ON THE COMPLEXITY OF ENUMERATING AND DECIDING PREDICATES

MATHEMATICS

1970

SovietRxiv

Abstract

Full Text

UDC 517.11

MATHEMATICS

M. I. KANOVICH

ON THE COMPLEXITY OF ENUMERATING AND DECIDING PREDICATES

(Presented by Academician P. S. Novikov on 26 V 1969)

The article considers certain questions connected with estimates of the complexity of algorithms performing a prescribed task for objects from a certain finite set.

We shall use the terminology and concepts introduced in works ([2–5], [8]). In particular, the length of the representation of a normal algorithm $\mathfrak{B}$ in the alphabet B will be called the **complexity** of the algorithm $\mathfrak{B}$ and denoted by the symbol $\mathfrak{B}\}$.

1. Let A be an alphabet; $\mathfrak{A}$ and $\mathfrak{B}$ normal algorithms over the alphabet A; n a natural number. By the symbol $\rho(n, \mathfrak{B}, \mathfrak{A})$ we denote the following proposition: "the algorithm $\mathfrak{B}$ is applicable to all words in the alphabet A whose lengths do not exceed the number n, and annihilates those and only those of them to which the algorithm $\mathfrak{A}$ is applicable." A general recursive function f will be called a **lower estimate of the complexity of deciding** the algorithm $\mathfrak{A}$, if for every natural number n and for every normal algorithm $\mathfrak{B}$ in the standard extension of the alphabet A, if $\rho(n, \mathfrak{B}, \mathfrak{A})$, then
$$\mathfrak{B}\} \geq f(n).$$

1.1. Let the letter a not be a letter of the alphabet A. By the symbol $\sigma_j(P)$, where j is a positive integer not exceeding the length of the word P, we shall denote the j-th letter of the word P. Let the word S be an a-system in the alphabet A; then by the symbol $\delta_0(S)$ we shall denote the number of its (a, A)-elements, and by the symbol $\delta_i(S)$, where i is a positive integer not exceeding $\delta_0(S)$, we shall denote the i-th (a, A)-element of the a-system S.

We shall say that normal algorithms $\mathfrak{C}$ and $\mathfrak{D}$ over the union of the alphabets aA and $0|$ **reduce** the normal algorithm $\mathfrak{B}$ over the alphabet A to the normal algorithm $\mathfrak{A}$ over the alphabet A, if for every word P in the alphabet A the following conditions are satisfied: 1) the algorithm $\mathfrak{C}$ is applicable to the word P, and $\mathfrak{C}(P)$ is an a-system in the alphabet A; 2) if some word R in the alphabet $0|$ is such that $[R^\circ = \delta_0(\mathfrak{C}(P))]$ and that for every positive integer j not exceeding

the length of the word R, one has

$$\sigma_j(R) \doteq | = !\mathfrak{A}(\delta_j(\mathfrak{C}(P))),$$

then the algorithm $\mathfrak{D}$ is applicable to the word PaR and

$$!\mathfrak{B}(P) \equiv \mathfrak{D}(PaR) \doteq \Lambda.$$

A normal algorithm $\mathfrak{A}$ over the alphabet A will be called **universal with respect to reducibility** if for every normal algorithm $\mathfrak{B}$ over the alphabet A one can indicate normal algorithms $\mathfrak{C}$ and $\mathfrak{D}$ over the union of the alphabets aA and $0|$ which reduce the algorithm $\mathfrak{B}$ to the algorithm $\mathfrak{A}$.

The indicated kind of reducibility was introduced in ([9]).

Theorem 1. *A normal algorithm $\mathfrak{A}$ over the alphabet A is universal with respect to reducibility if and only if there exists an unbounded general recursive function that is a lower estimate of the complexity of deciding the algorithm $\mathfrak{A}$.*

2.

Consider the problem of enumeration and decision for one-place predicates (in a suitable language) over words in the alphabet $0|$.

Let ν be a predicate, n a natural number. A Φ-algorithm $\mathfrak{B}$ will be called (ν, n)-enumerating if, for every word P of length not greater than n in the alphabet $0|$,

$$!\mathfrak{B}(P) \equiv \nu(P).$$

A Φ-algorithm $\mathfrak{C}$ will be called (ν, n)-deciding if, for every word P of length not greater than n in the alphabet $0|$,

$$!\mathfrak{C}(P) \;\&\; (\mathfrak{C}(P) \simeq \Lambda \equiv \nu(P)).$$

2.1.

The following two theorems show the close connection between the notions introduced.

Theorem 2. There exists a natural number C such that, for every Φ-algorithm $\mathfrak{B}$, one can construct two Φ-algorithms $\mathfrak{C}$ and $\mathfrak{D}$, each of complexity not exceeding $2\mathfrak{B} + C$, such that, whatever the predicate ν and the natural number n may be, if the algorithm $\mathfrak{B}$ is (ν, n)-deciding, then the algorithm $\mathfrak{C}$ is (ν, n)-enumerating, and the algorithm $\mathfrak{D}$ is $(\neg\nu, n)$-enumerating.

Theorem 3. There exists a natural number C such that, for every pair of Φ-algorithms $\mathfrak{C}$ and $\mathfrak{D}$, one can construct a Φ-algorithm $\mathfrak{B}$ of complexity not

exceeding $2(\mathfrak{C} + \mathfrak{D}) + C$ such that, whatever the predicate ν and the natural number n may be, if the algorithm $\mathfrak{C}$ is (ν, n)-enumerating and the algorithm $\mathfrak{D}$ is $(\neg\nu, n)$-enumerating, then the algorithm $\mathfrak{B}$ is (ν, n)-deciding.

2.2.

Nevertheless, estimates of the complexity of (ν, n)-enumerating and (ν, n)-deciding algorithms do not always have the same value. This is shown by the following two theorems.

Theorem 4. For every unbounded general recursive function f one can specify a predicate ν such that, for every natural number n:

1) every (ν, n)-deciding Φ-algorithm has complexity not less than the number $n/3$;
2) there exists a natural number m, not less than n, such that there quasi-exists a (ν, m)-enumerating Φ-algorithm of complexity not exceeding the number $f(m)$;
3) there exists a natural number k, not less than n, such that there quasi-exists a $(\neg\nu, k)$-enumerating Φ-algorithm of complexity not exceeding $f(k)$.

Theorem 5. One can specify a predicate ν such that:

1) for every natural number n, every (ν, n)-deciding Φ-algorithm has complexity not less than the number $n/3$;
2) there is no such unbounded general recursive function f that, for every natural number m, the complexity of every (ν, m)-enumerating Φ-algorithm is not less than $f(m)$;
3) there is no such unbounded general recursive function g that, for every natural number k, the complexity of every $(\neg\nu, k)$-enumerating Φ-algorithm is not less than $g(k)$.

2.3.

In order to formulate theorems on lower estimates of the complexity of enumeration for predicates satisfying certain conditions, we shall need the following definitions.

Let us define a method of writing Φ-algorithms in the alphabet $0|$ as follows.

$$[0^3 \rightleftarrows 000, \qquad [|^3 \rightleftarrows 00|,$$

$$[a^3 \rightleftarrows 0|0, \qquad [b^3 \rightleftarrows 0||, \qquad [c^3 \rightleftarrows |00,$$

$$[x^3 \rightleftarrows |0|, \qquad [y^3 \rightleftarrows ||0, \qquad [z^3 \rightleftarrows |||.$$

We assume

$$[\Lambda \rightleftarrows \Lambda,$$

$$[P_\xi \rightleftarrows [P[\xi$$

(P is a word in the alphabet $0|abcxyz$, ξ is a letter of the alphabet $0|abcxyz$).

Let $\mathfrak{A}$ be a Φ-algorithm. The word $[\mathfrak{A}$ will be called the **record** of the algorithm $\mathfrak{A}$ and denoted by the symbol $\mathfrak{A}$.

Let ν be a predicate, and let $\mathfrak{A}$ and $\mathfrak{B}$ be Φ-algorithms. The predicate ν will be called $(\mathfrak{A}, \mathfrak{B})$-**nontrivial** if, for every Φ-algorithm $\mathfrak{C}$ equivalent, relative to the alphabet $0|$, to the algorithm $\mathfrak{A}$, $\nu(\mathfrak{C})$ holds, while for every Φ-algorithm $\mathfrak{D}$ equivalent, relative to the alphabet $0|$, to the algorithm $\mathfrak{B}$, it is false that $\nu(\mathfrak{D})$. The predicate ν will be called **strictly** $(\mathfrak{A}, \mathfrak{B})$-**nontrivial** if it is $(\mathfrak{A}, \mathfrak{B})$-nontrivial and, for any word P in the alphabet $0|$, from $!\mathfrak{A}(P)$ it follows that $\mathfrak{B}(P) \simeq \mathfrak{A}(P)$.

Theorem 6. *There exists a natural number C such that, whatever the predicate ν and the Φ-algorithms $\mathfrak{A}$ and $\mathfrak{B}$ may be, provided that ν is strictly $(\mathfrak{A}, \mathfrak{B})$-nontrivial, for every natural number n the complexity of any (ν, n)-enumerating Φ-algorithm is not less than the number*

$$n/6 - 2(\mathfrak{A} + \mathfrak{B}) - C.$$

We give an example showing that Theorem 6 cannot be substantially strengthened.

Denote by $\nu_0(P)$ the following predicate: "there exists an empty Φ-algorithm $\mathfrak{A}$ such that $\mathfrak{A} \simeq P$."

There exists a natural number C such that, for every natural number n, there quasi-exists a (ν_0, n)-deciding Φ-algorithm of complexity not greater than $n + C$.

Let $\mathfrak{A}$ be a Φ-algorithm. By the symbol $\mathfrak{A}_m$ (where m is a natural number) we shall denote the union of the algorithm $\mathfrak{A}$ and the normal algorithm in the alphabet $0|$ defined by the scheme

$$\begin{cases} 0 & \to & |, \\ |^{m+1} & \to & |^{m+1}, \\ | & \to & \end{cases}$$

It is not difficult to see that, whatever the natural number m and the Φ-algorithm $\mathfrak{A}$ may be, for all words P in the alphabet $0|$ whose lengths do not exceed the number m, the equality

$$\mathfrak{A}_m(P) \simeq \mathfrak{A}(P),$$

holds, while to words in the alphabet $0|$ whose lengths are greater than m, the algorithm $\mathfrak{A}_m$ is inapplicable.

Let ν be a predicate, $\mathfrak{A}$ a Φ-algorithm, and k a natural number. The predicate ν will be called **strongly** $(\mathfrak{A}, k)$-**nontrivial** if:

1) the number k is the Gödel number of an unbounded general recursive function;

2) whatever the natural number m may be, the predicate ν is $(\mathfrak{A}, \mathfrak{A}_{\{k\}(m)})$-nontrivial.

Theorem 7. *There exists a natural number C such that, whatever the predicate ν, the Φ-algorithm $\mathfrak{A}$, and the natural number k may be, provided that ν is strongly $(\mathfrak{A}, k)$-nontrivial, for every natural number n the complexity of any (ν, n)-enumerating Φ-algorithm is not less than the number*

$$2^{2n/3-\mathfrak{A}-2\log_2(k+1)-C} - 1.$$

2.4. It is not difficult to see that *there exists a natural number C such that, whatever the predicate ν may be, for every natural number n there quasi-exists a (ν, n)-deciding Φ-algorithm of complexity not greater than $2^n + C$.*

Theorem 7 indicates the impossibility of a substantial improvement of this estimate.

2.5. According to Theorem 2, one can obtain theorems on lower bounds for the complexity of deciding predicates analogous to Theorems 6 and 7 (an example of a predicate "difficult" to decide is given in [6]).

3. Analogous results also hold for other criteria of algorithm complexity (for the definition of a complexity criterion, see [7]). As a consequence, one can obtain the theorem on a lower bound for the complexity of decision for nontrivial invariant properties stated in [1].

The author expresses his deep gratitude to his supervisor A. A. Markov.

Moscow State University
named after M. V. Lomonosov

Received
15 V 1969

REFERENCES

[1] Ya. M. Barzdin, DAN, **182**, No. 6, 1249 (1968).
[2] S. K. Kleene, *Introduction to Metamathematics*, Moscow, 1957.
[3] A. A. Markov, Tr. Matem. inst. im. V. A. Steklova AN SSSR, **42** (1954).
[4] A. A. Markov, Izv. AN SSSR, ser. matem., **27**, 101 (1963).
[5] A. A. Markov, ibid., **31**, 161 (1967).
[6] N. V. Petri, DAN, **185**, No. 1, 37 (1969).
[7] B. A. Trakhtenbrot, *Complexity of Algorithms and Computations*, Novosibirsk, 1967.
[8] N. A. Shanin, Tr. Matem. inst. im. V. A. Steklova AN SSSR, **52**, 226 (1958).
[9] R. Friedberg, H. Rogers jr., Zs. Math. Logik und Grundlagen d. Math., **5**, 117 (1959).

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