An approach to automata schemes synthesis

A. S. PODKOLZIN, Š. M. UŠĆUMLIĆ

The function of a finite automaton can be described on both abstract and structural levels. In the first case, only some interrelation between input and output sequences of an automaton is pointed out without showing its structure (we call such automata — abstract). In the second case, an automaton is represented by a scheme, constructed of a set of "elementary" automata, defining a process of conversion of input sequences into output ones (we call such automata structural) [1]. One of the widely used languages, describing the functioning of finite automata on an abstract level is the language of regular expressions [2, 3], which can define time events, distinguishable by automata. In considering structural automata, we shall limit ourselves to schemes constructed out of the following elements: disjunction, conjunction, negation and delay. The problem of constructing a structural automaton V representing an event defined by a regular expression R (in this paper called the problem of synthesis of an automaton V), can be solved by constructing, in an intermediate stage, a Moore diagram of the automaton V [4]. The number of vertices of this diagram in many cases considerably exceeds both the complexity of the scheme of the automaton V and the length of the regular expression R. Corresponding estimation of the number of vertices depends exponentially on the given parameters. In such cases, regardless of a relatively simple scheme definition of an automaton V and a regular expression R, the given method of synthesis becomes in fact, non- applicable, and so arises the necessity of developing "direct" methods of synthesis of the automaton V, which are not based on the construction of its Moore diagram. A direct method of synthesis of automaton schemes, which is in fact an improvement of the synthesis method from [5], is suggested. (From now on these two methods will be reffered to as S_2 and S_1, respectively).

Let us describe the method S_1. The source information in this method is a regular expression R, obtained by application of operations $\cup$, $\cdot$, and $< >$ over the alphabet $A=\{a_1, a_2, ..., a_m\}$ (see [3]). Symbols of the alphabet A are supposed to be encored by binary strings of length $m'=[\log_2 m]$, so that a symbol a_i is encoded by the string $\tilde{a}_i=\tilde{a}_{i1}, ..., a_{im'})$; $a_{ij}\in\{0, 1\}$; $i=1, 2, ..., m$; $j=1, ..., m'$. The regular event defined by a regular expression R is denoted by $|R|$. Let us also introduce the notation $\|R\|=\{\tilde{a}_{i1}...\tilde{a}_{is}|a_{i1}...a_{is}\in|R|,\ s\geqq 1\}$. The problem of synthesis consists of constructing an automaton scheme Σ with m' inputs $x_1, ..., x_{m'}$ and one output y, having $\vee$, & — and delay elements (with 0 or 1 initial state) and representing the event $\|R\|$ by means of the output symbol 1. It means that the appearance of 1 as an output of the scheme Σ, at a moment t, is equivalent to the belonging of the word $\tilde{a}_{i1}\tilde{a}_{i2}\dots\tilde{a}_{it}$, to the event $\|R\|$, where $\tilde{a}_{ij}$ is a string arriving at the input of the scheme Σ at a moment j; $j=1, \dots, t$ (we assume that the first moment of time is assigned number 1). Note that in such an approach the empty word e, which generally speaking could be included in $|R|$, is not taken into account.

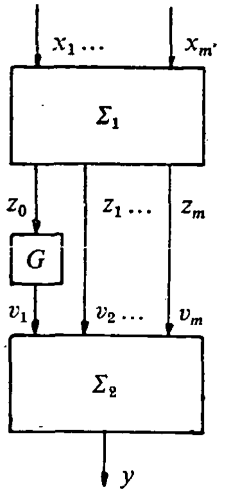

Fig. 1.

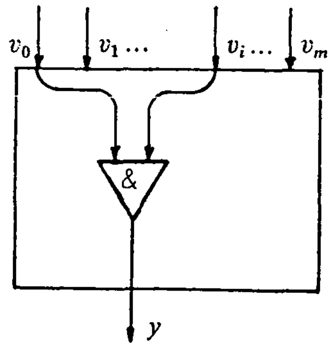

Fig. 2.

According to the method S_1, the scheme Σ is presented in the form given in Fig. 1. The scheme Σ_1 is constructed without using delay elements, its output z_0 is identical to 0, and an output z_i $(i=1, \dots, m)$ equals 1 if and only if $(x_1, \dots, x_{m'})=\tilde{a}_i$. Let us denote $\hat{a}_i^{\alpha}=(\alpha, 0, \dots, 0, 1, 0, \dots, 0)$; $\alpha\in\{0, 1\}$; $i=1, \dots, m$. By means of the symbol 1, the scheme Σ_2 represents the event $R=\{\hat{a}_{i1}^{\alpha_1}\hat{a}_{i2}^{\alpha_2}\dots\hat{a}_{ir}^{\alpha_r}\hat{a}_{ir+1}^{\alpha_r+1}\dots\hat{a}_{is}^{\alpha_s}|a_{ir}\dots a_{is}\in|R|;\ s\geqq r\}$ and it is defined by induction on the construction of the regular expression R. Let us further denote $\Sigma_2=\Sigma_2(R)$. The initial state of the delay element in Figure 1 equals 1. Construction of the scheme $\Sigma_2(R)$ is implemented in the following way.

1. R has the form a_i. In that case $\Sigma_2(R)$ has the form shown in Fig. 2.

2. R has the form $(R_1\cup R_2)$. In that case $\Sigma_2(R)$ has the form shown in Fig. 3.

3. R has the form $\langle R_1\rangle$. In that case $\Sigma_2(R)$ has the form shown in Fig. 4.

4. R has the form $(R_1\cdot R_2)$. In that case $\Sigma_2(R)$ has the form shown in Fig. 5.

If the regular expression R consists of k_1 occurrences of symbols from the alphabet A, k_2 occurrences of the symbol $\cup$, k_3 occurrences of the symbol $\cdot$ and k_4 occurrences of the symbol $<\ >$, then the scheme $\Sigma_2(R)$ has the least $k_1+k_2+k_3+2k_4$ elements.

The scheme Σ_1, which is essentially a decoder, can be constructed, for instance, as follows:

1) An input of the negation element is joined with each input $x_1, \dots, x'_m$. As a result, the values $\bar{x}_1, \dots, \bar{x}_{m'}$ are computed.

2) Let us consider the oriented tree T shown in Fig. 6. Every vertex w of the tree T, differing from w_0, w_1, w_2 is connected to the conjunction element. Let an edge marked x_i^σ lead from a vertex w' to a vertex w (where $x_i^0=\bar{x}_i$ and $x_i^1=x_i$). Then the first input of this conjunction element is connected to an input x_i, if $\sigma=1$, and to an output element computing $\bar{x}_i$, if $\sigma=0$. If $w'\notin\{w_1, w_2\}$, then the second input of the conjunction element is connected to an output of the element associated

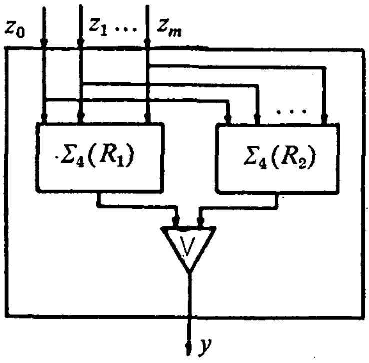

Fig. 3.

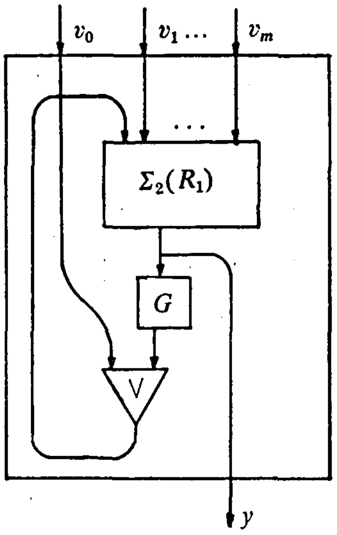

Fig. 4.

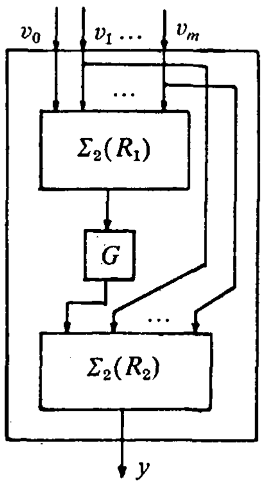

Fig. 5.

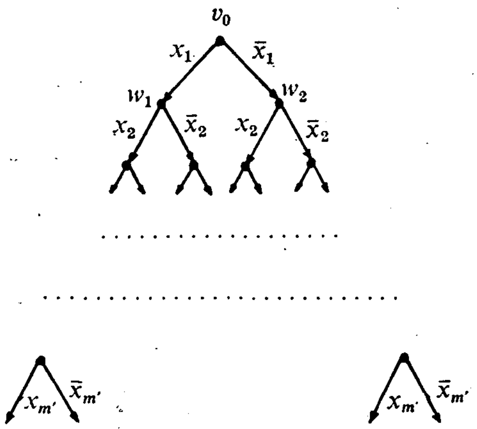

Fig. 6.

with the vertex w'. Otherwise, the second output of the conjunction element is connected to an input x_1, if $w=w_1$, and to an output of element computing $\bar{x}_1$, if $w=w_2$.

3) Conjunction elements, associated with terminating vertices of the tree T, compute all the possible functions of the logic algebra $x_1^{\sigma_1}\dots x_{m'}^{\sigma_{m'}}$. Let us remove those elements for which $(\sigma_1, \dots, \sigma_{m'}) \notin \{\tilde{a}_1, \dots, \tilde{a}_m\}$. The output of the conjunction element, computing the function $x_1^{a_{i1}}\dots x_{m'}^{a_{im'}}$, is the output z_i of the scheme Σ_1 $(i=1, 2, \dots, m)$.

4) A conjunction element whose inputs are connected to the input x_1 and to the output of the negation element computing $\bar{x}_1$, is introduced. This conjunction element computes the 0 — function, and its output is the output z_0 of the scheme Σ_1.

It is not difficult to check out that the number of elements of the scheme Σ_1 is $m'+(2^{m'}-4)+m+1$ and that it is not larger then $3m+\log_2 m-2$.

Let us describe the method S_2. Unlike the method S_1, let us assume that the initial regular expression R may contain an empty word symbol e and that $|R| \neq \{e\}$. In the preliminary step, a regular expression R is simplified by the use of the following transformations:

a) $eR' \to R'$; $R' \cdot e \to R'$.

b) $R'' \to e$, if $\|R''\| = \{e\}$ and $R'' \neq e$.

c) Let the expression R include R', which is of the form $\langle R_1 \cup \dots \cup R_n \rangle$; $n \geqq 1$. If any R_i equals e, then expression $\langle R_1 \cup \dots \cup R_{i-1} \cup R_{i+1} \cup \dots \cup R_n \rangle$ is substituted for the expression R'. If an R_i has the form $\langle R_i' \rangle$, then R' is replaced by $\langle R_1 \cup \dots \cup R_{i-1} \cup R_i' \cup R_{i+1} \cup \dots \cup R_n \rangle$. If any R_i has the form $R_i' R_i''$ and $e \notin |R_i|$, then $\langle R_1 \cup \dots \cup R_{i-1} \cup R_i' \cup R_i'' \cup R_{i+1} \cup \dots \cup R_n \rangle$ is substituted for R'.

d) $R_1 \cdot R_2 \cup R_1 \cdot R_3 \to R_1 \cdot (R_2 \cup R_3)$; $R_2 \cdot R_1 \cup R_3 \cdot R_1 \to (R_2 \cup R_3) \cdot R_1$ (In order to apply this transformation, it is possible to preliminary reorganize disjunctive elements of the expression).

Let us denote using $\tilde{R}$, the result of the described process of simplification of the expression R'. It is not difficult to notice that for any subexpression having the form $\langle Q \rangle$ in $\tilde{R}$ we have $e \notin |Q|$.

A scheme Σ, representing an event $|\tilde{R}|$ is constructed as shown in Fig. 7. where Σ_1 is a scheme constructed in the description of the method S_1 and Σ_3 is a scheme

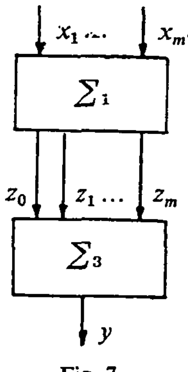

Fig. 7.

representing, using the symbol 1, the event

$$B(\tilde{R}) = \hat{a}_{i1}^{\alpha_1}\hat{a}_{i2}^{\alpha_2}\dots\hat{a}_{is}^{\alpha_s}|a_{i1}\dots a_{is}\in|\tilde{R}|,\ s\geqq 1\}\cup C(\tilde{R}),$$

$$C(\tilde{R}) = \hat{a}_{i1}^{\alpha_1}\hat{a}_{i2}^{\alpha_2}\dots\hat{a}_{ir}^{\alpha_r}\hat{a}_{ir+1}^{\alpha_{r+1}}\dots\hat{a}_{is}^{\alpha_s}|\hat{a}_{i2}\dots\hat{a}_{is}\in|\tilde{R}|;\ s\geqq r\}.$$

To construct the scheme Σ_3, let us define, by induction over the construction of the regular expression $\tilde{R}$, an axilliary scheme $\Sigma_4(\tilde{R})$ representing, by the use of 1, the same event $B(\tilde{R})$ as the scheme Σ_3:

1) $\tilde{R}$ has the form e. Then the scheme $\Sigma_4(\tilde{R})$ has the form given in Fig. 8.

2) $\tilde{R}$ has the form a_i; $i\in\{1, \dots, m\}$. Then $\Sigma_4(\tilde{R})$ has the form given in Fig. 9, where the initial state of the dealy element equals 1.

3) $\tilde{R}$ has the form $(R_1\cup R_2)$. Then $\Sigma_4(\tilde{R})$ has the form given in Fig. 10.

4) $\tilde{R}$ has the form $\langle R_1\rangle$. Then $\Sigma_4(\tilde{R})$ has the form given in Fig. 11.

5) $\tilde{R}$ has the form $R_1\cdot R_2$. Then, in case that $e\in|R_1|$, the scheme $\Sigma_4(\tilde{R})$ has the form given in Fig. 12 and in case that $e\notin|R_1|$, it is derived from the scheme in Fig. 12

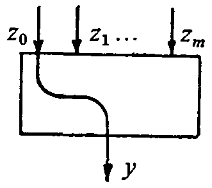

Fig. 8.

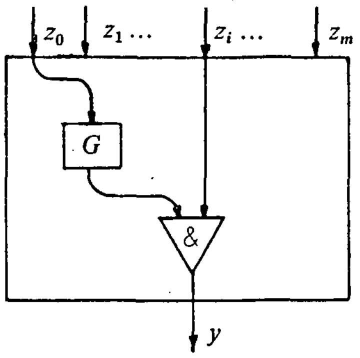

Fig. 9.

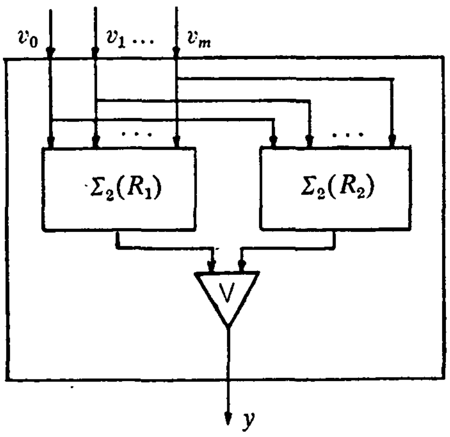

Fig. 10.

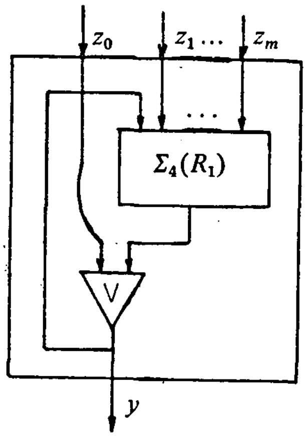

Fig. 11.

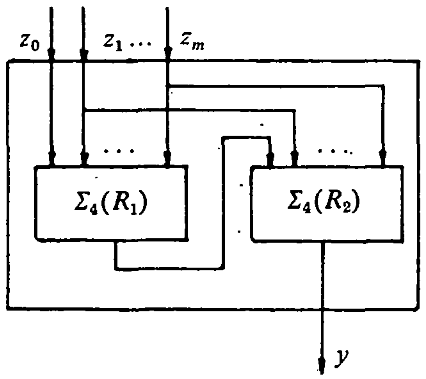

Fig. 12.

by substituting all the 1 — initial states of dealy elements in block $\Sigma_4'(R_2)$ for 0 — initial state. It is not difficult to see that the resulting block Σ (R_2) represents, by means of 1, an event $C(R_2)$.

The scheme Σ_3 derives from the scheme $\Sigma_4(\tilde{R})$ by means of the following transformations:

a) All the delay elements, having the same initial state, inputs of which are associated with an output of the same element or with the same input z_0 of the scheme $\Sigma_4(\tilde{R})$ become identical.

b) All conjunction elements whose inputs are connected to the same output of a delay element and to an input Z_i, become identical.

c) If one of the inputs of the disjunction element E turns out to be associated with the input z_0 of the scheme $\Sigma_4(\tilde{R})$, then the second input of E is identified with the output of this element, and the element itself is removed. If no input of an element of the scheme Σ_3 has been associated with the output z_0 of the scheme Σ_1, then the conjunction element whose output is z_0, is removed.

Let us denote the results of the application of the synthesis methods S_1 and S_2 to a regular expression R by $S_1(R)$ and $S_2(R)$, respectively. We call the number of elements of the scheme Σ the complexity of that scheme and denote it by $L(\Sigma)$. In order to compare the complexity of the schemes $S_1(R)$ and $S_2(R)$, let us define a few auxiliary notions. (Such comparison is possible only for regular expressions R, not containing the symbol e.)

An occurrence an expression of the form $\langle 0 \rangle$ in a regular expression $\tilde{R}$. Let a regular expression $R_1 \cup \dots \cup R_s$, $s \geqq 2$, occur in $\tilde{R}$, where each R_i has the form $\langle Q \rangle$ or $\langle Q \rangle R_i'$. Then an occurrence of $\langle Q \rangle$ in $\tilde{R}$ we call a disjunctive occurrence of iteration. All other occurrences of iterations in $\tilde{R}$ we call no-disjunctive.

Theorem 1. If a regular expression R does not contain the symbol e, then the following inequality holds: $L(S_2(R)) \leqq L(S_1(R)) - N$, where N is the number of nondisjunctive occurrences of iteration in a regular expression $\tilde{R}$.

Proof. If a regular expression R does contain the e symbol, then the transition to expression $\tilde{R}$ is performed only by means of the transitions given in (c) (see above).

The number of iterations doesnot increase, and total number of operations $\cup$ remains the same. Therefore, $L(S_1(\tilde{R}))\leqq L(S_1(\tilde{R}))$. Let us denote by k_1 — the number of occurrences of symbols from the alphabet A in a regular expression R, by k_2 — the number of occurrences of the symbol $\cup$, by k_3 — the number of the occurrences of the symbol $\cdot$, and by k_4 — the number of the occurrences of the symbol $< >$. We have

$$L(S_1(\tilde{R}))\geqq L(\Sigma_1)+k_1+k_2+k_3+2k_4+1;$$

$$L(S_2(R))\leqq L(\Sigma_1)+2k_1+k_2+k_4-k_1';$$

where k_1' is the number of delay elements removed from the scheme $\Sigma_4(\tilde{R})$ during its transformation into the scheme Σ_3. It is not difficult to notice that the number of occurrences of letters in the regular expression is one more than the number of occurrences of binary operations, i.e., $k_1=k_2+k_3+1$. Therefore $L(S_1(\tilde{R})\geqq L(\Sigma_1)+2k_1+$ $+2k_4$. Thus, for the proof of the theorem it is enough to prove the inequality $L(\Sigma_1)+2k_1+k_2+k_2+k_4-k_1'\leqq L(\Sigma_1)+2k_1+2k_4-N$, or after reduction: $k_2-k_1'\leqq$ $\leqq k_4-N$. Let Π be an occurrence of the expression $\tilde{R}$ having the form $R_1\cup R_2$; $i\in\{1, 2\}$, where each R_i has the form $R_{i1}\cdot R_{i2}\dots R_{i_s}$; $s\geqq 1$; R_{i1} is not of the form of $R'\cdot R''$. In that case we say that the occurrence of the expression R_{i1} in $\tilde{R}$ is subordinated to the occurrence of Π. Let us connect, with a regular expression $\tilde{R}$, an oriented graph G, whose vertices are the occurrence of the expression $R_1\cup R_2$ in R and also the occurrences subordinated to them. An edge leads from a vertex v to a vertex w in the graph G, if and only if the occurrence w is subordinated to the occurrence v. It is not difficult to notice that every occurrence in a regular expression $\tilde{R}$ is subordinated to not more than one occurrence. Therefore, the graph G represents the union of a finite number of oriented trees $G_1, \dots, G_r$. Let q_i denote the number of nonterminating vertices of the tree G_i; $i=1, \dots, r$. Evidently, $\sum_{i=1}^{r} q_i=k_2$. As from each nonterminating vertex of the tree G_i exactly two edges leave, the number of terminating vertices of the tree G_i equals q_i+1. Let q_i' be the number of terminating vertices of the tree G_i representing the occurrences of letters from the alphabet A. Then q_i+1-q_i' is the number of terminating vertices of the tree G_i representing occurrences of iterations and all these occurrences are disjunctive. It is not difficult to see that delay elements corresponding to the occurrences of letters from the alphabet A in $\tilde{R}$, which are terminating vertices of the tree G_i, become identical through the transformation of the scheme $\Sigma_4(\tilde{R})$ into the scheme Σ_3. Therefore, denoting $q_i''=q_i'$ when $q'=0$, and $q_i''=q_i'-1$ when $q_i>0$, we have: $\sum_{i=1}^{r} q_i''\leqq q_i'$. Consequently, $q_2-k_1'\leqq$ $\leqq \sum_{i=1}^{r}(q_i-q_i'')$. On the other hand, k_4-N is equal to the number of occurrences of disjunctive iterations in $\tilde{R}$, and consequently, $\sum_{i=1}^{r}(q_i+1-q_i')\leqq k_4-N$. When $q_i'=0$ we have: $q_i-q_i''=q_i\leqq q_i+1=q_i+1-q_i'$; however, if $q_i>0$ then $q_i-q_i'=$ $=q_i-q_i'+1$. This implies the inequality $\sum_{i=1}^{r}(q_i-q_i'')\leqq\sum_{i=1}^{r}(q_i+1-q_i')$. The theorem is proved.

Construction of the scheme Σ_3 by the method S_2 is realized by induction on the structure of a regular expression $\tilde{R}$, with further transformations of the scheme. Let us give a more convenient constructing procedure of the scheme Σ_3, based on a preliminary marking of a regular expression $\tilde{R}$. The marking gives the possibility to follow directly a series of removals of elements from the scheme $\Sigma_4(\tilde{R})$ during its transformation into the scheme Σ_3, as well as the joining together of the elements of that scheme.

Let us define a series of auxiliary notions connected to a regular expression $\tilde{R}$. Occurrences of letters Π_1 and Π_2 from the alphabet A in the regular expression $\tilde{R}$ are called similar if there exist such sequences of occurrences $\Pi_0, \Pi_1, \ldots, \Pi_s = \Pi$ and $\Pi_0, \Pi_1', \ldots, \Pi_r' = \Pi$ in $\tilde{R}$, in which every next occurrence is subordinated to the previous one. Similar occurrences of the same letter in $\tilde{R}$ we call adjacent. Therefore, every class of similar occurrences of letters in $\tilde{R}$ is devided into a number of subclasses of adjacent occurrences.

Let Π be an occurrence of a regular expression R' in the regular expression $\tilde{R}$. We shall now define occurrences in $\tilde{R}$, referred to as the basis and the predecessor of the occurrence Π:

1. If $R' = e$, then the basis of Π is equal to the predecessor of Π.

2. If $R' = a_i$, $i \in \{1, \ldots, m\}$, then the basis of Π represents a distinguished element of the class of occurrences of the letter a_i, occurrences being adjacent with Π (all elements of this class have the same basis). The occurrence of a letter is called basic if this occurrence is included in the basis.

3. If $R' = R_1 \cup R_2$ and if the predecessor of Π is defined or both expressions R_1 and R_2 differ from e, then the basis of the occurrence Π is Π. If the predecessor of the occurrence Π is undefined and $R_{i1} = e$; $\{i_1, i_2\} = \{1, 2\}$, then the basis of the occurrence Π is equal to the basis of the occurrence of R_{i2} in $\tilde{R}$. In all the enumerated cases the predecessors of the occurrences of R_1 and R_2 in $\tilde{R}$ are equal to the predecessor of the occurrence Π.

4. If $R' = \langle R_1 \rangle$, and the predecessor of the occurrence Π is defined, then the basis of the occurrence Π equals Π; otherwise it is equal to the basis of the occurrence of R_1 in $\tilde{R}$. If the predecessor of the occurrence Π is defined, then the predecessor of the occurrence of in $\tilde{R}$ is Π; otherwise it is equal to the basis of the occurrence of R_1 in $\tilde{R}$.

5. If $R' = R_1 \cdot R_2$, then the basis of the occurrence Π is equal to the basis of the occurrence of R_2 in $\tilde{R}$. The predecessor of the occurrence R_2 in $\tilde{R}$ is equal to the basis of the occurrence R_1 in $\tilde{R}$; the predecessor of the occurrence R_1 in $\tilde{R}$ is equal to the predecessor of the occurrence Π.

6. If $R' = \tilde{R}$, then the predecessor of the occurrence Π is undefined (at the some time all the predecessors as well as basis of the occurrences in $\tilde{R}$, which, according to p. 1—5 are equal to the predecessor of the occurrence Π, are undefined).

It is not difficult to check out that, in accordance with p. 1—6, for any occurrence of a regular expression in $\tilde{R}$, it is possible to unambigously find (in a finite number of steps), the basis and the predecessor of that occurrence, or their absence can be confirmed.

Next, let us define initial occurrences in the expression $\tilde{R}$:

1. An occurrence of the expression $\tilde{R}$ in itself is initial.

2. If an occurrence of the expression $\langle R_1 \rangle$ in $\tilde{R}$ is initial, then an occurrence of the expression R_1 in $\tilde{R}$ is also initial.

3. If an occurrence of the expression $R_1 \cup R_2$ in $\tilde{R}$ is initial, then the occurrences of the expressions R_1, R_2 in $\tilde{R}$ are initial.

4. If an occurrence of the expression $R_1 \cdot R_2$ in $\tilde{R}$ is initial, then the occurrence of the expression R_1 in $\tilde{R}$ is initial. If at the same time, $e \in |R_1|$, then the occurrence of the expression R_2 in $\tilde{R}$ is initial.

Let us describe, using the notions given above, the process of construction of the scheme Σ_3' obtained from $\Sigma_4(\tilde{R})$ by applying a portion of the transformations a)—c). The scheme Σ_3 will be obtained from Σ_4 by applying the unused of the transformations a) and b). The scheme Σ_3 is constructed in the following way:

1) For each class $\varkappa$ of similar occurrences of letters in $\tilde{R}$, a delay element, denoted by $G(\varkappa)$, is introduced. If all the occurrences from $\varkappa$ are initial then the initial state of this element equals 1, otherwise it is euqal to 0.

2) For each basis occurrence Π of a letter in an expression $\tilde{R}$, an element of conjunction, denoted by $E(\Pi)$, is introduced.

3) For each occurrence Π in the expression $\tilde{R}$ of the expression $R_1 \cup R_2$, such that $R_1 \neq e$ and $R_2 \neq e$ or the predecessor of Π is defined, an element of disjunction, denoted by $E(\Pi)$, is introduced.

4) For each occurrence Π having a predecessor, *in expression* $\tilde{R}$ of the expression $\langle R_1 \rangle$, an element of disjunction, denoted by $E(\Pi)$, is introduced.

5) The input of the element $G(\varkappa)$ is associated with the output of the element $E(\Pi')$ where Π' is a predecessor of an arbitrary occurence Π belonging to the class $\varkappa$. If the occurrence from $\varkappa$ do not have predecessors, then the input of the element $G(\Pi)$ is associated with the input z_0.

6) If Π is a basic occurrence of the letter a_i, which belongs to the class $\varkappa$ of similar occurrences of letters, then one of the inputs of the element $E(\Pi)$ is joined to the *output of the element* $G(\varkappa)$ *and the other is joined to the input* z_i.

7) If Π is an occurrence of the expression in $\tilde{R}$ and the element $E(\Pi)$ is defined, then the inputs of that element are connected with the outputs of the elements $E(\Pi_1)$ and $E(\Pi_2)$, where Π_1 and Π_2 are the basis of the occurrences of R_1 and R_2 in $\tilde{R}$.

8) If Π is an occurrence of the expression $\langle R_1 \rangle$ in $\tilde{R}$ and the element $E(\Pi)$ is defined, then the inputs of this element are connected with the inputs of the elements $E(\Pi_1)$ and $E(\Pi_2)$, where Π_1 is the predecessor of occurrence of Π and Π_2 is the basis of occurrence of R_1 in $\tilde{R}$.

9) The output of the scheme Σ_3' is the output of the element $E(\Pi)$, where Π is the basis of occurrence of $\tilde{R}$ in $\tilde{R}$ (this basis is defined as $|\tilde{R}| = \{e\}$).

Let us illustrate the method S_2 for a regular expression $R = \langle\langle a \rangle b \cup c \langle a \rangle\rangle$ in a three — letter alphabet $A = \{a, b, c\}$. Let us encode the symbols a, b, c, by binary strings 00, 01, 10 respectively. In this case the decoder Σ_1 has the from given in Fig. 13. There are no similar occurrences of letters in the expression R and therefore, delay and conjunctinn elements correspond to each occurrence of a letter. The form of the scheme Σ_3' is given in Fig. 14. Scheme Σ_3 is obtained from the scheme Σ_3' by the unifi-. cation of delay element corresponding to the initial occurrences of letter a and letter b. Since no element in Σ_3 is connected to the output z_0 of Σ_1, the corresponding conjunction element in Σ_1 is to be removed. The form of the scheme Σ is given in Fig. 15.

Let us note by $L(\tilde{R})$ the least complexity of the schemes, representing, by the symbol 1, an event $|R|$ defined by a regular expression R. The complexity of the regular expression R is the number of occurrences of the letters and operation symbols

$\cup, \ldots, \langle\,\rangle$ within this expression. Let us consider the Shenon function $L(m, n) = \max\limits_{R \in R_{m,n}} \tilde{L}(R)$, where $R_{m,n}$ is the class of all regular expressions in an m — letter alphabet A with complexity not greater than n. According to the synthesis method S_2, an estimation $\tilde{L}(m, n) \leq 3m + \log_2 m - 2 + 2n$ holds. In the paper [6], there is an example of a regular expression — with complexity n in a two — letter alphabet, for

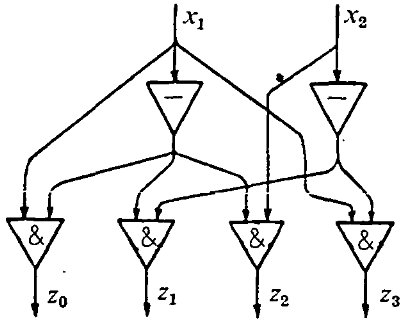

Fig. 13.

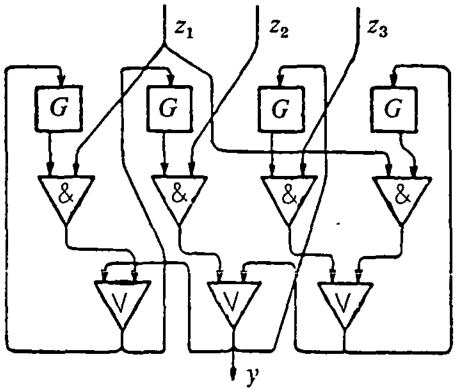

Fig. 14.

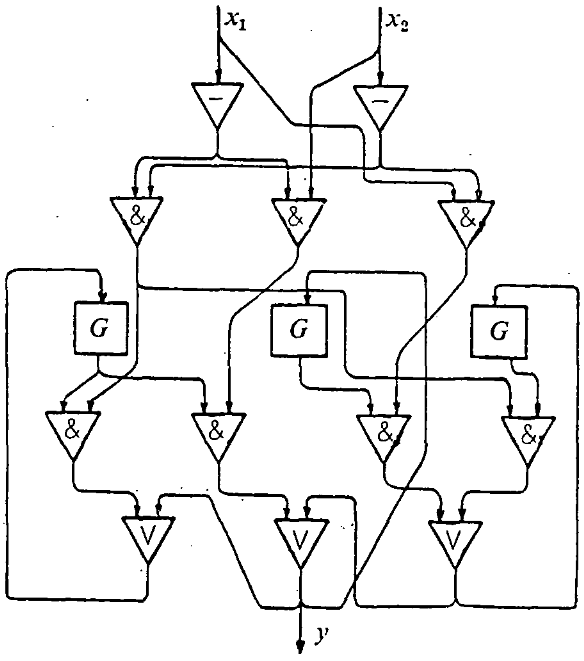

Fig. 15.

which the minimal number of delay elements in the corresponding scheme is asymptotically not less than $\frac{n}{5}$. It turns out that, for a fixed $m \geqq 2$ and $n \to \infty$, the following asymptotical inequalities hold: $\frac{n}{5} \lesssim \tilde{L}(m, n) \leqq 2$, i.e., the order of the entity $\tilde{L}(m, n)$ equals n.

As a conclusion, let us describe one of the methods of simplifying the scheme $\Sigma_4(\tilde{R})$, used in the process of realization of the method S_2 without marking the expression $\tilde{R}$. Let $\Pi_1, \ldots, \Pi_k$ be non — initial occurrences of the same regular expression R' in $\tilde{R}$. We call the occurrences $\Pi_1, \ldots, \Pi_k$ alternative, if for any letter p in the alphabet A, the set of pre-main positions of the expression $\tilde{R}$, p — following its starting positions (see [3]) has an empty intersection with the set of pre-main positions of not more then one of the occurrences $\Pi_1, \ldots, \Pi_k$. For example, the first and the second occurrences of the expression $(b\langle a\rangle \cup c)$ in the regular expression $a(b\langle a\rangle \cup c) \cup \cup \langle b\langle a\rangle\rangle \cup c\rangle$ are alternative.

Let $\Sigma_4(\tilde{R})$ be a scheme constructed by using the method S_2 for a regular expression $\tilde{R}$ and let $\Pi_1, \ldots, \Pi$ (be the alternative occurrences of a regular expression R' in $\tilde{R}$. Let us determine, in the scheme $\Sigma_4(\tilde{R})$, the blocks $B_1, \ldots, B_k$ corresponding to the occurrences $\Pi_1, \ldots, \Pi_k$. Each such block represents the scheme $\Sigma_4'(R')$ and at any instant of time t, all the blocks $B_1, \ldots, B_k$, except possibly one, have a zero state of delay elements. This enables us to use, in the scheme $\Sigma_4(\tilde{R})$, only one block $\Sigma_4(\tilde{R}')$ instead of k samples of such blocks. This block switches its input z_0 and output according to the positions of blocks B_i; $i=1, \ldots, k$. This switching will be realized using the scheme W given in Fig. 16. If $e \notin |R'|$ then the first input of the conjunction

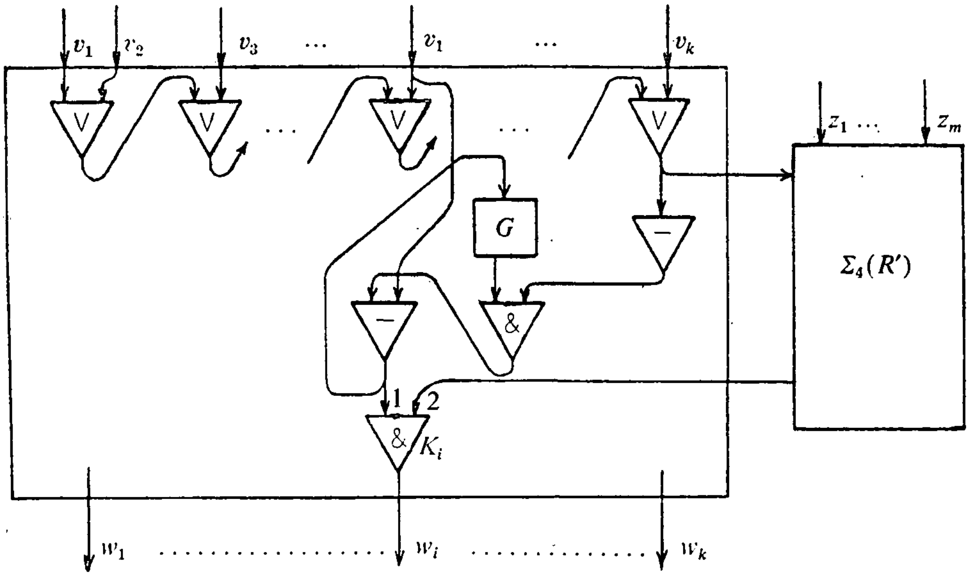

Fig. 16.

element K_i is joined to the output of the element z_i, $i=1,\dots,k$. The input v_i of this scheme is joined either to the same element, or the input of $\Sigma_4(\tilde{R})$, as well as the input $\tilde{z}_0$ of the block B_i. The input w_i is joined to the inputs of the same elements or the output of the scheme $\Sigma_4(\tilde{R})$, as well as the output of the block B_i; $i=1, \dots, k$. At the initial instant of time, states of the dalay element $z_1', \dots, z_k$ of the scheme W equal 0. When 1 arrives to the input v_i, the block $\Sigma_4'(R')$ begins to participate in the functioning of the scheme $\Sigma_4(\tilde{R})$ as a block B_i, while the delay element z_i turns into state 1 and others into state 0. The complexity of the scheme W is $5k+L(\Sigma_4'(R'))$ and therefore, applying the described process of exchange of block $B_1, \dots, B_k$ to the scheme W is useful only in case that $5K+L(\Sigma_4'(R'))\leq k\cdot L(\Sigma_4'(R'))$, i.e., when $L(\Sigma_4'(R'))\geq\frac{5k}{k-1}$.

АЛЕКСАНДАР СЕРГЕЕВИЧ ПОДКОЛЗИН
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. М. В. ЛОМОНОСОВА
МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ
СССР, 117234 МОСКВА, В-234, ЛЕНИНСКИЕ ГОРЫ, МГУ

ŠĆEPAN UŠĆUMLIĆ
UNIVERSITET U BEOGRADU
TEHNOLOŠKO-METALURŠKI FAKULTET
KATEDRA ZA MATEMATIČKE NAUKE
JUGOSLAVIJA, 11000 BEOGRAD, KARNEDŽIJEVA 4

Bibliography

[1] Кудрявцев, В. Б., Алёшин, С. В., Подколзин, А. С.: Элементы теории автоматов, Москва, *Изд-во Моск. университета*, 1978.

[2] Клини, С. К.: Представление событий в нервных сетях и конечных автоматах, *В книге «Автоматы»*, Москва, ИЛ, 1956, стр. 15—67.

[3] Кудрявцев, В. Б., Подколзин, А. С., Ушчумлич, Ш. М.: Введение в теорию абстрактных автоматов, Москва, *Изд-во Моск. университета*, 1985.

[4] Глушков, В. М.: Синтез цифровых автоматов, Москва, *Физмат.*, 1962.

[5] Копи, И. М., Элгот, К. С., Райт, Д. Б.: Реализация событий логическими сетями. *В книге Кибернетический сборник*, Вып. 3, М. ИЛ, 1961, 147—166.

[6] Гринберг, В. С.: Детерминизация систем графов и синтез конечных автоматов, *Сиб. матем. журнал*, Том 7, №. 6, 1966, 1260—1267.

(*Received Dec. 8, 1987*)