10. K. Knopp, *Problem book in the theory of functions* II, Dover, New York, 1952, 138 pp. [A good collection of interesting exercises and problems for second and third semester courses in classical function theory. (Vol. I is rather elementary.)]

11. J. G. Krzyz, *Problems in complex variable theory*, American Elsevier, New York, 1971, xix + 283 pp. [For supplementing a regular course. Mostly exercises, but some real problems.]

12. Ya. I. Rivkind, *Problems in mathematical analysis*, Noordhoff, Groningen, ca 1965, v + 98 pp. [Meant to supplement real variable courses; mostly routine on the hard side.]

13. D. O. Shklarsky, N. N. Chentzov, and I. M. Yaglom, *The USSR Olympiad problem book*, Freeman, San Francisco, Calif., 1962, xvi + 452 pp. [An outstanding collection of problems on elementary mathematics.]

14. W. Sierpiński, 250 *problems in elementary number theory*, American Elsevier, New York, 1970, vii + 125 pp. [About half routine, half challenging problems, a few quite challenging.]

15. G. Szász et al., *Contests in higher mathematics, Hungary* 1949–1961, Akadémiai Kiadó, Budapest, 1968, 260 pp. [A collection of highest quality.]

16. A. M. Yaglom and I. M. Yaglom, *Challenging mathematical problems with elementary solutions*, Holden-Day, San Francisco, Calif.; Vol. I: *Combinatorial analysis and probability theory*, 1964, ix + 231 pp.; Vol. II: *Problems from various branches of mathematics*, 1967, xi + 214 pp. [A collection of outstanding problems at the U. S. university level.]

HARLEY FLANDERS

BULLETIN OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 84, Number 1, January 1978

Rekursive Funktionen in der Komputer Theorie, by Rózsa Péter, Akadémiai Kiadó, Budapest, Hungary, 1976, 190 pp., $12.00.

The Theory of Recursive Functions developed in its present form in the decades following 1930. Pioneered by the work of Turing, Post and Church, it has aimed at making precise and at studying the notions of algorithm and computation.

A (partial) function from the set of natural numbers into natural numbers is *recursive* if it can be represented by an expression formed from certain *base functions* and the operations of *substitution*, *primitive recursion*, and *minimization*. The base functions comprise the *successor function* ($S(x) = x + 1$), the *null function* ($N(x) = 0$), and *projection functions* ($U_i^n(x_1, \ldots, x_n) = x_i$, where $1 \leqslant i \leqslant n$). Primitive recursion is used to define a function $h(z, x_1, \ldots, x_n)$ from recursive functions $f(x_1, \ldots, x_n)$ and $g(z, y, x_1, \ldots, x_n)$ by the pair of equations

$$h(0, x_1, \ldots, x_n) = f(x_1, \ldots, x_n),$$

$$h(S(z), x_1, \ldots, x_n) = g(z, h(z, x_1, \ldots, x_n), x_1, \ldots, x_n).$$

The operation of minimization defines a (possibly partial) function $f(x_1, \ldots, x_n)$ from a total recursive function $g(y, x_1, \ldots, x_n)$ as the "smallest y such that $g(y, x_1, \ldots, x_n) = 0$," and is written

$$f(x_1, \ldots, x_n) = (\mu y)[\, g(y, x_1, \ldots, x_n) = 0].$$

Note that all recursive expressions can be enumerated and, hence, all recursive functions.

A. Church conjectured in 1936 that this class of functions was precisely the class of all effectively computable functions [**1**]. More accurately, to every effective rule for computing a sequence of natural numbers there exists a recursive expression with number e such that the function defined by the rule

has the same value as the recursive function

$$\varphi_e(x) = U\big[(\mu y)(T(e, x, y))\big]$$

in Kleene's notation [2].

To date all attempts to define effective procedures which are not expressible relative to a suitable coding in terms of recursive functions have failed, and Church's thesis has gained wide-spread acceptance. In the sequel the study of recursive functions was soon considered to provide the theoretical foundation for computer science and, in this aspect, concerned itself with questions of effective computability, i.e. whether there exists an effective procedure for a given class of problems.

Early research, interested in finding evidence for and against Church's thesis, studied a wide spectrum of models of computational processes and found that by using suitable encoding techniques it could be shown that all models had the same computational power, thus corroborating Church's thesis.

Péter's book is written for the nonexpert and argues more often by example or intuition than by complete detailed proofs. The unifying concept of the text is to give evidence for Church's thesis. To this end various applied areas of computer science are surveyed. It is shown that each has an effective translation into the class of recursive functions and vice versa, thus establishing the computational equivalence of these areas.

To illustrate this "translation" process establishing the equivalence of two models of computation let us take the class of recursive functions as one system, and a simple computer model as the other. We assume that our computer has an unlimited number of storage cells each capable of holding one natural number, and that it can manipulate these cells incrementing the content of a cell by one, or decrementing it by one unless it is zero, or interrogating if the cell contains zero. At the beginning, the input argument(s) are deposited in the first (n) cells. The computer then executes a finite program of instructions manipulating the cell contents, repeating possibly some instructions, and eventually may reach a HALT instruction. At that point the "result" of the computation is stored in the first cell. Other cells may have been used for intermediate storage.

Showing that our computer model is computationally at least as powerful as the class of recursive functions amounts to exhibiting programs which can compute the base functions and can simulate all operations involved in defining a recursive function. Except for the operation of primitive recursion which has to be transformed into an equivalent iteration schema first, the task is quite straightforward. Once the programs have been found, they provide the tools for an effective translation of any recursive function f into a program P_f on our computer which evaluates the function value for each argument assignment input to P_f.

The key to showing that recursive functions can "simulate" the computations of our computer lies in coding strings of symbols and numbers into a single number. Usually a "Gödel numbering" is chosen: The sequence of numbers $n_1, \ldots, n_k$ is encoded by the product $p_1^{n_1} \cdot p_2^{n_2} \cdot \cdots \cdot p_k^{n_k}$ where p_i is the ith prime number. It is clear that we can encode sequences of such *Gödel*

numbers in the same way, so that arbitrary (tree-like) structures of numbers can be uniquely represented by a single natural number. We then enumerate the alphabet of our computer model and encode the entire program and used storage contents along with a description of the next step to be executed by a suitably structured Gödel number.

We have to show that the encoding and decoding of these numbers, e.g. extracting a specific exponent, can be done by recursive functions. These functions are the building blocks in defining a recursive function which simulates a step in the computation in our model. In this manner we can show that to every program P of our computer there exists a recursive function f_P which, given the inputs to P as argument, has the corresponding output of P as value.

Despite the fact that the essence of equivalence proofs always amounts to a programming effort, it is clear that it may require considerable ingenuity to find an elegant and economic coding permitting the translation. Also, the formal details of a translation often may obscure the intuitive idea of the underlying strategy. The book tries to score well on both accounts. The author takes great pains at motivating and explaining the proof strategy, and gives a wealth of different encodings for a variety of problems: Translation of arithmetic expressions, recursion and iteration schemata, two-level grammars, interpreters for LISP and decision-table programs, etc. Each subject is well explained, and for each it is investigated how to translate problems in the particular area into terms of recursive functions. Over the past twenty years the author has also published a number of articles giving each topic a rigorous and formal treatment.

The book shows the connection between computer science and recursion theory. It can be taken in two ways: On the one hand, the mathematician who perhaps is inclined to see recursive functions as the essence of what he understands effective computability to be, may read this book in order to school his eye for the different guises under which these functions appear in the world of computing. The computer scientist, on the other hand, perhaps interested in what he considers the practical issue in his field, could read this book to learn about the structural unity underlying a diversity of questions he has considered.

Judging from the style of presentation, the author addresses the second view point more directly. A large part of the book is devoted to motivate the reader to become interested in the mathematical façet of the problems arising in computer science, and it is apparent that the author can draw on considerable pedagogical experience for this.

As indicated, Church's thesis cannot be proved or disproved unless the notion "effective rule" is given a formal definition. Short of doing so, one can speculate whether accepting the thesis would limit our view of the potential capabilities of computers as we might succeed in designing. In light of the material found since the formulation of the thesis, this appears unlikely. The last chapter in the book briefly mentions this and also cites without further discussion an incompleteness result by Kalmár [3] as an intuitive counterargument.

Theoretical research in computer science has passed from recursion theory

to computational complexity. Instead of investigating if there exists an algorithm to solve a given problem, more attention is given to study how economical (space or time intensive) an algorithm for a given problem potentially can be, thereby classifying decidable problems into (sub) hierarchies of difficulty. To the computer scientist, then, the book compiles material which is well understood for some time now. This is in line with the tutorial level at which the book is written.

REFERENCES

1. A. Church, *An unsolvable problem of elementary number theory*, Amer. J. Math. **58** (1936), 345–363.

2. S. Kleene, *Introduction to metamathematics*, Van Nostrand, Princeton, N.J., 1952. MR **14**, 525.

3. L. Kalmár, *An argument against the plausibility of Church's thesis*, Constructivity in Mathematics: North-Holland, Amsterdam, 1959, pp. 72–80. MR **21** #5567.

CHRISTOPH M. HOFFMANN

BULLETIN OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 84, Number 1, January 1978

Applied functional analysis, By A. V. Balakrishnan, Springer-Verlag, New York, Heidelberg, Berlin, 1976, vii + 309 pp., $19.80.

For present purposes "functional analysis" will mean the study of Hilbert spaces and of (not necessarily continuous) linear operators between such spaces. Let us recall that Hilbert spaces are characterized among general Banach spaces by any one of numerous geometric properties, such as the parallelogram property of the norm (Jordan-von Neumann, 1935), or the existence of a norm-one projector onto every (closed linear) subspace (Kakutani, 1939; spaces of dimension $\leqslant 2$ are exceptional). These two conditions can be jointly utilized to show that any Banach space of dimension > 2 having sufficiently many finite dimensional linear metric projectors is a Hilbert space (Rudin-Smith, 1961). More recently, isomorphs of Hilbert spaces have been characterized in several spectacular ways; for example, as those Banach spaces in which every subspace is complemented [**15**], or those which obey a Central Limit Theorem property (due to several authors, see [**1**] for one presentation).

The operator theory for Hilbert spaces is dominated by the interplay between an operator and its adjoint which, thanks to the Riesz representation theorem, can be defined on the codomain of the given operator. Major achievements in this theory include the spectral theorem for normal operators (Hilbert, Riesz, von Neumann, Stone, Gelfand-Naimark, Segal, et al., 1906–1951), the polar decomposition of closed operators (von Neumann, 1932), the dilation theory of Halmos and Sz.-Nagy (1950–1955), which in turn has led to the characteristic function and canonical model approach to the study of contractive operators (Livšic, Sz.-Nagy-Foiaş, et al., 1946–1967), and the triangular representation of compact operators (Livšic, Brodskiĭ, et al., 1954–1969). Presentations of these theories and much more are given in [**7**], [**10**], [**18**], [**19**]. Note that we are for the most part leaving completely aside the vast subject of operator algebras.