Mathematics

1969

SovietRxiv

Abstract

Full Text

Mathematics

K. V. Shakhbazyan

On Schematic Programming Languages

(Presented by Academician L. V. Kantorovich, 21 I 1969)

Recently, the problem of formalizing the syntax and semantics of languages has become especially urgent ([1]). The present work is devoted to solving this problem for a certain special type of languages—schematic languages, which until now have existed in the form of insufficiently formalized input languages for programming programs ([2–8]).

Suppose that some arbitrary set of objects Ω is given. We shall call it the semantic set. Below we shall be concerned with the notation, in schematic languages, of algorithms that realize mappings of some subsets of the set Ω into others.

Let there be a grammar $\Gamma p'$ generating certain sign constructions in the union of the alphabets A and M, where M is the basic alphabet, whose symbols we shall call labels, and A is an alphabet of auxiliary symbols. Let a word α be an admissible construction of the grammar $\Gamma p'$, and let $p = (M_1, \dots, M_n)$ be the list of labels occurring in α, written without repetitions in the order of their occurrence. In particular, this list may be empty. To each label there may correspond an object from some subset $\Omega_0 \subset \Omega$. We shall assume that a semantic algorithm $F'(\alpha, p)$ is given, which realizes (for each fixed admissible $\alpha = \alpha_0$) a mapping of sequences of objects from the set Ω_0 onto some set $\Omega' \subset \Omega$.

Let us now consider a grammar Γp generating constructions of the form

$$M_1 \sim \alpha_1, \quad M_2 \sim \alpha_2, \dots, M_m \sim \alpha_m. \tag{1}$$

The symbols $\sim$ do not belong to A; $M_1, \dots, M_m$ are distinct and belong to the alphabet M; $\alpha_1, \dots, \alpha_m$ are constructions generated by the grammar $\Gamma p'$. We shall call words of the form (1) schemes, and the grammar Γp a schematic grammar.

Each subword of the word (1) of the form

$$M_i \sim \alpha_i \tag{2}$$

is called a line of the scheme. The symbol M_i is called the external label of line (2). The labels occurring in α_i are called the internal labels of line (2). The

labels $M_1, \dots, M_m$ are called the results of scheme (1). The internal labels of lines included in scheme (1) that are not results are called the arguments of scheme (1).

Let us now examine in more detail the structure of scheme (1). To it one may associate a graph $G(X, \Gamma)$, where X is the set of all labels occurring in the scheme, and the mapping Γ is defined as follows:

$$\Gamma_M = \begin{cases} \{M_1, \dots, M_r\}, & \text{if } M \text{ is the external label of some line of scheme (1),} \\ & \text{and } M_1, \dots, M_r \text{ are the internal labels of that same line;} \\ \emptyset, & \text{if } M \text{ is not the external label of any line of the scheme.} \end{cases}$$

If, for some vertex M of the graph G, there is no path issuing from it and leading back to M, then such a vertex is called a **simple vertex**. If, however, for the vertex M there exists at least one such path, then it is called a **recursive vertex**.

Let M be a recursive vertex. The set of all vertices of the graph G lying on paths issuing from the vertex M and leading back to M will be called a **strong component** and denoted by $\psi(M)$.

Obviously:

1. Every vertex belonging to a strong component is recursive.
2. If M_1, M_2 are vertices belonging to the same strong component, then $\psi(M_1) = \psi(M_2)$.
3. If two strong components intersect, then they coincide.
4. The set of vertices of the graph decomposes into disjoint sets, each of which either consists of a simple vertex of the graph or is a strong component. We shall call these sets **components** of the graph G.

Thus,

$$X = \bigcup_{i=1}^{N} Y_i, \qquad Y_i \cap Y_j = \emptyset \quad \text{for } i \neq j,$$

where Y_i $(i = 1, \dots, N)$ is a component, and N is the number of components.

5. The set of components is partially ordered: Y_i precedes Y_j if there exist vertices $M' \in Y_i$, $M'' \in Y_j$, connected by a path from M' to M''.

Using the concepts introduced, we shall now define the semantics of scheme (1), i.e., we shall define the dependence between the values of the results of the scheme and the values of the arguments.

Introduce some order on the set of components that preserves the indicated partial order. Let the numbering $Y_1, \dots, Y_N$ reflect precisely this arbitrarily chosen order.

Let the arguments of scheme (1) be $M_1^0, \dots, M_\nu^0$. Assign to them certain concrete values from Ω: $\mathfrak{Z}_1^0, \dots, \mathfrak{Z}_\nu^0$. We shall define the process of computing the values corresponding to the results of scheme (1) from the values $\mathfrak{Z}_1^0, \dots, \mathfrak{Z}_\nu^0$. This process consists of N steps—according to the number of components of the graph G, and at the i-th step the values are determined for those results that belong to the set Y_i.

Suppose that, after the execution of i steps, values have been determined for all results of the scheme belonging to the sets $Y_1, \dots, Y_i$. We now describe the $(i+1)$-st step, at which the values of the results belonging to Y_{i+1} are determined. Here two cases are possible:

1) $Y_{i+1} = \{M_j\}$, where M_j is a simple vertex of the graph G. Then either M_j is an argument of the scheme and some value is assigned to it, or M_j is the external label of the row $M_j \sim \alpha_j$.

Let $M_j^1, \dots, M_j^{s_j}$ be the internal labels of the row $M_j \sim \alpha_j$. They may be either arguments of the scheme or results belonging to the sets $Y_1, \dots, Y_i$. And then, by assumption, certain values from Ω have already been assigned to all of them. Denote them by $\mathfrak{Z}_j^1, \dots, \mathfrak{Z}_j^{s_j}$. Then to the result M_j is assigned the value

$$\mathfrak{Z}_j = F\left(\alpha_j, \left(\mathfrak{Z}_j^1, \dots, \mathfrak{Z}_j^{s_j}\right)\right).$$

2) Y_{i+1} is a strong component. Let $Y_{i+1} = \{M_{i1}, \dots, M_{ir_i}\}$.

Consider the rows of scheme (1) with external symbols $M_{i1}, \dots, M_{ir_i}$. These will be the rows

$$\begin{gathered} M_{i1} \sim \alpha_{i1}, \\ \dots \\ M_{ir_i} \sim \alpha_{ir_i}. \end{gathered}$$

Let the row $M_{il} \sim a_{il}$ $(l = 1, \dots, r_i)$ contain the internal labels $M_{il}^1, \dots, M_{il}^{s_i l}$. They may be either arguments of the scheme, or labels belonging to the sets $Y_1, \dots, Y_i$ (in these two cases their values have already been determined), or symbols belonging to Y_{i+1}. Then, from the recursive relations

$$\begin{gathered} z_{i1} = F(a_{i1}, (z_{i1}^1, \dots, z_{i1}^{s_{i1}})), \\ \dots\dots\dots\dots\dots \\ z_{ir_i} = F(a_{ir_i}, (z_{ir_i}^1, \dots, z_{ir_i}^{s_{ir_i}})), \end{gathered} \tag{3}$$

if possible, $z_{i1}, \ldots, z_{ir_i}$ are determined—the values of the results belonging to the set Y_{i+1}.

At the last, N-th step of this process, all values for the results of the scheme are determined from the given values of the arguments. If at some step of the process the algorithm F proves inapplicable, or the system of recursive equations (3) is unsolvable, or has a nonunique solution, then the process is regarded as non-executable.

Let us emphasize especially the fact that the process of determining the values of the results of a scheme begins with the introduction of a certain ordering, preserving partial order, which is rigidly determined by the scheme, on the set of components of the graph. Obviously, the values of the results do not depend on the choice of this ordering, and, consequently, this order is not essential and determining in the semantics of the scheme. This shows that the process of computing the results of a scheme decomposes into parallel branches: if Y_i and Y_j are not in the precedence relation, then the values of the results belonging to Y_i can be computed independently of the values of the results from the set Y_j, and this computation may be carried out on devices operating in parallel.

The process described above, with due account of the possibility of its parallel execution, will be called a **scheme algorithm**.

A language with grammar Γ_p, generating schemes, and with semantics determined by a scheme algorithm, will be called a **scheme language**.

Concerning schemes and scheme languages one can state the following propositions:

1. Each result M_i ($M_i \in Y_k$) of scheme (1) is completely determined by some part of the scheme, which can be obtained by the following construction: from the scheme one deletes all those rows whose external labels belong to sets Y_j such that Y_j is not connected with Y_k by the precedence relation. The scheme obtained as a result of this deletion will be called the scheme for the label M_i. If the original scheme (1) is denoted by S, then the derived scheme for the label M_i will be denoted by S_{M_i}.
2. Each scheme S_{M_i} describes a certain functional dependence between the values of the arguments of S_{M_i} and the value of the label M_i. In this sense the label M_i may be regarded as the functional sign of this function.

Thus, on the one hand, the label M_i determines a certain scheme S_{M_i}, and, on the other hand, a function whose description is S_{M_i}.

Let us now dwell on the question of the objects of scheme languages, i.e. on what values labels may take. Obviously, these values may be objects of quite arbitrary nature—numbers, strings of symbols, arrays, structures, and so on.

Up to now we have been speaking of the first level of scheme languages. Scheme languages of the next level of complexity may have as their basic objects scheme

algorithms themselves and schemes as records of scheme algorithms. In such a language a scheme will describe a scheme algorithm,

arguments and values of which will be schemes and scheme algorithms of a lower level. The unification in one language of schemes and scheme algorithms will undoubtedly lead to a complication of the above definition of scheme languages in the sense that it will be necessary to distinguish different occurrences in the line of a scheme of one and the same label, since it may denote both a scheme algorithm described by the scheme S_{M_i}, and this scheme itself—the notation of the algorithm.

Leningrad Branch
of the V. A. Steklov Mathematical Institute
Academy of Sciences of the USSR

Received
30 XII 1968

REFERENCES

1. A. P. Ershov, A. A. Lyapunov, *Kibernetika*, No. 5 (1967).
2. L. V. Kantorovich, DAN, 113, No. 4, 738 (1957).
3. V. A. Bulavskii, *Izv. vyssh. uchebn. zaved.*, Mathematics, No. 5 (1958).
4. L. T. Petrova, *Zhurn. vychislit. matem. i matem. fiz.*, 1, No. 3, 513 (1961).
5. O. K. Daugavet, E. F. Ozerova, *Zhurn. vychislit. matem. i matem. fiz.*, 1, No. 4, 747 (1961).
6. K. V. Shakhbazyan, *Tr. Matem. inst. im. V. A. Steklova AN SSSR*, 66, 45 (1962).
7. T. N. Smirnova, *Carrying Out Polynomial Expansions on an M-20 Computer by Means of Probes*, L., 1967.
8. K. V. Shakhbazyan, M. M. Lebedinskii, *Tr. Matem. inst. im. V. A. Steklova AN SSSR*, 96, 16 (1968).

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