BULLETIN OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 77, Number 4, July 1971

BOOK REVIEWS

Recursiveness, by Samuel Eilenberg and Calvin C. Elgot. Academic Press, New York and London, 1970. vii+89 pp.

Jorge Luis Borges (see e.g., Borges [**1962**]) has made impressive use of an unusual literary device—the review of a nonexistent book.There are ideas better conveyed by hinting at their elaboration elsewhere than by any such elaboration. In reading Eilenberg and Elgot (henceforth, E²), one senses the lack of such a review, for by elaborating mathematical details to the exclusion of all motivation, E² have robbed the reader of many of their insights.

The preface of this book makes a bold claim:

"On the one hand, it appears that a more algebraic, less arithmetic point of view [than is usual in the study of computable (recursive) functions] has been sought, and on the other hand, a theory of programs for digital computers appears to be the long-range aim. · · · This short monograph is a contribution to the latter activity. Its algebraic flavour is more or less obvious. Its connection with computer science is less obvious."

And then it promises that little aid will be given the reader in evaluating this claim:

"Because of the novelty of the approach, known theorems often appear in unfamiliar guises. Because of this we have not attempted to assign authorship to the theorems."

By writing the book instead of the review, E² have made it hard for us to accept their claim—for while they have recast the *elements* of recursive function theory in the spirit of modern algebra, and have made minor use of operations somewhat like those of automata theory, they advance no evidence that their approach contributes more to computer science than other approaches. In fact the words 'computer' and 'program' do not appear to recur in the volume after the preface! However, one knows that Elgot has thought much about theories of programming—see, for example, Elgot and Robinson [**1964**], which is *not* cited by E²—and so why Elgot chose to give no hint of his expertise in this monograph is a mystery. A good theory of programs probably will be heavily algebraic—perhaps E² were not so far from the spirit of Borges when they claimed their book as a forerunner!

E²'s aim, though nowhere explicated in the monograph, seems to have been presented by Eilenberg in lectures as follows: "Much of the power of modern mathematics has arisen with the development of

algebraic methods and terminology which contribute to almost all branches of mathematics. Recursive function theory is notable for an almost complete lack of this algebraic approach. If only we could recast it using this language, the subject should be immensely revitalized." E² succeed in showing that an algebraic recasting is possible, but their results are all elementary—for example, the only undecidability result is in an appendix which rephrases (with due credit) Péter's [e.g. **1967**, §9] presentation of a recursive function which is not primitive recursive. It thus remains to be seen whether the algebraic approach of the present monograph when wedded to the motivation of an approach such as Rogers' [**1967**] can yield dramatic advances that could not have been achieved without it. Perhaps it is reasonable to expect that algebraic methods—with the resultant increase in the ability to compute with symbolic expressions—will help us to systematise the labyrinthine diagonal arguments of recursive function theory. This may lead to generalisations on which a whole new level of theory will be built. It may not. Algebra will help, but whether the *details* of the approach here reviewed will make much difference is problematical. So much for rhetoric. What does the paper actually do?

The usual setting for recursive function theory is within the class of *partial* functions $f: N^n \to N$, i.e. functions whose domain is a *subset* of N^n, and whose range falls within N. A number of authors—e.g. Asser [**1960**], Hu [**1960**], Vučković [**1960**], Péter [**1961**], [**1962**], Shepherdson and Sturgis [**1963**], Schwenkel [**1965**] and Arbib [**1969**]—have taken a more automaton-theoretic approach and instead considered partial functions $f: W^r \to W$ where W is some (usually finitely generated) free semigroup $W = X^*$. E² note in their preface that a more algebraic, less arithmetic point of view is sought, but nowhere admit their debt to other authors for the transition from N to X^*. In fact, Péter [**1961**], [**1962**] has gone even further and set up recursive function theory for *any* free algebra, and one wonders why E² did not extend their treatment to this case. If X has but one element, we may identify $W = X^*$ with N by matching the element of W of length k with the integer k. Within this setting, we may present the standard definitions for finite X:

For each $x \in X$, let $L_x: W \to W$ be the *left-successor* $w \mapsto xw$.

For each $n \in N$, let $U^n: W^n \to W: (w_1, \cdots, w_n) \mapsto 1$ be the function which replaces n arguments by the *unit* 1.

For each $n \in N$ and $1 \leqq j \leqq n$, let $\Pi^n_j: W^n \to W: (w_1, \cdots, w_n) \mapsto w_j$ be the function which *projects* the jth of n arguments.

Composition is the operation which, when applied to the partial

functions $h: W^m \to W$ and $g_j: W^n \to W$, $1 \leqq j \leqq m$, yields the function $f: W^n \to W$ defined by

$$f(w_1, \cdots, w_n) = h(g_1(w_1, \cdots, w_n), \cdots, g_m(w_1, \cdots, w_n)).$$

Recursion is the operation which, when applied to the functions $g: W^{n-1} \to W$ and $h_x: W^n \to W$, for each $x \in X$, yields the function $f: W^n \to W$ defined *recursively* by

$$f(1, w_2, \cdots, w_n) = g(w_2, \cdots, w_n),$$

$$f(xw, w_2, \cdots, w_n) = h_x(f(w, w_2, \cdots, w_n), w, w_2, \cdots, w_n) \quad \text{for } x \in X.$$

Minimisation in $\{x\}^*$, where x is a given element of X, is the operation which, when applied to $g: W^n \to W$, yields the function $f: W^{n-1} \to W$ defined by

$$f(w_1, \cdots, w_{n-1}) = \mu_x w[g(w_1, \cdots, w_{n-1}, w) = 1],$$

the string x^k for which k is the smallest integer such that $g(w_1, \cdots, w_{n-1}, x^k) = 1$, while $g(w_1, \cdots, w_{n-1}, x^l)$ is defined, but non-null, for $0 \leqq l < k$.

Let Γ be any class of partial functions $f: W^r \to W$ (with various exponents r). We denote by $\hat{\mathfrak{E}}_\Gamma(W^*)$ the smallest class of partial functions containing Γ and the L_x, U^n and Π_j^n which is closed under composition and recursion. We denote by $\hat{\mathfrak{P}}_\Gamma(W^*)$ the smallest class of partial functions containing Γ and the L_x, U^n and Π_j^n which is closed under composition, recursion and minimisation. We say a function is *primitive recursive* if it belongs to $\hat{\mathfrak{E}}_\phi$ (also written $\hat{\mathfrak{E}}_0$), and we say that a partial function is a *partial recursive* if it belongs to $\hat{\mathfrak{P}}_\phi$ (also written $\hat{\mathfrak{P}}_0$).

Before contrasting these definitions with those of E² we should note that E^2 work with relations. They write a relation as $f: C \to B$ and identify it with an additive function from subsets of C to subsets of B. We recapture partial functions if $|f(c)| \leqq 1$ for each $c \in C$, and we recapture ordinary (often referred to in recursive function theory as total) functions if we have $|f(c)| = 1$. $\mathfrak{R}(A^*)$ is then the category of all relations of the form $f: A^p \to A^q$, $p, q \geqq 0$, under the obvious composition; and the partial functions and functions form subcategories $\mathfrak{P}(A^*)$ and $\mathfrak{F}(A^*)$. By always considering subcategories of $\mathfrak{R}(A^*)$, E² can make use of composition and identity functions without explicitly mentioning them. In giving their definitions, let us restrict attention to the case $A = W = X^*$ for finite X. E² note various places at which their theory is available more generally. Recursive function theorists usually work only with functions of the form $f: W^r \to W$ and so must use the somewhat clumsy form of composition we pre-

sented above. E^2, working within the category $\mathfrak{R}(W^*)$ and its subcategories, use the natural composition of $f: W^r \to W^s$ and $g: W^m \to W^r$ as $f \circ g: W^m \to W^s: w \mapsto f(g(w))$. However, to recapture the other composition one must then be able to explicitly select from any $f: W^r \to W^s$ a desired $\Pi_j^n \circ f: W^r \to W$ and obtain

$$g = \langle g_1, \cdots, g_m \rangle: W^n \to W^m: w \mapsto (g_1(w), \cdots, g_m(w))$$

from the set of functions $g_j: W^n \to W$, $1 \leqq j \leqq m$. To do this, we simply need the functions Π_j^n and the operation of *pairing* which assigns to any pair of functions $f: W^s \to W^u$ and $g: W^s \to W^v$ the function $\langle f, g \rangle$: $W^s \to W^{u+v}: w \mapsto (f(w), g(w))$. However, E^2 do not tell us of this necessity, but instead state baldly the unmotivated definition:

A subcategory $\mathfrak{A}$ of $\mathfrak{R}(W^*)$ is called *admissible* if it satisfies the following axioms:

(1) The objects of $\mathfrak{A}$ are all the objects W^r, $r \geqq 0$, of $\mathfrak{R}(W^*)$.

(2) *Cylindrification.* If $f: W^r \to W^s$ is a relation in $\mathfrak{A}$, then so is the relation

$$W \times f: W^{1+r} \to W^{1+s}: (x, y) \mapsto (x, fy).$$

(3) *Transposition.* The function $\theta_k: W^k \to W^k$ $(k \geqq 2)$ defined by

$$\theta_k(w_1, w_2, w_3, \cdots, w_k) = (w_2, w_1, w_3, \cdots, w_k)$$

is in $\mathfrak{A}$.

(4) *Diagonal.* The function $\Delta: W \to W^2: w \mapsto (w, w)$ is in $\mathfrak{A}$.

(5) *Projection.* The unique function $\Pi: W \to W^0: w \mapsto 1$ is in $\mathfrak{A}$.

Clearly, $\mathfrak{A}$ is admissible just in case it satisfies the need we have motivated, and also contains the projection Π. To get the primitive recursive functions, then, we must add something like L_x and U^n and the operation of recursion. E^2 do this as follows (they give other definitions and prove them equivalent, but one will do to give the flavour):

An admissible subcategory $\mathfrak{A}$ of $\mathfrak{F}(W^*)$ is called *primitive* if it satisfies the following axioms:

(1) *Unit.* The function $U: W^0 \to W$ with value 1 is in $\mathfrak{A}$. [In my copy, W^0 was misprinted as W^*. Note that $U^n = U \circ \Pi \circ \Pi_1^n$.]

(2) *Left successors.* For each $x \in X$, the left successor function L_x: $W \to W$ is in $\mathfrak{A}$.

(3) *Left exponentiation.* Suppose each function $k_x: W^r \to W^r$ is in $\mathfrak{A}$ for each x in X. Then so is their *left exponential* $k: W^{r+1} \to W^r$ defined inductively as follows:

$$k(1, w) = w$$

$$k(xw_1, w) = k_x(k(w_1, w)) \qquad \text{for } x \in X.$$

Clearly recursion can do anything left exponentiation can do, and it is a straightforward exercise to verify that left exponentiation—with the aid of some of the given functions and other operations—can do what recursion can do. And thus E² are justified when they say: The class of all primitive subcategories of $\mathfrak{F}(W^*)$ is to be denoted by $\boldsymbol{E}(W^*)$. If Γ is any class of functions $f: W^r \to W^s$ (with various exponents r and s) we denote the smallest element of $\boldsymbol{E}(W^*)$ containing Γ by $\mathfrak{E}_\Gamma(W^*)$. The functions in $\mathfrak{E}_0(W^*) \underline{\underline{\Delta}} \mathfrak{E}_\phi(W^*)$ are called primitive recursive functions.

Just as in the ordinary theory we say a set $A \subset W^r$ is primitive recursive just in case its characteristic function χ_A is a primitive recursive function, so do E² say that the set A belongs to the category $\mathfrak{A}$ just in case χ_A does.

Since this monograph provides no motivation—E² refer the reader to other texts for the standard development and all intuition—the reader is probably so adept at proving such results as "all Turing-computable functions are recursive" that stating and proving the theorems which show the equivalence of old and new definitions should not tax him unduly. The real problem he faces with such theorems in the current monograph is that they are scattered, with virtually no label, motivation or demarcation, throughout the text, with the result that the development of the algebraic theory seems to take far longer than it really does. When E² introduce a new operation, the reader must indulge in much detective work to determine whether it is a component of their algebraic approach, or merely a standard operator in algebraic guise brought in only to be subsumed in the new approach, but then never to be used again.

In E²'s treatment of recursive isomorphisms in Chapter III, I found the algebraic *approach* (irrespective of the actual choice of definitions) to be vindicated. My admittedly subjective criterion for this was that E², in doing almost the same work that I did in Arbib [**1969**, § 6.1], read out far deeper results in the process. Having defined what it was for a function to be partial recursive with respect to a given finite alphabet, one wants to verify that the choice of alphabet was immaterial to partial recursiveness. Regarding a string in a k-letter alphabet as a number in k-adic notation, I showed that—at least with this encoding—a function was partial recursive over any finite alphabet iff its numerical correspondent was also partial recursive. The elegance of E² was that, with essentially the same effort, they proved the far more interesting result that for *any* choice of a bijection between two free monoids $c: W \to V$, for which c and c^{-1} are (in some suitable sense) primitive recursive, one gets a bijection—independent of the choice of c-between $\boldsymbol{E}(W^*)$ and $\boldsymbol{E}(V^*)$ under

which $\mathcal{E}_0(W^*)$ corresponds to $\mathcal{E}_0(V^*)$ as well as bijections of the classes $\boldsymbol{R}(W^*)$ and $\boldsymbol{R}(V^*)$ of recursive categories which we shall define below. [Note that it is the correspondence between classes—not that between individual functions—which is independent of c.] Armed with this result, E^2 are free to prove many results for the simple case of the 1-letter alphabet. [It is a disturbing fact that the argument for other alphabets often seems hard to push through without the occasional use of a lemma which pushes back to the 1-letter case.]

To reconstruct E^2's motivation for their definition of recursive relations, it is useful to consider a very general model of a digital computer (the original definition, and its application to a stored program machine appeared in Arbib [**1969**, §6.2]). Presumably, such considerations (based on Elgot and Robinson [**1964**], perhaps) motivated E^2. However, the irony of our motivation will be that it argues for the conventional definition as much as for E^2's new definition, and so does not advance their claim that their algebraic recasting of recursive function theory does indeed contribute to a theory of programs.

Briefly, we introduce the notion of a *recursive computer* M to compute functions $f: W^r \to W^s$ as follows:

Firstly, M has a state-space $\hat{Q}$ of such a nature that we can speak of partial recursive functions from $\hat{Q}$ to $\hat{Q}$; and in fact we supply $\hat{Q}$ with a state-transition function $\lambda: \hat{Q} \to \hat{Q}$ which is indeed partial recursive.

Then, a computation of f by M proceeds as follows:

(i) We encode our data $w \in W^r$ as an initial state in $\hat{Q}$ by applying some partial recursive function $\alpha: W^r \to \hat{Q}$.[1]

(ii) We repeatedly update the state of M from its initial value q_0 by applying λ until computation halts in the sense that we reach a state q for which $\lambda(q) = q$. Let this final state (if it is ever achieved) be q_f, and let $\tilde{\lambda}$ be the (partial) function $q_0 \mapsto q_f$.

(iii) We decode q_f by some partial recursive function $\beta: \hat{Q} \to W^s$ to obtain our answer $f(w) = \beta(q_f)$.

Consider, then, the following data:

(i) The α, β and λ for a Turing machine are all primitive recursive.

(ii) To get $\tilde{\lambda}$ from λ it is natural to use minimisation:

Let $\lambda(q, n)$ be the result of applying λ n times to q. It is defined from λ by the recursion scheme:

$$\lambda(q, 0) = q, \qquad \lambda(q, n + 1) = \lambda(\lambda(q, n)).$$

---

[1] Definitions of (primitive) recursive functions easily generalise to cases of mixed alphabets, but we shall not expose the niceties here.

Then we find the first n such that $\lambda(q, n) = \lambda(q, n+1)$ by *minimisation*

$$p(q) = \mu n[\lambda(q, n) = \lambda(q, n + 1)],$$

it being a standard exercise to recast this in the form of a "kosher" minimisation with the aid of some auxiliary primitive recursive functions. But then

$$\tilde{\lambda}(q) = \lambda(q, p(q))$$

and so must be partial recursive if λ is primitive recursive (or even partial recursive).

Thus if a partial function is computable by a Turing machine it can be obtained using primitive recursive functions and one minimisation, and so is partial recursive by our "old-fashioned" definition which obtains partial recursive functions by adding minimisation to the list of operations used to build up primitive recursive functions from the basis functions. The converse—that all partial recursive functions are Turing computable—is easy. This then is the machine-theoretic justification for the operation of minimisation.

Let us approach our recursive computer via an alternate tack to build toward E²'s characterisation. We view a *relation* $f: W^r \to W^s$ as being recursive just in case the relation

$$\begin{aligned} \tau\gamma f: W^{r+s} \to W^0: (w_1, w_2) &\mapsto 1, \qquad \text{if } w_2 \in f(w_1), \\ &\mapsto \varnothing, \qquad \text{if not,} \end{aligned}$$

is the set of all input strings for which some Turing machine will eventually terminate.

Now given any relation $g: W^r \to W^r$ with equal domain and co-domain we define its *closure* to be the relation

$$g^* = W^r \cup g \cup g^2 \cup g^3 \cdots : w \mapsto \bigcup_{n \geqq 0} g^n(w).$$

[Incidentally, one should note that this g^* is *not* the usual g^* of automata theory, inasmuch as here we consider closure under repeated composition whereas there we consider closure under repeated convolution.]

Now given α and λ for our recursive computer let us set

$$g = \lambda \times \lambda: \hat{Q}^2 \to \hat{Q}^2 \quad \text{and} \quad \hat{\alpha}: W^r \to \hat{Q}^2: w \mapsto (\alpha(w), \lambda\alpha(w)).$$

Then $g^* \circ \hat{\alpha}(w) = \{(\lambda^n\alpha(w), \lambda^{n+1}\alpha(w)) \mid n \geqq 0\}$.

Clearly, the computation initiated by w halts just in case at least

one element of $g^* \circ \hat{\alpha}(w)$ is of the form (q, q). If we compose $g^* \circ \alpha$ with the relation

$$\tau\gamma\hat{Q}: (q_1, q_2) \mapsto 1, \quad \text{if } q_1 = q_2,$$
$$\mapsto \varnothing, \quad \text{if } q_1 \neq q_2,$$

we then get 1, rather than $\varnothing$, iff the computation terminates. Thus, if $(\tau\gamma f)_{(\alpha,\lambda)}$ is the relation determined by our recursive computer (α, λ, β), we have the equality

$$(\tau\gamma f)_{(\alpha,\lambda)} = \underset{\text{check for termination}}{\tau\gamma\hat{Q}} \circ \underset{\text{update as often as possible}}{g^*} \circ \underset{\text{read in initial data}}{\hat{\alpha}}.$$

Since

$$\tau\gamma f = W^{r+s} \xrightarrow{\langle W^r, f\rangle^{-1}} W^r \xrightarrow{\Pi} W^0,$$

and

$$f = W^r \xrightarrow{\Pi_1^{-1}} W^{r+s} \xrightarrow{\langle \tau\gamma f, W^{r+s}\rangle} W^{r+s} \xrightarrow{\Pi_2} W^s,$$

we have that if $\mathfrak{A}$ is an admissible category which is closed under the taking of inverses, then a relation $f: W^r \to W^s$ is in $\mathfrak{A}$ iff the relation $\tau\gamma f: W^{r+s} \to W^0$ is in $\mathfrak{A}$.

Given the additional lemma that with closures and inverses we can reconstitute recursions (or left exponentiations) we are prepared to believe E²'s following elegant definitions:

A subcategory $\mathfrak{A}$ of $\mathfrak{R}(W^\cdot)$ is called *distinguished* if it is admissible and if it satisfies the following axioms:

1. *Unit.* The function $U: W^0 \to W$ is in $\mathfrak{A}$.
2. *Successors.* The functions $L_x: W \to W$ are in $\mathfrak{A}$ for each x in X.
3. *Closure.* If $f: W^r \to W^r$ is in $\mathfrak{A}$, then so is $f^*: W^r \to W^r$.
4. *Inverse.* If $f: W^r \to W^s$ is in $\mathfrak{A}$, then so is $f^{-1}: W^s \to W^r$.

The class of all distinguished subcategories of $\mathfrak{R}(W^\cdot)$ will be denoted by $\boldsymbol{R}(W^\cdot)$.

Let Γ be a class of functions $f: W^r \to W^s$. We then denote by $\mathfrak{R}_\Gamma$ the smallest element of $\boldsymbol{R}(W^\cdot)$ containing Γ and further set

$$\mathcal{P}_\Gamma(W^\cdot) = \mathfrak{R}_\Gamma(W^\cdot) \cap \mathcal{P}(W^\cdot),$$
$$\mathfrak{F}_\Gamma(W^\cdot) = \mathfrak{R}_\Gamma(W^\cdot) \cap \mathfrak{F}(W^\cdot).$$

The relations in $\mathfrak{R}_\Gamma$ are called Γ-*recursive*. Thus $\mathfrak{F}_\Gamma$ consists of the Γ-recursive functions. As usual, the verification of equivalence with

the earlier definition will be regarded as a standard exercise by the student with any reasonable background in recursive function theory.

The point of our lengthy discussion is that while E^2's definition has more algebraic elegance than the standard definition, it seems to be, if anything, less directly tied to simple principles of digital computer operation. Their definition of R takes us through the first half of E^2's book. There seems little point in retailing the contents of the second half. The results contain no surprises and can easily be verified by standard methods—which is not to deny that the algebra gives some of the statements a pleasing elegance.

In discussing the book under review with my colleagues, I have gained the following picture of Eilenberg's mathematical philosophy: "Mathematical intuition is built not from half-baked verbalisations and nonmathematical analogies, but rather from the study of austere and elegant proofs. Thus we must boil down the old treatments of automata theory, extract the mathematical residue, and polish and tailor this residue before presenting it to the next generation so that automata theory will become a truly mathematical subject." With this approach, one does not seek surprisingly new theorems *at this stage*, but simply tries to provide the right algebraic setting on which later work should build. To the extent that this review is critical it is because I feel that an important component of a mathematician's intuition is paramathematical and all traces of this component have been carefully excised. But this criticism should not blind us to the fact that such intuitions cannot become mathematics unless refined by the sort of elegant polishing presented by E^2. For example, much of my work seeks to establish the right setting for automata theory. However, in stopping when paramathematical intuition seemed sufficiently developed, I have often given proofs whose roughness an Eilenberg could rightly decry. Thus, while I would not use E^2 as the text for a course on recursion theory, it would seem to me highly desirable that a few weeks of such a course be devoted to E^2, with exercises designed to encourage the student to refine his presentation of proofs, thus complementing the less formal tools he will have acquired elsewhere in the course.

In short, the book does not touch (save in the one appendix) on the real meat of recursion theory—the proof of subtle undecidability results. The algebraic setting is not as general as that given to us by Rosza Péter, so that the chance is lost to provide the proper setting for undecidability studies for, say, second-order arithmetic logic or group theory. The motivation that might allow the computer theorist to see better how to make the transition back and forth between pro-