boldsymbol{u}) = \nu y R(y, \boldsymbol{u})$. Then $\mathrm{f}(\boldsymbol{u})$ is in $SC(A)$, say $\mathrm{f}(\boldsymbol{u}) = \{f\}_\nu(\boldsymbol{u})$, hence

$$P(\boldsymbol{u}) \Leftrightarrow \mathrm{f}(\boldsymbol{u}) \downarrow \Leftrightarrow (Ey)T(f, \boldsymbol{u}, y).$$

The theorem follows for the other classes by taking negations and quantifying.

THEOREM 4. (a) $\sigma_n^0(A) \cup \pi_n^0(A) \subset \delta_{n+1}^0(A)$. (b) $\sigma_n^0 - \boldsymbol{\pi}_n^0 \neq \varnothing$; $\pi_n^0 - \boldsymbol{\sigma}_n^0 \neq \varnothing$; *hence in particular, for each* A, $\sigma_n^0(A) - \pi_n^0(A) \neq \varnothing$, $\pi_n^0(A) - \sigma_n^0(A) \neq \varnothing$. (*Hierarchy theorem.*)

Proof. (a) is immediate. To prove (b), say for $n=1$, consider the predicate $(Ey)T(f, f, y)$ which is in σ_1^0 and assume it were also in $\boldsymbol{\pi}_1^0$. Then for some m we would have

$$(Ey)T(f, f, y) \Leftrightarrow (y)\overline{T}(m, f, y)$$

which is contradictory when $f = m$.

REMARK 13. Let us define the classes of predicates $\sigma_n^0(A; C)$, $\pi_n^0(A; C)$ $(C \subset B^*)$ to consist of those predicates on C which are restrictions of predicates in $\sigma_n^0(A)$, $\pi_n^0(A)$ respectively. The classical diagonal argument above shows that $\sigma_n^0(A; A^*) - \pi_n^0(A; A^*) \neq \varnothing$, so we get the usual hierarchy theorem in this "normal" case. However this argument says nothing about the interesting classes $\sigma_n^0(\varnothing; B)$, $\pi_n^0(\varnothing; B)$, $\sigma_n^0(B; B)$, $\pi_n^0(B; B)$. General theorems of the type we treat here cannot be proved about these classes, since e.g. if B is finite there are clearly only finitely many distinct $\sigma_n^0(B; B)$ classes.

THEOREM 5. (a) *A predicate* $P(\boldsymbol{u})$ *is search computable from* A, *if and only if* $P(\boldsymbol{u}) \in \delta_1^0(A)$.

(b) *A predicate* $P(\boldsymbol{u})$ *is in* $\delta_{n+1}^0(A)$ *if and only if* $P(\boldsymbol{u})$ *is search computable from* A *in predicates in* $\sigma_n^0(A) \cup \pi_n^0(A)$. (*Post's theorem.*)

Proof. That a predicate search computable in predicates in $\sigma_n^0(A) \cup \pi_n^0(A)$ must be in $\delta_{n+1}^0(A)$ follows immediately from Theorem 4(a) and Lemma 36. Conversely, if $P(\boldsymbol{u})$ is in $\delta_{n+1}^0(A)$, then

$$P(\boldsymbol{u}) \Leftrightarrow (Ey)R(y, \boldsymbol{u}) \Leftrightarrow (y)Q(y, \boldsymbol{u})$$

and then the characteristic function of $P(\boldsymbol{u})$ is given by

$$\begin{aligned} \chi_P(\boldsymbol{u}) &= 0 \quad \text{if } 0 \cdot \nu y R(y, \boldsymbol{u}) \to 0, \\ &= 1 \quad \text{if } 0 \cdot \nu y \overline{Q}(y, \boldsymbol{u}) \to 0 \end{aligned}$$

which shows that $P(\boldsymbol{u})$ is search computable in $R(y, \boldsymbol{u})$, $Q(y, \boldsymbol{u})$.

Proof of (a) is similar, though a little simpler.

BIBLIOGRAPHY

1. R. Fraïssé, *Une notion de récursivité relative*, Infinitistic methods, Proc. Sympos. Foundations of Math., 1959, New York, 1961, pp. 323–328.

2. R. O. Gandy, *Proof of Mostowski's conjecture*, Bull. Acad. Polon. Sci. **8** (1960), 571–575.

3. R. O. Gandy, *General recursive functionals of finite type and hierarchies of functions*, mimeographed copy of paper presented at the Symposium on Mathematical Logic, University of Clermont-Ferrand, June 1962.

4. S. C. Kleene, *Introduction to metamathematics*, Van Nostrand, New York, 1952.

5. ———, *On the forms of the predicates in the theory of constructive ordinals*. II, Amer. J. Math. **77** (1955), 405–428.

6. ———, *Recursive functionals and quantifiers of finite types*. I, Trans. Amer. Math. Soc. **91** (1959), 1–52.

7. ———, *Recursive functionals and quantifiers of finite types*. II, Trans. Amer. Math. Soc. **108** (1963), 106–142.

8. G. Kreisel, *Model theoretic invariants: applications to recursive and hyperarithmetic operations*, The theory of models, Proc. 1963 Internat. Sympos. Berkeley, North-Holland, Amsterdam, 1965, pp. 190–205.

9. G. Kreisel and Gerald E. Sacks, *Metarecursive sets*, J. Symbolic Logic **30** (1965), 318–338.

10. Saul Kripke, *Transfinite recursions on admissible ordinals*. I, II and *Admissible ordinals and the analytic hierarchy*, Abstracts of papers presented at the April 1964 meeting of the Assoc. for Symbolic Logic in New York, J. Symbolic Logic **29** (1964), 161–162.

11. D. Lacombe, *Deux généralisations de la notion de récursivité*, C. R. Acad. Sci. Paris **258** (1964), 3141–3143.

12. ———, *Deux généralisations de la notion de récursivité relative*, C. R. Acad. Sci. Paris **258** (1964), 3410–3413.

13. W. M. Lambert, *A notion of effectiveness in arbitrary structures*, (to appear).

14. Azriel Levy, *A hierarchy of formulas in set theory*, Mem. Amer. Math. Soc. No. 57, 1965.

15. M. Machover, *The theory of transfinite recursion*, Bull. Amer. Math. Soc. **67** (1961), 575–578.

16. R. Montague, *Recursion theory as a branch of model theory*, Proc. Third Internat. Congr. Logic, Methodology and Philosophy of Science, Amsterdam, 1967.

17. Yiannis N. Moschovakis, *Hyperanalytic predicates*, Trans. Amer. Math. Soc, **129** (1967), 249–282.

18. ———, *Generalization of the theory of recursive and hyperarithmetic relations*, (in preparation).

19. ———, *Post's theorem in the analytic hierarchy*, (in preparation).

20. R. Platek, *Foundations of recursion theory*, Ph.D. Thesis, Stanford Univ., Stanford, Calif., 1966.

21. C. Spector, *Recursive well-orderings*, J. Symbolic Logic **20** (1955), 151–163.

22. ———, *Hyperarithmetical quantifiers*, Fund. Math. **48** (1960), 313–320.

23. G. Takeuti, *On the recursive functions of ordinal numbers*, Math. Soc. Japan **12** (1960), 119–128.

24. H. Wang, *Alternative proof of a theorem of Kleene*, J. Symbolic Logic **23** (1958), 250.

UNIVERSITY OF CALIFORNIA,
LOS ANGELES, CALIFORNIA