PROGRAMMING IN MEANINGFUL NOTATIONS

1964

SovietRxiv

Abstract

Full Text

CYBERNETICS AND CONTROL THEORY

A. L. BRUDNO

PROGRAMMING IN MEANINGFUL NOTATIONS

(Presented by Academician P. S. Novikov on 14 IX 1963)

1. **Arrangement and order of writing a program.** The program sheet is divided into two parts—the **meaningful** (left) and the **coded** (right). The program itself is written in the meaningful part, in special **meaningful** notations. A **dictionary** (memory allocation) is attached to the program for coding. The completed program is checked, coded, and punched.

2. **Notation of cells.** If the address of a cell is mentioned in the commands of a program, then it receives a notation coinciding with, or reminiscent of, the notation of the contents of the cell. Thus, for example, the addresses of cells containing the letters a, b, the constants c_1, c_2, the variables x, y, the numbers 0.253 or 1953, and so on, are denoted respectively by a, b, c_1, c_2, x, y, 0.253, 1953, etc. In a three-address machine, (i, j, k) denotes a cell containing the integers i, j, k in their corresponding addresses. If in this address there is the letter ω, this means that all binary digits of the address are occupied by ones. Working cells for short-term storage of unnamed intermediate results are denoted by $R1$, $R2, \ldots$. Variable cells of the program are denoted by ρ_1, $\rho_2, \ldots$. Standards are denoted by 1, 2,.... Transfer of control and sending of contents are indicated by an arrow, but if it is difficult to reach the cell, then its address must also be denoted somehow. The proper address of a command may be denoted by the letter . If a cell has the notation, for example, $s3$, then $s3-1$ denotes the preceding cell, and $s3+1$ the cell following it. The notations $\rho 1+5$, $+3, \ldots$ have an analogous meaning. It is obvious that **one and the same memory cell receives several different notations if, in the course of computation, differently named results enter it**. If some command address has the meaning not of a cell number but precisely of a number, and the possibility of confusion arises, then such a number is put in quotation marks.

3. **Notation of commands.** Each machine command receives a strictly definite notation. It is chosen so as to be easy to read, but it **preserves the order of arrangement of addresses in the machine**. Only the operation code is replaced by a sign written in a place familiar to the mathematician. It is forbidden to write commands that are not present in the machine codes, i.e., commands that cannot be coded.

4. **Dictionary.** Compiling the dictionary is not a mechanical task—it is necessary to choose places for different blocks of the program, to keep track of cells being freed, not to occupy needed ones, and so on.

5. **Coding** is a mechanical task that requires no mathematical qualification. It is performed by laboratory assistants. Coding errors are rare, and after checking none remain. The time required for coding is ten times less than the programming time.

6. **Examples of programs.** Let the list of commands and the arrangement of commands and numbers in a memory cell of a conventional three-address machine be as follows:

	l digits	k digits	k digits	k digits
Command	Code	III address	II address	I address
Floating-point number	sign	Characteristic exponent	Mantissa	Mantissa
Integer	Code	Number	Number	Number

The sign + (−) is represented by the digit 0 (1). An indelible zero is stored in the zero cell.

Code	Designation	Code	Designation
00	$a \to b;\quad c$	10	Stop
01	$a \leftrightarrow b;\quad c$	11	Input $b;\quad c$
02	$\lvert a\rvert < \lvert b\rvert;\quad c$	12	$a \leftarrow b = c$
03	$a < b;\quad c$	13	$a \wedge b = c$
04	$a + b = c$	14	$a+, b = c$
05	$a - b = c$	15	$a-, b = c$
06	$a \cdot b = c$	16	$a\cdot, b = c$
07	$a : b = c$	17	$a :, b = c$

Description of commands. 00 transfers the contents of cell a to cell b and passes control to cell c. In particular, the command $(0;\ 0, 0, c)$ gives an unconditional transfer. 01 enters into cell b the information $(0;\ 0, 0, a)$ (which is also the integer a with zero code and the command $(0 \to 0;\ a)$) and passes control to cell c. 02 and 03 transfer control to cell c (or to the next cell) if the indicated inequalities are satisfied (not satisfied) for the contents of cells a and b, as floating-point numbers. These same commands work (in the obvious way) when comparing commands or integers with equal codes. 11 is input or output into cell b from external devices with transfer of control to c. The types of input and output are indicated in the left address. 12 is a shift of the entire contents of cell b by the number $a \leqslant 3k + l$. In the semantic part the direction of the shift is indicated by an arrow; in coding, a shift to the right is specified

in complementary code. 13 is logical intersection. 14–16 are operations on positive integers modulo 2^{3k}. The code part is transferred to the result without change from the cell indicated in the left address. The result of division 17 is the integer part of the quotient.

a) A short standard program for $V = \sqrt{a}$ (or 0) for $a > 0$ (for $a \leqslant 0$). The iterative formula is $2V_1 = a + 1$ and $2V_{n+1} = a : V_n + V_n$. Continue the iterations until $V_{n+1} < V_n$.

Handwritten semantic notation:

```
0 < a
0 $\rightarrow$ √
a + 1 = 2√
2√ $\rightarrow$ R1
(2√) : 2 = √
a : √ = a/√
a/√ + √ = 2√
2√ < R1
output
```

Machine program:

Address	Code	III address	II address	I address
2536	03	0000	0077	2540
2537	00	0000	0100	2546
2540	04	0077	0021	0102
2541	00	0102	0101	2542
2542	07	0102	0022	0100
2543	07	0077	0100	0102
2544	04	0102	0100	0102
2545	03	0102	0101	2541
2546			–	

DICTIONARY

Designation	Address	Contents
1	0021	41 4000 0000 0000
2	0022	42 4000 0000 0000
a	0077	–
$2\sqrt{};\ a/\sqrt{}$	0102	–
$R1$	0101	–
$\sqrt{}$	0100	–

b) Send to cell a the maximum of the numbers $a_1, a_2, \ldots, a_n$, arranged in consecutive cells.

$\rho,\ \rho+1$
$a_1 < a$
$a_n < a$
$\rho + (1,0,0) = \rho$
$a_i < a$
$0 + \rho = (\ +1)$
$a_i \to a$
$|\rho| < |\ \ |$

c) Matrix multiplication by a vector: $y_i = a_{i1}x_1 + ... + a_{in}x_n$. The matrix is written by rows, and within a row by columns.

variant 1

1) ρ_{i1}
$x_1 \cdot a_{11} = R1$
$x_n \cdot a_{11} = R1$

2) ρ_i
$s + 0 = y_1$
$s + 0 = y_n$
$\rho_{i1} + (0,n,0) = \rho_{i1}$
$\rho_i + (0,0,1) = \rho_i$
$0 \to s$

ρ_{i1}) ρ_{ik}
$x_i \cdot a_i = R1$
$\rho_{ik} + (1,1,0) = \rho_{ik}$
$x_k \cdot a_{ik} = R1$
$s + R1 = s$
$|\rho_{ik}| < |\ \ 1|$

ρ_i) $s + 0 = y_i$
$|\rho_i| < |\ \ 2|$
Stop

variant 2

$\rho 1)$ $0 \to i$
$0 \to y_1$
$0 \to k$

$\rho 2)$ $a_{11} \cdot x_1 = R1$
$\rho 3)$ $y_1 + R1 = y_1$
$k + (0,0,1) = k$
$\rho 2 + (1,1,0) = \rho 2$
$|k| < |(0,0,n+1)|$
$i + (0,0,1) = i$
$\rho 1 + (0,1,0) = \rho 1$
$\rho 2 - (0,n,0) = \rho 2$
$\rho 3 + (1,0,1) = \rho 3$
$|i| < |(0,0,n+1)|$
$\rho 1 - (0,n,0) = \rho 1$
$\rho 2 - (n^2,0,0) = \rho 2$
$\rho 3 - (n,0,n) = \rho 3$
Stop

d) Compute and print

$$\exp\left(-\sqrt{r^2+\rho^2-2r\rho\cos\varphi}\right)$$

for m equally spaced values of ρ from ρ_1 to ρ_2, with n equally spaced values of φ from φ_1 to φ_2.

$r \cdot r = r^2$
$\rho_2 - \rho_1 = \Delta\rho$
$\Delta\rho : m = \Delta\rho$
$\Delta\rho : 2 = R1$
$\rho_2 - R1 = \quad \rho$
$\varphi_2 - \varphi_1 = \Delta\varphi$
$\Delta\varphi : n = \Delta\varphi$
$\Delta\varphi : 2 = R1$
$\varphi_2 - R1 = \quad \varphi$

S) $\rho_1 \to \rho$
$\rho + \Delta\rho = \rho$
$\rho \cdot \rho = R1$
$r^2 + R1 = (r^2 + \rho^2)$
$r \cdot \rho = R1$
$R1 + R1 = 2r\rho$
$\varphi_1 \to \varphi$
$\varphi + \Delta\varphi = \varphi$
$\varphi \to \alpha$
cos
$(2r\rho) \cdot \beta = R1$
$(r^2 + \rho^2) - R1 = \alpha$
$\sqrt{}$
$0 - \beta = \alpha$
exp
$\beta \to \alpha$
output
$|\varphi| < | \quad \varphi|$
$|\rho| < | \quad \rho|; \ S$
Stop

The "command"cos for a call to the standard cosine program is deciphered by the coder as "$(+1) \leftrightarrow$ cos output; cos start."This is done according to an established convention, which may also be different if, for example, all standard programs have a common output. The "commands" $\sqrt{}$, exp, and output (conversion to decimal form and printing) are deciphered analogously. In addition, it is assumed that the arguments of all standard programs must be located in the common cell α, and the results are obtained in β.

The examples given are sufficient for the reader to be able to continue them and form his own opinion about substantive programming. Since 1953, programming has been done in this way at the Institutes of Electronic Control Machines and of Theoretical and Experimental Physics of the Academy of Sciences of the USSR. Subsequently several more groups connected with us adopted this method.

In the generally accepted **coding** programming, a program is written in memory-cell numbers with brief explanations. The information being transformed is indicated not explicitly, but as the contents of explicitly indicated memory-cell numbers. In practice, reading each program requires its own **dictionary**. Conversely, in **substantive notation** one gives **operations on explicitly indicated information**, written in the language of the problem (formulas, a computational scheme). Thanks to this, a substantive program has the form **customary** for a mathematician: it contains the usual operations on the usual letters and digits. It is easier to write, read, check, and modify—different parts of the program can be written independently (without memory allocation). Some errors (slips) disappear altogether; for example, transferring control to a floating-point number. A programmer quickly becomes accustomed to substantive notation, but for the rapid reading of other people' s programs it is necessary to use a single symbolism for a given machine.

At the same time, substantive programming, unlike automatic programming, possesses the flexibility of coding programming: it makes it possible to use nuances of the operation set, to economize on operations, memory cells, computation time, and so on. The possibility of debugging at the console is preserved, since the encoded program is written on the right side of the sheet in the same form and in the same cells in which the program is located in the machine, while the left (substantive) part speeds up the analysis of the right part.

The author is aware that substantive programming removes a number of difficulties of coding programming (in addresses), thereby stimulating the development of automatic programming (programming programs, Algol, etc.) and diminishing the practical usefulness of the latter.

Institute of Electronic Control
Machines

Received
11 IX 1963

Note: Figure translations are in progress. See original paper for figures.