6

Autonomous Finite Automata and Sequential Machines

6.1. WHAT AUTONOMOUS FINITE AUTOMATA AND SEQUENTIAL MACHINES "CAN DO"

This and the following chapter will deal with the capabilities of finite automata and sequential machines. We shall establish what they "can do" and shall determine what is in principle beyond the "capabilities" of any such machine. In the present chapter we shall determine the capabilities of the autonomous finite automaton.

Recall that a finite automaton is *autonomous* if the variable ρ under function F is fixed. Recall also that from this automaton, whose equation is

$$\varkappa^{p+1} = F(\varkappa^p, \rho^p),$$

one can compile r different autonomous automata (see Section 3.7)

$$\varkappa^{p+1} = F(\varkappa^p, \rho_*), \tag{6.1}$$

because ρ_* can coincide with any of the r symbols of alphabet $\{\rho\}$. The constant ρ_* may be treated as a symbol-parameter.

If $\varkappa^0$ is given, then by virtue of (6.1) we can find $\varkappa^1$. Then, substituting $\varkappa^1$ into (6.1), we can determine $\varkappa^2$, and so on. Thus, an autonomous finite automaton started at some initial symbol $\varkappa^0$ can generate an infinite sequence of symbols $\varkappa$:

$$\varkappa^0, \varkappa^1, \varkappa^2, \ldots, \varkappa^p, \ldots$$

The number of symbols in alphabet $\{\varkappa\}$ is finite and equal to k. For this reason, the automaton will generate, after a finite number of time units q $(q \leqslant k)$, a symbol that has already appeared in the sequence, for example, the symbol $\varkappa_6$ in the sequence

$$\varkappa_2\varkappa_3\varkappa_6\varkappa_5\varkappa_4\varkappa_7\varkappa_6 \ldots$$

Thus, from the $(q+1)$-th time unit onward, the automaton will simply periodically repeat this symbol sequence (the sequence of our example is $\varkappa_2\varkappa_3\underline{\varkappa_6\varkappa_5\varkappa_4\varkappa_7}\varkappa_6\varkappa_5\varkappa_4\varkappa_7\varkappa_6\varkappa_5\varkappa_4\varkappa_7 \ldots$ where the recurring sequence is underlined).

It follows that an autonomous automaton, started up at any initial state will, after at most k time units, periodically repeat a sequence of symbols $\varkappa$ (whose length does not exceed k). With any other initial state, this automaton would also periodically repeat a sequence of symbols $\varkappa$ after some time. However, this sequence may not coincide with that generated before.

A special case occurs when the sequence consists of a simple symbol. This will occur when, for example,

$$\varkappa_2\varkappa_3\varkappa_6\varkappa_6\varkappa_6 \ldots$$

or

$$\varkappa_2\varkappa_2\varkappa_2\varkappa_2 \ldots$$

A symbol that appears twice in succession in a sequence shall be called a *stable* symbol. We shall say that the automaton becomes stable during the time unit in which the stable symbol first appears.

Symbol $\varkappa_j$ is stable if

$$\varkappa_j = F(\varkappa_j, \rho_*), \tag{6.2}$$

that is, at $\rho = \rho_*$, symbol $\varkappa_j$ is translated by Eq. (6.1) into itself.

All these statements are illustrated by the diagram of the autonomous automaton. This diagram contains k circles, corresponding to all the possible symbols $\varkappa$ that can be generated. Several arrows may converge on each circle, but only one may be drawn from it. For this reason, if we proceed in the direction of the arrows, we are bound to return after no more than k steps to one of the circles previously crossed (the arrows form a *cycle*, Fig. 6.1), or, alternatively, we shall arrive to a circle at which the leaving arrow forms a loop, that is, indicates equilibrium (see Fig. 6.2).

Figure 6.3 shows various diagrams for $k = 12$.

In the case of Fig. 6.3, the cycle involves all the available circles. Here the machine generates from the outset a periodically

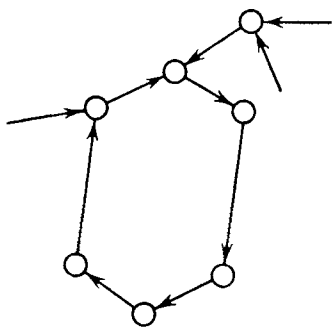

Fig. 6.1.

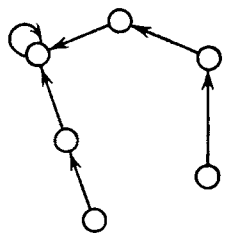

Fig. 6.2.

recurring sequence consisting of all k symbols.* The initial state determines only the order of the symbols in the sequence.

In Fig. 6.3,b the cycle encompasses only five circles; the arrows drawn from the remaining circles converge on one of the circles of the cycle. Here the automaton generates a sequence of length 5; this sequence will be generated after a maximum of one time unit, regardless of the initial state of the automaton.

For the case of Fig. 6.3,c the maximum number of time units preceding the generation of the periodic sequence is four.

Figure 6.3,d shows an autonomous automaton that, depending on its initial state, generates one of three sequences (of lengths 2, 3, and 5, respectively).

The automaton of Fig. 6.3,e can, depending on its initial state, either attain equilibrium after a maximum of three time units, or

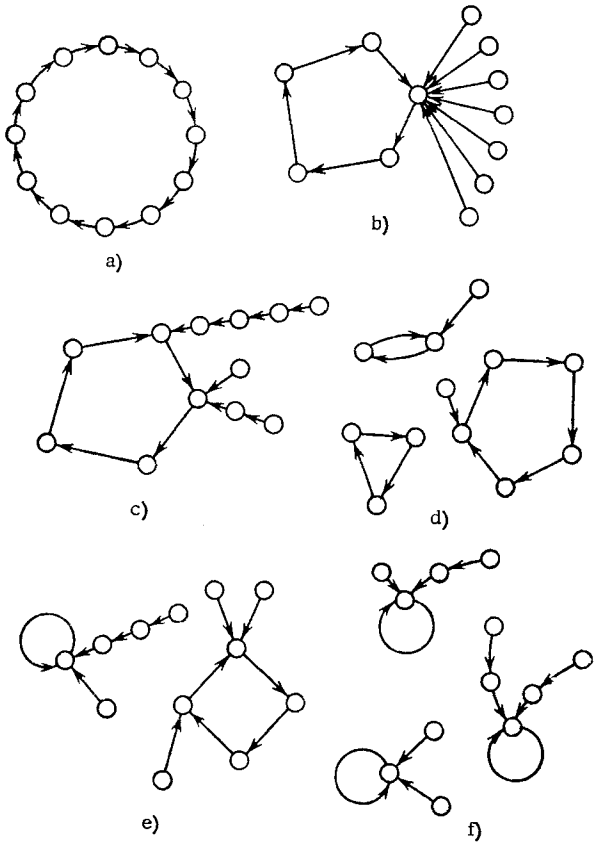

Fig. 6.3.

*In this and the following case we are assuming that it is immaterial what the initial symbol of the sequence is.

can start to generate periodic sequence of length 4 after a maximum of one time unit.

Finally, Fig. 6.3,f shows an autonomous automaton with three equilibrium states: depending on its initial state, some equilibrium is attained in at most two discrete time units.

If we want to synthesize an autonomous automaton generating a periodic sequence of some given length q and we impose some additional conditions as to the time preceding the generation of this sequence, all we need to do is to draw a diagram satisfying these conditions. This is because the diagram defines the autonomous automaton uniquely [the basic table can be directly reconstructed from it (Chapter 3)].

Suppose that we want to generate periodic sequences of lengths 2, 4, and 6. Which of the sequences shall be generated shall be determined by the initial state of the automaton; in no case can the generation of the sequence begin later than one time unit after start up.

The minimum k satisfying these conditions is 12. If $k = 12$ (Fig. 6.4), then the appropriate diagram is drawn by joining into cycles any of two, four, and six circles. If, however, $k > 12$, for example, if $k = 16$ (Fig. 6.5), then, to satisfy the conditions, the arrows emerging from circles not included in the cycles are directed to any of the circles of the three cycles.

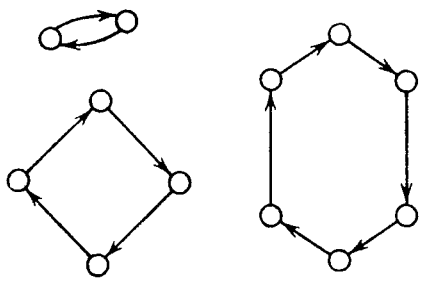

Fig. 6.4.

Problems involving variation of the parameter ρ_* are solved in a similar manner. For each $\rho = \rho_*$ there is an autonomous automaton, that is, a diagram. However, all these automata operate in the same alphabet $\{\varkappa\}$. For this reason, the diagrams of all autonomous automata obtained by varying ρ_* contain the same number k of circles (nodes). Thus assume as before that the automaton generates periodic sequences of lengths 2, 4, and 6. But now it will be the parameter ρ_* (ρ_1, ρ_2 or ρ_3) which will determine which of these sequences shall be generated. As before, the generation of the sequence

at any ρ_* shall begin no later than one discrete time unit after start up of the automaton.

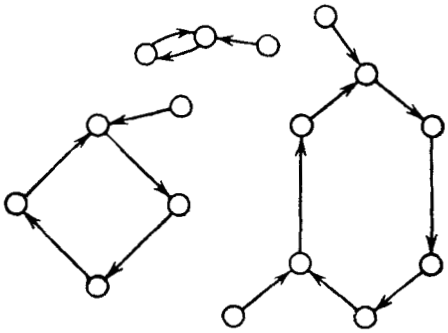

Fig. 6.5.

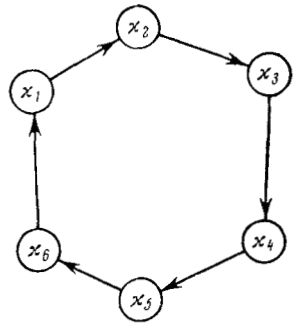

Fig. 6.6.

Obviously, k must be at least equal to the length of the longest given sequence; otherwise it would be impossible to design an autonomous automaton generating that sequence at any ρ. Now, assuming, for example, that $k = 6$, we construct a diagram for $\rho_* = \rho_3$ (Fig. 6.6), connecting all the six nodes into a cycle. Then the graph for $\rho_* = \rho_2$ also contains six nodes, but only four of them (any four) need to be connected into a cycle. The arrows drawn from the remaining two circles are therefore made to converge on the nodes of the cycle (Fig. 6.7).

Finally, only two nodes are connected into a cycle for $\rho_* = \rho_1$, while the arrows from the remaining circles are made to converge on these nodes (Fig. 6.8).

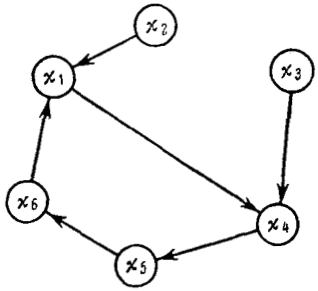

Fig. 6.7.

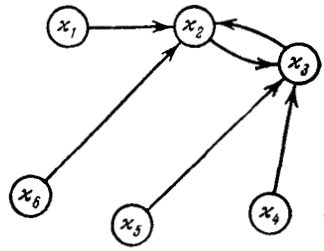

Fig. 6.8.

These three diagrams define the automaton, and its basic table can be readily derived from them (Table 6.1).

Now let us proceed to the sequential machine, where the automaton is supplemented by an output converter.* If the automaton periodically generates a sequence of symbols $\varkappa$, then the converter also must periodically generate a sequence of symbols λ of the same length. The sequence of $\varkappa$ cannot contain two identical symbols, but the output sequence of λ may do so. For example, if the automaton generates the recurrent sequence $\varkappa_1\varkappa_7\varkappa_2\varkappa_5\varkappa_3$, while the converter embodies table

Table 6.1

$\varkappa$ \ ϱ_*	ϱ	ϱ_1	ϱ_3
$\varkappa_1$	$\varkappa_2$	$\varkappa_4$	$\varkappa_2$
$\varkappa_2$	$\varkappa_3$	$\varkappa_1$	$\varkappa_3$
$\varkappa_3$	$\varkappa_2$	$\varkappa_4$	$\varkappa_4$
$\varkappa_4$	$\varkappa_3$	$\varkappa_5$	$\varkappa_5$
$\varkappa_5$	$\varkappa_3$	$\varkappa_6$	$\varkappa_6$
$\varkappa_6$	$\varkappa_2$	$\varkappa_1$	$\varkappa_1$

$\varkappa$	$\varkappa_1$	$\varkappa_2$	$\varkappa_3$	$\varkappa_5$	$\varkappa_7$
λ	λ_1	λ_1	λ_2	λ_2	λ_1

,

the sequence $\lambda_1\lambda_1\lambda_2\lambda_2\lambda_1$ will periodically recur at the output of the converter.

If the desired sequence of λ is given, then one synthesizes an s-machine producing this sequence by first synthesizing an automaton generating any sequence of $\varkappa$ of the same length as the given sequence of λ. Then, by writing these two sequences one below the other, we have produced the converter table.

The above discussion has important practical implications.

If a practical device can be considered described by the abstraction "autonomous finite automaton," and if the symbols $\varkappa$ characterize the possible states of this device, then the device can either attain equilibrium in a finite time, or it can periodically repeat a finite sequence of states. It can do nothing else. It also follows that given any finite number of sequences of states of any finite length, we can always synthesize an automaton generating any of these sequences by a judicious choice of its initial state. Alternatively, we can synthesize this automaton by identifying the parameter which can remain constant throughout the operation and fixing its value at the very outset.

To show the practical significance of this technically important conclusion we shall now discuss the synthesis of a bistable abstract structure substituting for an autonomous sequential machine, after which we shall generalize our reasoning to other abstract structures.

---

*Since ϱ_* is fixed, in an autonomous automaton, "sequential machine" and "finite (autonomous) automaton with an output converter" mean the same thing.

6.2. SYNTHESIS OF THE BISTABLE STRUCTURE OF AN AUTONOMOUS SEQUENTIAL MACHINE

Consider first the case when parameter $\rho = \rho_*$ cannot be varied; that is, we do not wish to synthesize a one-parameter family of finite automata or sequential machines, but just one such machine. We formulate the problem as follows: given the number of initial states which determines the number of possible variants of the operation of the machine; for each of these states we are given a sequence of 0 and 1 that the machine must generate, starting the generation no later than one time unit after the beginning of the operation. It is desired to synthesize the bistable structure of the finite automaton and the bistable (logical) converter satisfying the given conditions. We must determine not only the logical functions in the right-hand sides of the equations for the bistable abstract structure, but also the number n of such equations, whereby it is required to obtain a solution with the minimum n. We shall consider the problem solved if we can synthesize a bistable abstract structure with a minimum number of equations, and shall forego further simplification of these equations by means of propositional calculus to meet optimization criteria.

This problem may be divided into for subproblems:

1. Determination of the minimal number n.
2. Synthesis of a finite automaton whose output consists of sequences of the given length.
3. Synthesis of a bistable abstract structure which can substitute for this finite automaton.
4. Synthesis of the required output converter.

Consider first the case when we are given only one sequence of 0 and 1 of length q; it is required to generate it periodically (assuming that the first output symbol can be any of these symbols) from any initial state, the s-machine producing this sequence beginning with the second discrete time unit after the start of its operation.

Let us select the smallest n satisfying the inequality

$$2^n \geqslant q$$

and consider an automaton having $k = 2^n$ states. We shall make the alphabet of its states $\{\varkappa\}$ to coincide with the series of integers $\{0, 1, 2, \ldots, 2^n - 1\}$. Assume that the diagram of this automaton shows the first q nodes connected into a cycle; if $2^n > q$, then each of the remaining nodes is directly connected (by an arrow) to some node of

the cycle. For example, let us connect these "extra" nodes with the node denoted by 0 (Fig. 6.9 shows an example for $n = 3$, $q = 5$).

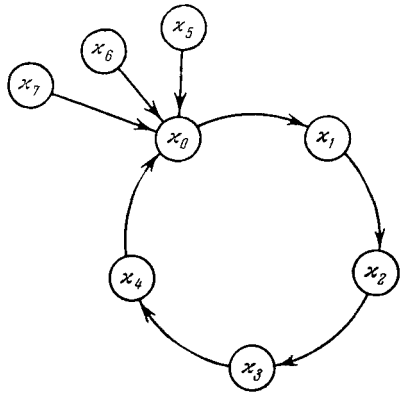

Fig. 6.9.

Table 6.2

$\varkappa$ \ p	p_*
$\varkappa_0$	$\varkappa_1$
$\varkappa_1$	$\varkappa_2$
$\varkappa_2$	$\varkappa_3$
$\varkappa_3$	$\varkappa_4$
$\varkappa_4$	$\varkappa_0$
$\varkappa_5$	$\varkappa_0$
$\varkappa_6$	$\varkappa_0$
$\varkappa_7$	$\varkappa_0$

This diagram immediately gives the basic table of the autonomous automaton. Thus, we obtain Table 6.2 for Fig. 6.9.

Now let us synthesize the bistable abstract structure of this automaton. We do this by the method of Section 4.2, by selecting n logical coordinates $x_1, x_2, \ldots, x_n$ (each of which may assume values only from the alphabet {0, 1}), and completing Table 6.3. The table contains 2^n rows and is divided in two parts by columns headed x^p and x^{p+1}.

Column $\varkappa^p$ is filled from top to bottom with numbers 0, 1, 2, ..., $2^n - 1$, that is, the subscripts of $\varkappa$ in the first column of the basic Table 6.2, while column $\varkappa^{p+1}$ contains the subscripts $\varkappa$ from the second column of that basic table.

Now we fill out the left-hand part of Table 6.3 (columns x^p) with binary representation of numbers contained in column $\varkappa^p$. It is convenient to use the rule already described in Chapter 1. Thus the last column on the right (for x_0) is filled with alternating 0 and 1. Column x_1 has alternating pairs of 0 and 1, column x_2— alternating quartets of 0 and 1, and so on. Thus the number of 0 and 1 in the alternating groups doubles in each succeeding column from the right.

We also fill the right part of the table (column x^{p+1}) with binary representation of numbers contained in column $\varkappa^{p+1}$.

Figure 6.4 shows the completion of the generalized Table 6.3 for the particular automaton of Fig. 6.9 and Table 6.2.

Now we turn to the x_0 column on the right of Table 6.3 (or 6.4), and underline all rows where $x_0 = 1$. We write out a conjunction of all x_k contained in the first such row, putting a negative sign over those x which are 0 in the same row of the left part of the table. In the same way, we write out conjunctions for the remaining underlined rows.

Table 6.3

x^p					χ^p	χ^{p+1}	x^{p+1}				
x_{n-1}	...	x_2	x_1	x_0			x_{n-1}	...	x_2	x_1	x_0
0	...	0	0	0	0	1	0	...	0	0	1
0	...	0	0	1	1	2	0	...	0	1	0
0	...	0	1	0	2	3	0	...	0	1	1
0	...	0	1	1	3	4	0	...	1	0	0
0	...	1	0	0	4	0	0	...	0	0	0
0	...	1	0	1	5	0	0	...	0	0	0
0	...	1	1	0	6	0	0	...	0	0	0
0	...	1	1	1	7	0	0	...	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
1	...	1	1	1	2^n-1	...	...	...	...	...	...

Table 6.4

Value of x^p			χ^p	χ^{p+1}	Value of x^{p+1}		
x_2	x_1	x_0			x_2	x_1	x_0
0	0	0	0	1	0	0	1
0	0	1	1	2	0	1	0
0	1	0	2	3	0	1	1
0	1	1	3	4	1	0	0
1	0	0	4	0	0	0	0
1	0	1	5	0	0	0	0
1	1	0	6	0	0	0	0
1	1	1	7	0	0	0	0

The conjunctive groups thus obtained are then joined by disjunction symbols. The full disjunctive form of the conjunctive groups thus obtained is the right-hand part of the first of the desired relations of the binary abstract structure. For example, for Table 6.4 this relationship is

$$x_0^{p+1} = [\overline{x_0^p} \,\&\, \overline{x_1^p} \,\&\, \overline{x_2^p}] \vee [\overline{x_0^p} \,\&\, x_1^p \,\&\, \overline{x_2^p}] \equiv \overline{x_0^p} \,\&\, \overline{x_2^p}. \qquad (6.3)$$

We shall now treat the x_1 column on the right of Table 6.3 (or 6.4) in an analogous manner and generate the right-hand part of the

second required relation. For Table 6.4, this becomes

$$x_1^{p+1} = [x_0^p \& \overline{x_1^p} \& \overline{x_2^p}] \vee [\overline{x_0^p} \& x_1^p \& \overline{x_2^p}] \equiv [x_0^p \sim \overline{x_1^p}] \& \overline{x_2^p}. \tag{6.4}$$

For column x_2, we obtain

$$x_2^{p+1} = x_0^p \& x_1^p \& \overline{\overline{x_2^p}} \tag{6.5}$$

and so on, until all the n rows have been examined and all the required relations of the desired binary abstract structure have been generated.

Now we synthesize the output converter

$$y = L(x_0, x_1, \ldots, x_{n-1}) \tag{6.6}$$

in the following manner. The q nodes of the cycle in the diagram of our autonomous automaton (see Fig. 6.9) are already numbered consecutively from 0 to $q - 1$. We assign the same numbers (from 0 to $q - 1$) to the 0 and 1 symbols comprising our given sequence (whose length is q). Thus, for example, if our given sequence has a length of 5 and the form 10010, we create the following one-to-one correspondence:

1 0 0 1 0
0 1 2 3 4

Let us separate the nodes which correspond to the symbol 1 of the given sequence. In our example, these nodes are 0 and 3, and they correspond to states $\varkappa_0$ and $\varkappa_3$.

We now compile a conjunction characterizing the values of all the coordinates of one of the states we separated, for example, $\varkappa_*$. To do this we compile a conjunction of all the x_i and place a negation sign over those which are 0 in the row $\varkappa_*$ in the left-hand part of Table 6.3. We form similar conjunctions for all the states that we have separated, and we join these conjunctions by means of disjunction symbols. In our particular example, we separated states $\varkappa_0$ and $\varkappa_3$. The row for $\varkappa_0$ in Table 6.4 contains only zeros, and thus we put a negation sign over all three coordinates x_0, x_1 and x_2. To obtain the conjunction

$$\overline{x_0} \& \overline{x_1} \& \overline{x_2}.$$

State $\varkappa_3$ yields (from the fourth row of Table 6.4) the conjunction

$$x_0 \& x_1 \& x_2.$$

Joining these conjunctions, we get the logical function

$$L(x_0, x_1, x_2) = (\overline{x_0} \& \overline{x_1} \& \overline{x_2}) \vee (x_0 \& x_1 \& \overline{x_2}). \tag{6.7}$$

It is immediately seen that this logical function equals 1 if and only if state x_i coincides with any of the above-separated states, that is, when 1 should be the output of the converter. Therefore, this expression $L(x_0, x_1, \ldots, x_{n-1})$ does indeed define the desired functional converter.

(These disjunctive descriptions of the operation of the automaton and the converter may, of course, be further simplified by the methods of propositional calculus to meet any criterion of optimality.)

The above binary abstract structure of the autonomous s-machine is a solution of our problem, and contains the necessary and sufficient number of states n. This number is sufficient because we were able to synthesize the required automaton with this value of n. It is necessary because at a smaller n the number of nodes in the diagram would be smaller than q, and therefore even if all the nodes were connected into a cycle, the number of time units in the period would be less than q; in that case, only a sequence smaller than q could be generated.

Now let us solve another problem. Assume we are given m different sequences of length q_i $(i = 1, 2, \ldots, m)$. Among them there may also be sequences of length 1. We select the smallest n satisfying the inequality

$$2^n \geqslant \sum_{i=1}^{i=m} q_i. \tag{6.8}$$

If

$$2^n = \sum_{i=1}^{i=m} q_i,$$

then the diagram of the automaton can have precisely m cycles, each of length q_i $(i = 1, 2, \ldots, m)$. If, however,

$$2^n > \sum_{i=1}^{i=m} q_i,$$

the diagram will contain nodes not connected into the cycles, and we shall draw arrows from each of them to one of the nodes of the cycle. After this, the bistable abstract structure of the automaton and the output converter are synthesized in precisely the same manner as in the case when one sequence was generated. In synthesizing the

converter we write out consecutively all the given m sequences and we number consecutively all the symbols (that is, 0 and 1) contained in these sequences.

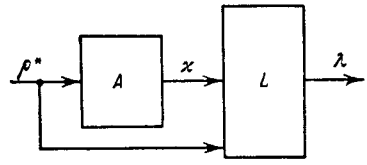

Fig. 6.10.

Now let us consider the case when the controlling parometers of the sequential machine ρ_*, can be varied, that is, when the binary structure

$$x_i^{p+1} = L_i[x_0^p, x_1^p, \ldots, x_{n-1}^p; u_1^p, u_2^p, \ldots, u_s^p], \quad i = 0, 1, 2, \ldots, n-1, \tag{6.9}$$

contains s binary parameters allowing 2^s possibilities.

Assume we are given m sequences. It is required to construct a binary abstract structure of an s-machine in accordance with Fig. 6.10, that is, we require a binary structure of the automaton A [Eq. (6.9)] and of the converter

$$y = L[x_0, x_1, \ldots, x_{n-1}; u_1, u_2, \ldots, u_s] \tag{6.10}$$

and we want the machine to generate the given m sequences. The additional requirement is that n and s should be minimum. The sequence to be generated is selected by choosing one of the parameters $u_1, u_2, \ldots, u_s$.

Let us separate the longest of the m given sequences. Let its length be $q_{\max}$, that is, let it contain $q_{\max}$ symbols. We find the minimum n and s satisfying the inequalities

$$2^n \geqslant q_{\max}, \tag{6.11}$$

$$2^s \geqslant m. \tag{6.12}$$

Let us call these values of n and s respectively $n_{\min}$ and $s_{\min}$. We now draw m diagrams, each of which contains $2^{n_{\min}}$ nodes. Then we number our given sequences consecutively. In the first diagram, we connect into a circle as many nodes as there are symbols in the first sequence, and draw arrows from the "extra" nodes to some node of the cycle. We do the same in the second diagram, except that now the cycle contains as many nodes as there are symbols in the second sequence, and so on. There will be a diagram for all our m sequences because, by virtue of (6.11), the number of nodes in each diagram is at least as large as the number of symbols in any given sequence.

Now we use the previously described method to develop a table such as 6.3 for each diagram; that is, we determine the right-hand

parts in relations

$$x_i^{p+1} = L_i\left[x_0^p, x_1^p, \ldots, x_{n_{\min}}^p\right], \quad i = 0, 1, \ldots, n_{\min} - 1, \qquad (6.13)$$

for each diagram of the binary abstract structure.

Assume that for the jth diagram ($j = 1, 2, \ldots, m$) these relations are

$$x_i^{p+1} = L_{ij}\left[x_0^p, x_1^p, \ldots, x_{n_{\min}}^p\right], \quad i = 0, 1, \ldots, n_{\min} - 1, \qquad (6.14)$$

and that m relations are defined in this manner.

Now we introduce $s_{\min}$ binary variables u_k ($k = 1, 2, \ldots, s_{\min}$) and compile a table of all possible combinations of u_k. This table (Table 6.5) is developed in the same manner as the left-hand part of Table 6.3. Table 6.5 contains $2^{s_{\min}}$ rows. The extreme right-hand column contains numbers of the first m rows [this is possible since by virtue of (6.12) the number of rows is not smaller than m].

Table 6.5

u_k					Number of u
$u_{s_{\min}-1}$	$u_{s_{\min}-2}$	...	u_1	u_0	
0	0	...	0	0	1
0	0	...	0	1	2
0	0	...	1	0	3
...	...	...	...	...	...
...	...	...	...	...	m
...	...	...	...	...	...
1	1	...	1	1	$2^{s_{\min}}$

Returning to relation (6.14) for the jth diagram, we add to the right-hand parts of all the $n_{\min}$ relations involved, conjunctions of those symbols u_k for which we have 1 in the jth row of Table 6.5. Thus we replace (61.4) with a relation

$$x_i^{p+1} = L_{ij}\left[x_0^p, x_1^p, \ldots, x_{n_{\min}-1}^p\right] \& u_{\alpha_1} \& u_{\alpha_2} \& \ldots \& u_{\alpha_k}, \qquad (6.15)$$
$$i = 0, 1, 2, \ldots, n_{\min} - 1.$$

By consecutively assuming $j = 1, 2, \ldots, m$, we can write m systems of relations such as (6.15).

We now combine by a disjunction the right-hand sides of all the first relations obtained from these m systems, then proceed likewise with respect to the right-hand sides of the second relations in these systems, and so on. As a result we get a single system of relations

$$\left.\begin{aligned} x_i^{p+1} = {} & \{L_{i1}[x_0^p, x_1^p, \ldots, x_{n_{\min}-1}^p] \& \mathfrak{U}_1\} \vee \\ & \vee \{L_{i2}[x_0^p, x_1^p, \ldots, x_{n_{\min}-1}^p] \& \mathfrak{U}_2\} \vee \\ & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & \vee \{L_{im}[x_0^p, x_1^p, \ldots, x_{n_{\min}-1}^p] \& \mathfrak{U}_m\}, \\ & i = 0, 1, 2, \ldots, n_{\min} - 1, \end{aligned}\right\} \tag{6.16}$$

where $\mathfrak{U}_j$ $(j = 1, 2, \ldots, m)$ denotes the conjunction of all the terms u_k which are equal to 1 in the jth row of Table 6.5.

Let us treat all the u_k as parameters and let us select them, for example, in accordance with the third row of Table 6.5. Then $\mathfrak{U}_3 = 1$, and all remaining $\mathfrak{U}$ are zero; this means that only the third conjunctive term will remain in each of relations (6.16). Equation system (6.16) then coincides with system (6.14) for $j = 3$. The diagram of this system consists of a cycle containing as many nodes as there are symbols (that is, 0 and 1) in the third of our given sequences.

Relations (6.16) are therefore an abstract structure substituting for a finite automaton from which, by selecting vector $\mathfrak{U}$ in accordance with the first m rows of Table 6.5, one can regenerate m diagrams of the same type as above, ensuring the generation of m sequences of given lengths.

In order for these sequences to coincide with the given ones, it remains to synthesize the output converter. To do this we first synthesize (by the methods described above) an output converter for each system of relations (6.14) separately, so that the jth relation (6.14) can generate the jth of the given m sequences.

Let these output converters be

$$y = L_j(x_1, \ldots, x_{n_{\min}-1}). \tag{6.17}$$

Then the desired converter to be added to (6.16) is specified by the relation

$$\begin{aligned} y = {} & [L_1(x_0, x_1, \ldots, x_{n_{\min}-1}) \& \mathfrak{U}_1] \vee \\ & \vee [L_1(x_0, x_1, \ldots, x_{n_{\min}-1}) \& \mathfrak{U}_2] \vee \ldots \\ & \ldots \vee [L_m(x_0, x_1, \ldots, x_{n_{\min}-1}) \& \mathfrak{U}_m]. \end{aligned}$$

It can readily be seen that this synthetic concept and procedure hold not only for a binary abstract structure, but also for any other structure.

In every case the abstract structure must yield a diagram in which there are as many modes in a cycle as there are characters in the given sequence. The transition from diagrams to the structure is accomplished by means of a table such as 6.3, but the completion of such a table and the number of its rows are determined by the particular properties of the synthesized structure, that is, the alphabet of its symbols.