SovietRxiv

Soviet-era science, translated into English

COMPLETENESS UP TO ENCODING OF SYSTEMS OF FUNCTIONS OF \(K\)-VALUED LOGIC AND THE COMPLEXITY OF RECOGNIZING IT

1968

SovietRxiv

View the original and related papers at https://sovietrxiv.org/items/ru-196801.64001

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

SovietRxiv

Abstract

Full Text

UDC 519.95

MATHEMATICS

R. V. FREIVALD

COMPLETENESS UP TO ENCODING OF SYSTEMS OF FUNCTIONS OF K-VALUED LOGIC AND THE COMPLEXITY OF RECOGNIZING IT

(Presented by Academician A. I. Mal' tsev, 17 VIII 1967)

The present paper consists of two parts. In 1° a nonstandard approach to the concept of "completeness of systems of functions of k-valued logic" is considered, and criteria of completeness are formulated. In 2° the complexities of recognizing completeness in the usual sense [(1)] and in the sense of 1° are compared.

1°. J. von Neumann [(2)], for computing functions in a logical network [(3)], used the so-called method of paired lines. Generalizing this method, let us consider the computation of functions of k-valued logic with encoding of the arguments and of the value of the function. The formulation of the problem is due to Ya. M. Barzdin' . He also predicted the possibility of the effect discussed in the conclusion of the note.

We shall say that a logical network computes the function $f(x_1, \dots, x_n)$ of k-valued logic with encoding

$$\left\{\begin{array}{cccc} 0 \to a_{01} & a_{02} & \dots & a_{0r} \\ 1 \to a_{11} & a_{12} & \dots & a_{1r} \\ \dots & \dots & \dots & \dots \\ k-1 \to a_{k-1,1} & a_{k-1,2} & \dots & a_{k-1,r} \end{array}\right.$$

where $a_{ij} \in \{0, 1, \dots, k-1\}$, if to each argument x_i there is assigned an r-tuple of inputs a_{ij} $(j \in \{1, 2, \dots, r\})$, the network has r outputs b_l $(l \in \{1, 2, \dots, r\})$, and it operates as follows: in order to compute $f(m_1, m_2, \dots, m_n)$, to each input a_{ij} one must apply $\alpha_{m_i j}$, and then the network produces at the outputs b_l the results $\beta_l = \alpha_{f(m_1, m_2, \dots, m_n) l}$ (see Fig. 1).

Let, for a fixed natural $k \geq 2$, a system of functions of k-valued logic be given. We shall say that to the given system A of functions there corresponds a system B of functional elements if B contains all those and only those memoryless functional elements that realize functions from the system A.

SovietRxiv

Fig. 1

Figure 1: Fig. 1

We shall say that a system A of functions of k-valued logic is **complete under a fixed encoding** if every function of k-valued logic is computable with this encoding by a logical network over the system of functional elements corresponding to the system A.

We shall say that a system A of functions of k-valued logic is **complete up to encoding** (complete u.e.) if for every function f of k-valued logic there exists an encoding (depending on f) with which f is computable by a logical network over the system of functional elements corresponding to the system A. It is easy to show that if a system of functions is complete u.e., then in fact the encoding can be chosen independently of f.

An encoding S will be called **universal** for k-valued logic if, for any system of functions of k-valued logic, completeness u.e. implies completeness under the encoding S.

Theorem 1. *There exists a universal encoding for two-valued logic*

$$\begin{cases} 0 \to 0011, \\ 1 \to 0101. \end{cases}$$

For $k \geq 3$ there is no universal encoding for k-valued logic.

A class of functions of k-valued logic is called **precomplete** with respect to definability if the closure (in the usual sense (1)) of this class is not complete with respect to definability, but, on adding to the class any function of k-valued logic not belonging to this class, we obtain a class whose closure is complete with respect to definability.

A system of functions of k-valued logic is complete with respect to definability if and only if it is not contained entirely in any one of these precomplete classes.

Theorem 2. *There are exactly three precomplete classes of functions of two-valued logic: 1) linear functions, 2) conjunctions of all essential arguments, 3) disjunctions of all essential arguments.*

Fig. 1. Example of computing the function

$$x_1 \,\&\, x_2 \quad \text{with the encoding} \quad \begin{cases} 0 \to 01, \\ 1 \to 10. \end{cases}$$

The class of preservation of a 4-place predicate $R(y_1, y_2, y_3, y_4)$ in P_k (1) is the class of all functions $f(x_1, \dots, x_n)$ of k-valued logic for which, for any four tuples

SovietRxiv

of arguments

$$(c_{11}, c_{12}, \dots, c_{1n}), \quad (c_{21}, c_{22}, \dots, c_{2n}), \quad (c_{31}, c_{32}, \dots, c_{3n}), \quad (c_{41}, c_{42}, \dots, c_{4n}),$$

the following is true:

$$(R(c_{11}, c_{21}, c_{31}, c_{41}) \,\&\, R(c_{12}, c_{22}, c_{32}, c_{42}) \,\&\, \dots \,\&\, R(c_{1n}, c_{2n}, c_{3n}, c_{4n})) \to R(f(c_{11}, \dots, c_{1n}),\ f(c_{21}, \dots, c_{2n}),\ f(c_{21},$$

Let $\mathfrak{A}_1$ be the set of all classes of preservation of 4-place predicates in P_k. Let $\mathfrak{A}_2$ be the set of all classes that are the intersection of any number of classes from $\mathfrak{A}_1$. Finally, let $\mathfrak{A}$ be the set of all classes from $\mathfrak{A}_2$ that are not complete with respect to definability and are not contained entirely in another (distinct from the given class itself) class from $\mathfrak{A}_2$. It is easy to see that the number of classes in $\mathfrak{A}_1, \mathfrak{A}_2$, and $\mathfrak{A}$ is finite.

Theorem 3. *For every natural $k \geq 2$ there is only a finite number of precomplete, with respect to definability, classes of functions of k-valued logic. They are precisely all the classes of the set $\mathfrak{A}$, and no others.*

2°. Let us estimate, in terms of time signalizing functions ([4],[5]), the complexity of recognizing on Turing machines the completeness (in the usual sense (1)) of systems of functions of k-valued logic.

On the tape of a Turing machine, the functions of a system are given in tabular form, i.e. by the string of values

$$f(0, \dots, 0, 0)\, f(0, \dots, 0, 1)\ \dots\ f(0, \dots, 0, k-1)\, f(0, \dots, 1, 0)\, f(0, \dots, 1, 1)\ \dots$$

$$\dots\ f(0, \dots, 1, k-1)\, f(0, \dots, 2, 0)\ \dots\ f(k-1, \dots, k-1, k-1).$$

The function of 3-valued logic $\max(x_1, x_2)$, for example, is given by the string 012112222. Systems of functions are given in the following way: each function is given tabularly, and the records of different functions are separated by an asterisk. For example, the system of functions of k-valued logic

$$\max(x_1, x_2), \quad x_1 + 1 \pmod 3, \quad \min(x_1, x_2)$$

is given as follows:

$$012112222 * 120 * 000011012.$$

For each natural $k \geq 2$, consider the complexity of computing on Turing machines the predicate Π_k on words in the alphabet $\{0, 1, \dots, k-1, *\}$ ("the word is a record of some complete system of functions of k-valued logic"). The predicate Π_k is false on words that are not records of systems of functions. However, it can be shown that none of the predicates differing–

SovietRxiv

differing from Π_k only on words that are not records of systems of functions is not computably essentially simpler than the predicate Π_k.

By the method developed by Ya. M. Barzdin [5], one can prove lower bounds for the complexity of recognizing membership of functions in certain precomplete classes. On the basis of these estimates the following is proved.

Theorem 4. *For every $k \geq 2$ and every Turing machine computing Π_k, the time function satisfies*

$$t(l) > l^2.$$

Theorem 5. *There exist Turing machines computing Π_2 and Π_3 with time function*

$$t(l) \asymp l^2.$$

Theorem 6. *For every $k \geq 3$ there exist Turing machines computing Π_k with time function*

$$t(l) \asymp l^{k-1}.$$

Next let us estimate the complexity of computing on Turing machines the predicates Π'_k, on words in the alphabet $\{0, 1, \dots, k-1, *\}$: "the word is a record of some system of functions of k-valued logic that is complete up to admissible coding."

It is easy to see that the predicates Π'_k are not computable by finite automata; therefore, from (4) it follows that for no $k \geq 2$ is there a Turing machine computing Π'_k with time function

$$t(l) < l \log l.$$

Theorem 7. *There exists a Turing machine computing Π'_2 with time function*

$$t(l) \asymp l \log l.$$

From Theorem 3 it follows that

Theorem 8. *For every $k \geq 3$ there exist Turing machines computing Π'_k with time function*

$$t(l) \asymp l^4.$$

SovietRxiv

Conclusion. Theorems 4, 5, and 7 show that recognition of completeness up to admissible coding, for $k = 2$, is essentially simpler than recognition of completeness in the ordinary sense.

For some precomplete (in the ordinary sense) classes of functions (for example, for some T-classes [1]) no machines are known that recognize membership in them of functions of k-valued logic with time function

$$t(l) \prec l^{\log_k C_{k-1}^{[(k-1)/2]}} \asymp l^{k/\log k\, O(1)}.$$

From any Turing machine M recognizing completeness one can easily construct machines recognizing membership of a given function in any precomplete class; moreover, the resulting machines have time functions of order no greater than the time function of the machine M. Hence it follows that recognition of completeness of systems of functions is no easier than recognition of membership of functions in any precomplete class.

Therefore, comparison of the last estimate with Theorem 8 may be regarded as an indirect argument in favor of the assertion that, also for large k, recognition of completeness up to admissible coding is essentially simpler than recognition of completeness in the ordinary sense.

Institute of Mathematics
Siberian Branch of the Academy of Sciences of the USSR

Received
4 VII 1967

REFERENCES

1. S. V. Yablonskii, *Tr. Matem. inst. im. V. A. Steklova AN SSSR*, **51** (1958).

2. J. Neumann, *Collection: Automata*, Moscow, 1956.

3. N. E. Kobrinskii, B. A. Trakhtenbrot, *Introduction to the Theory of Finite Automata*, Moscow, 1962.

4. B. A. Trakhtenbrot, *Algebra and Logic*, Seminar, **3**, no. 4, 1964.

5. Ya. M. Barzdin, *Problems of Cybernetics*, issue 15, 1966.

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

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