BULLETIN DE L'ACADÉMIE
POLONAISE DES SCIENCES
Série des sciences techniques
Volume VIII, No. 11-12 — 1960

On Realization of Recursive Schemes in the Address-free Computer

by

Z. PAWLAK

Presented by P. SZULKIN on October 18, 1960

In the papers [1] and [2] a certain method of recording arithmetical formulae was given as well as the organization of a digital computer based on this method.

The present paper gives a simple method of recording recursive schemes in the formalized language, given in [2]. The aforementioned method has been chosen with a view to application to a small serial computer.

Several, different, more general, interesting methods, were given by A. Ehrenfeucht [3] (lecture).

Parenthesis-free notations with "blank space"

Let R be a symbolic language defined as follows: The primitive expressions of the language R are

a) $x, y, z, \ldots$, are independent variables,

b) $\bar{x}, \bar{y}, \bar{z}, \ldots$, — inductive variables,

c) $\Delta_1, \Delta_2, \ldots, \Delta_n$ — symbols of dyadic operations,

d) $*$("blank space") — a place, within which a partial result is to be registered,

e) π denotes iteration of formula computation.

If α and β are independent variables or inductive variables, then $\alpha\beta\Delta_i$ is a well-formed expression in R.

If α and β are well-formed expressions in R, then $\alpha\beta(*)$ is also a well-formed expression in R, where $\beta(*)$ denotes an expression obtained by substitution of an asterisk for an arbitrary variable in β.

In turn, if α is the well-formed expression in R, then $\alpha*$ is a formula in R; and if α is a formula in R containing at least one inductive variable, then $\alpha n\,\pi$ is the recursive formula in R.

The formulae in the symbolism presented above should be read as follows.

We divide the whole formula into triples of symbols, beginning from the left-hand. We read the formula:

Carry out on the given arguments the indicated operation and put the result in the nearest blank space at the righthand.

If one of the arguments is the inductive variable, then, as its value, take the number occurring in the place of the last asterisk.

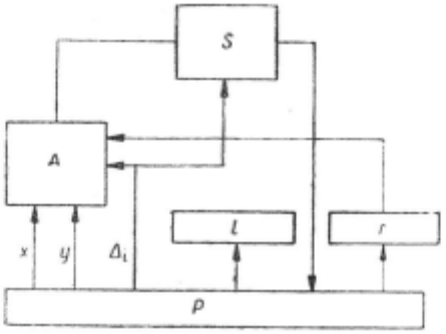
The expression $a n \pi$ denotes n-fold repetition of the a-formula computation. For instance, raising to a power of x^y, defined by recursion, takes the form:

$$\text{a)}\quad x^0 = 1, \qquad \text{b)}\quad x^{y+1} = x^y \cdot x.$$

This scheme, in the introduced symbolism, can be written as follows: $zx \cdot * y \pi$, where the initial value of z is 1.

Organization of address-free machine for calculating of recursive schemes

A simplified scheme of computer for calculation of recursive schemes by means of the above presented symbolism is shown in the Figure. The



General block scheme of address-free computer for calculation of recursive formulae

machine consists of the following elements: memory P, arithmometer A, register of inductive variable r, iteration counter l and control unit S. The procedure of computation of non-recursive formulae is the same as is given in [2], i.e. the formula is recorded in the memory of the computer; the control unit scans subsequent pairs of arguments and sets the appropriate operations in the arithmometer, and sends the result of calculation to the nearest blank space.

In the case when the recursive formula is to be computed, the result of computation of the latter is always located in the register r and the number of iterations — in the iteration counter l. The register r is not necessary for realization of recursive schemes, it is merely applied for the sake of simplicity of the construction of the machine.

If the control unit finds in the memory the inductive variable, then, as a value of the latter it takes the number located in the register r.

After each iteration of computation, the number in the iteration counter l decreases its value by one. The computation is finished, if in the iteration counter l number zero appears.

All the partial results should be erased from the memory before each new iteration. This operation requires certain indications, whether the given number in the memory represents initial data or partial results.

For simplification in construction of the machine, the question of the above mentioned indication has been disregarded.

In machine computations iterative schemes often occur, in which the number of iterations is not known "in advance", and the end of the iteration depends on the fulfilment of certain relations.

In order to include such schemes into the formalized language under consideration, the primitive symbols should be supplemented by symbols of the following relations $=, \neq, >, \leqslant$.

If α and β are formulae, then the expressions $\alpha\beta =, \alpha\beta \neq, \alpha\beta >, \alpha\beta \leqslant$, become predicates.

If α is a formula and β — a predicate, then $\alpha\beta$ is also a recursive formula. (We assume that the formula contains at least one inductive variable).

INSTITUTE OF MATHEMATICS, POLISH ACADEMY OF SCIENCES
(INSTYTUT MATEMATYCZNY, PAN)

REFERENCES

[1] Z. Pawlak, *The organization of digital computers and computable functions*, Bull. Polon. Sci., Sér. sci. techn., 8 (1960), 41.

[2] — , *Organization of the address-free digital computer for calculatitng simple arithmetical expression*, Bull. Acad. Polon. Sci., Sér. sci. techn., 8 (1960), 253.

[3] A. Ehrenfeucht, Lecture in the Institute of Mathematics of the Polish Academy of Sciences, 1960.