FUNCTIONAL EQUIVALENCE OF AUTOMATA WITH A FINAL STATE

1969

SovietRxiv

Abstract

Full Text

UDC 519.95

CYBERNETICS AND CONTROL THEORY

A. A. LETICHEVSKII

FUNCTIONAL EQUIVALENCE OF AUTOMATA WITH A FINAL STATE

(Presented by Academician V. M. Glushkov, 3 VII 1968)

The scheme introduced by V. M. Glushkov for the interaction of two automata —control and operational automata ([1])—defines a new approach to the study of the fundamental problems of the applied theory of algorithms, which includes questions of designing the structures of computing machines and a substantial part of programming theory. One of the important directions in this field consists in the study of various kinds of equivalence of control automata; some results were obtained in the author' s works ([2], [3]). Continuing these investigations, we define here one of the strong forms of equivalence of control automata —functional equivalence. This concept is introduced on the basis of the syntactic structure of the language in which the operators implemented by the output signals of the control automaton, and the conditions determining its transitions, are specified. The problem of functional equivalence turns out to be algorithmically undecidable. The author repeatedly discussed this problem with D. A. Zaslavskii, who, in a somewhat different form, posed it for flowcharts with memory ([4]). The algorithmic undecidability of an analogous problem was proved in ([5]) for the calculus of flowcharts. However, the result of ([5]) cannot be applied directly to the case considered here, since the concept of a flowchart used in ([5]) is not expressed in terms of automata with a final state.

Let R be a set of variables, Ω a set of functional symbols, each of which has a definite (finite) arity, and let T be the language of terms constructed from these symbols, i.e., T is the smallest set of expressions containing R and, for every n-ary functional symbol ω, containing together with the expressions $t_1, \dots, t_n$ also the expression $\omega(t_1, \dots, t_n)$. An expression $r := t$, where $r \in R$, $t \in T$, will be called an operator. The set of all operators will be denoted by Y.

Let P be a set of predicate symbols, each of which has a definite (finite) arity. An expression $u = p(t_1, \dots, t_n)$, where p is an n-ary predicate symbol and $t_1, \dots, t_n$ are terms, will be called an elementary condition, and the set of all elementary conditions will be denoted by U. Let X be the set of all mappings of the set U into the set $\{0, 1\}$. We shall consider X-Y-automata A with a final state ([2]), whose transition functions have the following property: for any two states $a, a' \in A$ there exists a Boolean function $f(z_1, \dots, z_n)$ and a set of elementary conditions $u_1, \dots, u_n$ such that, for any $x \in X$, $ax = x'$ if and only if

$$f(x(u_1), \dots, x(u_n)) = 1.$$

An automaton A will be called finite if its set of states is finite.

Although the input and output alphabets X and Y are infinite, every finite automaton uses only a finite number of symbols of the alphabet Y, and its transition function depends only on the values of the input signal x on a finite set of elementary conditions. This makes it possible in each particular case to narrow the input and output alphabets to finite ones,

leaving only those symbols of the alphabet Y which are values of the output functions, and identifying symbols of the alphabet X which differ in their values only on those elementary conditions on which the transition function does not depend.

Fix some algebraic system M with signature of operations Ω and signature of predicates P. In this system, each term $t(r_1, \dots, r_n)$ will be interpreted as a function with values in M of variables $r_1, \dots, r_n$ taking values in M, and an elementary condition $p(t_1, \dots, t_n)$ is interpreted as a predicate function of the variables occurring in the terms $t_1, \dots, t_n$ and taking values in M. If some distribution of values of the variables from R is fixed, i.e., a function $b : R \to M$, then each term t receives the value $b(t)$ in M, and each elementary condition u receives the value $b(u)$ in the set $\{0, 1\}$. The set B_M of all functions $b : R \to M$ can be regarded as an operational $Y - X$-automaton if the transition function is defined by the relation $by = b'$, where for $y = (r := t)$, $b'(s) = b(s)$ if $s \neq r$ and $b'(r) = b(t)$, and the output function μ is defined by the condition $\mu(b)x$, where $x(u) = b(u)$.

Every controlling $X - Y$-automaton A, for fixed M, defines a certain partial transformation f_A of the set B_M of states of the operational automaton, and two automata A_1 and A_2 are equivalent with respect to M if $f_{A_1} = f_{A_2}$. The $X - Y$-automata A_1 and A_2 are called functionally equivalent if A_1 and A_2 are equivalent with respect to any algebraic system M. In the terminology of [2], functional equivalence is equivalence with respect to the class $\mathfrak{B}$ of all initial subautomata $B_M(b)$ of all operational automata B_M.

Theorem 1. *If the set R is countable, Ω contains at least one unary functional symbol, and P contains at least one unary predicate symbol, then the problem of functional equivalence of finite $X - Y$-automata with a final state is algorithmically undecidable.*

The set of terms T may be regarded as the free universal algebra with operations Ω, and the set G of all mappings from R into T, which leave in place all but a finite number of elements of the set R, as a Y-automaton, assuming that $gy = g'$, where for $y = (r := t)$, $g'(s) = g(s)$ if $s \neq r$, and $g'(r) = g(t)$. As the initial state g_0 of the automaton G we choose the identity mapping $g_0(r) = r$. Let L be the set of all mappings $\mu : G \to X$ such that, if $u =$

$p(t_1, \dots, t_n)$, $g(t_i) = g'(t_i)$ $(i = 1, \dots, n)$, $\mu(g) = x$ and $\mu(g') = x'$, then $x(u) = x'(u)$. Every initial subautomaton $B_M(b)$ of any operational automaton B_M is a homomorphic image of some automaton G_μ, where $\mu \in L$ (G_μ is the $Y-X$-automaton G with output function μ); therefore the automata A_1 and A_2 are functionally equivalent if and only if $A_1 \sim A_2(G, L)$.

The further proof of Theorem 1 is based on a special method of representing an arbitrary partially recursive function by means of mappings induced by controlling automata.

Let ω be a unary functional symbol, and p a unary predicate symbol. We shall say that the variable r represents the number n with respect to the pair (g, μ) $(g \in G,\ \mu \in L)$ if the mapping $x = \mu(g)$ is such that, for every condition u of the form $u = p(\omega^k(g(r)))$ $(k = 0, 1, \dots)$, $x(u) = 1$ if and only if $k = m(2n+m+1)/2$ $(m = 0, 1, \dots)$.

Order all variables in the sequence $r_0, r_1, \dots$. We shall say that the pair (g, μ) $(g \in G,\ \mu \in L)$ represents the tuple of nonnegative integers $z_1, \dots, z_n$, if the variables $r_1, \dots, r_n$ represent the numbers $z_1, \dots, z_n$, respectively, and the variable r_0 represents the number 0 with respect to the pair (g, μ).

Each X-Y automaton A induces a partial mapping $u_A : L \to G$ [2], and we shall say that automaton A represents a partial recursive function $f(z_1, \dots, z_n)$ if, whenever the pair (g_0, μ) represents a set of numbers $z_1, \dots, z_n$, A is applicable to μ if and only if $f(z_1, \dots, z_n)$ is defined, and, if $f(z_1, \dots, z_n)$ is defined, then the pair $(u_A(\mu), \mu)$ represents the number $f(z_1, \dots, z_n)$.

Lemma 1. *For every partial recursive function $f(z_1, \dots, z_n)$ there exists a finite X-Y automaton A that represents this function.*

If automata A_1 and A_2 represent one and the same partial recursive function, they still need not be functionally equivalent. However, the following proposition holds.

Lemma 2. *There exists a constructive procedure which, for any pair of finite X-Y automata A_1 and A_2 and for each number $n \geq 0$, makes it possible to construct finite automata A_1 and A_2 possessing the following property: if A_1 and A_2 represent the partial recursive functions $f_1(z_1, \dots, z_n)$ and $f_2(z_1, \dots, z_n)$, respectively, then the automata A_1 and A_2 are functionally equivalent if and only if the functions $f_1(z_1, \dots, z_n)$ and $f_2(z_1, \dots, z_n)$ coincide on the intersection of their domains of definition.*

From Lemma 2 the undecidability of the problem of functional equivalence follows immediately.

Institute of Cybernetics
Academy of Sciences of the Ukrainian SSR

Received
3 VII 1968

REFERENCES

1. V. M. Glushkov, *Kibernetika*, No. 1 (1965).

2. A. A. Letichevsky, *Kibernetika*, No. 4 (1965).

3. A. A. Letichevsky, *Kibernetika*, No. 1 (1967).

4. D. A. Zaslavsky, *Tr. Matem. inst. im. V. A. Steklova AN SSSR*, 72 (1964).

5. H. Thiele, *Wissenschaftstheoretische Untersuchungen in algorithmischen Sprachen*, Berlin, 1966.

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