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

COMPUTERS

On the Application of the Rule of Substitution in the Organization of an Address-free Computer

by

Z. PAWLAK

Presented by P. SZULKIN on October 19, 1960

Papers [1] and [2] described the organization of an address-free computer. The present paper deals with setting by substitution arithmetical functions (linking of sub-programmes) in parenthesis-free symbolics and the resulting organization of the machine.

Definition of the rule of substitution in parenthesis-free symbolics

Arithmetical functions noted in the formal language L defined below will be considered here.

The language L contains the following primitive symbols:

a) $x, y, ..., z$ variables,

b) $+, -, \cdot, /$, symbols of dyadic operations,

c) $*$, symbol denoting the place, where the partial result is located.

If α and β are variables, then $\alpha\beta+$, $\alpha\beta-$, $\alpha\beta\cdot$, $\alpha\beta/$ are well formed formulae.

If α and β are well-formed formulae, then $\alpha\beta(*)$ is a well-formed formula also, where $\beta(*)$ denotes the expression obtained by replacing the arbitrary variable with the symbol $*$ in β. If α is a well-formed formula, $\alpha *$ is an arithmetical formula.

Rule of substitution. If $\alpha, \beta_1, \beta_2, ..., \beta_n$ are well-formed formulae and the variables $x_1, x_2, ..., x_n$ occur in the expression α in the stated order, and the formulae $\beta_1, \beta_2, ..., \beta_n$ are to be correspondingly substituted for the variables $x_1, x_2, ..., x_n$, then the formulae $\beta_1\ \beta_2 \ldots \beta_n\ \alpha$ are also well-formed formulae. We shall call the variables $x_1, x_2, ..., x_n$ substitutional variables and denote them by $\bar{x}_1, \bar{x}_2, ..., \bar{x}_n$.

It is easy to note that, if the formula takes the form $\alpha_1, \alpha_2, ..., \alpha_n$, the result of each calculation of the formula α_i is located in the place of the nearest right substitutional variable (and of all its further equivalents).

Organization of a computer with external linking of sub-programmes

The linking of sub-programmes in a computer provided with only a small memory takes place on the outside of the machine. This means that each elementary function is programmed and computed separately. The organization of a computer of this kind is shown in Fig. 1. The computer consists of an operating memory *Pr*, an arithmometer *A*, the control *S*, the auxiliary memory *Pp* and of the external memory *Pz*. The operating memory together with the arithmometer operates on the principle described in [2], i.e. computes the values of the subsequent

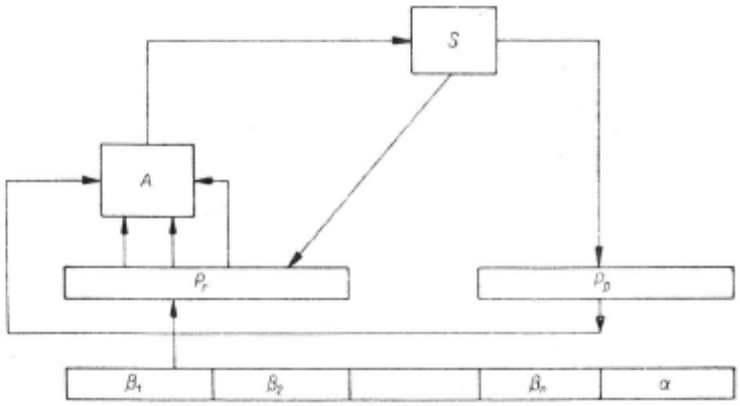

Fig. 1. Organization of address-free computer with external linking of sub-programmes

elementary formulae $\beta_1, \beta_2, \ldots, \beta_n$. The result of the computation of each elementary formula β_i is located in the successive places of the auxiliary memory *Pp*. If one of the arguments contained in the operating memory *Pr* is a substitutional variable, its value is taken from the successive location of the auxiliary memory. At the same time this value is located in all the places of the operating memory denoted by the same symbol of the substitutional variable. The external memory *Pz* (e.g. the paper tape) is used for collecting the successive expressions $\beta_1, \beta_2, \ldots, \beta_n$. After the expression α_1 is computed in the operating memory *Pr*, the next expression β_{i+1} from the external memory *Pz* is located in the operating memory.

Organization of a computer with an internal memory of sub-programmes

A computer with large internal memory may possess often used elementary functions (sub-programmes) permanently noted in the internal memory. An example of an address-free computer operating on this principle is shown in Fig. 2. Its scheme is similar to the one shown in Fig. 1 with the only difference that the elementary functions are not noted in the external memory *Pz*, but are permanently contained in the internal sub-programme memory *Ps* in a manner providing sufficiently

rapid access to each particular elementary function, so that the entire range of the sub-programme memory is superfluous. The computer is, moreover, equipped with a control memory *Pc*, where the order of computing the componential functions is given. The control *S* takes up

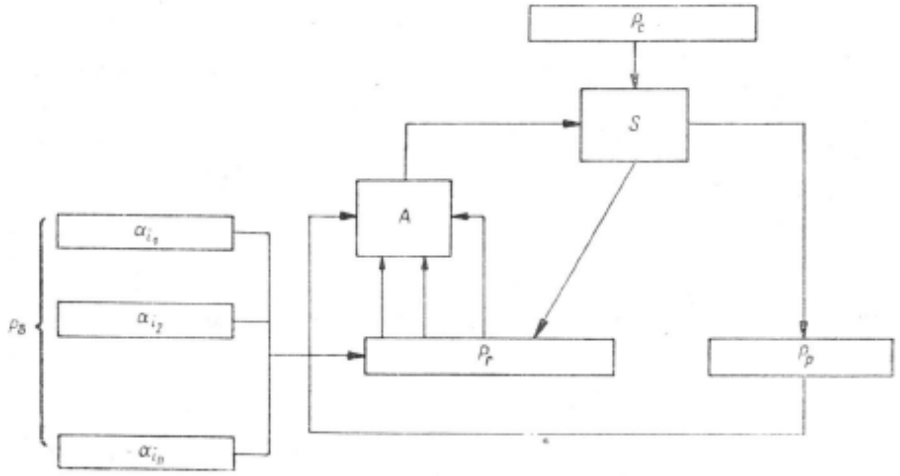

Fig. 2. Organization of address-free computer with internal memory of sub-pragrammes

the elementary functions contained in the control memory *Pc* according to information supplied by the control memory *Pc* and places them in the operating memory *Pr*. The value of the elementary functions is computed in the manner described in the former example.

Organization of an address-free computer for multiple computation of the function values for different arguments

Practice requires often computing the value of the same function for different values of arguments. A simplified scheme of a computer designed for this purpose is shown in Fig. 3, operating on exactly the same

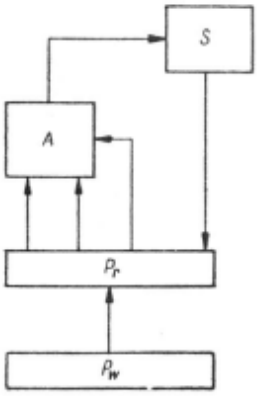

Fig. 3. Organization of address-free computer for manifold computation of function values

principle as the computer described in [2], with the only difference consisting in its being provided with an additional input memory *Pw* (input register), where the values of all the successive arguments are contained in the order in which they appear in the formula. This is a particular case of organization shown in Fig. 1, where to the formula α containing the variables $x_1, x_2, ..., x_k$ are substituted not the expressions $\beta_1, \beta_2, ..., \beta_k$, but the numerical values of the arguments $n_1, n_2, ..., n_k$. This notation may be treated as one similar to the λ-system of Church. Here α is the symbol of the function, while $\alpha\ n_1, n_2, ..., n_k$ is the value of the function for the values $n_1, n_2, ..., n_k$ of the variables $x_1, x_2, ..., x_k$, respectively.

I avail myself of this opportunity to express my gratitute to Mr. Ehrenfeucht and Mr. Grzegorczyk for their aid in elaborating the present subject.

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

REFERENCES

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

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