

ON ALGORITHMS CONNECTED WITH PREDICATES AND BOOLEAN FUNCTIONS

N. V. Petri

1969

SovietRxiv

View the original and related papers at <https://sovietsrxiv.org/items/ru-196901.15876>

Source: Math-Net.Ru and CyberLeninka. Machine translation. Verify with the original.

Abstract

Full Text

UDC 517.11

MATHEMATICS

N. V. PETRI

ON ALGORITHMS CONNECTED WITH PREDICATES AND BOOLEAN FUNCTIONS

(Presented by Academician A. N. Kolmogorov, July 5, 1968)

The article is devoted to lower and upper estimates of the complexity of normal algorithms connected with the decision of predicates and the computation of Boolean functions.

1. One of the ways of studying algorithmic problems is to establish estimates for algorithms that perform a prescribed task on finite sets. In this note some results are presented on estimates of the complexity of normal algorithms. The following problems are considered: a) recognition of the applicability of normal algorithms to words of bounded length; b) recognition of the M -computability of Boolean functions and the construction of minimizing algorithms ⁽²⁾; c) decision of predicates of the form $\forall\exists$, $\exists\forall$, and $\neg\neg\exists\forall$, given on finite sets.
 2. We use here the terminology and concepts introduced in the works ⁽¹⁻³⁾. In particular, we shall call the length of the representation of a normal algorithm $\mathfrak{A}$ the **complexity** of the algorithm $\mathfrak{A}$ and denote it by $\mathfrak{A}$; normal algorithms in the alphabet $o|abc$ will be called Φ -algorithms; by the **quasi-feasibility** of an object with given properties we shall understand the possibility of deriving a contradiction from the assumption of its nonexistence.
 3. Let B be an alphabet; $\mathfrak{A}$ and $\mathfrak{B}$ normal algorithms over the alphabet B ; n a natural number. We shall say that the algorithm $\mathfrak{B}$ is **n -deciding** for the algorithm $\mathfrak{A}$ with respect to the alphabet B , if the algorithm $\mathfrak{B}$ is applicable to all words of length not greater than n in the alphabet B and annuls those and only those of them to which the algorithm $\mathfrak{A}$ is applicable.
- 3.1. One can indicate such natural numbers C_0 and C_1 that, for any alphabets A and B (where $|B^0| \geq 2$ and $B \subset A$), any algorithm $\mathfrak{A}$ in the alphabet A , and an arbitrary natural number n , there quasi-exists a normal algorithm $\mathfrak{B}$ in a two-letter extension of the alphabet B , which is n -deciding for the algorithm $\mathfrak{A}$ with respect to the alphabet B and is such that

$$\mathfrak{B}; \geq n + \frac{\log(|A^\partial| + 2)}{\log |B^\partial|} \cdot \mathfrak{A}; + C_1 \cdot |B^\partial| + C_0.$$

It can be shown that the estimate given by this theorem* is close to optimal. At the same time, the question is of interest as to which general recursive (i.e., computable) functions can serve as estimates of the complexity of recognizing the applicability of normal algorithms to words of bounded length. We shall consider the problem of the applicability of normal algorithms over the alphabet $o|$ to words in the alphabet $o|$.

4. Let a general recursive function φ of one argument and a normal algorithm $\mathfrak{A}$ over the alphabet $o|$ be such that, for every natural number n , there quasi-exists a Φ -algorithm $\mathfrak{B}$ of complexity not greater than $\varphi(n)$, which is n -deciding for $\mathfrak{A}$ with respect to $o|$. Then we shall

* A similar result is contained, apparently, in a somewhat earlier and as yet unpublished work of Ya. M. Barzdin.

to say that the function φ is an upper estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$.

We shall call a general recursive function φ a lower estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$ if, for every natural number n and every Φ -algorithm $\mathfrak{B}$ that is n -deciding for the algorithm $\mathfrak{A}$ with respect to the alphabet $o|$, the inequality $\mathfrak{B}_3 > \varphi(n)$ holds.

4.1. Let f be a nondecreasing general recursive function such that, for every natural number n , $f(n) \leq n$. Then one can construct an algorithm $\mathfrak{A}$ over the alphabet $o|$ and indicate a natural number C such that the following assertions are true:

- a) the function f^* , defined by the equality $f^*(n) = f(n) + C$, is an upper estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$;
- b) for every function φ that is an upper estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$, there are infinitely many natural numbers for which the inequality

$$\varphi(n) > \frac{1}{3}f(n) - C$$

is violated;

- c) there is no unbounded lower estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$.

4.2. Let a pair of nondecreasing, unbounded general recursive functions f_0 and f_1 be such that, for every natural n , the inequalities $f_0(n) \leq f_1(n) \leq n$ hold. Then one can construct a normal algorithm $\mathfrak{A}$ over the alphabet $o|$ and indicate a natural number C such that the following assertions are true:

- a) the function f_1^* , defined by the equality $f_1^*(n) = f_1(n) + C$, is an upper estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$;
- b) the function f_0^* , defined by the equality $f_0^*(n) = \frac{1}{3}f_0(n) - C$, is a lower estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$;
- c) for every function φ that is an upper estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$, there are infinitely many natural numbers for which the inequality

$$\varphi(n) > \frac{1}{3}f_1(n) - C$$

is violated;

- d) for every function φ that is a lower estimate of the complexity of recognizing the applicability of the algorithm $\mathfrak{A}$, there are infinitely many numbers for which the inequality $\varphi(n) \leq f_0(n) + C$ is violated.

Theorems 4.1 and 4.2 show that, from the point of view of recognition complexity, enumerable sets are quite diverse.

5. The method of proof of Theorem 3.1 turns out to be very useful in estimating the complexity of normal algorithms connected with the computation of Boolean functions. In (2) it is proved that, for $M < \frac{1}{3}2^N$, the complexity of every Φ -algorithm recognizing M -noncomputability of Boolean functions of N arguments is greater than $\frac{5}{14}M - 433$. A substantial improvement of this estimate is impossible in view of the validity of the following theorem.

5.1. There exists a natural number C such that, for any natural numbers M and N , there exists a Φ -algorithm $\mathfrak{A}$ recognizing M -noncomputability of Boolean functions of N arguments and satisfying the condition:

$$\mathfrak{A}_3 \leq 3M + C.$$

Alongside the problem of recognizing M -noncomputability of Boolean functions of N arguments, it is natural to consider the problem of recognizing M -noncomputability of Boolean functions of no more than N arguments. Here, too, it is possible to obtain close upper and lower estimates, as is seen from the following propositions.

5.2. There exists a natural number C such that, for any natural numbers M and N , there exists a Φ -algorithm $\mathfrak{A}$ recognizing M -noncomputability of Boolean functions of no more than N arguments and satisfying the condition

$$\mathfrak{A}_3 \leq 3M + \log N + C.$$

5.3. One can specify a number C such that for any numbers M and N satisfying the condition $M < \frac{1}{3}2^N$, and for any Φ -algorithm $\mathfrak{A}$ recognizing the

M -noncomputability of Boolean functions of no more than N arguments, the inequality holds

$$\mathfrak{A} \ni > \frac{5}{28}M + \frac{1}{6} \log N - C.$$

6. The problem of finding a Φ -algorithm of least complexity computing a given Boolean function leads us to the problem of constructing minimizing algorithms. In [2] it is proved that every M -algorithm (i.e., an algorithm in the alphabet $o|abcsyuziv|$) minimizing for N arguments has complexity greater than $\frac{2}{21} \cdot 2^N - 648$. This estimate is close to optimal by virtue of the following theorem.

6.1. One can specify a natural number C such that for every natural N there quasi-exists an M -algorithm $\mathfrak{A}$, minimizing for Boolean functions of no more than N arguments, and such that $\mathfrak{A} \ni \leq 2^N + C$.

7. Let μ be a predicate (in a suitable language) on words in the alphabet $o|$, let n be a natural number, and let $\mathfrak{A}$ be an algorithm over the alphabet $o|$ such that the algorithm $\mathfrak{A}$ is applicable to every word P of length no more than n in the alphabet $o|$ and annihilates those and only those of them for which $\mu(P)$ is true. In this case we shall say that the algorithm $\mathfrak{A}$ is n -deciding for the predicate μ .

One can prove the existence of a number C such that, for any predicate μ and any natural number, there quasi-exists a Φ -algorithm $\mathfrak{A}$ that is n -deciding for the predicate μ and such that $\mathfrak{A} \ni \leq 2^{n+1} + C$. As Theorem 3.1 shows, the complexity of deciding so-called one-quantifier predicates cannot grow so rapidly with the growth of n . The following theorem shows that already in the class of two-quantifier predicates (and even in a certain narrower subclass of it) such growth of the complexity of decision with increasing n can occur.

7.1. One can specify a predicate μ on words in the alphabet $o|$, for which the following conditions hold:

- a) whatever the natural number n , the complexity of every Φ -algorithm that is n -deciding for the predicate μ is greater than $\frac{1}{3}2^n$;
- b) the predicate μ can be constructively represented in the forms $\forall x \exists y \mathfrak{R}_1(P, x, y)$ and $\neg \neg \exists x \forall y \mathfrak{R}_2(P, x, y)$, where $\mathfrak{R}_1$ and $\mathfrak{R}_2$ are decidable predicates on words P in the alphabet $o|$ and natural numbers x and y .

Let us note that in condition b) of the theorem one cannot replace the predicate $\neg \neg \exists x \forall y \mathfrak{R}_2(P, x, y)$ by the predicate $\exists x \forall y \mathfrak{R}_2(P, x, y)$, by virtue of the following theorem.

7.2. If the predicate μ is constructively representable in the forms $\forall x \exists y \mathfrak{R}_1(P, x, y)$ and $\exists x \forall y \mathfrak{R}_2(P, x, y)$, where $\mathfrak{R}_1$ and $\mathfrak{R}_2$ are decidable predicates on words P in the alphabet $o|$ and natural numbers x and y , then

one can specify a number C such that, for every natural n , there quasi-exists a Φ -algorithm $\mathfrak{A}$ that is n -deciding for the predicate μ and such that $\mathfrak{A} \ni \leq n + C$.

The comparison of Theorems 7.1 and 7.2 illustrates the difference between the classical and constructive approaches in solving the problem of the complexity of algorithms.

Moscow State University
named after M. V. Lomonosov

Received
4 VII 1968

REFERENCES CITED

1. A. A. Markov, Tr. Matem. inst. im. V. A. Steklova AN SSSR, **42** (1954).
2. A. A. Markov, Izv. AN SSSR, ser. matem., **31**, 161 (1967).
3. N. A. Shanin, Tr. Matem. inst. im. V. A. Steklova AN SSSR, **52**, 226 (1958).

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

Source: Math-Net.Ru and CyberLeninka. Machine translation. Verify with the original.