7

Representation of Events in Finite Automata and Sequential Machines

7.1. STATEMENT OF THE PROBLEM

Chapter 3 introduced the concept of tapes of a finite automaton (ρ, $\varkappa$ tape, Table 7.1) and of a sequential machine (ρ, $\varkappa$, λ tape, Table 7.2). Thus, we know that a tape represents the operation of a finite automaton (or an s-machine) when the input sequence of ρ's and the initial state $\varkappa^0$ are given. The sequences of ρ from 0 time to time p is finite; but since there is no limitation on the operating time of the automaton, the length of each sequence, even though finite, may be as long as desired, and the number of possible sequences of ρ is

Table 7.1

Discrete moment	0	1	2	3	4	5	...	p	...
ρ							...		...
$\varkappa$									

Table 7.2

Discrete moment	0	1	2	3	4	5	...	p	...
ρ							...		...
$\varkappa$							...		...
λ									

infinite. The automaton (or s-machine), which starts from an initial state $\varkappa^0$, establishes a correspondence between each sequence of ρ and some sequence of $\varkappa$ (or λ, in the case of an s-machine); that is, it transforms a sequence of symbols of one alphabet into a sequence of symbols from another alphabet.

What then are the general rules governing this transformation?

Let K be the set of all possible sequences of $\varkappa$, and E be the set of all possible sequences of ρ. The two sets are equipollent. This means that each sequence from K can be placed in correspondence with a sequence from E. Now, if such a correspondence is established in some *arbitrary* way, is it possible to devise a finite automaton embodying this correspondence? Alternatively, is it possible to indicate those correspondences between sequences that can be embodied in a finite automaton, and those that cannot? If there are correspondences that cannot be embodied in any finite automaton, can these be separated from those that can? Identical problems arise with the sequential machines.

These problems can also be formulated in other terms. Assume a finite automaton with a fixed initial state $\varkappa^0$, and consider some state $\varkappa^*$. In examining the ρ, $\varkappa$ tape of the automaton, we mark all instances where $\varkappa^*$ appears. Then we write out all the sequences of ρ (beginning with discrete time 0) that lead to the generation of $\varkappa^*$. Assume that we could analogously process all the conceivable $\rho, \varkappa$ tapes of the same automaton. Assume also that in a set E of all conceivable input sequences of some automaton, we can distinguish a subset G^* of all input sequences that lead to the generation of $\varkappa^*$ in our first automaton. We shall then say that the automaton with initial state $\varkappa^0$ *represents* the input sequences of subset G^* by producing the symbol $\varkappa^*$ at the output. Similarly, an s-machine *represents* input sequences of a subset G^* by generating the symbol λ^* at the output. Our problem then is: Can any subset of input sequences be represented in an automaton or s-machine? If not, what are the conditions for representability of a set of input sequences? What are the properties of representable sets?

To answer these questions we shall first have to formulate the problem more precisely. Therefore, we shall introduce the term "event," and define the classification of events.

7.2. EVENTS. REPRESENTATION OF EVENTS

Let us examine the top strip, that is, the ρ sequence of the tape of an automaton (or a sequential machine). Let us call this strip the *input tape* of the machine (example: Table 7.3).

Table 7.3

Discrete moment	0	1	2	3	4	5	6	7	...
ρ	ρ_3	ρ_1	ρ_5	ρ_7	ρ_1	ρ_{12}	ρ_1	ρ_3	...

Let G be the set of all conceivable input tapes of a given automaton. Further, let us agree that there is some criterion for distinguishing subset G^* from the set G. Whatever this criterion, we shall say that *event G^* occurs* whenever (that is, at all the p's such that) the input tape of the automaton from time 0 to time p belongs to subset G^*.

With time, that is, as p increases, the tape may cease to belong to subset G^*; that is, an unfolding input tape may belong to G^* at some values of p and not belong to G^* at other p's.

Example 1. The event occurs if ρ_5 and ρ_3 are consecutive inputs at sampling instant p and the preceding instant $p-1$. For example, the event occurs at time p if the input tapes are

1

0	1	$p=2$
ρ_3	ρ_5	ρ_3

2

0	1	2	3	4	$p=5$
ρ_2	ρ_1	ρ_7	ρ_4	ρ_5	ρ_3

3

0	1	2	3	4	5	6	7	8	$p=9$
ρ_4	ρ_1	ρ_5	ρ_3	ρ_7	ρ_5	ρ_3	ρ_2	ρ_5	ρ_3

4

0	1	2	3	4	5	6	7	8	9	$p=10$
ρ_3	ρ_5	ρ_3	ρ_5	ρ_3	ρ_5	ρ_3	ρ_5	ρ_3	ρ_5	ρ_3

and it does not occur with, for example, the input tapes

5

0	1	$p=2$
ρ_5	ρ_3	ρ_5

6

0	1	2	3	4	$p=5$
ρ_2	ρ_1	ρ_7	ρ_4	ρ_3	ρ_5

7

0	1	2	3	4	5	6	7	8	$p=9$
ρ_4	ρ_1	ρ_5	ρ_3	ρ_7	ρ_5	ρ_3	ρ_2	ρ_4	ρ_1

8

0	1	2	3	4	5	6	7	8	9	$p=10$
ρ_3	ρ_5	ρ_3	ρ_5	ρ_3	ρ_5	ρ_3	ρ_5	ρ_3	ρ_3	ρ_5

If the tape is

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	...
ρ_3	ρ_1	ρ_4	ρ_8	ρ_6	ρ_5	ρ_3	ρ_7	ρ_9	ρ_1	ρ_5	ρ_3	ρ_5	ρ_3	ρ_5	ρ_2	ρ_4	ρ_3	ρ_5	ρ_5	ρ_3	...

then the event occurs at $p = 6, 11, 13$, and 20, and it does not occur at all other p's.

Example 2. The event occurs if prior to the sampling instant p there is at least one ρ_5, ρ_3 input sequence. This condition is met by all the above tapes except 6; therefore, after some initial time, all these tapes belong to subject G^*. Thus tape 2 belongs to G^* at $p \geqslant 5$, tape 3 at $p \geqslant 3$, tape 4 at $p \geqslant 4$, and so on.

In other typical formulations, the event occurs if:

Example 3. The input tape contains the sequence $\rho_2 \rho_0 \rho_1$ between p-5 and p.

Example 4. At least one of the sequences $\rho_1\rho_2\rho_6$, $\rho_2 \rho_4 \rho_5$, or $\rho_3 \rho_2 \rho_1$ appears on the tape between p-2 and p.

Example 5. There is no input ρ_3 between p-2 and p.

Example 6. No input ρ_1 is encountered between p-4 and p unless it follows ρ_3.

Example 7. Input ρ_5 appears at least once prior to p.

Example 8. Input ρ_7 is followed by ρ_3 at least once prior to p.

Example 9. There is no ρ_7 prior to p.

Example 10. Prior to p, there is no input sequence of three symbols ρ with odd subscripts after a ρ with an even subscript.

Example 11. At $p = 2$, the input is ρ_7.

Example 12. At $p = 4$, the input is ρ_1, ρ_6, or ρ_2.

Example 13. At $p = 2, 6$, and 8 the input is a ρ with an odd subscript.

Example 14. The input value is divisible by three (e.g., at $p = g$).

Example 15. $p = k$.

Note: In this case, the set G^* is equal to the set of all tapes of length k.

Example 16. A symbol ρ with an even subscript is the input at all p's whose value is a square of an integer (e.g., 1, 4, 9, 16, 25, ...).

Example 17. Either ρ_5 or ρ_9 appears at the input whenever the p is a prime number (e.g., 2, 3, 5, 7, 11, and so on).

In considering the occurrence of events we assumed that there is someone examining the input tape and deciding whether the sequence of inputs ρ appearing on the tape at time p belongs to the given set G^* or not. Now, couldn't this very task be performed by an automaton or an s-machine? To formulate this question precisely, we shall introduce the concept of representation of events by an automaton and an s-machine.

Consider the set K of all possible states $\varkappa_1, \varkappa_2, \ldots, \varkappa_k$ of a finite automaton (which at $t = 0$ is in an initial state $\varkappa^0$), and select a nonempty subset M (in particular, M may contain only one state). We shall say that *this automaton represents a given event G^* by states from set M if, and only if, at $t = p + 1$ it is in a state of set M solely because of occurrence of the input event G^* at $t = p$.*

If the automaton is associated with an output converter, the latter can always be so designed that it generates an output of 1 when $\varkappa$ belongs to M, and a 0 for other $\varkappa$. In this case the automaton represents an event if, and only if, it generates an output or 1 at $t = p + 1$ solely as a result of an event occurring at $t = p$.

Similarly, a sequential machine represents an event by generating an output symbol 1 at $t = p + 1$ if, and only if, the event occurs at the input at $t = p$.

We shall say that an event G^* is representable in a finite automaton if there exists a finite automaton A that represents the event G^*.

There is an important consequence of the general theorem proven in Section 4.3: from the representability (or nonrepresentability) of an event by an automaton follows its representability (or nonrepresentability) by a sequential machine. The converse statement is also true. For this reason, in discussing representability of events, we need only to consider the case of finite automata.

7.3. OPERATIONS ON SETS OF INPUT SEQUENCES

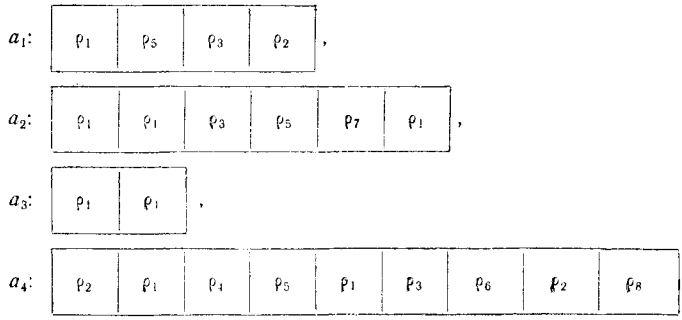
Regular Events

So far, when speaking of an input tape we meant the top strip of the tape of the automaton, that is, the sequence of input symbols

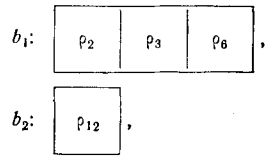
associated with a discrete time sequence and starting from $t = 0$. However, in the first part of this section we shall deal merely with sequences of input symbols ρ without in any way associating the beginning and the end of any such sequence with a specific time. We shall denote specific, finite input sequences by a, b, c, and so on or a_1, a_2, a_3, and so on.

Let A and B be sets (finite or infinite) of input sequences, composed of elements $a_1, a_2, \ldots$ and $b_1, b_2, \ldots$ respectively. We shall form new sets from sets A and B by means of three operations.

First operation: *union of sets.* The set C, containing all sequences of set A and of set B, will be called the *union* (or sum) of A and B and will be denoted by $C = A \vee B$. Thus, for example, if the set A consists of the four sequences



and the set B of the two sequences



then the set C will consist of all the above six sequences; that is,

$$c_1 = a_1, \quad c_2 = a_2, \quad c_3 = a_3, \quad c_4 = a_4, \quad c_5 = b_1, \quad c_6 = b_2.$$

Second operation: *multiplication.* From the sequences contained in sets A and B, we form a new set of sequences via the following rule: we add to the right-hand side of any sequence of set A, for example, a_i, any sequence of set B, for example, b_j, thus forming a new sequence $c_k = a_i b_j$. Let us form all possible sequences of this type, in turn adding to each sequence of A one of the possible sequences

of B. The new set of sequences C is

$$c_k = a_i b_j.$$

It shall be called the *product* of the sets A and B, and the operation yielding C is the *multiplication* of A by B. It is written as

$$C = A \cdot B.$$

If, for example, set A consists of the four and set B of the two sequences of the preceding example, then set $C = A \cdot B$ consists of the following eight sequences:

$c_1 = a_1 b_1 =$ | p_1 | p_5 | p_3 | p_2 | p_2 | p_3 | p_6 |,

$c_2 = a_1 b_2 =$ | p_1 | p_5 | p_3 | p_2 | p_{12} |,

$c_3 = a_2 b_1 =$ | p_1 | p_1 | p_3 | p_5 | p_7 | p_1 | p_2 | p_3 | p_6 |,

$c_4 = a_2 b_2 =$ | p_1 | p_1 | p_3 | p_5 | p_7 | p_1 | p_{12} |,

$c_5 = a_3 b_1 =$ | p_1 | p_1 | p_2 | p_3 | p_6 |,

$c_6 = a_3 b_2 =$ | p_1 | p_1 | p_{12} |,

$c_7 = a_4 b_1 =$ | p_2 | p_1 | p_4 | p_5 | p_1 | p_3 | p_5 | p_2 | p_8 | p_2 | p_3 | p_6 |,

$c_8 = a_4 b_2 =$ | p_2 | p_1 | p_4 | p_5 | p_1 | p_3 | p_6 | p_2 | p_8 | p_{12} |.

Third operation: *iteration*. The first two operations were binary; that is, a new set C was formed from two sets A and B. The third operation is unary; that is, it forms a new set C from some single set.

Consider, for example, set B, and select a sequence from it. Then add to its right side any sequence from the same set B (it may also be the first sequence itself). Then add to this new sequence another sequence from set B, and so on, any (finite) number of times.

This process of adding sequences from set B one after the other may be interrupted at any point, in particular after the first step, that is, after selection of the first sequence from B. At any time during this process one can select any sequence from B, including any of those already utilized before.

In forming all possible new sequences from sequences belonging to B, that is, in running through all the possibilities of attaching one sequence of B to another sequence from B, and interrupting this process at every conceivable finite step, we form a new set of sequences C. If we now add to set C an "empty"* sequence Λ (containing no symbols), we shall obtain set C', which is known as the *iteration* of B, and is written as

$$C' = B^*.$$

Even if set B is finite, set $C' = B^*$ is infinite. Thus in our example set B consisted of two elements

b_1: | p_2 | p_3 | p_6 |,

b_2: | p_{12} |.

The elements of the infinite set $C' = B^*$ are, for example, the sequences

$c_0 = \Lambda$ ("empty" sequence),

$c_1 = b_1$: | p_2 | p_3 | p_6 |,

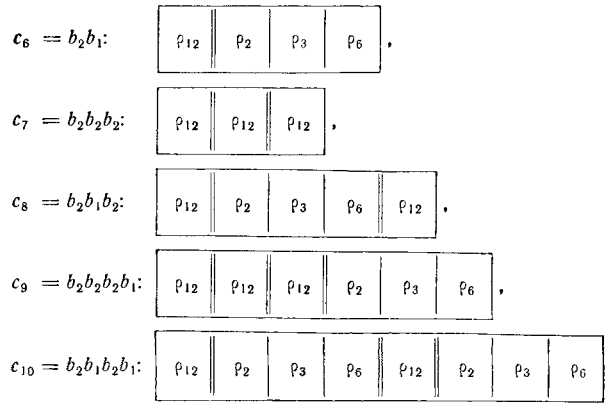
$c_2 = b_2$: | p_{12} |,

$c_3 = b_1b_1$: | p_2 | p_3 | p_6 | p_2 | p_3 | p_6 |,

$c_4 = b_1b_1b_1$: | p_2 | p_3 | p_6 | p_2 | p_3 | p_6 | p_2 | p_3 | p_6 |,

$c_5 = b_1b_2$: | p_2 | p_3 | p_6 | p_{12} |,

*Multiplication of any sequence a by an "empty" sequence Λ yields a: $a\Lambda = \Lambda a = a$. Introduction of "empty" sequences is convenient for writing of regular expressions (see below, Example 2).



and so on.

We now see the relationship between multiplication and iteration: iteration is the result of union of all the sets obtained by multiplying set B by itself some (finite) number of times. Accordingly, iteration may be represented as an "infinite series"*

$$B^* = \Lambda \vee B \vee (B \cdot B) \vee ((B \cdot B) \cdot B) \vee (((B \cdot B) \cdot B) \cdot B) \vee \ldots$$

The new sets generated from A and B can also be treated as initial sets: they can be operated upon to form new sets, and so on. Even if the initial sets are finite (and even if each contains only one sequence consisting of one symbol), one iteration operation will produce an infinite new set.

Let us introduce a *universal set* E, containing some input sequence of some (finite) length.

If the set $\bar{A}$ is a set of sequences containing only one symbol ρ, that is, if it is assumed that $\bar{a}_1 = \rho_1$, $\bar{a}_2 = \rho_2$, ..., $\bar{a}_r = \rho_r$, then the universal set E can be obtained from the elements of $\bar{A}$ by adding these one after another in any desired number and in any desired order, that is,

$$E = \bar{A}^*.$$

*These three operations just discussed were first introduced by Kleene [42]. Our exposition differs from that of Kleene in two respects: 1) Instead of the unary operation of iteration, Kleene uses the binary operation $C = A * B$, which may be expressed by means of our three operations: $A * B = A \vee A \cdot B^*$; 2) In contrast to Kleene, we add sequences during multiplication and iteration on the right- and not the left-hand side.

In the following we shall call *basic** a) any sets consisting of one input sequence of finite length (a, b, c, ...) and b) the universal set E.

We shall call a *regular set* of input sequences:

1) any basic set;

2) any set of sequences that may be formed from the basic ones by using union of sets, multiplication, or iteration over a finite number of times.

Regular sets, of which a few examples follow, will be denoted by R.

Example 1.

$$R = [(a \vee b) \vee c].$$

In this case the regular set is the union of the three basic sequences.

Example 2.

$$R = [b \cdot (a)^*].$$

The regular set is the set of all sequences starting with b, followed only by element (sequence) a, which may be repeated any number of times. For example:

$$b,\ ba,\ baa,\ baaa, \ldots\ .$$

Note that because the definition of $(a)^*$ includes empty sequence Λ, set R also contains symbol b by itself.

Example 3.

$$R = \{[(a \vee b)^*]\, c\}.$$

This set contains all sequences consisting of sequences a and b repeated any desired number of times and in any order, and terminating in sequence c. For example:

$$c,\ abc,\ bac,\ aabc,\ baabc,\ baaaabbc\ ,$$

and so on.

*It should be pointed out that it would be logical to treat as basic sequences not those of finite length but sequences of length 1: $a_1 = p_1$, $a_2 = p_2$, ..., containing only one symbol. Indeed, we can obtain finite sequences from these merely by multiplication. However, in proving Kleene's theorems, we shall find that basic sequences of finite length are more convenient. Later, in Chapter 8, we shall give another proof of Kleene's theorems, and there we shall use basic sequences containing only one symbol.

Example 4.

$$R = (E \cdot a).$$

Here R is the set of all sequences terminating in sequence a.

Example 5.

$$R = |(E \cdot a) \cdot b|.$$

This set contains all sequences terminating in a sequence of sequences a and b.

Example 6.

$$R = \{|(a \cdot E) \cdot (c \cdot E)| b\}.$$

In this case R contains all sequences starting with a, terminating in sequence b, and containing sequence c somewhere in between.

Expressions such as in the examples, that is, formed from the basic sequences (a, b, c, ... and E) connected by the signs for union of sets, multiplication, and iteration ($\vee$, $\cdot$, and *), shall be called *regular expressions*.

In regular expressions, each sign for an operation may be used only with a pair (in binary operations) or a single basic element (in iteration), or with parentheses (brackets) containing the result of such an operation. For this reason, a regular expression may contain "parentheses within brackets" (see examples above).

The highest number of "parentheses (brackets) within brackets" in a regular expression (counting the external brackets) is known as the depth of the regular expression.* In the above examples the depth is equal to 1 only in Example 4; it is 2 in Examples 1, 2, and 5, and 3 in Examples 3 and 6.

We shall say that the depth of a regular expression is zero if it contains no operations, for example, $R = a$, $R = b$, $R = E$, and so on.

Now, the same regular set of input sequences may be represented by several different regular expressions. For example, expressions

$$R = [(a \cdot b) \cdot (c \cdot d)] \text{ and } R = \{[(a \cdot b) \cdot c] \cdot d\},$$

certainly describe the same set, but they are of different depth. For

*The operations $\vee$, $\cdot$, and * could be used as the basis for algebraic operations on sets of input sequences. In particular, we could obtain identities, for example: $(E \cdot B)^* = EB$ (where B is an arbitrarily given set), which would enable us to simplify regular expressions. However, we shall not need such an algebra in this book.

this reason depth relates to a specific regular expression rather than to a regular set.

Let us also point out that a subset of a regular set of input sequences may not necessarily be regular. This follows from the simple fact that a universal set is regular by definition, whereas irregular sets do exist (for an example of an irregular set, see Section 7.6).

The reader will recall that the sequences treated so far in this section did not carry a number identifying the discrete time at which they appeared at the input.

Let us now consider a set S of such sequences and identify each element of this set with a number describing t. We start with $t = 0$. Then we obtain a set of input tapes from a set of input sequences. Thus, for example, a set consisting of the three input sequences

p_1	p_5	p_{12}	p_3	p_1	p_4	p_6	,

p_2	p_5	p_3	,

p_2	p_1	p_2	p_5	p_6	p_1	p_2	p_8	p_1	,

becomes a set consisting of the three input tapes

Discrete moment	0	1	2	3	4	5	6
p	p_1	p_5	p_{12}	p_3	p_1	p_4	p_6

Discrete moment	0	1	2
p	p_2	p_5	p_3

Discrete moment	0	1	2	3	4	5	6	7	8
p	p_2	p_1	p_2	p_5	p_6	p_1	p_2	p_8	p_1

.

A set of input tapes formed by such a method from a regular set of input sequences will be called a *regular set of input tapes*. We shall associate with each regular set of input tapes the same regular

expression that describes the corresponding regular set of input sequences.

We can now proceed to the classification of events at the input of the automaton.

An input event of the automaton is said to be regular if the set of input tapes (each examined completely from $t = 0$ *to* $t = p$*) defining the occurence of the event is a regular set.*

Among regular events we may distinguish a subclass of events that are described by a regular expression

$$R = E \cdot A,$$

where A is any set of finite input sequences containing not more than q symbols. Such a regular event is known as a specific event *of length* q or simply a *specific event*. With a specific event, one need not examine the entire input tape from $t = 0$ to $t = p$ in order to specify it: one merely looks over a length q of the "tail end" of the tape, corresponding to $t = p, p-1, p-2, \ldots, p-q$. One can visualize this process as one in which the input tape lies under a transparent runner (similar to hairline-carrying runner of a slide rule) through which one can observe only some number q of tape positions. With each discrete instant the runner is displaced one position to the right, so that the extreme right position seen through the runner always corresponds to $t = p$.

Specific events are distinguished by the fact that we can so select the runner (a specific one for each specific event) that at any time p the occurrence or nonoccurrence of an event is indicated simply by those positions on the input tape that can be seen through that runner.

7.4. REPRESENTABILITY OF REGULAR EVENTS

We can now formulate and prove the following fundamental theorems.

Kleene's first theorem. Assuming a suitable initial state of the automaton, any regular event can be represented in a finite automaton equipped with an output converter by generation of 1 at the output of the converter.

We shall prove this theorem by showing one of the possible methods for synthesizing an automaton representing a regular event defined by an arbitrarily chosen regular expression.*

*Another proof of this theorem is given in Chapter 8 in connection with the description of Glushkov's method.

We shall now introduce *auxiliary automata* with output converters (the converter output may only be 0 or 1) having, in addition to input p, auxiliary inputs for symbols α from the alphabet {0, 1}.*

Assume that a regular set of input sequences R is given. We shall say that an *event* R_α occurs at the input of the auxiliary automaton at time p if there exists a time t $(0 \leqslant t \leqslant p)$ such that:

1) the symbol $\alpha = 1$ appears at the auxiliary input at t; and

2) the sequence of symbols ρ that appear between times t and p belongs to R.

For example, let the given set R include the three sequences

$$\rho_1\rho_2\rho_3,\ \rho_3\rho_2\rho \text{ and } \rho_5.$$

Then, given the input tape of our auxiliary automaton (Table 7.4), the event R_α will only occur at times 3, 5, 7, 14, and 18.

We shall say that the generation of symbol 1 at the output of the converter for the auxiliary automaton represents the event R_α if, and only if, the event R_α occurs at the input of this automaton.

Table 7.4

Discrete moment	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
ρ	ρ_1	ρ_3	ρ_2	ρ_1	ρ_2	ρ_3	ρ_5	ρ_5	ρ_3	ρ_4	ρ_5	ρ_2	ρ_1	ρ_2	ρ_3	ρ_5	ρ_3	ρ_2	ρ_1
Auxiliary input	0	1	1	1	0	1	0	1	1	0	0	0	1	1	1	0	1	0	1

Now let us imagine that there is an autonomous automaton with an output alphabet {0, 1} and that the output of this automaton coincides with the wire α of the auxiliary automaton (Fig. 7.1). The whole network so obtained constitutes an automaton with a single input ρ. Further, let the autonomous automaton generate an output of $\alpha = 1$ at time zero, and $\alpha = 0$ at all subsequent times. Then our network will represent the event R by generating an output of 1.

Such an autonomous automaton can, indeed, be synthesized: this may be, for example, a binary delay element whose input is always zero, while its initial state is 1. Therefore, if we can show that an auxiliary automaton representing the event R_α can be synthesized

*In other words, the input symbols of such an automaton are symbol pairs $(\rho, 1)$ or $(\rho, 0)$.

regardless of the type of regular event R, then we shall have proven the existence of an automaton representing any regular event R. Our proof will be inductive and be based on the depth of the regular expression defining the regular event R_α being represented. We shall show first a way of representing all regular events R_α with a zero depth of the defining regular expression R. Then we shall prove that if a regular event R_α defined by a regular expression R of arbitrary depth ν can be represented, then any regular event R_α defined by a regular expression R of depth $\nu + 1$ can also be represented.

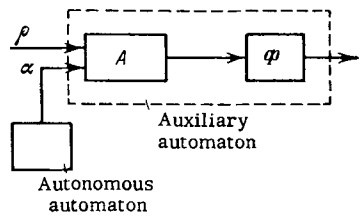


Fig. 7.1.

First inductive step. Recall that regular expressions of depth zero are simply symbols corresponding to the given input sequences a, b, c, ..., as well as the symbol E, corresponding to the universal set of input sequences.

Representation of the event R_α denoted by symbol E means devising an automaton with an output of 1 maintained from the first instance when $\alpha = 1$ regardless of what the input sequence may be. Such an automaton may be synthesized from, among others, a binary delay unit that is in its zero state at $t = 0$. The input to this unit, via a disjunction element, is a symbol α and an output symbol* (Fig. 7.2.).

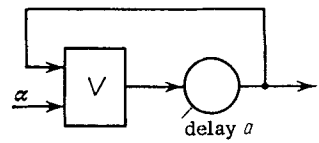


Fig. 7.2.

Let us now construct an automaton representing the event R_α when the set R contains only one input sequence of finite length q. This machine consists of two delay lines (each with q delay units) and a symbol converter (Fig. 7.3). The first (main) delay line operates in alphabet $\{\rho\}$, and the second (auxiliary) in binary alphabet $\{\alpha\}$. The converter output is 1 if, and only if, the symbols ρ at the outputs of all the delay units of the main line form one given sequence of length q, counting from the end of the delay line toward its beginning. The initial state of the delay units in the main line is immaterial, but all such units in the auxiliary line must be in state zero. The output of the automaton is a conjunction of the outputs of the converter and the auxiliary delay line.

The output will be 1 if, and only if: 1) at q discrete instants prior to the sampling instant p there is an input of 1 at α (and therefore

*To make this automaton conform to our other automata, we can assume that it also has an input ρ, but that its output does not depend on ρ.

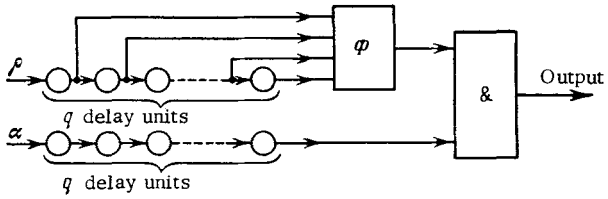


Fig. 7.3.

the auxiliary line has an output of 1 at $t = p$) and 2) there is an input of the given sequence of symbols during the q discrete instants preceding ρ (and therefore there is a converter output of 1 at $t = p$). It is seen from Fig. 7.3 that there cannot be an output of 1 until q discrete instants after $t = 0$ (since at $t = 0$ all delay units in the auxiliary line are in state zero). It follows from this that the states of the delay units in the main line do not affect the output of the system at $t = p$; that is, q instants after $t = 0$, the states of these delay units are determined only by the inputs.

This completes the first part of our proof. That is, we showed that the system of Fig. 7.3 represents an event R_α where R is any regular expression of depth zero. We now proceed to the second part of the proof.

Induction. We shall now prove that if there exist auxiliary automata that represent events R_α specified by any desired regular expressions of depth ν ($\nu \geqslant 0$), then we can synthesize an automaton that will represent the event R specified by any desired regular expression of depth $\nu + 1$.

From the definition of the concept "depth of a regular expression" it follows that any regular expression of depth $\nu + 1$ is obtained from one or two regular expressions of depth ν via a single multiplication, iteration, or union of sets.

In this connection, and in order to complete the induction, it must be proved that if events R_α, specified by regular expressions R_1 and R_2, are representable, then events R_α, specified by expressions

$$R = (R_1 \vee R_2),$$
$$R = (R_1 \cdot R_2),$$
$$R = (R_1^*) \quad \text{or} \quad R = (R_2^*)$$

are also representable.

Let us consider these three operations separately. For the sake of brevity and wherever there is no risk of ambiguity, we shall denote the regular expression, the event corresponding to it, and the automaton representing the event R_α by the same letter R.

Union of sets. An automaton representing a union

$$R = (R_1 \vee R_2),$$

is obtained from two automata, respectively representing R_1 and R_2. This is done by connecting their inputs and feeding their outputs to a common disjunction element (Fig. 7.4).

Multiplication. An automaton representing a product

$$R = (R_1 \cdot R_2),$$

is obtained from two automata, representing R_1 and R_2, respectively, by feeding the output of automaton R_1 into the auxiliary input of R_2 via a delay unit (Fig. 7.5). Then the auxiliary input line of R_1 becomes the auxiliary input to the system, while the output of R_2 is the output of the system. Indeed, the output of R_2 will be 1 if R_2 represents an event that has begun during the instant following an output of 1 in R_1. But an output of 1 in R_1 means that the event R_1 had already been represented prior to that instant, and that this event had begun at the instant of occurrence of 1 at the auxiliary input of R_1 (and therefore at the input of our entire automaton system). Consequently the automaton system as a whole represents the event R_α which is described by: "the event R_2 occurred directly after the event R_1," that is, the product $R_1 \cdot R_2$.

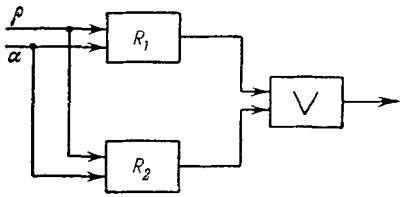


Fig. 7.4.

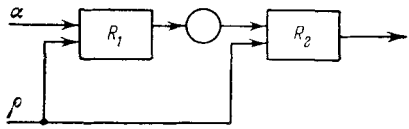


Fig. 7.5.

Iteration. If the automaton R_1 representing the event R_1 is given, then the automaton representing an R_α event: $R = (R_1^*)$ may be obtained by coupling R_1 to one disjunction element (Fig. 7.6). The disjunction element output is made to coincide with the auxiliary input to R_1, while the input of disjunction element consists of the auxiliary input α to R_1 as well as the R_1 output, which is made to pass through a delay unit. Indeed, the output of this system is 1 whenever the event R_1 is represented, starting with the appearance of 1 at the auxiliary input α, or from the instant directly following the representation of the event R_1. Therefore, there is a system output of 1

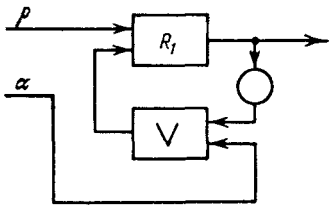


Fig. 7.6.

whenever the event R_1 occurs (this can happen any number of times in succession), which is what indeed constitutes the representation of event $R = (R^*)$.

This concludes the induction process based on the depth of the regular expression, and therefore completes the proof of the entire theorem.

First note. Our formulation of Kleene's first theorem included the statement about the choice of a suitable initial state. It can be seen from the proof that a suitable choice of the initial state reduces to: a) ensuring a state of 0 in all units of the auxiliary delay line during representation of the given set consisting of one input sequence (Fig. 7.3); b) ensuring a state of 1 in the delay unit of the automaton representing a universal event (Fig. 7.2); and c) ensuring a state of 1 in the delay unit of the autonomous automaton representing the event $t = 0$ (see above).

Second note. The representation of the event ER differs from that of the event R only in the method of utilizing the auxiliary input of the automaton which represents the event R_a. To be precise, in order to represent the event R, this input is connected to the autonomous automaton which produces 1 only at $t = 0$; however, in order to represent the event ER, this input consists of the output of the automaton representing the event E (Fig. 7.2); that is, this input is always 1. This applies, in particular, to the representation of a specific event $E \cdot a$.

7.5. REGULARITY OF REPRESENTABLE EVENTS

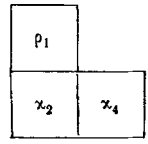
In this section we shall prove a theorem that is the converse of the one proved above.

Kleene's second theorem. Only regular events are representable in a finite automaton.

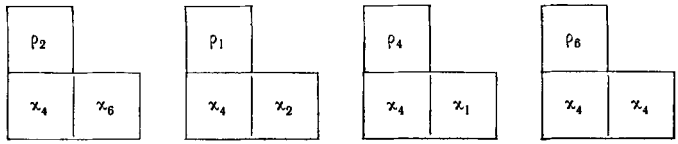
To prove this theorem we shall first introduce the concept of regular sets of triad chains, then we shall prove an auxiliary lemma, and finally with the help of this lemma, we shall prove the theorem.

The triad labyrinth and labyrinthine paths. The infinite set of tapes which may be generated in a finite automaton contains only a finite number kr of different triads. Let us denote these by μ_1, μ_2, ..., μ_{kr} and match each triad μ_i to a point in a plane.

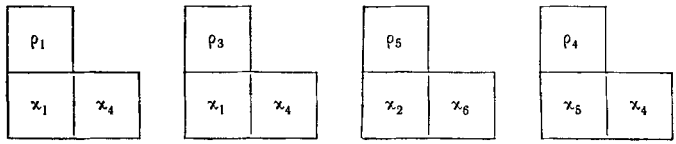
Two adjacent triads must have one common symbol. For this reason, the triad



must be followed by a triad containing x_4 at lower left, for example,



so that triads with any other symbols in this position, for example,



are not allowed.

Now, we draw arrows from the point corresponding to triad μ_i to all points corresponding to triads which can follow μ_i on the tape (Fig. 7.7); we do the same for all triads μ_i ($i = 1, 2, \ldots, kr$). We obtain (see Fig. 7.8) a plane diagram, which we shall call the *triad labyrinth*.

Now let some triad μ_i be the starting point from which we proceed along any desired arrow, writing out the sequence of triads encountered on the way. This sequence is called the *path in the triad labyrinth*, or simply a *path*. The length of a path is measured by the number of triads contained in it. Thus, if μ_2 is the starting point, Fig. 7.8 has paths described by triad chains.

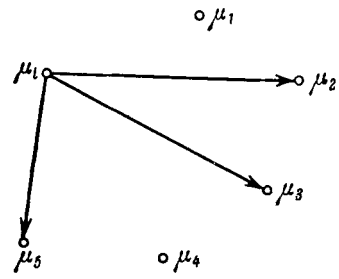


Fig. 7.7.

$\mu_2\ \mu_3\ \mu_4\ \mu_5\ \mu_6\ \mu_1\ \mu_2$,

$\mu_2\ \mu_4\ \mu_3\ \mu_1\ \mu_4\ \mu_5$,

$\mu_2\ \mu_1\ \mu_3\ \mu_4$, etc.

However, path

$$\mu_2\ \mu_3\ \mu_6$$

is not allowed since there is no arrow from μ_3 to μ_6.

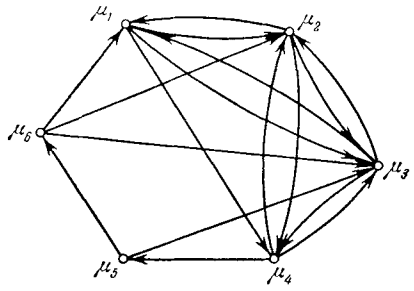


Fig. 7.8.

Now let us proceed to the concept of a *regular set of triad chains*. By definition, all sets containing only one triad are regular; if A and B are regular sets of triad chains, then, by induction, the following are also regular:

a) their union $(A \vee B)$, that is, a set containing both A and B;

b) their product $(A \cdot B)$, that is, a set of chains obtained by adding some chain from B to the right of each chain from A;

c) their iteration (A^*), that is, a set of chains obtained by adding to the right-hand sides of all chains from A any and all chains from A (including itself or any of those already added) and doing that any desired number of times.

Lemma. The set of all paths leading from triad $\bar{\mu}$ to triad $\tilde{\mu}$ in the labyrinth of any finite automaton is a regular set of triad chains.

We shall prove this lemma by induction based on q, where q is the number of different triads on the path from $\bar{\mu}$ to $\tilde{\mu}$.

For $q = 1$, the lemma is obvious: μ coincides with μ and only two subcases are possible:

a) if all symbols $\varkappa$ in the lower line of this triad $\bar{\mu} = \tilde{\mu}$ are not the same, then the set of paths consists of only one path containing this one triad $\bar{\mu}$;

b) if all $\varkappa$ in $\bar{\mu}$ are the same, then the set of paths comprises all triad chains repeating the same triad $\bar{\mu}$, that is, the iteration $(\bar{\mu})^*$.

Consequently the set is regular for $q = 1$.

Now let the lemma hold for any set of paths with $q \leqslant m$. We shall prove that it also holds for any set of paths with $q = m + 1$. Thus let us consider any path where $q = m + 1$. We write out all the triads $\bar{\mu}$ as well as the triad $\tilde{\mu}$ at the end of the path, and replace the remaining triads of this path (including any triads $\tilde{\mu}$ that may occur at points other than the end) by groups of dots:

$$\bar{\mu} \ldots \bar{\mu} \ldots \bar{\mu} \ldots \bar{\mu} \ldots \bar{\mu} \ldots \tilde{\mu}.$$

Each group of dots stands for a path that does not contain the triad $\bar{\mu}$. For this reason, the number of different triads contained in each path replaced by a group of dots is smaller than $m+1$ by at least 1, that is, $q \leqslant m$. By induction, each path replaced by dots is a regular set of triad chains. We shall denote these replaced paths by $\bar{r}_i$, $\bar{r}_j$, $\bar{r}_g$, and so on.

To start with, let $\tilde{\mu} = \bar{\mu}$. Then one can imagine the entire path from $\bar{\mu}$ to $\tilde{\mu} = \bar{\mu}$ as consisting of triads $\bar{\mu}$, with the regular sets $\bar{r}$ interspersed among them. We write this as follows:

$$\bar{\mu}\bar{r}_i\,\bar{\mu}\bar{r}_j\,\bar{\mu}\bar{r}_g\,\bar{\mu}\,\bar{\mu}\bar{r}_s \ldots \bar{\mu}\bar{r}_q\,\bar{\mu}.$$

In this notation, the entire path consists of pairs of consecutive paths. Each such pair is the product of two regular sets: a set consisting of a single element $\bar{\mu}$, and the corresponding set $\bar{r}$. Consequently, such a pair must itself belong to a regular set $\bar{\mu} \cdot \bar{R}$. Therefore, all paths leading from $\bar{\mu}$ to $\tilde{\mu} = \bar{\mu}$ are elements of the iteration of $\mu \cdot \bar{R}$

$$(\bar{\mu} \cdot \bar{R})^* \cdot \bar{\mu};$$

that is, they constitute a regular set of triad chains.

If $\bar{\mu} \neq \tilde{\mu}$ then by the same reasoning the set of all paths leading from $\bar{\mu}$ to $\tilde{\mu}$ is

$$(\bar{\mu} \cdot \bar{R})^* \cdot \tilde{\mu};$$

that is, it is also regular. The lemma is thus proved.

This lemma can be readily explained on a triad labyrinth.

Consider the triad labyrinth of Fig. 7.9 (where some arrows have been omitted for clarity), and run through the possible paths from $\bar{\mu}$ to $\tilde{\mu}$.

First, there is a choice of the two paths

$$\bar{\mu}\ 1\ 2\ \ 3\ \tilde{\mu}$$

and

$$\bar{\mu}\ 8\ 9\ 10\ \tilde{\mu}.$$

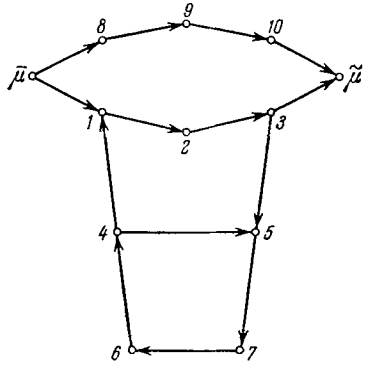


Fig. 7.9.

Second, the first path may be lengthened by including a loop, for example,

$$\bar{\mu}\ 1\ 2\ 3\ 5\ 7\ 6\ 4\ 1\ 2\ 3\ \tilde{\mu}.$$

The looping paths may be traversed any desired finite number of times, for example,

$$\bar{\mu}\ 1\ 2\ 3\ 5\ 7\ 6\ 4\ 1\ 2\ 3\ 5\ 7\ 6\ 4\ 1\ 2\ 3\ 5\ 7\ 6\ 4\ 1\ 2\ 3\ \tilde{\mu},$$

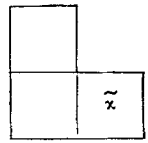
or the path may be complicated still further by including "loops within loops," for example,

$$\bar{\mu}\ 1\ 2\ 3\ 5\ 7\ 6\ 4\ 1\ 2\ 3\ 5\ 7\ 6\ 4\ 5\ 7\ 6\ 4\ 5\ 7\ 6\ 4\ 1\ 2\ 3\ \tilde{\mu},\ \text{etc.}$$

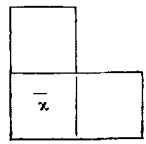
This type of routing is not confined to our example. Each set of paths from $\bar{\mu}$ to $\tilde{\mu}$ contains, first of all, several variants of simple paths (a union!), and it may also consist of arrows traversed one after another (a product!) and of loops traversed any desired finite number of times along anywhere in the labyrinth (an iteration!). And each path from $\bar{\mu}$ to $\tilde{\mu}$ may consist of these three components. This fact, almost obvious from the examination of the labyrinth, is really what the lemma is stating.

From the lemma immediately follows the proof of Kleene's second theorem.

Proof of Kleene's second theorem. Consider a finite automaton, and write out all its triads (a finite number). Let M be a set of several symbols $\varkappa$ (we shall denote an element of this set by $\tilde{\varkappa}$), and let the automaton (which starts with an initial state $\varkappa^0 = \bar{\varkappa}$) represent an input event by the appearance of a state $\tilde{\varkappa}$. Now, let us select a triad $\tilde{\mu}$ such that the $\tilde{\varkappa}$ in its lower right belongs to M (the other two symbols of the triad need not be part of M):



Let us also select a triad $\bar{\mu}$ containing $\bar{\varkappa}$ in its lower left, the other two symbols being completely arbitrary:



Then, by virtue of the lemma just proved, the set of all paths leading from $\bar{\mu}$ to $\tilde{\mu}$ is a regular set of triad chains.

Consider sets $\bar{A}$ and $\tilde{A}$ of all possible triads $\bar{\mu}$ and $\tilde{\mu}$. The set of all paths connecting any triad $\bar{\mu}$ of $\bar{A}$ with any triad $\tilde{\mu}$ of $\tilde{A}$ is the union of the sets of paths leading from $\bar{\mu}$ to $\tilde{\mu}$; that is, it is regular.

To each triad there corresponds a symbol ρ in the top strip of the tape, and to a chain of triads there corresponds a sequence of ρ. If a set of triad chains is regular then the set of corresponding sequences of ρ is a regular set of sequences, as defined in Section 7.3. Indeed, in devising new paths (from the given ones) by union, multiplication and iteration of chains of triads, we used operations analogous to those introduced in Section 7.3 for obtaining regular sets of sequences, and applied those to the ρ sequences in these chains.

We have not imposed any restrictions on the automaton, that is, on the number and the nature of its triads, or on the choice of the initial state. We have therefore proved that in any automaton, starting from any initial state, the set of input sequences leading from any initial state $\varkappa^0$ to a state comprised in M is regular; that is, we have proved Kleene's second theorem.

7.6. DO IRREGULAR (UNREPRESENTABLE) EVENTS EXIST?

Irregular events, which cannot be represented in an automaton, do exist, as we shall prove by examples. Moreover, we shall show that there exists an entire class of events, *a priori* known to be irregular.

Suppose we are given an infinite sequence of ρ, for example,

$$p_5\ p_2\ p_1\ p_5\ p_4\ p_3\ p_2\ p_1 \cdots$$

which we shall call $\mathscr{L}$. We will say that $\mathscr{L}$ is *ultimately periodic* if, starting with the qth symbol from the left, the sequence contains a periodically recurring segment of finite length.

Consider an event such that the input sequence of ρ coincides at at all times p with the segment of $\mathscr{L}$ bracketed by its 0th and the pth symbols. If such an event can be represented by an automaton, we shall say that *the automaton represents sequence* $\mathscr{L}$.

Theorem. A given infinite sequence $\mathscr{L}$ *of input symbols* p *is representable in a finite automaton if and only if* $\mathscr{L}$ *is "an ultimately periodic sequence."*

Proof. First, we shall prove that all ultimately periodic sequences are representable. Let such a sequence consist of an "initial segment" A of finite length h and then a periodically recurring "segment" B of finite length T. Consider all the initial sections

$B_1, B_2, \ldots, B_{T-1}$ of segment B (B_1 contains only the first symbol of B; B_2 contains the first two symbols, and so on), as well as the initial sections $A_1, A_2, \ldots$ of segment A. Thus the ultimately periodic sequence—that is, the input event—really consists of segment A followed by segment B, which is then repeated a number of times; the sequence terminates in one of the sections $B_1, B_2, \ldots, B_{T-1}, B$. Therefore $\mathscr{L}$ can be written as a regular expression

$$R = \{[A(B)^*] \cdot (B_1 \vee B_2 \vee \ldots \vee B_{T-1} \vee B)\} \vee (A_1 \vee A_2 \vee \ldots),$$

and, by virtue of Kleene's first theorem, such an event is representable in a finite automaton.

Now let us prove the converse statement, that is, that all representable sequences are ultimately periodic. Let sequence $\mathscr{L}$ be represented by a finite automaton which has k internal states. With $\mathscr{L}$ as the input, let us examine what happens during the first $k + 1$ sampling instants. Since $t = k + 1$, and the machine can have only k states, there must be at least one internal state $\bar{x}$ that will be repeated at least twice during this time. Let that occur in instants t' and t'' ($0 \leqslant t' < t'' \leqslant k + 1$). Now let us compile a sequence $\overline{\mathscr{L}}$, which differs from $\mathscr{L}$ in that the segment from t' to t'' has been thrown out; we then make $\overline{\mathscr{L}}$ the input of the same automaton. Thus, if with $k = 8$ sequence $\mathscr{L}$ corresponds to a tape in which $t' = 3$, $t'' = 6$ (Table 7.5), the tape for $\mathscr{L}$ then would be that of Table 7.6.

Table 7.5

t	0	1	2	$t' = 3$	4	5	$t'' = 6$	7	8	9	...
ρ	ρ_1	ρ_3	ρ_5	ρ_1	ρ_7	ρ_6	ρ_{12}	ρ_{10}	ρ_1	ρ_5	...
x	...	...	...	$\bar{x}$	...	...	$\bar{x}$	...	...	...	...

Table 7.6

t	0	1	2	3	4	5	6	7	8	9	...
ρ	ρ_1	ρ_3	ρ_5	ρ_{12}	ρ_{10}	ρ_1	ρ_5	...	...	...	...
x	...	...	...	$\bar{x}$	...	...	...	...	...	...	...

After t'' (that is, from instant 7), the sequence of $\varkappa$ on the first tape coincides with the $\varkappa$ sequence appearing on the second tape after t' (that is, from instant 4). This is because from these instants on, the inputs ρ are the same in both cases, as are the initial states $\varkappa$ of the automaton. And since $\mathscr{S}$ is represented by the automaton, $\overline{\mathscr{S}}$ must also be represented by it. This, however, does not contradict our definition of representability, according to which $\mathscr{S}$ and $\overline{\mathscr{S}}$ can be represented by the same automaton only if they coincide. For two infinite sequences of symbols, $\mathscr{S}$ and $\overline{\mathscr{S}}$, where $\overline{\mathscr{S}}$ differs from $\mathscr{S}$ by the fact that it lacks all symbols between t' and t'', may coincide only if beginning from t', $\mathscr{S}$ periodically repeats the segment occurring between t' and t''.

The theorem is thus proved.

This theorem enables us to specify an infinite number of examples of unrepresentable events, but we shall give only two such examples.

Example 1. An event occurs if the automaton input at $t = \alpha^2$ ($\alpha = 1, 2, 3, \ldots$) is ρ_1, and if the input is ρ_2 at all other times.

Example 2. An event occurs if the number of symbols ρ_2 between two nonadjacent symbols ρ_1 of the input sequence always doubles. Thus the event takes place if the sequence has the form

$$\rho_1\,\rho_2\,\rho_1\,\rho_2\,\rho_2\,\rho_1\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_1\,\rho_1\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_1\,\rho_1\,\rho_1\,\ldots,$$

and does not take place when the sequence is

$$\rho_1\,\rho_2\,\rho_1\,\rho_2\,\rho_2\,\rho_1\,\rho_2\,\rho_2\,\rho_2\,\rho_2\,\rho_1\,\rho_1\,\rho_2\,\rho_2\,\rho_1\,\ldots\,.$$

The events of each of the above examples are unrepresentable since the corresponding sequences are not ultimately periodic.

It must not be assumed, however, that all not representable events are embraced by this theorem. For example, we may have an input alphabet ρ consisting of two symbols $\{0, 1\}$, and we may define an event so that at $t = p$, the number of symbols 1 in the output sequence is equal to the number of symbols 0. This event does not belong to the class considered above (since the occurrence of this event is determined not by one specific sequence but by an entire class of such sequences), but at the same time it is not representable. We shall prove this.

Assume we are given some finite automaton A, and that its input sequence consists of zeros only. Then at least one internal state $\varkappa$ of the automaton must sooner or later recur. Let that happen at t' and t'' ($0 \leqslant t' < t''$). Now consider the following two input sequences:

1) $\mathscr{L}$, consisting of t'' zeros followed by t'' symbols 1 and 2) $\mathscr{L}^*$, consisting of t' zeros followed by t'' symbols 1. The corresponding tapes are shown in Tables 7.7 and 7.8.

Table 7.7

t	0	1	2	...	$t'-1$	t'	...	$t''-1$	t''	...	$2t''-1$
ρ	0	0	0	...	0	0	...	0	1	...	1
$\varkappa$	...	...	...	...	...	...	...	...	$\bar{\varkappa}$	...	$\varkappa^*$

Table 7.8

t	0	1	2	...	$t'-1$	t'	...	$t'+t''-1$
ρ	0	0	0	...	0	1	...	1
$\varkappa$	...	...	...	...	...	$\bar{\varkappa}$	...	$\varkappa^*$

Beginning at t' and ending at $t'+t''$, the sequence of $\varkappa$ in the $\mathscr{L}^*$ tape coincides with that $\varkappa$ sequence of the $\mathscr{L}$ tape which starts at t'' and ends at $2t''$. This is because the initial states (at t' and t'', respectively) and the input sequences (t'' symbols 1 in succession) coincide in the two cases. Therefore, given tape $\mathscr{L}^*$, the automaton will attain state $\varkappa^*$ at $t'+t''$; this state will coincide with the state occurring at $2t''$ with tape $\mathscr{L}$. However, the event does occur with input sequence $\mathscr{L}$ (the number of symbols 1 is equal to the number of zeros), whereas it does not occur with $\mathscr{L}^*$ (the number of symbols 1 exceeds that of zeros), even though the automaton A achieves the same state in both cases. This means that automaton A cannot represent this event. Q.E.D.

It is easy to perceive why the events of these examples can not be represented. The reason is that, by definition, the finite automaton (with a finite number of states) has a "finite memory." But in the above examples, the amount of information which the machine must "remember" to be able to "decide" at any given instant whether an event does take place, becomes infinite with time (in the first example, the machine must "remember" how many instants have elapsed prior to sampling; in the second example, the machine

must "count" the zeros occurring between two nonconsecutive symbols of 1, and this number also goes to infinity).

Let us point out that a set of sequences specifying an irregular (nonrepresentable) event may itself be a subset of a regular set of sequences. For example, an event specified as "two symbols 1 never appear consecutively at the input" is representable. However, the input sequences of Example 1 (see above) which are a subset of this regular set, constitute an irregular set.

7.7. WHAT A FINITE AUTOMATON "CAN DO"

In the preceding chapter we found out what an autonomous automaton "can do." Now we have mastered the representability of events and we can thus answer that question for *nonautonomous* automata.

Let A be an automaton with an output converter, or an s-machine representing a regular input event by generating a symbol (say, a 1) at the output. Further, let A stop as soon as 1 is generated at its output and let it trigger an autonomous automaton B that generates a predetermined sequence M. The output of B then becomes the input of automaton C which represents the event M, and the generation of a 1 at the output of C is used to trigger A and stop B (Fig. 7.10). This is done by the automaton Φ, which represents a simple event: the appearance of a given input symbol. Therefore this entire system constitutes a finite automaton. Such a finite automaton responds to any regular event by generating at the output any predetermined finite sequence of states (symbols), after which it is again ready to receive external stimuli, that is, to respond to events.

The automaton of Fig. 7.10, does not respond to those input symbols ρ that appear during the operation of B. If the (discrete) timing of B is so fast that its entire periodic output sequence is generated

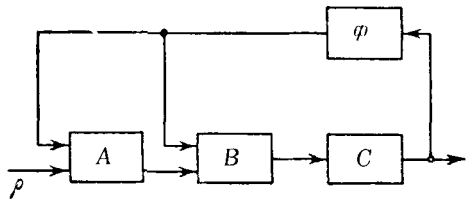


Fig. 7.10.

between two sampling instants of A, then there is no need for automaton Φ, and the output of A can be used as the input of B. Such an automaton maintains an output of any given periodic sequence of symbols throughout the representation of the event, and will generate another predetermined periodic sequence of symbols when no event is represented.

Naturally, other combinations of automata are also possible. But the above two combinations already show that an automaton can respond to any regular event by generating any predetermined cycle.

Can an automaton do something more than this? And if so, what? The answers to such questions depend on what language is used to formulate the laws for the handling of sequences by a finite automaton. Kleene's theorems formulate these laws in the language of representation of events. So far, there are no other convenient languages capable of describing (in terms of necessary and sufficient conditions) what a finite automaton "can do" and what, in principle, it "cannot do." This raises a number of new problems discussed in the next chapter.