Tanulmányok

63/1977

MTA Számítástechnikai és Automatizálási Kutató Intézet Budapest

ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
И АВТОМАТИЗАЦИИ ВЕНГЕРСКОЙ АКАДЕМИИ НАУК

Д О К Л А Д Ы С Е М И Н А Р А

проводимого 20-21 мая 1976 г. в Риге
во время совещания по теме 1-15.1.

"Разработка общей теории автоматов"

Будапешт
1977

A kiadásért felel:

DR VÁMOS TIBOR

ISBN 963 311 043 2

Készült az
ORSZÁGOS MŰSZAKI KÖNYVTÁR ÉS DOKUMENTÁCIÓS KÖZPONT
Sokszorosító üzemében, Budapest, VIII., Reviczky u. 6.
F. v.: Janoch Gyula

С О Д Е Р Ж А Н И Е

E L Ő S Z Ó

A KGST keretében müködő "Automaták általános elméletének kidolgozása" /1-15.1/ szakbizottság 1976. május 18-23 között a Szovjetunióban tartotta ülését az LTA Elektronikai és Számitástechnikai Intézete szervezésében. A szakbizottsági ülések a résztvevő országok munkabeszámolóinak és a következő időszakra vonatkozó munkatervnek a megvitatása után mint kollokvium folytatódtak, ahol a témához kapcsolódó, az egyes országokban elért legujabb eredményekről és a jövőbeni elképzelésekről előadások hangzottak el. Az eddigi gyakorlat szerint az elhangzott előadásokról kiadvány készül. Jelen kiadvány a második.

ПРОБЛЕМЫ ИСПЫТАНИЙ И ДИАГНОСТИКИ СМОНТИРОВАННЫХ ПЕЧАТНЫХ ПЛАТ НА НЕКОТОРОЙ ФАЗЕ ЭЛЕКТРОННОГО ПРОЕКТИРОВАНИЯ ПРИ ПОМОЩИ ЭВМ

К. ПАСТОР, И. ИВИЧ

Исследовательский Институт Вычислительной Техники и Автоматизации Венгерской Академии Наук

Введение.

В будущем нам бы хотелось работать над этой важной и актуальной проблемой электронного проектирования при помощи ЭВМ, и в связи с этим познакомить вас с нашими представлениями о ее решении, и несколькими вытекающими из нее проблемами, которые уже были решены или будут решены.

1. В мире сложилась проблема автоматизации испытаний, точнее тест-диагностики, более сложных электронных сетей с помощью одиночных устройств и систем управляемых малой или большой ЦВМ.

Учитывая наши сегодняшние возможности, мы считаем осуществимой ниже описанную систему.

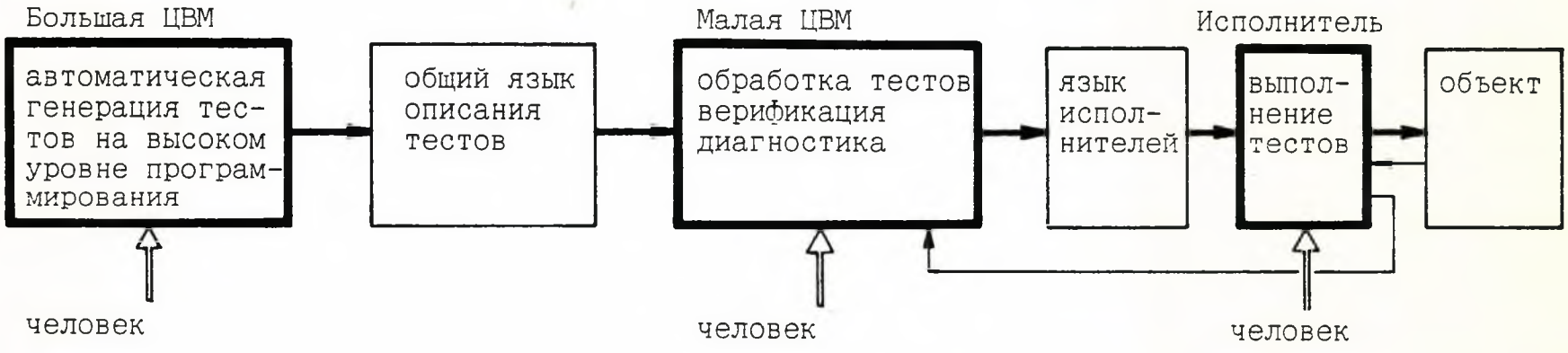
Требования к системе:
основным является ее общность

1.1 Многоуровневое построение и доступ к отдельным уровням:

- большая ЦВМ;
- малая ЦВМ;
- ручной ввод в исполнитель;

с точки зрения режима диалога:

- "малая ЦВМ - исполнитель";
- "человек - малая ЦВМ - исполнитель".


Большая ЦВМ
автоматическая генерация тестов на высоком уровне программирования
общий язык описания тестов
Малая ЦВМ
обработка тестов верификация диагностика
язык исполнителей
Исполнитель
выполнение тестов
объект
человек
человек
человек

С обеспечением возможности создания - на уровнях доступа - гибкого, расширяемого-сужаемого входного языка.

1.2 Возможность подключения любого типа исполнителя.

1.3 Гибкость системы. Она заключается в возможности расширяемости ее т.е. системы созданные на различных уровнях не требуют отдельной конструкции, а включаются иерархичести или по другому, подмножества входного языка соответствуют различным исполнителям.

1.4 Возможность встроения системы в проектирующую-испытывающую- производящую систему, управляемой ЦВМ.

2. На основании этих требований ясно, что фундаментальным средством для осуществления системы требуется создание общей базы данных ее и соответственно программное обеспечение для обработки этой общей базы данных.

Для этой цели была разработана система обработки общей базы данных с помощью списковой структуры, ориентированная на проектирование при помощи ЭВМ, в нескольких вариантах.

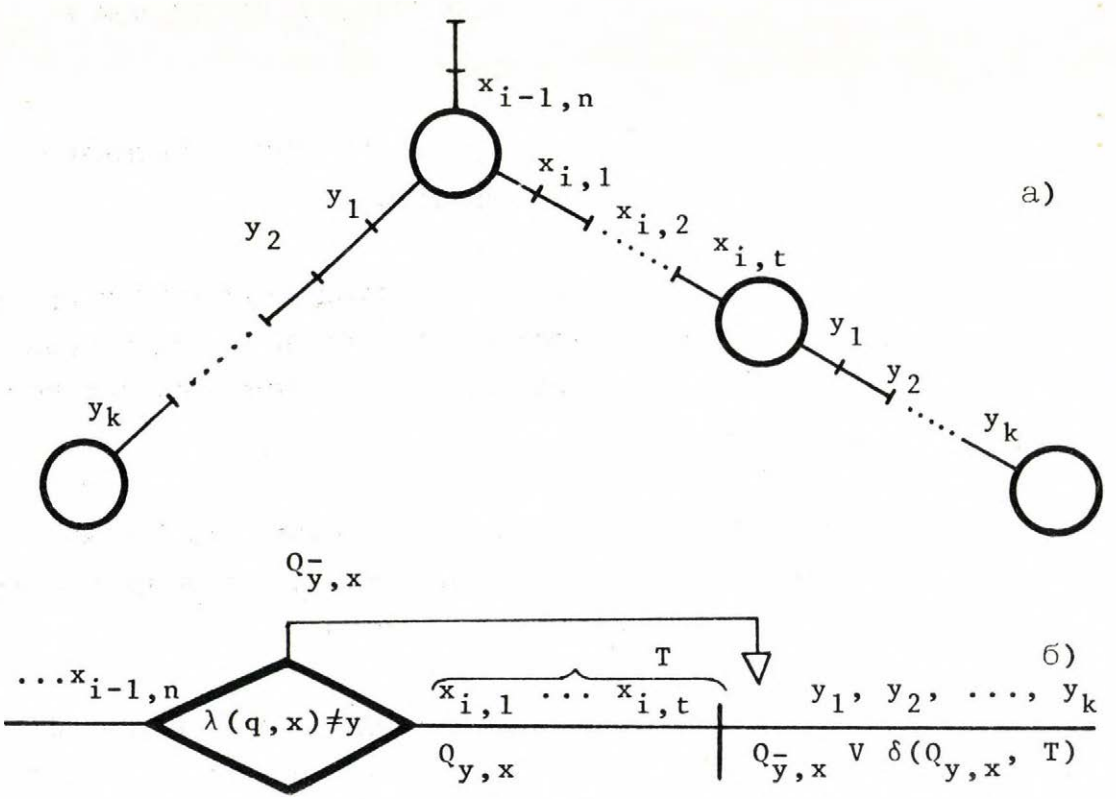
Одна из основных целей создания этой системы, обеспечение обмена информацией между частями проектирующей системы, а также решение проблем связанных с объемом памяти машины.

3. В области испытаний и диагностики решены ряд общих задач, и ряд специальных задач, как то комбинационная схема, одиночные логические ошибки.

Новейшие проблемы вызывают большие размеры схем и трудности, связанные с последовательностными схемами.

Одну такую проблему представляет генирация синхронизирующих последовательностей. Эти последовательности называются линейными, если не зависят от значения выходного сигнала автомата.

Последовательность считается разветвляющейся или адаптивной, если в зависимости от значения выходного сигнала, разветвляется на несколько ветвей. Устройства управляемые перфолентой могут обрабатывать только линейные последовательности. Разветвляющиеся последовательности можно преобразовать в квази-линейные, если мы можем обеспечить то, что некоторая ветвь, выходящая из узла разветвления, заканчивается на другой ветви, исходящей из того же узла разветвления /рис. а, б/.



Мы разработали алгоритм, выполняющий квази-линеаризацию.

А/ Пусть предположим, что состояние q автомата, является произвольным элементом подмножества Q состояний.

Алгоритм генерирует такой входной сигнал X, при котором численность множества состояний $Q' = \delta(Q,X)$, достижимых из множества состояний Q будет меньше и это продолжается до того, пока численность $Q'=1$.

Если такого входного сигнала нет, т.е. численность множества $Q'=\delta(Q,x)$, достижимого из множества Q не уменьшается, то значение выходного сигнала дает возможность для уменьшения числа состояний, например по следующему способу.

В/ 1. Выбираем состояние $a\in Q$, такое, что $\lambda(a,x)=y$, пусть подаем на вход автомата сигнал x.

2/ Если значение выходного сигнала y, то генерируем такую входную последовательность T, которая переводит состояние a в такое состояние $r\in Q$, для которого $\lambda(r,x)\neq y$

3/ Генерируем следующий элемент синхронизирующей последовательности, предполагая множество состояний $R = (Q_{y,x}^{-} \vee \delta(Q_{y,x},T))$ численность которого хотя бы на единицу меньше численности множества Q по сказанному в пункте А.

$$\lambda(q,x) \quad \begin{matrix} = y, \text{ если } q\in Q_{y,x} \\ \neq y, \text{ если } q\in Q_{y,x}^{-} \end{matrix}$$

Если кроме выполнения условий рассмотренного алгоритма имеется возможность при заданной входной последовательности для итерации заданного входного сигнала, то мы можем существенно упростить часть алгоритма обеспечивающую разветвление по ветвям алгоритма. Упрощение мы можем осуществить с помощью специального циклового анализа граф-состояний автомата, в ходе чего мы ищем длину цикла, полученного при интерации заданного входного сигнала, т.е. мы ищем решение соотношения $r=\delta(r,x^n)$ по n при заданном входном сигнале x.

При машинной реализации такого типа алгоритмов дальнейшие проблемы возникают из-за объема памяти и больших затрат времени, что в этом случае вынуждает использование функции переходов состояний автомата вместо таблицы переходов.

Машинное обслуживание и имитация свойств, вытекающих из машинного обслуживания и технологических данных сложных схем, влечет за собой, в частности, решение - при новых условиях - ряда ранее решенных проблем и, в частности, появление новых проблем. Далее нам бы хотелось это проиллюстрировать несколькими примерами.

При совместном упрощении неполностью определенных функций в том случае, если корреллируют области неопределенности компонентных функций /т.е. функции реализованного вида остальных компонентных функций/.

Не решен ряд проблем описания, например, описание в закрытой форме элементов, имеющих специальные входы /установка, гашение, перезапись и т.д./ Этот круг вопросов направляет нас на то, чтобы более гибко применять временный параметр.

Дать функциональное описание элементов схемы таким образом, что бы в этом выражалось и их временное поведение. Это является актуальной темой для исследования в мире. Полезность темы не оспорима /например: моделирование и исследования состязаний/. В ходе этих исследований, описание временного характера решено дискретизацией времени, которое для схемы означает то, что мы разделим ось времени на довольно маленькие интервалы, на основании данных каталога для отдельных составляющих элементов схемы и в то же время мы не решим проблемы общего подхода элементов, управляемых фронтами сигналов и уровнями сигналов. Например, по этой теме выступил с докладом Сифакис на симпозиуме ИФАК в 1974 г.

Очевидно, что высказанные проблемы не только наши проблемы, вероятно известно кроме этих проблем еще ряд решаемых задач. Нашей целью явилось не стремление к полноте, а ознакомление с несколькими проблемами, и мы надеемся, что решение их реализуется или о возможных достигнутых решений мы получим сведения.

MULTITAPE AUTOMATA AND LANGUAGES

Peter H. Starke
Sektion Mathematik
der Humboldt-Universität
DDR-1086 Berlin, PSF 1297

In this paper we give a review of some of the recent results achieved when multitape (resp. multihead) automata are considered as accepting devices for languages. The study of such devices was begun by RABIN and SCOTT (5) and continued by ROSENBERG et al. (2), (6) motivated by the fact that the accepting capacity of multitape resp. multihead automata is high compared with the simplicity of these devices.

1. Multitape automata

In this section we introduce the basic notions and notations and review some of the main results on multitape automata.

Let be X a finite nonempty alphabet and $n \geq 2$ a natural number. The set of all n-tuples $\underline{p} = (p_1,\ldots,p_n)$ of words $p_j \in W(X)$ is denoted by $\times_n W(X)$. Catenation is defined componentwise, i.e.

$$(p_1,\ldots,p_n)(q_1,\ldots,q_n) = (p_1q_1,\ldots,p_nq_n),$$

thus, $\times_n W(X)$ forms a semigroup with identity $\underline{e} = (e,\ldots,e)$ which is obviously not free. For $\underline{p} = (p_1,\ldots,p_n)$ let

$$\nu(\underline{p}) = \{ i \mid 1 \leq i \leq n \wedge p_i \neq e \}.$$

Moreover we put

$$X_n = \bigtimes_n (X \cup \{e\}) \setminus \{\underline{e}\}.$$

Clearly, X_n and

$$X_n^1 = \{\underline{x} \mid \underline{x} \in X_n \wedge \mathrm{card}(\nu(\underline{x})) = 1\}$$

are sets of generators for $\bigtimes_n W(X)$. Finally we put for $\underline{p} \in \bigtimes_n W(X)$, $1 \leq j \leq n$, $\underline{p} = (p_1, \ldots, p_n)$

$$(\underline{p})_j = p_j \quad \text{and} \quad l(\underline{p}) = \sum_{j=1}^{n} l(p_j)$$

whereby $l(p)$ is the length of p.

Definition.

The system $\underline{A} = (n, X, Z, \tau, f, Z_1, M)$ is said to be a (weakly-initial) nondeterministic n-tape automaton (shortly: ND-n-TA) if

a) $n \geq 2$ is a natural number (of tapes)

b) X and Z are finite nonempty sets (of inputs and states)

c) τ is a function from Z into $\underline{P}^+(\{1, \ldots, n\})$, the set of all nonempty subsets of $\{1, \ldots, n\}$

d) f is a function from $Z \times X_n$ into $\underline{P}(Z)$ such that for $z \in Z$, $\underline{x} \in X_n$ it holds

$$f(z, \underline{x}) \neq \emptyset \longleftrightarrow \nu(\underline{x}) = \tau(z)$$

e) $\emptyset \neq Z_1 \subseteq Z$, $M \subseteq Z$.

Interpretation. The input of $\underline{A}$ consists of n tapes (numbered by $1, \ldots, n$) with a one-way (left to right) read-only head on each of it. If at time t the system has the state $z \in Z$ it first computes the nonempty set $\tau(z)$. Then exactly those heads working on tapes with numbers in $\tau(z)$ will scan one letter and move one square to the right. The result of the scanning procedure is a $\underline{x} \in X_n$ with $\nu(\underline{x}) = \tau(z)$. Now $\underline{A}$ chooses a state z' from $f(z, \underline{x})$ and goes to state z' at time $t+1$.

If $\underline{A}$ is started within a state from Z_1 with a n-word $\underline{p}$ on its

tapes and if there is at least one way to reach a state from M reading the whole n-word $\underline{p}$ then $\underline{p}$ is accepted by $\underline{A}$, i.e. $\underline{p}$ is an element of the n-ary relation $R(\underline{A})$ represented by $\underline{A}$. Formally

$$R(\underline{A}) = \{\underline{p} \mid \underline{p} \in \bigtimes_n W(X) \wedge \bar{f}(Z_1,\underline{p}) \cap M \neq \emptyset\},$$

where for $Z' \subseteq Z$, $\underline{p} \in \bigtimes_n W(X)$

$$\bar{f}(Z',\underline{p}) = \{z_{m+1} \mid \exists m \exists z_1 \ldots \exists z_m \exists \underline{x}_1 \ldots \exists \underline{x}_m (0 \leq m \in nz \wedge z_1 \in Z' \wedge \wedge \underline{x}_1 \ldots \underline{x}_m = \underline{p} \wedge \bigwedge_{j=1}^{m} z_{j+1} \in f(z_j,\underline{x}_j))\}.$$

The automaton $\underline{A}$ is called initial, if $Z_1 = \{z_1\}$ is a singleton and deterministic (D-n-TA) if moreover alle the sets $f(z,\underline{x})$ are empty or singletons. In this case the transition function f can be described by a function δ which uniquely maps

$$D_\delta = \{(z,\underline{x}) \mid z \in Z \wedge \underline{x} \in X_n \wedge \tau(z) = \nu(\underline{x})\}$$

into Z such that

$$f(z,\underline{x}) = \{\delta(z,\underline{x})\}$$

for all $(z,\underline{x}) \in D_\delta$.

For n-ary relations $R,R' \subseteq \bigtimes_n W(X)$ catenation and catenation closure are defined by

$$R \cdot R' = \{\underline{pq} \mid \underline{p} \in R \wedge \underline{q} \in R'\}$$

$$\langle R \rangle = \{\underline{e}\} \cup \{\underline{p}_1 \ldots \underline{p}_m \mid m > 0 \wedge \bigwedge_{i=1}^{m} \underline{p}_i \in R\}.$$

Thus, in a natural way, $\underline{R}_n$, the set of all n-regular relations is defined as the least class of relations containing all the finite relations (i.e. at least the relations $\emptyset$, $\{\underline{x}\}$ for $\underline{x} \in X_n^1$) and closed under union, catenation and catenation closure.

It is an interesting fact that a n-ary relation $R \subseteq \bigtimes_n W(X)$ is representable by a ND-n-TA iff it is n-regular (cf. (1),(4),(8)). But in constrast to the "classical" case n=1, for $n \geq 2$ the set

$\underline{R}_n$ is not a BOOLEan algebra of sets, since $\underline{R}_n$ is neither closed under intersection nor under complementation. The only positive result in that direction says that the complement

$$\bar{R} = \bigtimes_n W(X) \setminus R$$

of a relation R representable by a deterministic n-TA is always n-regular, i.e. representable by a ND-n-TA. On the other hand, the class $\underline{R}_d^{(n)}$ of all n-ary relations representable by D-n-TA is neither closed under union, nor under catenation nor under catenation closure, i.e. $\underline{R}_d^{(n)}$ is not a KLEENEan algebra (cf. (12)).

This it makes obvious that $\underline{R}_d^{(n)} \subset \underline{R}_n$ and a characterization for $\underline{R}_d^{(n)}$ is needed. Such a characterization is given in the paper (9) by means of the language of regular expressions over the alphabet $Y_n = X \times \{1,\ldots,n\}$ whereby the letter (x,i) is used to describe the elementary relation

$$\underline{x}(x,i) = \{(e,\ldots,e,\underset{i}{x},e,\ldots,e)\}.$$

Then each regular expression T over Y_n describes a n-regular relation and vice versa, and, a decidable property of regular expressions is given such that a relation R is representable by a D-n-TA iff it is describable by a regular expression having that property (cf. (8),(9)).

Let us remark that the behavior of infinite n-tape automata (not considered here) is characterized for the deterministic case in (9); obviously each relation is representable by a infinite ND-n-TA.

Finally we consider the common decision problems for n-tape automata. It has been shown (cf. (2),(12)) that most of them are unsolvable. In detail, if X is not a singleton, only the

emptiness problem ($R(\underline{A}) = \emptyset$?) and the finiteness problem (Is $R(\underline{A})$ finite ?) are solvable while the universe problem ($R(\underline{A}) = \bigtimes_n W(X)$?), the co-finiteness problem (Is $\overline{R(\underline{A})}$ finite ?), the disjointness problem ($R(\underline{A}_1) \cap R(\underline{A}_2) = \emptyset$?) and the containment problem ($R(\underline{A}_1) \subseteq R(\underline{A}_2)$?) are all unsolvable even for D-n-TA. The equivalence problem ($R(\underline{A}_1) = R(\underline{A}_2)$?) is still open for D-n-TA and is unsolvable for ND-n-TA. For autonomous ND-n-TA (i.e. $X = \{x\}$ is a singleton) all the problems are solvable (cf. (13)).

2. Languages

In this section we consider the relation between languages on the one hand and n-regular relations on the other hand. First we have

Theorem 1 (see (1))

A relation R is n-regular iff there exist a finite alphabet Y, a regular language L over Y and (alphabetic) homomorphisms $h_1,\ldots,h_n : Y \to X \cup \{e\}$ such that

$$R = \{(h_1(q),\ldots,h_n(q)) \mid q \in L\}.$$

The theorem can be generalized to the case when regular substitutions $\sigma_1,\ldots,\sigma_n : Y \to \mathcal{L}_3(X)$ are used in place of $h_1,\ldots,h_n$.

This theorem is a useful tool in the theory of n-regular relations but it gives no insight how to use n-tape automata to accept (i.e. to decide upon) languages. The first and very natural way is to consider the case when the relation R itself can be considered as a language, i.e. as a subset of a free

semigroup. This is the case, e.g., when R is synchronous, i.e.

$$\forall \underline{q} \forall i \forall j (\underline{q} \in R \wedge 1 \leq i,j \leq n \longrightarrow l((\underline{q})_i) = l((\underline{q})_j))$$

since then R is a subset of $W(\underset{n}{\asymp} X)$ which is a free semigroup. But applying ND-n-TA to synchronous relations we get nothing new:

Theorem 2 (see (4))

A synchronous n-ary relation R over W(X) is n-regular iff R is a regular language over $\underset{n}{\asymp} X$.

Another possibility to relate a language with a relation consists in the study of the projections

$$P_i(R) = \{ p \mid \exists p_1 \ldots \exists p_{i-1} \exists p_{i+1} \ldots \exists p_n (p_1,\ldots,p_{i-1},p,p_{i+1},\ldots,p_n) \in R \} .$$

But again we get nothing new:

Theorem 3 (see (2),(5))

If R is a n-regular relation over W(X) then all the projections $P_i(R)$ are regular languages over X.

Let us finally consider diagonalization. By the diagonal D(R) of the relation R we understand the language

$$D(R) = \{p \mid (p,\ldots,p) \in R\}.$$

It is easy to see (cf. (11)) that the binary relation

$$R = \{(0^m10^n10^k, 0^l10^m10^n) \mid k,l,m,n \geq 0\}$$

is representable by a deterministic two-tape automaton, thus R is 2-regular, but

$$D(R) = \{0^m10^m10^m \mid m \geq 0\}$$

is a context-sensitive language which is not context-free. Therefore we shall study the classes $\mathcal{L}_d^{(n)}$ resp. $\mathcal{L}_{nd}^{(n)}$ of all languages L such that there is a deterministic resp. nondeter-

ministic n-tape automaton $\underline{A}$ with $L = D(R(\underline{A}))$.

Considering only the diagonals of the relations represented by n-tape automata we are able to simplify the interpretation of the tuple $\underline{A} = (n,X,Z,\tau,f,Z_1,M)$. We can imagine $\underline{A}$ as the description of a system operating with n independently running one-way read-only heads on one tape in the manner described above. In the beginning all the heads are concentrated on the first square and necessary for acception is that all the heads are assembled again on the square just right of the last letter of the word. Under this interpretation $\underline{A}$ is called a n--head automaton (cf. (6)). In the sequel we review some of the results achieved in the study of n-head automata.

Theorem 4 (see (11), (6))
If R is a n-regular relation then D(R) is a deterministic context-sensitive language.

The converse is not true,

$$L = \{p\overleftarrow{p} \mid p \in W(X)\}$$

(where $\overleftarrow{p}$ denotes the mirror image of p) is a context-free (hence, deterministic context-sensitive) language which is in $\mathcal{L}_{nd}^{(n)}$ for no integer n.

If we restrict ourselfs to an one-letter alphabet we obtain the stronger result

Theorem 5 (see (11))
If $X = \{x\}$ then $\mathcal{L}_{nd}^{(n)} = \mathcal{L}_3$ for all $n = 1,2,\ldots$

where $\mathcal{L}_3$ denotes the class of all regular languages over X.

In the general case ($\mathrm{card}(X) \geq 2$, $n \geq 2$) we have

Theorem 6 (see (13))

$$\mathcal{L}_3 \subset \mathcal{L}_d^{(n)} \subset \mathcal{L}_{nd}^{(n)} \subset \mathcal{L}_1.$$

Let us remark that the classes $\mathcal{L}_d^{(n)}$ and $\mathcal{L}_{nd}^{(n)}$ really depend on n:

Theorem 7 (see (6),(13))

a) $\mathcal{L}_d^{(n)} \subset \mathcal{L}_d^{(n+1)}$ b) $\mathcal{L}_{nd}^{(n)} \subset \mathcal{L}_{nd}^{(n+1)}$

This follows from the fact that if R is a n-ary relation representable by a D-n-TA (resp. by a ND-n-TA) then

$$R' = \{(p_1,\ldots,p_n,p_{n+1}) \mid (p_1,\ldots,p_n) \in R \wedge p_n = p_{n+1}\}$$

is representable by a D-(n+1)-TA (resp. ND-(n+1)-TA). In (6) and (13) languages L_n are given which are in $\mathcal{L}_d^{(n+1)}$ but not in $\mathcal{L}_{nd}^{(n)}$. Moreover, the language

$$L'' = \{ psp \mid p,s \in W(\{o,1\}) \wedge p \neq e \}$$

is an element of $\mathcal{L}_{nd}^{(2)} \setminus \mathcal{L}_d^{(n)}$ for all n, hence, the two hierarchies of Theorem 7 are incomparable.

The results concerning the closedness properties of $\mathcal{L}_d^{(n)}$, $\mathcal{L}_{nd}^{(n)}$ and

$$\mathcal{L}_d^* = \bigcup_{i\geq 1} \mathcal{L}_d^{(i)}, \qquad \mathcal{L}_{nd}^* = \bigcup_{i\geq 1} \mathcal{L}_{nd}^{(i)}$$

are given by the following table (see next page). In addition to that table let us remark that it is still open whether one or the other of these classes is closed under catenation or catenation closure. We shall give a brief sketch of the proof.

- intersection with regular languages

That all the classes under consideration are invariant under

Theorem 8	$\mathcal{L}_d^{(n)}$	$\mathcal{L}_d^*$	$\mathcal{L}_{nd}^{(n)}$	$\mathcal{L}_{nd}^*$
intersection with regular languages	yes	yes	yes	yes
union	no	no	yes	yes
intersection	no	yes	no	yes
complementation	no	no	no	?
marked catenation	yes	yes	yes	yes
marked catenation closure	yes	yes	yes	yes
e-free homomorphisms	no	no	no	no
inverse homomorphisms	yes	yes	yes	yes
left-quotient with regular languages	no	no	no	no

under intersection with regular languages is nearly obvious since it is possible to feed additionally all the symbols the n-TA $\underline{A}$ scans on its first tape into an ordinary (one-tape) automaton representing the regular language L under consideration. The whole system is an n-TA $\underline{A}'$ with $D(R(\underline{A}')) = D(R(\underline{A})) \cap L$ if acception by $\underline{A}'$ means that both automata, $\underline{A}$ and the one-tape automaton, accept.

- union

From the non-closedness of $\mathcal{L}_d^*$ under union the non-closedness of $\mathcal{L}_d^{(n)}$ by Theorem 7 follows. It is easy to see that

$L_1 = \{0^k 1 0^k \mid k \geq 0\} \in \mathcal{L}_d^{(2)}$, and,

$L_2 = \{0^k 1 0^{2k} \mid k \geq 0\} \in \mathcal{L}_d^{(2)}$.

We show that $L_1 \cup L_2 \notin \mathcal{L}_d^*$.

Assume the contrary and let be $\underline{A} = (n,\{0,1\},Z,\tau,\delta,z_1,M)$ a D-n-TA with $D(R(\underline{A})) = L_1 \cup L_2$. Then for all $k \geq 0$ the state $\bar{\delta}(z_1,(0^k10^k,\ldots,0^k10^k))$ is defined and an element of M. Since Z is finite there are numbers i,j with $i < j$ and

$$z_i' = \bar{\delta}(z_1,(0^i10^i,\ldots,0^i10^i)) = \bar{\delta}(z_1,(0^j10^j,\ldots,0^j10^j)) = z_j'.$$

From $0^i10^{2i} \in L_2$ we obtain that $\bar{\delta}(z_1,(0^i10^{2i},\ldots,0^i10^{2i}))$ is defined and an element of M. Since $\bar{\delta}(z_1,(0^i10^i,\ldots,0^i10^i))$ is defined it holds

$$\begin{aligned}\bar{\delta}(z_1,(0^i10^{2i},\ldots,0^i10^{2i})) &= \bar{\delta}(z_i',(0^i,\ldots,0^i))\\ &= \bar{\delta}(z_j',(0^i,\ldots,0^i))\\ &= \bar{\delta}(z_1,(0^i10^{i+j},\ldots,0^i10^{i+j})).\end{aligned}$$

Therefore, $\bar{\delta}(z_1,(0^i10^{i+j},\ldots,0^i10^{i+j}))$ is defined and an element of M contradicting $0^i10^{i+j} \notin L_1 \cup L_2$.

- intersection

The proof that $\mathcal{L}_d^{(n)}$ and $\mathcal{L}_{nd}^{(n)}$ are not closed under intersection is contained in (13).

The closedness of $\mathcal{L}_d^*$ and $\mathcal{L}_{nd}^*$ under intersection is seen as follows. Let be $L = D(R)$, $L' = D(R')$ the languages under consideration. By Theorem 7 we can assume that R and R' are both n-ary relations for some integer n. Let be $\underline{A} = (n,X,Z,\tau,f,Z_1,M)$, $\underline{A}' = (n,X,Z',\tau',f',Z_1',M')$ two n-tape automata with $R = R(\underline{A})$, $R' = R(\underline{A}')$ and which are deterministic if $L,L' \in \mathcal{L}_d^*$. Now consider

$$\underline{A} \times \underline{A}' = (2n,X,Z \times Z',\tau^\times,f^\times,Z_1 \times Z_1',M \times M')$$

with

$$\tau^\times((z,z')) = \tau(z) \cup \{j+n \mid j \in \tau'(z')\}$$

and

$$f^{\times}((z,z'),(\sigma_1,\ldots,\sigma_n,\sigma_{n+1},\ldots,\sigma_{2n})) =$$
$$= f(z,(\sigma_1,\ldots,\sigma_n)) \times f'(z',(\sigma_{n+1},\ldots,\sigma_{2n})).$$

Obviously, $\underline{A} \times \underline{A}'$ is deterministic if $\underline{A}$ and $\underline{A}'$ are. One shows

$$D(R(\underline{A} \times \underline{A}')) = D(R(\underline{A})) \cap D(R(\underline{A}')) = L \cap L'.$$

- complementation

If $\mathcal{L}_d^{(n)}$ is closed under complementation then $\mathcal{L}_d^*$ is, thus, it is sufficient to prove that $\mathcal{L}_d^*$ is not closed under complementation. Obviously,

$$L = \{0^k 1 0^k \mid k \geq 0\} \in \mathcal{L}_d^{(2)}.$$

Now assume that $\bar{L} = W(\{0,1\}) \setminus L$ is an element of $\mathcal{L}_d^*$, then

$$L' = \bar{L} \cap \langle\{0\}\rangle \cdot \{1\} \cdot \langle\{0\}\rangle = \{0^k 1 0^l \mid k,l \geq 0 \wedge k \neq l\} \in \mathcal{L}_d^*.$$

Now let be $\underline{A} = (n,\{0,1\},Z,\tau,\delta,z_1,M)$ a D-n-TA with $D(R(\underline{A})) = L'$. Then for all $k > 0$ we have $0^k 1 \in L'$ thus, the state $\bar{\delta}(z_1,(0^k 1,\ldots,0^k 1))$ is defined and an element of M. Since Z is finite there are numbers i,j with $0 < i < j$ and

$$z_i^+ = \bar{\delta}(z_1,(0^i 1,\ldots,0^i 1)) = \bar{\delta}(z_1,(0^j 1,\ldots,0^j 1)) = z_j^+.$$

From $i < j$ we obtain that $\bar{\delta}(z_1,(0^i 1 0^j,\ldots,0^i 1 0^j))$ is defined and an element of M. Since $\bar{\delta}(z_1,(0^i 1,\ldots,0^i 1)) = z_i^+$ is defined it holds

$$\begin{aligned}\bar{\delta}(z_1,(0^i 1 0^j,\ldots,0^i 1 0^j)) &= \bar{\delta}(z_i^+,(0^j,\ldots,0^j)) \\ &= \bar{\delta}(z_j^+,(0^j,\ldots,0^j)) \\ &= \bar{\delta}(z_1,(0^j 1 0^j,\ldots,0^j 1 0^j)).\end{aligned}$$

Hence $\bar{\delta}(z_1,(0^j 1 0^j,\ldots,0^j 1 0^j))$ is defined and an element of M, i.e. $0^j 1 0^j \in D(R(\underline{A}))$, which is in contradiction with $D(R(\underline{A})) = L'$.

That $\mathcal{L}_{nd}^{(n)}$ is not closed under complementation is a consequence of its closedness under union, its non-closedness under

intersection and the DeMORGAN laws.

- marked catenation, marked catenation closure

Obviously it suffices to prove the closedness of $\mathcal{L}_d^{(n)}$ and $\mathcal{L}_{nd}^{(n)}$, i.e. to show that

$$L\cdot\{c\}\cdot L', \langle L\cdot\{c\}\rangle \in \mathcal{L}_d^{(n)} \text{ resp. } L\cdot\{c\}\cdot L', \langle L\cdot\{c\}\rangle \in \mathcal{L}_{nd}^{(n)}$$

if $L,L' \in \mathcal{L}_d^{(n)}$ resp. $L,L' \in \mathcal{L}_{nd}^{(n)}$, where c is a letter not contained in X.

Let be R,R' n-regular relations over W(X) with L = D(R) and L' = D(R'). Then $R\cdot\{(c,\ldots,c)\}\cdot R'$ and $\langle R\cdot\{(c,\ldots,c)\}\rangle$ are n-regular again and

$$D(R\cdot\{(c,\ldots,c)\}\cdot R') = L\cdot\{c\}\cdot L',\ D(\langle R\cdot\{(c,\ldots,c)\}\rangle) = \langle L\cdot\{c\}\rangle$$

From this the proposition on $\mathcal{L}_{nd}^{(n)}$ follows. To prove the proposition on $\mathcal{L}_d^{(n)}$ one shows that $R\cdot\{(c,\ldots,c)\}\cdot R'$ and $\langle R\cdot\{(c,\ldots,c)\}\rangle$ are representable by deterministic n-tape automata if R,R' are.

- e-free homomorphisms

In (11) it is shown that the language

$$L = \{ x_1x_2\bar{x}_1x_3\bar{x}_2\bar{x}_1x_4\bar{x}_3\bar{x}_2\bar{x}_1\ldots x_n\bar{x}_{n-1}\ldots\bar{x}_1x_nx_{n-1}\ldots x_1 \mid n\geq 2 \wedge \bigwedge_{i=1}^{n} x_i \in \{a,b\} \}$$

over the alphabet $\{a,\bar{a},b,\bar{b}\}$ is an element of $\mathcal{L}_d^{(2)}$. Now consider the e-free homomorphism h with

$$h(a) = a,\ h(b) = b,\ h(\bar{a}) = h(\bar{b}) = c.$$

We obtain

$$h(L) = \{ x_1x_2cx_3c^2x_4c^3\ldots x_nc^{n-1}x_nx_{n-1}\ldots x_1 \mid n\geq 2 \wedge \bigwedge_{i=1}^{n} x_i \in \{a,b\} \}$$

The same observations which show that

$$\{p\overleftarrow{p} \mid p \in W(\{a,b\})\} \notin \mathcal{L}_{nd}^{*}$$

will lead to the result that $h(L) \notin \mathcal{L}^*_{nd}$ (cf. (6)).

- inverse homomorphisms, left quotient

The proof given in (11) for the nondeterministic case applies to the deterministic case too.

From Theorem 8 we can conclude that all the classes $\mathcal{L}^{(n)}_d$, $\mathcal{L}^*_d$, $\mathcal{L}^{(n)}_{nd}$, $\mathcal{L}^*_{nd}$ considered as families of languages are pre-AFL's (cf. (7)).

Finally let us remark that all the common decision problems are unsolvable within the classes under considerartion (13).

3. References

(1) Berstel, J.: Memento sur les transductions rationelles. Inst. de Programmation, Univ. de Paris VI, N°IP 74-23 (1974).

(2) Fischer, P.C.; Rosenberg, A.L.: Multitape one-way nonwriting automata. J. Comp. Syst. Sci. 2 (1968) 88 - 101.

(3) Makarevski, A.; Stotskaja, E.V.: Representability in deterministic multitape automata. Kibernetika (Kiev) 1969, No. 4 (in russian).

(4) Mirkin, B.G.: On the theory of multitape automata. Kibernetika (Kiev) 1966, No. 5, 12 - 18 (in russian).

(5) Rabin, M.O.; Scott, D.: Finite automata and their decision problems. IBM J. Res. Devel. 3 (1959) 125 - 144.

(6) Rosenberg, A.L.: On multi-head finite automata. IBM J. Res. Devel. 10 (1966) 2, 61 - 75.

(7) Salomaa, A.: Formal Languages. Academic Press, New York 1973.

(8) Starke, P.H.: On the representability of relations by deterministic and nondeterministic multitape automata. Lecture Notes in Comp. Sci. 32 (1975) 114 - 124.

(9) Starke, P.H.: Über die Darstellbarkeit von Relationen in Mehrbandautomaten. Elektron. Informationsverarbeitung u. Kybernetik (EIK) 12 (1976) 1/2, 61 - 81.

(10) Starke, P.H.: Entscheidungsprobleme für autonome Mehrbandautomaten. Z. Math. Logik Grundlagen Math. 22 (1976) 131 - 140.

(11) Starke, P.H.: On the diagonals of n-regular relations. EIK 12 (1976) 6, 281 - 288. Correction forthcoming.

(12) Starke, P.H.: Closedness properties and decision problems for finite multitape automata. Kybernetika (Praha) 12 (1976) 2, 61 - 75.

(13) Starke, P.H.: Decision problems for multitape automata. To appear in Lecture Notes in Comp. Sci. (MFCS'76 Conf. Rec.).

(14) Stotskaja, E.V.: On deterministic multitape automata without endmarkers. Avtomatika i Telemehanika 9 (1971) 105 - 110 (in russian).

On Theory of state-identification experiments at finite non-deterministic automata

Hans-Dieter Burkhard, Berlin

The paper gives an abstract of some results in theory of state-identification experiments at finite non-deterministic automata. Diagnosing experiments for identification of the initial state and homing experiments for identification and control of the final state are considered. Experiments may be performed as preset or adaptive experiments. The existence of identification experiments is algorithmically decidable. If identification experiments exist, their length can be bounded and such experiments can be constructed.

1. Introduction

The starting point of the theory of identification experiments at automata is an "identification task": Given is an automaton (with a well-known structure) and the state of this automaton is to be identified with help of an experiment. Thereby an identification experiment is a suitable input sequence such that the experimentalist is able to determine the state of the automaton after observation of

the output word (which depends on the state to be identified). The main goals of the theory are algorithms for solving the existence and construction problems and also for solutions of certain complexity problems. By the existence problem is meant the problem of decision wether there exist identification experiments or not. More exactly, we consider a class $\mathfrak{K}$ of identification tasks $\mathfrak{F}$, thereby an identification task can be given by description of the automaton and the kind of experiments (which will be explained later on). Then the solution of the existence problem for $\mathfrak{K}$ is a recursive function (an algorithm) A from $\mathfrak{K}$ into the set $\{$yes, no$\}$ such that $A(\mathfrak{F}) =$ yes if identification experiments for $\mathfrak{F}$ exist and $A(\mathfrak{F}) =$ no otherwise.
The solution of the construction problem is a partial recursive function B out of the class $\mathfrak{K}$ into the set of experiments with domain A^{-1}(yes) such that $B(\mathfrak{F})$ is an identification experiment for $\mathfrak{F}$ if such experiments exist.
Note, that if the set of experiments is recursively enumerable and the set of identification experiments is recursively decidable (for each $\mathfrak{F}$ by universal methods for the whole class $\mathfrak{K}$), the construction problem can be solved by successive testing all experiments.
The solution of the complexity problem is a partial recursive function C out of $\mathfrak{K}$ into the set of complexity values such that $C(\mathfrak{F})$ is the complexity (for instance the length) of simplest identification experiments for $\mathfrak{F}$ if they exist.

The aim of this paper is to give descriptions of some solutions of these problems for experiments on finite non-deterministic automata. (For experiments on finite deterministic automata cf. /1/, /2/, /4/, /6/, /8/ - /11/.)

A finite synchronous total defined weakly-initial non-deterministic automaton (shortly NDA) is described by $\mathfrak{A} = [X, Y, Z, h, Z_0]$. Thereby X, Y, Z are finite non-empty sets (the sets of input symbols, outpu. symbols and internal states), h is a function from $Z \times X$ into the set $2^{Y \times Z} \setminus \emptyset$, (the set of all non-empty subsets of the set $Y \times Z$), and $Z_0 \subseteq Z$ describes the possible initial states of the automaton (one could also consider a set $\mathfrak{Z}_0 \subseteq 2^Z \setminus \emptyset$ of "non-deterministic" initial states).

The output function $v : Z \times X^* \to 2^{Y^*}$ is defined by

$$v(z, e) =_{Df} \{ e \}$$
$$v(z, xp) =_{Df} \bigcup_{z' \in Z} \{ y \,/\, [y, z'] \in h(z, x) \} \cdot v(z', p) ,$$

and the transition function with respect to the output is defined by

$$h_e(z, e) =_{Df} \{ z \}$$
$$h_{yq}(z, xp) =_{Df} \bigcup_{[y, z'] \in h(z, x)} h_q(z', p) ,$$

such that $h_q(z, p)$ denotes all states in which the NDA may be after starting in z, input p and output q (in the definitions we have made use of $z \in Z$, $x \in X$, $y \in Y$, $p \in X^*$, $q \in Y^*$, e the empty word). For functions which depend on Z we shall make use of extensions in 2^Z in the sense of union, i.e. for example $v(M, p) =_{Df} \bigcup_{z \in M} v(z, p)$ ($M \subseteq Z$, $p \in X^*$).

2. Diagnosing preset experiments

The aim of diagnosing experiments is the determination of the initial state, by a preset experiment is meant a fixed word. (For adaptive experiments , at which the input sequence is determined with respect to the output while the experiment is going on, see 4. of this paper. - We only consider simple experiments, a diagnosing multiple experiment may be considered as a set of simple experiments. For special methods due to diagnosing multiple preset experiments at NDA see /6/ .) For diagnosing experiments let be at least 2 states in Z_0.

(1) Definition

An input word $p \in X^*$ is a diagnosing preset experiment (d.p.e.) for $\mathfrak{Z} = [X, Y, Z, h, Z_0]$
iff there exists a function φ : $v(Z_0, p) \longrightarrow Z_0$
such that holds : $\forall z \forall q (z \in Z_0 \wedge q \in v(z,p) \longrightarrow \varphi(q) = z)$.

(2) Corollary

p is a d.p.e. for $\mathfrak{Z}$
iff $\forall z, z' \; (z, z' \in Z_0 \wedge z \neq z' \longrightarrow v(z,p) \cap v(z',p) = \emptyset)$.

In /6/, /7/, /12/ are also considered d.p.e. with the conditions $v(z,p) \neq v(z',p)$
and $v(z,p) \not\subseteq v(z',p) \wedge v(z',p) \not\subseteq v(z,p)$ respectively
instead of $v(z,p) \cap v(z',p) = \emptyset$ in Corollary (2).

For a given identification task $\mathfrak{Z}$ in the sense of diagnosing experiments as defined in (1) for a NDA $\mathfrak{A}$ we consider the set of all d.p.e. for $\mathfrak{A}$:

$$L_{\mathfrak{Z}} =_{Df} \{ p \,/\, p \in X^* \wedge p \text{ is d.p.e. for } \mathfrak{A} \}.$$

Then the following theorems hold:

(3) Theorem

(a) $L_{\mathfrak{Z}} = L_{\mathfrak{Z}} \cdot X^*$

(b) $L_{\mathfrak{Z}} \in \mathfrak{L}_3$ (CHOMSKY-Hierarchy)

(c) $e \notin L_{\mathfrak{Z}}$ (e denotes the empty word) .

(4) Theorem

For each $L \subseteq X^*$ which satisfies the conditions (a), (b), (c) of (3) there exists a NDA $\mathfrak{A}$ with $L_{\mathfrak{Z}} = L$.

(5) Theorem (on existence and construction problem)

For each NDA $\mathfrak{A}$ it is algorithmically decidable if there exists a d.p.e. (i.e. if $L_{\mathfrak{Z}} \neq \emptyset$) and such an experiment can be construsted if it exists.

(6) Theorem (on length problem)

If $L_{\mathfrak{Z}} \neq \emptyset$ then there is a d.p.e. with length $|p| < 2^{\binom{n}{2}}$, thereby n is the number of states in Z. This bound is strong.

For proving these theorems one can define a function F from X^* into the set of all symmetric and reflexive binary relations over Z as follows:

$$F(p) =_{Df} \{ [z,z'] \,/\, z,z' \in Z \wedge v(z,p) \cap v(z',p) \neq \emptyset \}.$$

Then one can show:

(7) Lemma

(α) $F(p) = F(p') \rightarrow F(rp) = F(rp')$

(β) $\{F(p) / |p|<i\} = \{F(p) / |p|\leq i\} \rightarrow$

$\rightarrow \{F(p) / |p|<i\} = \{F(p) / p \in X^*\}$

(γ) $p \in L_3 \leftrightarrow (Z_o \times Z_o) \cap F(p) = \{[z,z] / z \in Z_o\}$

(p, p', $r \in X^*$, $i \in nz$, $|p|$ the length of p).

From Lemma (7) follows that $\overleftarrow{L}_3 =_{Df} \{ x_n \ldots x_1 / x_1 \ldots x_n \in L_3\}$ can be accepted by the finite acceptor

$\mathcal{A} =_{Df} [X, S, \delta, s_o, S_f]$ with set of states $S = \{F(p) / p \in X^*\}$,

initial state $s_o = F(e)$,

accepting states $S_f = \{F(p) / p \in L_3\}$

and transition function $\delta: S \times X \rightarrow S$ with $\delta(F(p),x) = F(xp)$.

Since Z is finite, $\mathcal{A}$ must be finite. For each identification task $\mathfrak{Z}$ as considered here, an acceptor $\mathcal{A}$ for the set $\overleftarrow{L}_3$ can be constructed.

It is well known that with $\overleftarrow{L}_3$ the set L_3 is regular too, and it is also well known, that a language L accepted by an acceptor $\mathcal{A}$ is not empty if and only if in L is a word p with length $|p| < |S|$ ($|S|$ the number of states in S). Hence by construction of $\mathcal{A}$ the regularity of L holds, and also Theorems (5) and (6). In /6/ are constructed examples for proving Theorem (4), and in /12/ for Theorem (6) (that the bound is strong).

Note that many solutions of existence and construction problems for other classes of identification tasks can be given by using such functions F (from X^* in a finite set) in a very similar way. In /6/, /7/ was shown that for

construction of acceptors in this manner it is sufficient to show conditions like (α) and (γ) of Lemma (7) for F. For instance, the methods given by GILL in /2/ (diagnosing tree, homing tree) may be interpreted by such functions F.
For asynchronous automata ($h : Z \times X \rightarrow 2^{Y^* \times Z}$) such methods are not useful even in case h(z,x) finite for all $z \in Z$ and $x \in X$, thereby the sets of identification experiments must not be regular. For such asynchronous automata the existenceproblem is not decidable (/6/, /8/, on the base of /3/), but for finite asynchronous deterministic automata it is decidable (/6/, /8/). But even in case of such deterministic automata the sets of d.p.e. must not be context-free (/9/).

3. Homing preset experiments

(8) Definition

An input word $p \in X^*$ is a homing preset experiment (h.p.e.) for $\mathcal{Z} = [X, Y, Z, h, Z_o]$ and a given finish set $\mathfrak{M}_f \subseteq 2^Z \setminus \{M / Z_o \in M\}$
iff there exists a function $\varphi : v(Z_o, p) \rightarrow \mathfrak{M}_f$
such that holds: $\forall q \; (q \in v(Z_o,p) \rightarrow h_q(Z_o,p) \subseteq \varphi(q))$.

By using the finish set $\mathfrak{M}_f$ such h.p.e. can perform both: identification of the state after the end of the experiment and control of this state. For $\mathfrak{M}_f = \{\{z\} / z \in Z\}$ we have only identification, and if $\mathfrak{M}_f$ consists of only one element M we have only control (in a state out of M).

Of course, identification and control may be also connected by performing a h.p.e. . The trivial case $\mathfrak{M}_f \cap \{ M / Z_o \subseteq M \} \neq \emptyset$ will not be considered, therefore the empty word e can not be a h.p.e. .

(9) Theorem

Let be $L_{\mathfrak{Z}}$ the set of all h.p.e. for a identification task $\mathfrak{Z}$ with a given NBA $\mathfrak{A}$ and a given finish set $\mathfrak{M}_f$. Then in case of $\mathfrak{M}_f = \{\{z\} / z \in Z\}$ Theorems (3), (4) and (5) hold,

and, omitting condition (a) in (3), (4), they also hold for any finish sets $\mathfrak{M}_f$.

An upper bound (but probably not a strong) for the length of shortest h.p.e. is 2^{2^n} (n the number of states in Z).

For proving these Theorems for h.p.e. (cf. /6/, /7/) can be used the function $F : X^* \to 2^{2^Z}$

with $F(p) =_{Df} \{ h_q(Z_o, p) / q \in v(Z_o, p) \}$.

F satisfies the conditions

(α) $F(p) = F(p') \longrightarrow F(pr) = F(p'r)$ ($p, p', r \in X^*$),

(β) as in Lemma (7) (one can show that (β) follows from (α))

(γ) $p \in L_{\mathfrak{Z}} \leftrightarrow \forall M (M \in F(p) \to \exists N (N \in \mathfrak{M}_f \wedge M \subseteq N))$.

Thus, an acceptor $\mathfrak{a}$ for $L_{\mathfrak{Z}}$ can be constructed in a similar way like for d.p.e. (because of (α) hereby is to define $(F(p), x) = F(px)$, and thus $\mathfrak{a}$ directly accepts $L_{\mathfrak{Z}}$).

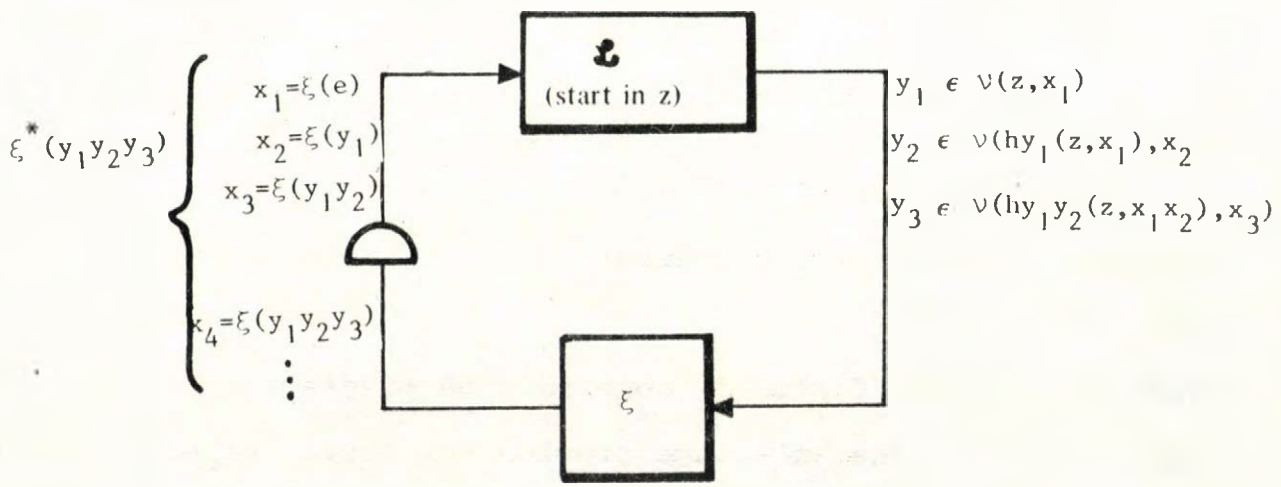
4. Adaptive experiments

An adaptive experiment can be described as a strategy ξ of the experimentalist for finding the next input symbol with respect to the output until this moment. Thus, ξ is a function from the set Y^* of output words into the set $X \cup \{\text{stop}\}$. The experimentalist begins with $x_1 = \xi(e)$ and later on after observation of the output $y_1 \ldots y_n \in Y^*$ he continues the experiment with input $x_{n+1} = \xi(y_1 \ldots y_n)$ or he stops the experiment if $\xi(y_1 \ldots y_n) = \text{stop}$.
The belonging sequential function we denote by ξ^*, we can define

$$\xi^*(e) =_{Df} e$$

$$\xi^*(qy) =_{Df} \xi^*(q) \cdot \xi(q) \quad , \quad q \in Y^*, \; y \in Y \; .$$

Note that by this definition $\xi^*(qy)$ not depends on y . $\xi^*(qy)$ is the input word that the experimentalist has put in the automaton when the automaton has put out the the whole word qy . The next action of the experimentalist then will be $\xi(qy)$ which also depends on y.



By $\varphi(z,\xi)$ we denote the set of all output words which the automaton can put out during the work of the strategy after the start in z :

$$\varphi(z,\xi) =_{Df} \{ q \;/\; q \in Y^* \wedge \xi^*(q) \in X^* \wedge q \in v(z,\xi^*(q)) \}$$

by $\varphi_f(z,\xi)$ we denote those of these words, for which the strategy stops:

$$\varphi_f(z,\xi) =_{Df} \{ q \;/\; q \in \varphi(z,\xi) \wedge \xi(q) = \text{stop} \} \quad .$$

(10) Definition

(a) A strategy ξ is an adaptive experiment (a.e.) for $\mathfrak{A} = [X, Y, Z, h, Z_o]$ iff the strategy stops for all states $z \in Z_o$ after a finite time (i.e. $\varphi(Z_o,\xi)$ is finite).

(b) An a.e. ξ for $\mathfrak{A}$ is a diagnosing adaptive experiment (d.a.e.) for $\mathfrak{A}$ iff there exists a function $\gamma : \varphi_f(Z_o,\xi) \to Z_o$ such that holds: $\forall z \forall q (z \in Z_o \wedge q \in \varphi_f(z,\xi) \to \gamma(q)=z)$.

(c) An a.e. ξ for $\mathfrak{A}$ is a homing adaptive experiment (h.a.e.) for $\mathfrak{A}$ and a given finish set $\mathfrak{M}_f$ iff there exists a function $\varphi : \varphi_f(Z_o,\xi) \to \mathfrak{M}_f$ such that holds: $\forall q (q \in \varphi_f(Z_o,\xi) \to h_q(Z_o,\xi^*(q)) \subseteq \varphi(q))$

We can remark that $h_q(z,\xi^*(q))$ is the set of all possible states after start in z and performing ξ when the output was q .

With help of the following construction of classes $\mathfrak{M}_1 \subseteq 2^Z \setminus \emptyset$ the existence problem for h.a.e. at NDA can be solved:

$$\mathfrak{M}_0 =_{Df} \{ M \;/\; \exists N (N \in \mathfrak{M}_f \wedge M \subseteq N)$$

$$\mathfrak{M}_{i+1} =_{Df} \mathfrak{M}_i \cup \{ M \;/\; \emptyset \neq M \subseteq Z \wedge$$

$$\wedge \exists x (x \in X \wedge \forall y (y \in v(M,x) \rightarrow h_y(M,x) \in \mathfrak{M}_i)) \}$$

Thereby $i = 0, 1, 2, \ldots$ and it can be proved:

(11) Lemma

(a) $\mathfrak{M}_i = \mathfrak{M}_{i+1} \longrightarrow \mathfrak{M}_i = \bigcup_{j=0}^{\infty} \mathfrak{M}_j$

(b) There exists a h.a.e. for iff $Z_0 \in \bigcup_{j=0}^{\infty} \mathfrak{M}_j$.

Since the considered automata are finite, we have $\mathfrak{M}_{i+1} = \mathfrak{M}_i$ after finite time and so we can solve the existence problem.

For $M \in \mathfrak{M}_0$ let be $\mu(M) =_{Df}$ stop and for other $M \in \mathfrak{M}_{i+1} \setminus \mathfrak{M}_i$ $(i = 1,2,3,\ldots)$ let be $\mu(M)$ a fixed input symbol x which satisfies the condition $\forall y (y \in v(M,x) \rightarrow h_y(M,x) \in \mathfrak{M}_i)$. Then in case of existence of a h.a.e. for $\mathfrak{L}$ and $\mathfrak{M}_f$ such h.a.e. ξ can be realized by the following finite partial defined deterministic initial MOORE-automaton $\mathfrak{A}_\xi =_{Df} [Y, X \cup \{\text{stop}\}, \mathfrak{M}, \delta, \mu, Z_0]$ with input set Y, output set $X \cup \{\text{stop}\}$, set of states $\mathfrak{M}$, initial state Z_0, partial defined transition function $\delta : \mathfrak{M} \times Y \xrightarrow{P} \mathfrak{M}$, output function $\mu : \mathfrak{M} \rightarrow X \cup \{\text{stop}\}$. Thereby we define $\delta(M, y) =_{Df} h_y(M, \mu(M))$ if $h_y(M, \mu(M)) \neq \emptyset$ and $M \notin \mathfrak{M}_0$, and δ not defined otherwise. $\mathfrak{M}$ is defined as the smallest subset of 2^Z which contains Z_0 and is closed under δ .

Then $\mathfrak{A}_\xi$ as experimentalist begins with $\xi(e) =_{Df} \mu(Z_0)$, and after the output of an output word $q \in Y^*$ by $\mathfrak{L}$ (in the

same time q is the input word for $\mathfrak{A}_\xi$) the state of $\mathfrak{A}_\xi$ is $\delta(z_0,q)$ and the next action of the experimentalist $\mathfrak{A}_\xi$ is $\xi(q) =_{Df} \mu(\delta(z_0,q))$ as input symbol for $\mathfrak{L}$ or stop respectively.

One can show that $\delta(z_0,q) = h_q(z_0,\xi^*(q))$. Thus, when $\mathfrak{A}_\xi$ is in a state $M \in \mathfrak{M}_0$ by definition of μ the experiment ξ stops and $\mathfrak{L}$ is in a state of M. By definition of $\mathfrak{M}_0$ for M a set $N \in \mathfrak{M}_f$ with $M \subseteq N$ exists, and this N is to be identified by the function φ. Because of $\delta(M,q) \neq M$ for all $q \in Y^*$ and $M \in \mathfrak{M}$ the automaton $\mathfrak{A}_\xi$ must come in a state $M \in \mathfrak{M}_0$ after finite time (no more then 2^n-2 input steps are needed when n is the number of states in Z).

The solution of the construction and existence problem for d.a.e. at NDA can be given in a similar way (cf. /6/).
So we have shown:

(12) Theorem

The existence and construction problem for h.a.e. or d.a.e. respectively at NDA is algorithmically solvable. The length of shortest experiments of these kinds can be bounded by $2^n - 2$, thereby n is the number of of states of the examined automaton. The bound is strong in case of h.a.e. (cf. /11/).

5. Further problems

To the task of an identification experiment can be added some conditions. For instance, one can forbid the automaton to arrive some states in some situations. By this condition only words out of a certain set $L \subseteq X^*$ can become an identification experiment, such that the set of those identification experiments is the set $L \cap L_3$ (L_3 in the earlier sense). Since L is regular in "evident" cases, no special problems have to be considered here.
Another condition may come from the power of the experimentalist to distinguish between some output words. This power can be given by a binary relation $U \subseteq Y^* \times Y^*$, such that $[q, q'] \in U$ means the ability of the experimentalist to distinguish between q and q'. In /6/ and also in /5/ and /7/ it was shown that for regular U (in the sense of only $U \subseteq (Y \times Y)^*$ which is sufficient for synchronous automata) the sets of identification preset experiments at NDA are regular too. The existence and construction problem can be algorithmically solved by using functions F like shown here for d.p.e. . For characterization of the sets of identification experiments at deterministic automata in case of context-free and other relations U cf. /10/ .

The study of identification experiments at partial defined automata can be reduced to the study of total defined automata with respect to the interpretation of the work of partial defined automata (cf. /6/).

Open problems are connected with complexity problems: It would be important to know some subclasses of automata for which the complexity of identification experiments is essentially simpler than in general cases.

Dr. H.D. Burkhard
Sektion Mathematik der Humboldt-Universität zu Berlin
DDR-1086 Berlin
Postfach 1297

Literatur:

/1/ MOORE, E.F., Gedanken-experiments on sequential machines. In: Automata studies; Princeton 1956; 129-153.

/2/ GILL, A., Introduction to the theory of finite state machines. New York 1962.

/3/ GRIFFITHS, T.V., The unsolvability of the equivalence problem for Λ-free nondeterministic generalized machines. J.ACM 15 (1968), 409-413.

/4/ STARKE, P.H., Abstrakte Automaten Berlin 1969.

/5/ STARKE, P.H., On diagnosing experiments with nondeterministic automata with final states. EIK 10 (1974) 8/9, 471-480.

/6/ BURKHARD, H.D., Konstruktionsalgorithmen für Identifizierungsexperimente an synchronen und asynchronen Automaten. Diss., Humboldt-Universität zu Berlin, 1974.

/7/ BURKHARD, H.D., Diagnose und Einstellung nicht-deterministischer Automaten bei regulären Unterscheidungsformen. EIK 10 (1974) 8/9, 455-469.

/8/ BURKHARD, H.D., Identifizierungsexperimente an asynchronen Automaten. EIK 12 (1976) 1/2, 45-59.

/9/ BURKHARD, H.D., Zustandsidentifizierung an asynchronen Automaten. EIK 11 (1975), 10-12, 653-658.

/10/ BURKHARD, H.D., Zum Längenproblem homogener Experimente an determinierten und nicht-deterministischen Automaten. EIK 12 (1976), 6, 301-306.

/11/ BURKHARD, H.D., Identifizierungsexperimente an determinierten Automaten mit Unterscheidungsformen für Ausgabewörter. Wiss.Zeitschr.der Humboldt-Universität zu Berlin, Math.-Nat.R. XXIV (1975) 6, 739-742.

/12/ BURKHARD, H.D., Über Experimente an nicht-deterministischen Automaten. EIK 5 (1969) 6, 347-376 und 6(1970) 1, 3-14.

КЁГСТ, Манфред; ФРАНКЕ, Гюнтер, ГДР, г. Дрезден

ВЛИЯНИЕ ДРЕБЕЗГА ВХОДНЫХ СИГНАЛОВ НА ПОВЕДЕНИЕ АСИНХРОННЫХ АВТОМАТОВ

При применении в технике контактных датчиков двоичных сигналов необходимо в общем случае считаться с их дребезгом. Это означает, что при включении /отключении/ датчика может происходить не однозначное изменение сигнала, а многократное изменение его, до тех пор, пока после окончания дребезга, он наконец примет желаемого значения.

Если элементы асинхронного управляющего автомата имеют меньшее или равное время срабатывание по сравнению с протяженностью дребезга, то возможны ошибки в работе автомата, т.е. автомат чувствителен к дребезгу на рассматриваемых входах.

Дребезг входных сигналов можно устранить, используя дополнительно некоторые технические устройства, например фильтры. С точки зрения уменьшения этих дополнительных затрат существенным является решение следующих задач:

1. Выявление множества входных сигналов, к дребезгу которых автомат чувствителен.
2. В практике проектирования автоматов устранение влияния дребезга обычно производится уже после построения его схемы, что приводит к большим дополнительным затратом. Поэтому существенной является задача разработки методов учета дребезга входных сигналов на этапе проектирования автомата, например на этапе составления таблицы переходов автомата, с целью построения автомата, чувствительного к дребезгу возможно меньшего числа входных сигналов.

Будем рассматривать автоматы, заданные таблицами переходов. Предполагаем, что одновременно может изменяться значение только одной переменной.

В данной работе таблица переходов рассматривается безотносительно к возможностям её минимизации, поэтому любые две разные строки её считаются несовместными.

В данной работе предлагается метод доопределения таблицы переходов с точки зрения устранения влияния дребезга, который, несмотря на внешнее сходство с методами устранения состязаний при кодировании внутренних состояний автомата [2, 3] существенно отличается от них. Различие это вызвано, например, тем, что число изменений сигналов при дребезге в принципе непредсказуемо в то время как возможные переходы при состязаниях в кодированном автомате могут быть точно описаны.

Определение I: Петёрка (X,Y,Z,f,g) описывает конечный автомат A, если $X=\{X_i\}, Y=\{Y_j\}, Z=\{Z_k\}$ есть не пустые конечные множества и f, соответственно g есть однозначное отображение из $X \times Z$ в Z соответственно Y.

Для f и g вводятся области определения $D_f, D_f \subseteq X \times Z$ и $D_g \subseteq X \times Z$, следующим образом

$$\forall X_1 \forall Z_1 (X_1 \in X \wedge Z_1 \in Z \rightarrow ((X_1,Z_1) \in D_f \leftrightarrow \exists Z_2 (Z_2 \in Z \wedge f(X_1,Z_1)=Z_2))) \quad \text{и}$$

$$\forall X_1 \forall Z_2 (X_1 \in X \wedge Z_2 \in Z \rightarrow ((X_1,Z_2) \in D_g \leftrightarrow \exists Y_1 (Y_1 \in Y \wedge g(X_1,Z_2)=Y_1)))$$

В дальнейшем рассматриваются асинхронные автоматы, т.е.

1. $\forall X_1 \forall Z_1 \forall Z_2 (X_1 \in X \wedge Z_1 \in Z \wedge Z_2 \in Z \wedge f(X_1,Z_1)=Z_2 \rightarrow f(X_1,Z_2)=Z_2)$
2. $D_g \subseteq D_f$

Если автомат имеет n двоичных входов, то это обозначается $X=\{0,1\}^n$, а множество входных переменных записывается как $E = \{x_1,\ldots,x_n\}$.

Следующие рисунки /рис. I и 2/ пояснают понятие чувствительности автомата к дребезгу входных сигналов.

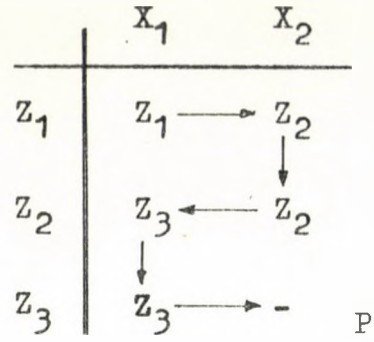


Рис. I

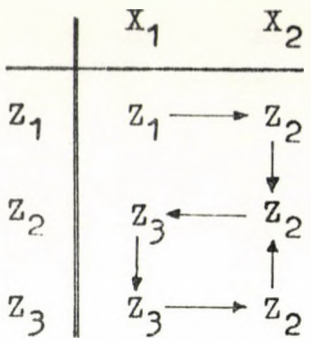


Рис. 2

При переходе от набора значений входных сигналов x_I и x_2 / левый рисунок/ при дребезге изменяемого сигнала произойдет переход не в состояние Z_2, как предполагается проектировщиком, а в состояние Z_3 и далее в неопределенное для проектировщика состояние. В данном случае однако возможно такое доопределение автомата /правый рисунок/, что автомат становится нечувствительным к дребезгу рассматриваемого сигнала.

Теперь можно точнее определить понятие чувствительности к дребезгу, используя понятие множества $\ddot{U}$ переходов между состояниями автомата, относительно которого определяется чувствительность автомата к дребезгу. Множество $\ddot{U}$ выбирается проектировщиком.

Определение 2: Автомат $A = (X,Y,Z,f,g)$ называется относительно $x_i, x_i \in E$ и $\ddot{U}, \ddot{U} \subseteq (X \times Z)^2$ нечувствительным к дребезгу, если имеет место:

1. $\ddot{U} \subset D_f^2$
2. $\forall((X_1,Z_1),(X_2,Z_2))(((X_1,Z_1),(X_2,Z_2)) \in \ddot{U} \longrightarrow$
   $f(X_1,Z_1)=Z_1 \wedge f(X_2,Z_1)=Z_2 \wedge f(X_2,Z_2)=Z_2)$
3. $\forall((X_1,Z_1),(X_2,Z_2))(((X_1,Z_2),(X_2,Z_2)) \in \ddot{U} \wedge (X_1,X_2) \in X^2(i)$
   $\longrightarrow \exists Z_3(Z_3 \in Z \wedge f(X_1,Z_2)=(Z_3 \wedge f(X_2,Z_3)=Z_2))$

При этом $X^2(i), X^2(i) \in X^2$, есть множество всех пар наборов значений входных переменных различающихся только значением переменной x_i. Используя это определение можно проверять заданный автомат на нечувствительность к дребезгу. Для автомата A, чувствительного дребезгу в множестве $\ddot{U}$ стремятся получить доопределение автомата A - автомат A', который относи-

тельно рассматриваемого множества Ü переходов был чувствителен для возможно меньшего числа переменных.

Определение 3: Автомат $A = (X,Y,Z,f,g)$ относительно $E', E' \subseteq E$ и $Ü, Ü \subset (X \times Z)^2$, называется доопределимым /для достижения нечувствительности к дребезгу/, если существует автомат $A' = (X,Y,Z,f',g')$ и выполняется следующее:

1. $Ü \subset D_f^2$
2. $\forall((X_1,Z_1),(X_2,Z_2))(((X_1,Z_1),(X_2,Z_2)) \in Ü \longrightarrow$
   $f(X_1,Z_1)=Z_1 \wedge f(X_2,Z_1)=Z_2 \wedge f(X_2,Z_2)=Z_2)$
3. $D_f \subset D_{f'}$, $D_g \subseteq D_{g'}$, $f \simeq f$, $g \simeq g'$ /совместимость/

4. Для каждого $x_i, x_i \in E'$, автомат A относительно x_i и множества Ü чувствителен к дребезгу, но автомат A' относительно x_i и множества нечувствителен к дребезгу.

A' называется доопределенным A относительно E' и Ü.

Предложение I: Если автомат A доопределим относительно E', $E' \subseteq E$, и Ü а также относительно $E'', E'' \subseteq E$ и Ü, то из этого в общем случае не следует, что автомат A доопределим также относительно $E' \cup E''$ и Ü.

Пример I служит доказательством этого предложения.

Пример I: Выделим из таблицы переходов, приведенной на рис. 3 множество переходов, существенных для доказательства предложения I:

$$Ü = \{((01,Z_3),(11,Z_4)),((01,Z_5),(11,Z_4)),((00,Z_1),(01,Z_2))\}$$

Автомат, заданный таблицей переходов рис. 4, чувствителен к дребезгу относительно обеих входных переменных. Он доопределим относительно x_1 и Ü /в клетку с координатами $(01,Z_4)$ нужно записать Z_2/, он доопределим также относительно x_2 и Ü /в клетку с координатами $(01,Z_4)$ нужно записать Z_3, Z_4 или Z_5/, но он недоопределим одновременно относительно x_1, x_2 и Ü.

x_2x_1 / z_i	00	01	11	10
z_1	z_1	z_2	z_4	z_1
z_2	z_4	z_2	-	z_5
z_3	z_3	z_3	z_4	-
z_4	z_4	-	z_4	z_1
z_5	z_3	z_5	z_4	z_5

Рис. 3: Таблица переходов автомата для примера I

При доопределении автомата A относительно множества $\ddot{U}$ можно последовательно доопределять автомат A относительно отдельных переходов из множества $\ddot{U}$. Доопределение автомата A относительно одного перехода осуществляется на основании определения 4.

Определение 4: Автомат $A = (X,Y,Z,f,g)$ называется относительно $((X_1,Z_1),(X_2,Z_2)),((X_1,Z_1),(X_2,Z_2))\in(X\times Z)^2$, доопределимым, если имеет место:

1. $(X_1,X_2)\in X^2(i)$
2. $f(X_1,Z_1)=Z_1 \wedge f(X_2,Z_1)=Z_2 \wedge f(X_2,Z_2)=Z_2$
3. $(X_1,Z_2)\notin D_f \vee \exists Z_3(Z_3\in Z \wedge f(X_1,Z_2)=Z_3 \wedge (X_2,Z_3)\notin D_f)$

При таком доопределении автомата, следует однако учесть, что в соответствии с предложением I может быть получено неоптимальное решение. Для улучшения этого решения можно использовать различные эвристические процедуры.

Так как при минимизации числа состояний с одной стороны некоторые возможности доопределения автомата могут быть утеряны, а с другой стороны появиться новые возможности доопределения с точки зрения устранения влияния дребезга, рекомендуется использование при проектирования следующей стратегии:

Данный автомат A приводится доопределением к автомату A_K . Производится минимизация числа состояний и получается автомат A_{KP} . Автомат A_{KP} вновь доопределяется до автомата A_{KPK} , чувствительного к дребезгу возможно минимального числа входов.

Проиллюстрируем изложенное выше на примере.

Пример 2: Имеются три наполняемых жидкостью сосуда. Наполнение сосуда контролируется специальным датчиком.

X_i	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
$x_3x_2x_1$	0 0 0	0 0 1	0 1 0	1 0 0	1 0 1	0 1 1	1 1 0	1 1 1
Z_1	1/000	2	3	4				
Z_2	1	2/001			5	7		
Z_3	1		3/010			8	9	
Z_4	1			4/100	6		10	
Z_5		?		4	5/001			11
Z_6		2		?	6/100			12
Z_7		?	3			7/001		16
Z_8		2	?			8/010		15
Z_9			?	4			9/010	14
Z_{10}			3	?			10/100	13
Z_{11}					?		10	11/001
Z_{12}					?	7		12/100
Z_{13}						8	?	13/100
Z_{14}					6		?	14/010
Z_{15}					5	?		15/010
Z_{16}						?	9	16/001

Рис. 4: Таблица переходов автомата для примера 2. Автомат чувствителен к дребезгу всех входных сигналов x_1, x_2, x_3.

При достижении высшего уровня значение сигнала на выходе датчика равно 0, при достижении нижнего уровня значение сигнала равно I, а в промежуточных точках сохраняется прежнее значение сигнала.

Три сосуда должны наполняться в той последовательности, в которой они опустошались.

X_i	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
$x_3x_2x_1$	0 0 0	0 0 1	0 1 0	1 0 0	1 0 1	0 1 1	1 1 0	1 1 1
Z_1	1/000	2	3	4				
Z_2	1	2/001			5	7		
Z_3	1		3/010			8	9	
Z_4	1			4/100	6		10	
Z_5		2		4	5/001			11
Z_6		2		4	6/100			12
Z_7		2	3			7/001		16
Z_8		2	3			8/010		15
Z_9			3	4			9/010	14
Z_{10}			3	4			10/100	13
Z_{11}					5		10	11/001
Z_{12}					6	7		12/100
Z_{13}						8	10	13/100
Z_{14}					6		9	14/010
Z_{15}					5	8		15/010
Z_{16}						7	9	16/001

Рис. 5: Таблица переходов автомата для примера 2. Таблица получена при доопределении автомата рис. 4. Автомат нечувствителен к дребезгу входных сигналов.

Необходимо построить автомат, управляющий вентилями для наполнения сосудов в соответствии с описанными условиями.

Пусть при проектировании получена таблица переходов /рис. 4/ без учета влияния возможного дребезга входных сигналов. Проверка таблицы в соответствии с определением 2 показывает, что автомат чувствителен к дребезгу всех входных сигналов x_1, x_2, x_3 относительно множества переходов $\ddot{U}$. Множество переходов $\ddot{U}$ соответствует здесь множеству всех возможных переходов в таблице рис. 4, поэтому его можно не выписывать.

Неопределенные переходы, которым в таблице рис. 4 соответствуют клетки, отмеченные знаком "?", определяют поведение автомата при дребезге. При доопределении автомата рис. 4 в соответствии с определением 4 будет получен нечувствительный к дребезгу автомат рис. 5.

Авторы благодарят к.т.н. Е. Пупырева за помощь при переводе.

Цитированная литература

/1/ Franke,G. und M.Koegst
Zum Einfluß von Prellerscheinungen der Eingangssignale auf das Verhalten asynchroner Automaten
AdW der DDR, ZKI-Informationen 2/1976, S.34-37

/2/ Unger,S.H.
A row assignment for delay-free realizations of flow tables without essential Hasards
IEEE Trans. on C.,Vol.C-17(1968)2,146-151

/3/ Zander,H.J.
Entwurf von Folgeschaltungen
Berlin, Verlag Technik 1974, Reihe Automatisierungstechnik Bd. 158

Работы по теории автоматов Познаньского политехнического института

Е. Гржимала-Буссе

А. Возняк

Познаньский политехнический институт

60-965 Познань, Польша

Р е з ю м е

В настоящей статьи будут реферированы работы ро теории автоматов Познаньского политехнического института за годы 1974-1975, но за исключением работ по теории автоматов с переменной структурой, которые реферированы в отдельной работе.

В статьи /I/ разработаны некоторые условия квазиуправляемости конечных автоматов. Понятие квазиуправляемости было введено в работе /4/.

В /I/ подано тоже критерий квазиуправляемости автоматов.

Две классы автоматов такие, что прямое произведение всякой пары автоматов такой, что первый автомат принадлежит к первой классе, а второй автомат к второму классу, сильно связное найдены в работе /2/ . В этой же работе показано некоторые алгебраические свойства прямого произведения таких автоматов.

Следующий вопрос: для заданного моноида M преобразований некоторого множества T найти все автоматы с множеством состояний T и моноидом всех эндоморфизмов равным M был разрешен в работе /3/ .

В работе /4/ кроме некоторых условий квазиуправляемости конечных автоматов подано тоже условия, при исполнении которых для заданного множества A автоматов существует такой автомат B , что для всякого автомата C из A существует подавтомат автомата B , связанный с преобразованием тактности.

Обобщения понятия эндоморфизма и конгруэнции автомата введено в /5/ . В работе /6/ подано некоторые свойства разбиений множества состояний автомата таких, которые не обязательно и конгруэнции.

Некоторые технические проблемы рассмотрены в /7/ , а именно какие классы автоматов реализованы в заданных классах сетей итеративных.

Список цитированной литературы

1. L. Bega. On the quasi-controllability of automata. 3rd Symp. Math. Found. Computer Sci., Jadwisin, June 17-22, 1974. Lecture Notes in Computer Science. Vol. 28, 23-25. Springer Verlag, Berlin-Heidelberg-New York,1975
2. J.W. Grzymała-Busse. On the strongly related automata. Proc Internat. Symp. IFAC "Discrete Systems", vol.4, 118-127 "Zinatne", Riga, 1974.
3. J.W. Grzymała-Busse. On the set of all automata with the same monoid of endomorphisms. 4th Symp. Math. Found.

Computer Sci., Marianske Lazne, September 1-5, 1975. Lecture Notes in Computer Sci.. Vol.32, 246-251. Springer Verlag, Berlin-Heidelberg-New York, 1975.

4. J.W. Grzymała-Busse. Problems of the change of operating time of finite automata. Proc. 5th Congress Gesel. Informatik, Dortmund, October 8-10, 1975. Lecture Notes in Computer Sci., vol. 34, 261-268. Springer Verlag, Berlin-Heidelberg-New York.

5. J.W. Grzymała-Busse. Generalized endomorphisms, congruences, and subautomata associated with the change of operating time of finite automata. Found. Control Engng 1, 1/1975/, 3-14.

6. Е.В. Гржимала-Буссе. О разбиениях множества состояний и отношениях на входной полугруппе автоматов. ЕІК II, I0/I2 /I975/, 58I-582.

7. Z. Miądowicz. The synthesis of automata realized by some iterative networks. Found. Control Engng 1, 1 /1975/, 37-45.

Дискретное управление медленным асинхронным процессом с помощью конечного автомата

Р.Новански
Институт по использованию вычислительной техники в управлении (ИИВТУ), ЧССР - Прага

В настоящем докладе описано одно применение модулярного синтеза, которое было разработано в нашем институте. Конкретно речь идет о синтезе дискретного управляющего устройства автоматического электрокардиографа, управляющего процессом автоматического съема и изготовления электрокардиограмм. Эти электрокардиограммы предназначены как для визуального анализа врачом, так и для обработки на вычислительной машине. Общая структура автоматического электрокардиографа указана на рисунке 1.

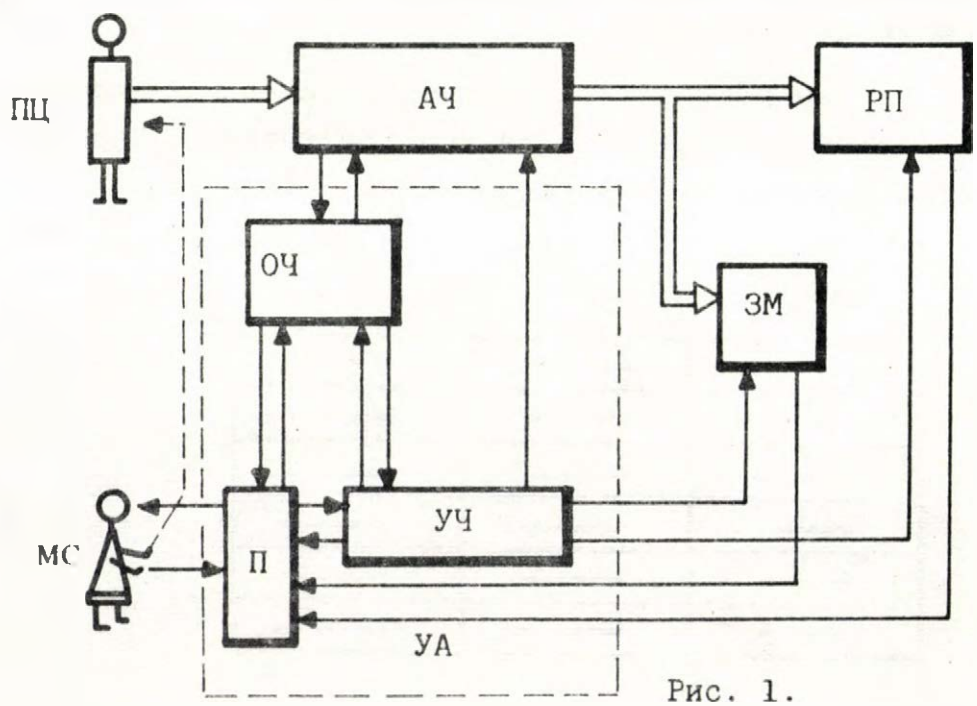


Рис. 1.

Как видно из чертежа, автоматический электрокардиограф состоит из :

- управляющего автомата (УА)
- аналоговой части (АЧ)
- регистрационного прибора (РП)
- записывающего магнитофона (ЗМ).

На рис.1 покавана кроме того глобальная структура управляющего автомата и его соединение с остальными частями электрокардиографа.

Управляющий автомат состоит из :

а) операционной части (ОЧ), в которой осуществляются операции над дискретными данными;

б) управляющей части (УЧ), служащей для управления действием всех частей автоматического электрокардиографа (АЭ);

в) панели (П), предназначенной для связи АЭ с обслуживающим персоналом (медсестрой МС). Панель позволяет производить контроль и управление процессом изготовления электрокардиограмм и сигнализацию особенно важных состояний системы АЭ.

Прежде, чем приступить к описанию основных принципов работы, следует остановиться на способе упорядочения данных в изготовляемых электрокардиограммах (см.рис.2).

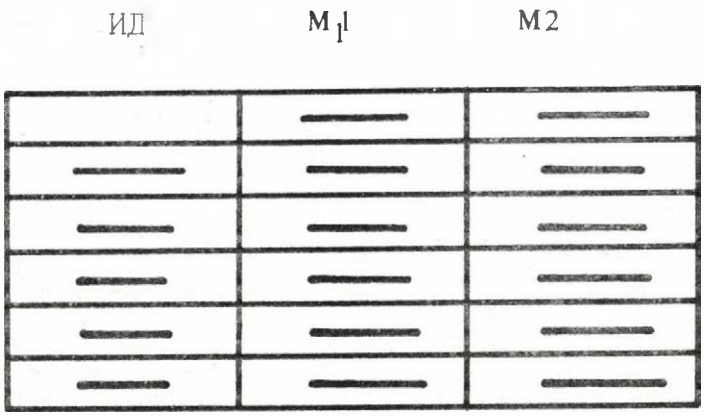


Рис. 2.

Как видно из рисунка, данные изображаются на шести дорожках. В первом отсеке (вертикальном) записываются идентификационные данные пациента (ПЦ) - (дата /ДТ/, год рождения /ГР/ и частота пульсирования сердца /ЧП/ - на 2-ой, 4-ой и 6-ой дорожках).

Во втором отсеке записываются кардиопотенциали конечностей (I,II,III, aVR, aVL, aVF).

В третьем отсеке записываются кардиопотенциали грудной клетки (V_1,V_2,V_3,V_4,V_5,V_6).

ДТ и ГР вводятся в прибор при помощи клавиши на панели, и ЧП вычисляется по значениям Р-волн кардиопотенциалов (см. рис.3).

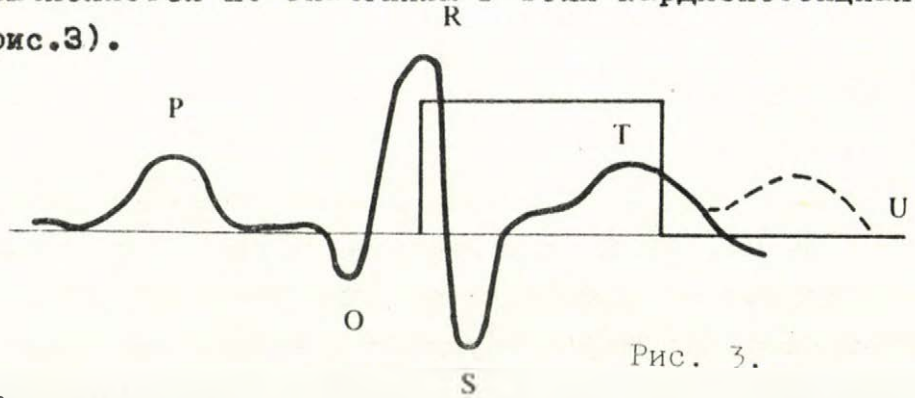


Рис. 3.

Реально изготовленная электрокардиограмма, полученная на автоматическом электрокардиографе, изображена на прилагаемой диаграмме.

Для изготовления электрокардиограмм необходимо, чтобы потенциалы требуемой формы появились на выходах усилителей АЧ. В связи с этим аналоговая часть сделана так, что обеспечивает переключение в четыре основные состояния - K, Z, M_1, M_2.

Состояние K служит для записи идентификационных данных (ИД).

Состояние Z для проверки подсоединения электродов к конечностям пациента.

Съем кардиопотенциалов осуществляется в состояниях M_1 и M_2, как показано на рис.4.

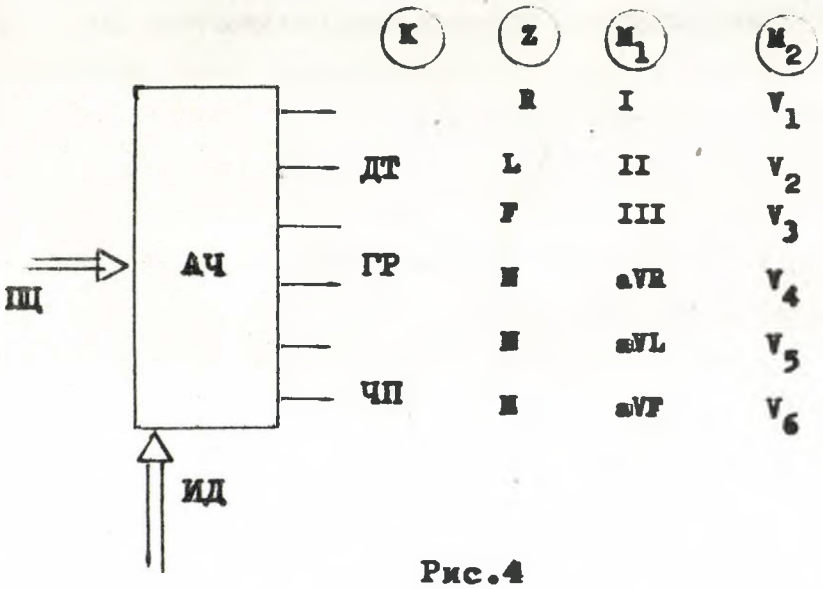


Рис.4

Запись этих потенциалов должна быть синхронизирована с R -волнами, чтобы при переходном процессе напряжение на усилителях не превышало рабочие значения и чтобы не осуществилось их перевозбуждение. Функционирование УА определяется требуемой структурой процесса подготовки и автоматического изготовления электрокардиограмм.

Действие УА можно проследить на рис.5, на котором указана упрощенная схема алгоритма управления АЭ. Более подробно работу АЭ можно наблюдать на рис.6, который прилагается только для иллюстрации.

Блок-схему алгоритма функционирования АЭ можно разделить на две части:
- левую, описывающую фазу подготовки электрокардиограммы;
- правую, которая представляет собой фазу автоматического изготовления электрокардиограммы (ЭКГМ) - автоматический цикл. УА переходит из фазы подготовки в фазу автоматического цикла после нажатия на освещенную кнопку СТ (только лишь после выполнения условий подготовки и ее окончания). Возвращение УА из автоматического цикла в фазу подготов-

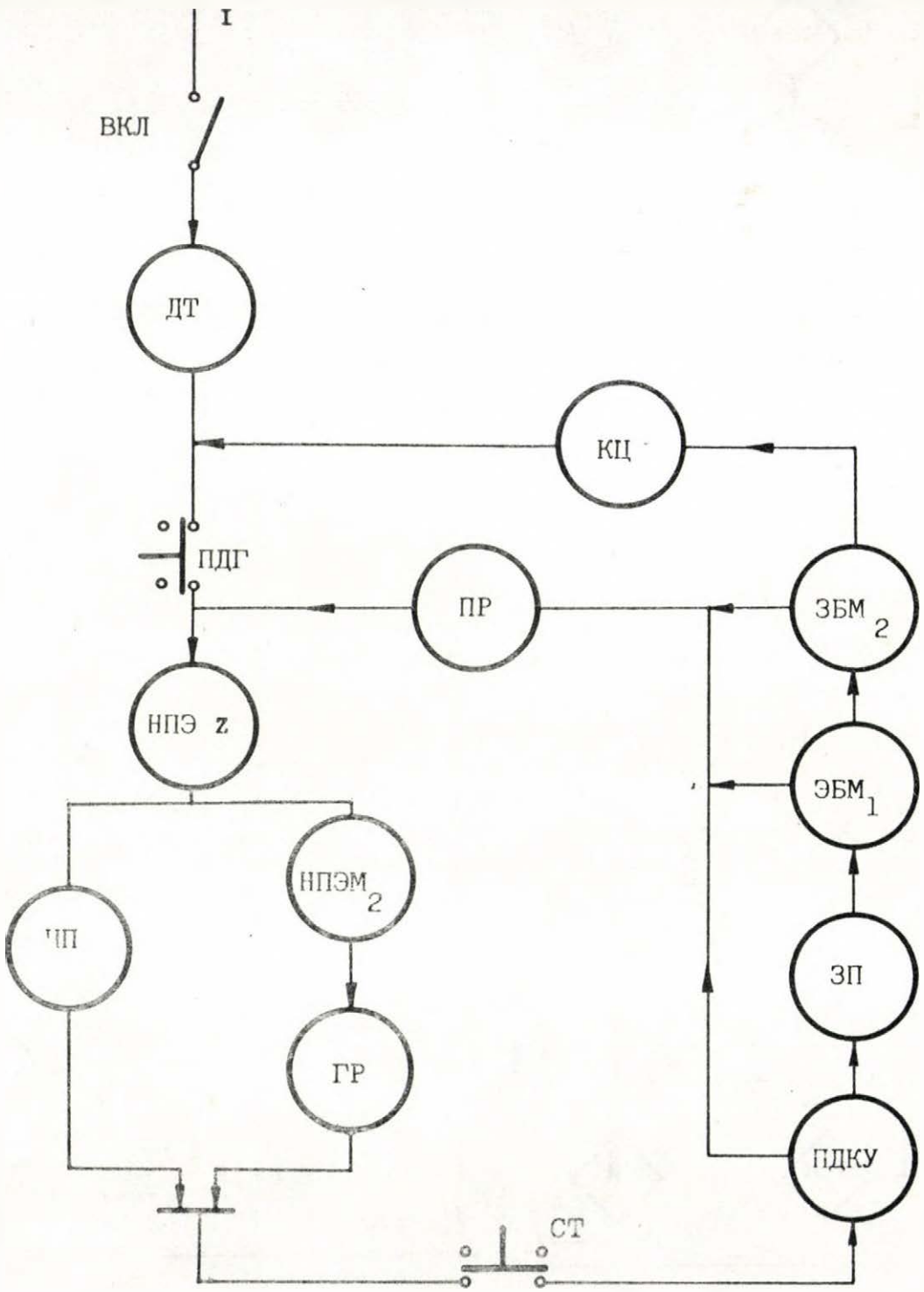


Рис. 5

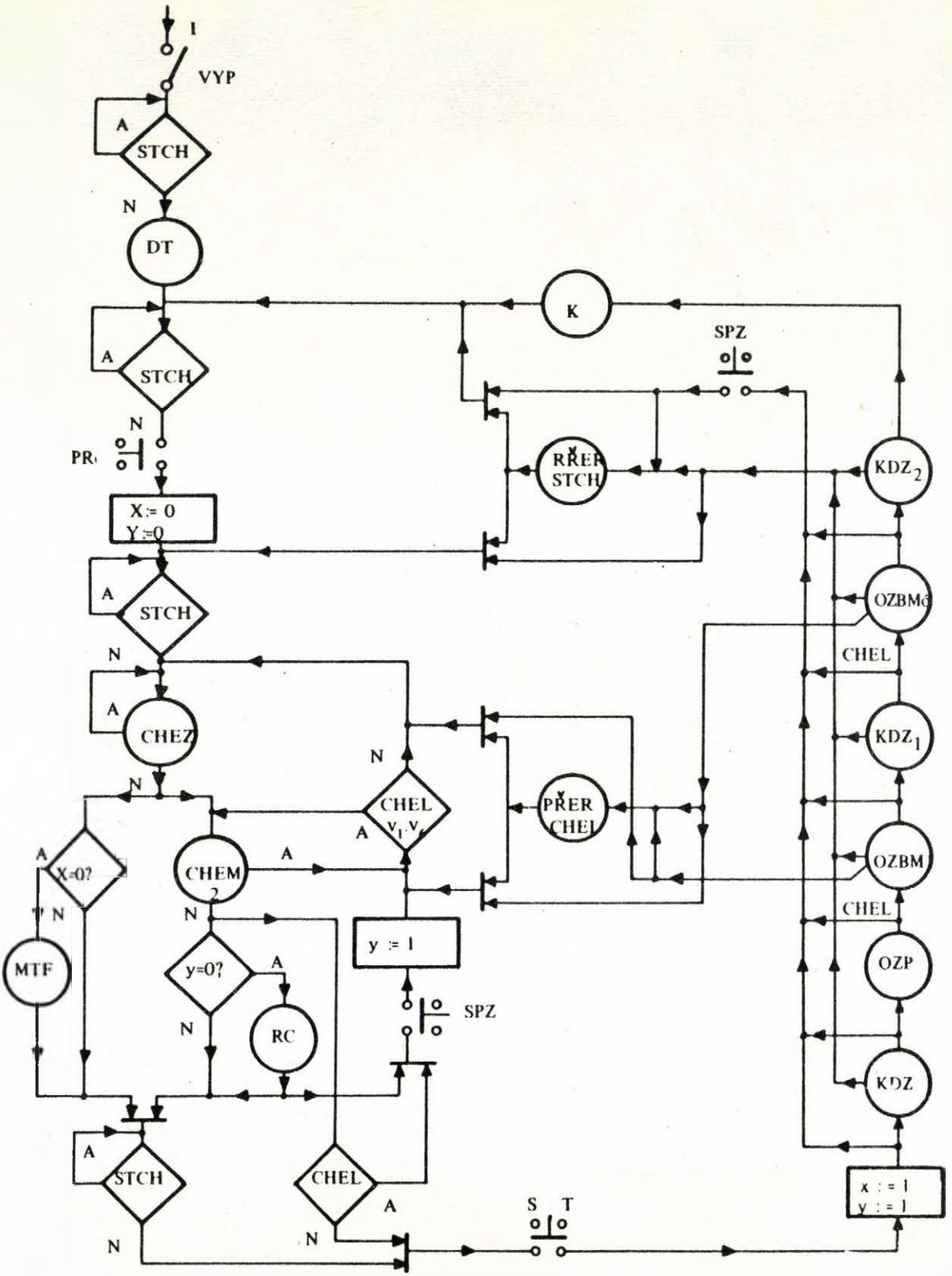


Рис. 6.

Принципиальная блок-схема функционирования автоматического ЭКГ

ки может происходить в случае прерывания (ПР) или в случае нормального окончания цикла (КЦ). Причинами прерывания могут быть автоматически определяемые неисправности в работе АЭ и нажатие некоторых кнопок на панели. К автоматически обнаруживаемым неисправностям относятся следующие :

- неправильное подключение электродов к пациенту (НПЭ z, НПЭ M_2)
- погрешности дрейфа и коэффициента усиления усилителей (ПДКУ),
- заканчивание ленты магнитофона итд.

Прерыванием автоматического цикла мы предотвращаем изготовление ошибочной электрокардиограммы.

Для объяснения фазы подготовки необходимо подчеркнуть следующие моменты:

а) вызов фазы путем нажатия освещенной кнопки "Подготовка" (ПДГ),
б) параллельное протекание операции измерения частоты пульсации сердца (ЧП) с операциями обнаружения неправильного подключения электродов (НПЭ) к пациенту и вложения даты рождения (ГР),
в) применение дисплея панели в различных операциях подготовительной фазы.

Пояснение сущности фазы автоматического цикла (АЦ) :

а) вызов фазы проводится путем нажатия освещенной кнопки "Старт" (СТ);
б) появление неисправности некоторой части прибора ведет к прерыванию АЦ
в) операции автоматического цикла осуществляются в следующей последовательности:
   1. Контроль погрешности дрейфа и коэффициента усиления усилителей АЧ (ПДКУ)
   2. Запись содержания памяти (ЗП) идентификационных данных

3. Запись биопотенциалов в состоянии M_1
4. Запись биопотенциалов в состоянии M_2
5. Конец цикла (КЦ).

Из того, что было сказано о действии и связи АЭ с обслуживающим персоналом вытекает, что АЭ вместе с обслуживающим персоналом образуют простую автоматизированную систему управления, поскольку в этой системе предполагается взаимодействие человека с автоматом.

Для того, чтобы объяснить, каким образом была осуществлена реализация выше описанного алгоритма, следует остановиться на структуре операционной и управляющей частей управляющего автомата. Структура операционной части определена составом регистров, анализаторов неисправностей, декодеров и некоторых других блоков и схемой их соединения. Структура управляющей части фактически определена подробной блок-схемой алгоритма управления в соответствии с рис. 6. Например, осуществление двух последовательных операций O_i, O_{i+1} показано на рис.7б.

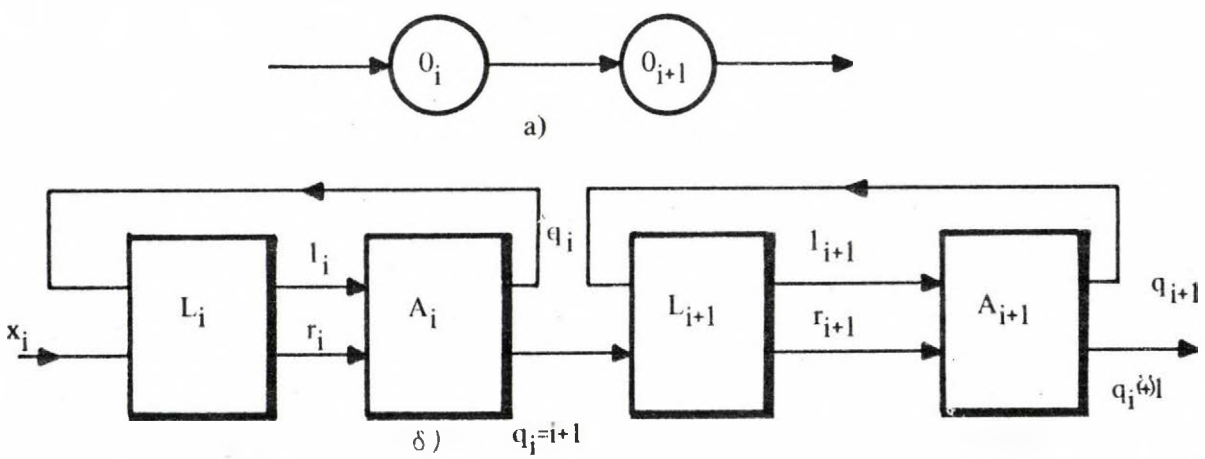


Рис. 7.

В данном случае мы предполагаем, что A_i и A_{i+1} представляют взаимонезависимые компоненты операционной части автомата, операционные модули L_i и L_{i+1} - специ-

альные управляющие модули. Работа этих управляющих модулей характеризуется временной диаграммой, указанной на рис.8.

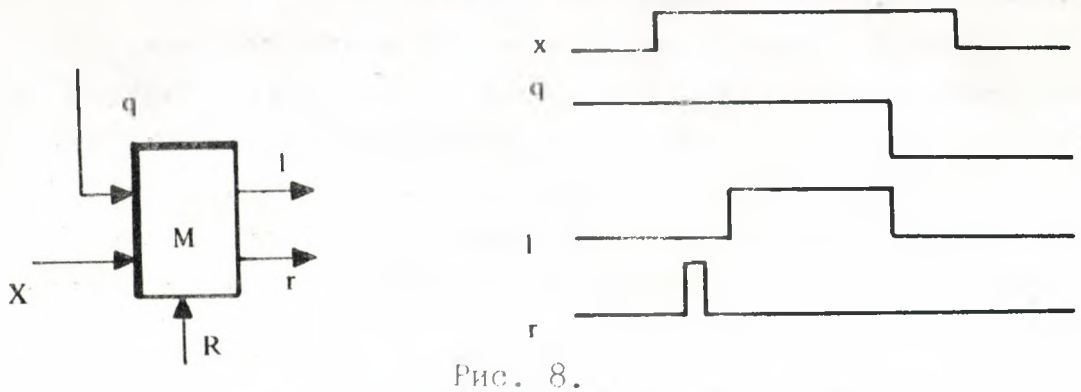


Рис. 8.

Импульс r_i должен привести соответствующий блок A_i осуществляющий операцию Q_i в начальное состояние (очистка регистров, начальное состояние триггеров, счетчиков и т.п.). Операция O_i в соответствующем блоке A_i осуществляется фактически при сигнале l=I.
После окончания операции значение $\bar{q}$ переходит с 1 → 0, что вызывает переход l: 1 → 0.

Управляющие модули работают в управляющей цепи последовательно так, что сначала работы изменится вход первого модуля x_1 с 0 на 1 и в этом состоянии вход модуля должен остаться, пока не нужно работу автомата начать снова. После окончания первой операции выход q_1 соответствующего операционного модуля изменится тоже из 0 → 1 и остается постоянно в этом состоянии. Изменение q_1: 0 → 1 является условием для изменения x_2: 0 → 1. Этого свойства можно использовать тоже для индикации тех операций, которые закончились. Таким образом с входа управляющего автомата постепенно расширяется в управляющей цепи единица на выход в следствии с последовательным окончанием предыдущих операций, пока не закончится последняя операция в управляющей цепи.

Следующим нажиманием подходящей кнопки ПАГ, СТ,

можно соответствующую управляющую цепь операций (смотри рис.5) спускать снова таким образом, что все операционные модули приводятся в нулевое состояние (кроме входного сигнала соответствующей части цепи).

Принцип реализации некоторого множества безусловно зависимых от времени синхронных операционных модулей A_i показан на рис.9. В таком операционном модуле счетчик насчитывает синхронизационные импульсы h и в зависимости от состояния счетчика генерируются декодером Д желаемые для осуществления и окончания операции А сигналы.

Состояние реализации прототипа.

В настоящее время разработка автомата закончена и проводятся его клинические испытания. Одновременно производится подготовка документации для введения установки в производство.

Внешний вид автоматического электрокардиографа представлен на предлагаемых фотографиях.

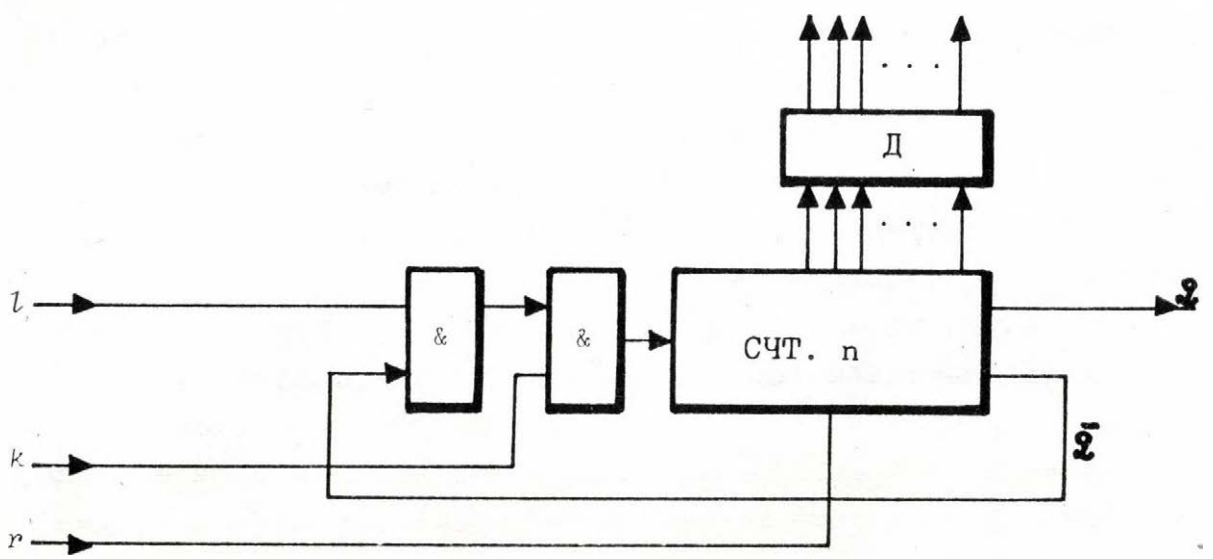


Рис. 9.

Возможные направления дальнейших работ.

а) При использовании АЭ в автономном режиме (off-line) в АСУ здравоохранения (субсистема диагностики и предупреждения заболеваний сердца) было бы желательно производить запись на магнитофонную ленту не в аналоговой форме, а в цифровой. Кроме того желательно, чтобы эти магнитофонные ленты можно было использовать непосредственно в магнитоленточных памятях вычислительных машин. Для этого было бы целесообразно снабдить АЭ аналого-цифровым преобразователем.

б) Реализация выше описанного алгоритма была бы возможна путем применения подходящего микропроцессора.

в) Съемка и запись электрокардиограмм является довольно медленным процессом, поэтому было бы целесообразно соединить АЭ с подходящим процессором для проведения предварительного анализа электрокардиограмм. Таким образом бы анализ электрокардиограмм происходил в месте съема и во время его проведения. На магнитофонную ленту можно было бы записывать лишь результирующую диагностическую информацию в сжатой форме.

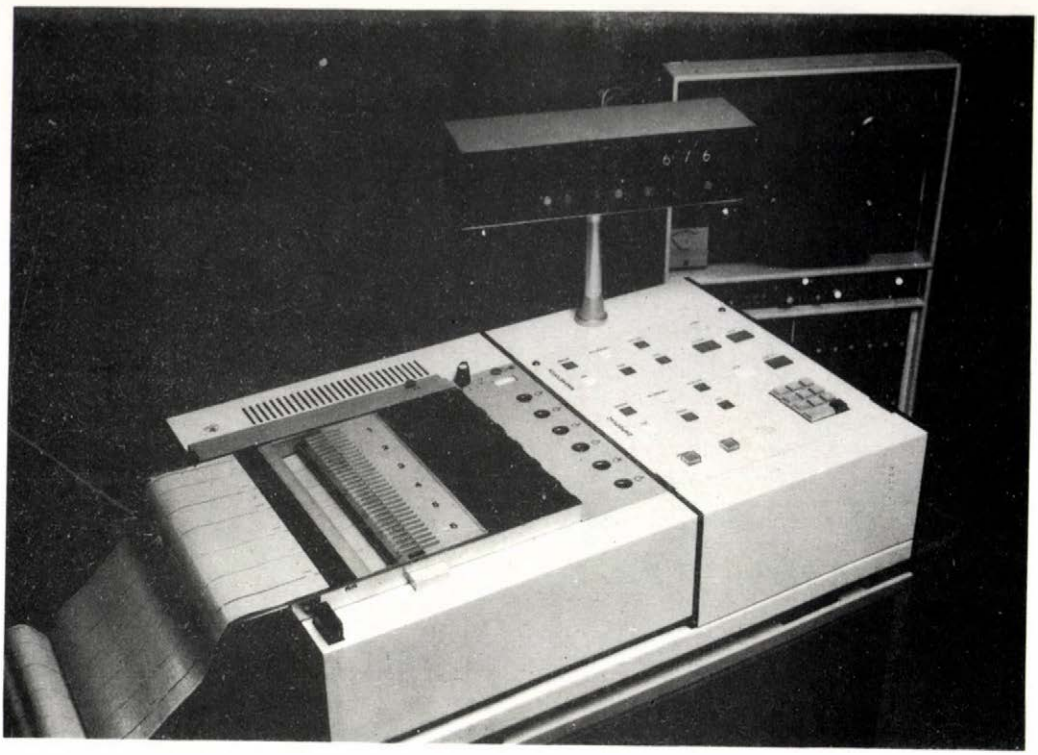
Литература :

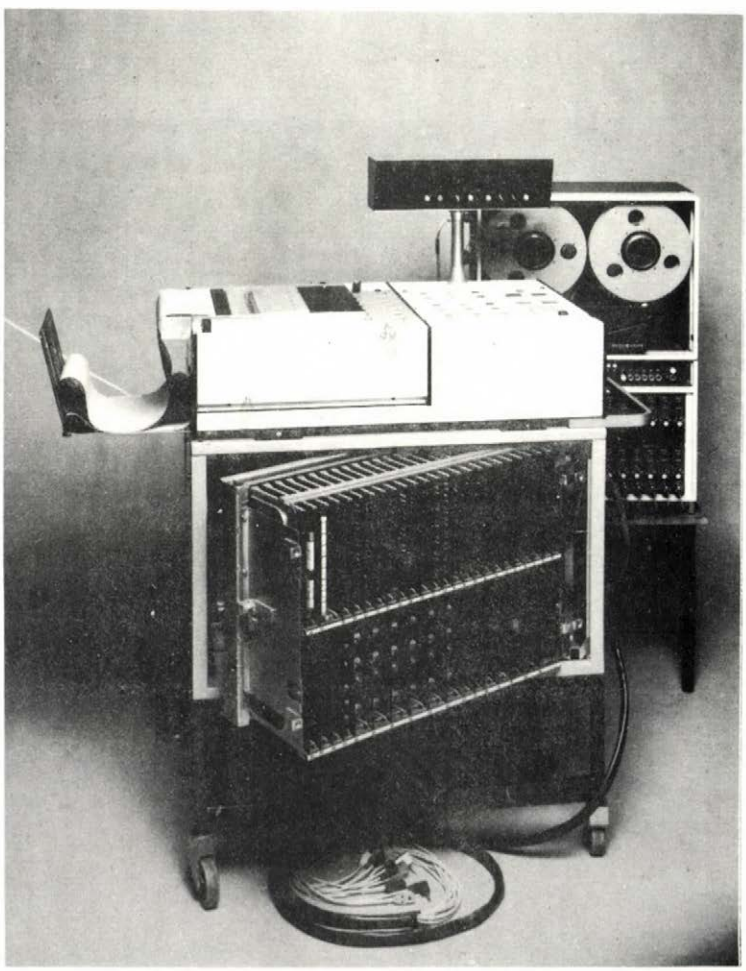
1. Nevanský R.: Diskrétní část automatického elektrokardiografu I. Algoritmus řízení a struktura realizace.
Научный отчет № ÚVVTŘ 75-032 011, ИИВТУ Прага 1975.

2. Nevanský R., Matula Z., Šretýř M.: Automatický EKG jako součást ASŘ zdravotnictví. Systémová studie.
Научный отчет № ÚVVTŘ 75-032 005, ИИВТУ Прага 1975.

3. Šretýř M.: Konečný automat pro řízení elektrokardiografu II. Analogová část. Научный отчет № ÚVVTŘ 73-125 002, ИИВТУ Прага 1973.


CHIKANA





СРАВНЕНИЕ ВЛИЯНИЯ КРИТЕРИЕВ S И R НА СЛОЖНОСТЬ СТРУКТУРЫ ЛОГИЧЕСКОЙ СЕТИ

Ян Коленичка

Кафедра вычислительных машин ПИ Брно, ЧССР

1. Введение

В работах [1,2] описывается метод поиска минимальной или близкой к минимальной формы неполностью определенной булевой функции. Метод предназначен для функций с большим количеством независимых логических переменных и обозначается тем, что множество всех заданных независимых переменных редуцируется на меньшее множество переменных. Это производится так, что из множества всех независимых переменных выбираются только такие переменные, которые по наибольшей вероятностьи будут в конечной минимальной форме. Собственная минимизация потом происходит тольло с этим редуцированным подмножеством независимых переменных, вследствие чего процесс минимизации значительно ускоряется.

В приведенных работах [1,2] для подбора переменных в редуцированное множество применяются критерии S_1 или S_0, которые для переменных $x_i \in \{x_1, x_2, \ldots, x_n\}$ определены следующим образом:

$$S_{1i} = \frac{n_1^1 \; n_0^0}{n_1^1 + n_0^0}$$

где n_1^1 - число рабочих состояний, в которых рассматриваемая переменная x_i принимает значение единицы, и

n_0^0 - число нерабочих состояаний, в которых рассматриваемая переменная x_i принимает значение нуля.

Величину S_1 будем называть критерием неинверсности переменной.

Аналогично величине S_1 был определен критерий инверсности S_0 :

$$S_{oi} = \frac{n_1^0 \, n_0^1}{n_1^0 + n_0^1}$$

где n_1^0 - число рабочих состояний, в которых переменная x_i принимает значение нуля , и n_0^1 - число нерабочих состояний, в которых переменная x_i принимает значение единицы.

В рассматриваемом методе предполагается, что неполностью определенная логическая функция $y=F(x_1,x_2,\ldots,x_n)$ задана таблицей состояний (обозначим ее напр. T1), которая разделена в два поля: множество рабочих состояний образует рабочее поле и подобно тому множество нерабочих состояний образует нерабочее поле функции. На первом этапе минимизации -коротко говоря- надо в табл.T1 искать обязательные переменные, столбцы которых потом надо из табл. T1 выбрать и включить в новую таблицу T2. Для каждого состояния табл. T2 потом надо проверить, находится ли оно в противоположном поле этой таблицы. Если его там нет, то соответсвующую строку в табл. T1 можно вычеркнуть. Но если обязательные переменные не существуют, или еще все строки табл. T1 не вычеркнуты, то новый столбец табл. T2 надо выбрать по максимальномузначению критериев S_{1i} или S_{oi} (S_{oi} тогда, когда мы хотим, чтобы в конечной схеме было наименшее число инверторов). Опять проверяется каждое состояние табл. T2 (с этим новым столбцом), находится ли оно в противоположном поле таблицы. Если его там нет, мы можем вычеркнуть соответствующую строку из табл. T1. Этот процесс создания табл. T2 повторяется до тех пор, пока не все строки из табл. T1 вычеркнуты.

Результатом этого процесса является то, что табл. T2 содержит только редуцированное множество k независимых логических переменных, при чем $k \leq n$. Создание минимальной формы происходит потом только при помощи табл. T2.

Автор настоящей статьи создал в [3] критерий R, который в рассматриваемом методе заменяет критерий S, и показал, что число переменных в табл. T2, определенноепри помощи критерия R может быть меньше чем при помощи критерия S.

Значение критерия R_i для переменной $x_i \in \{x_1,x_2,\ldots,x_n\}$ можно вычислить очень легко: оно равно числу состояний, которые мы можем вычеркнуть из табл. T1 включением переменной x_i

в табл. T2. В качестве нового столбца мы в табл. T2 окончательно включим столбец той переменной, у которой значение критерия R максимальное. Иначе, алгоритм минимизации с критерием R не отличается от алгоритма с критерием S.

Как мы уже сказали, собственная минимизация происходит потом известным способом (Гаврилов) только при помощи табл. T2. В результате этого мы получим логическую сеть, структуру которой можно в общем виде показать например на рис. 1, где 3-ий логический уровень представляет собой реализацию дополнительных функций (которые в рассматриваемом методе могут возникнуть в течение минимизации).

Замечание: Если мы предполагаем, что число входов логических элементов, которыми можно созданную минимальную форму реализовать, ограничено, то каждый логический уровень распадается в несколько подуровней, которые мы назовем ступенями; потом тоже число логических элементов на одном уровне будет больше. (В небольшой ниже приведенной статистической работе мы предполагали, что максимальное допустимое число входов элемента равно восьми).

2. Сравнение влияния критериев

В следующих абзацах мы хотим показать влияние критериев S и R на некоторые избранные параметры. Нам интересно, как проявится наличие критериев S и R в отношении к числу логических переменных в редуцированной таблице T2, далее какое влияние имеют эти критерии на общее число логических элементов в конечной логической сети, на число элементов в отдельных уровнях и на время вычисления конечной (минимальной) формы изображения логической функции.

В качестве основы этой малой статистики мы взяли 25 логических булевых функций, таблицы состояний которых случайно генерировались вычислительной машиной и которые были упорядочены по повышающейся сложности исходной таблицы состояний. Сложностью таблицы мы будем понимать произведение числа независимых переменных на число их состояний. Сложность у самой простой таблицы была 77 (в примерах она обозначена как задача 1),

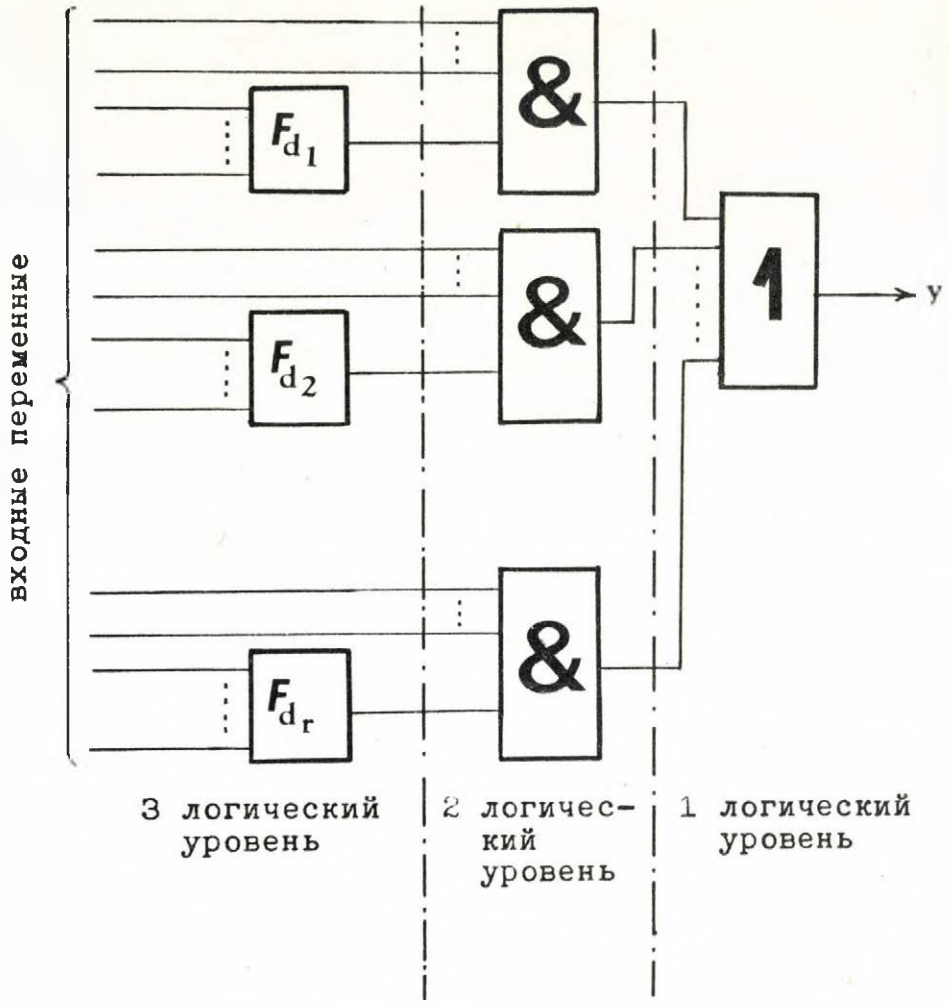


Рис. 1.

у самой сложной потом 4800 (в примерах она несет номер 25).

2.1. Число логических переменных в таблице T2

В тавл.1 находится для отдельных задач отношение x_S/x_R где x_S представляет собой число лог. переменных в табл. T2, построенной при помощи критерия S, и аналогично x_R представляет собой число переменных в тавл. T2, построенной при помощи критерия R.

Как мы видим, только в трех случаях одинаково число переменных в тавл. T2 по критериям R и S; в остальных случаях число переменных в табл. T2 больше по критерию S чем по критерию R.

2.2 Число всех логических элементов в конечной логической сети

На рис. 2 показано абсолютное число всех логических элементов в конечной логической сети для отдельных задач. Задачи упорядочены по повышающейся сложности. Интересно, что от сложности 880 уже в каждой из приведенных задач число всех элементов в сети, построенной при помощи критерия R, меньше чем в сети, построенной при помощи критерия S.

номер задачи	x_S/x_R	среднее значение
1	1,34	
2	1,25	
3	1,25	
4	1	
5	1,50	
6	1,16	
7	1,20	
8	1	
9	1,16	
10	1	
11	1,20	
12	1,45	1,25
13	1,50	
14	1,16	
15	1,75	
16	1,25	
17	1,30	
18	1,12	
19	1,30	
20	1,12	
21	1,34	
22	1,26	
23	1,11	
24	1,20	
25	1,23	

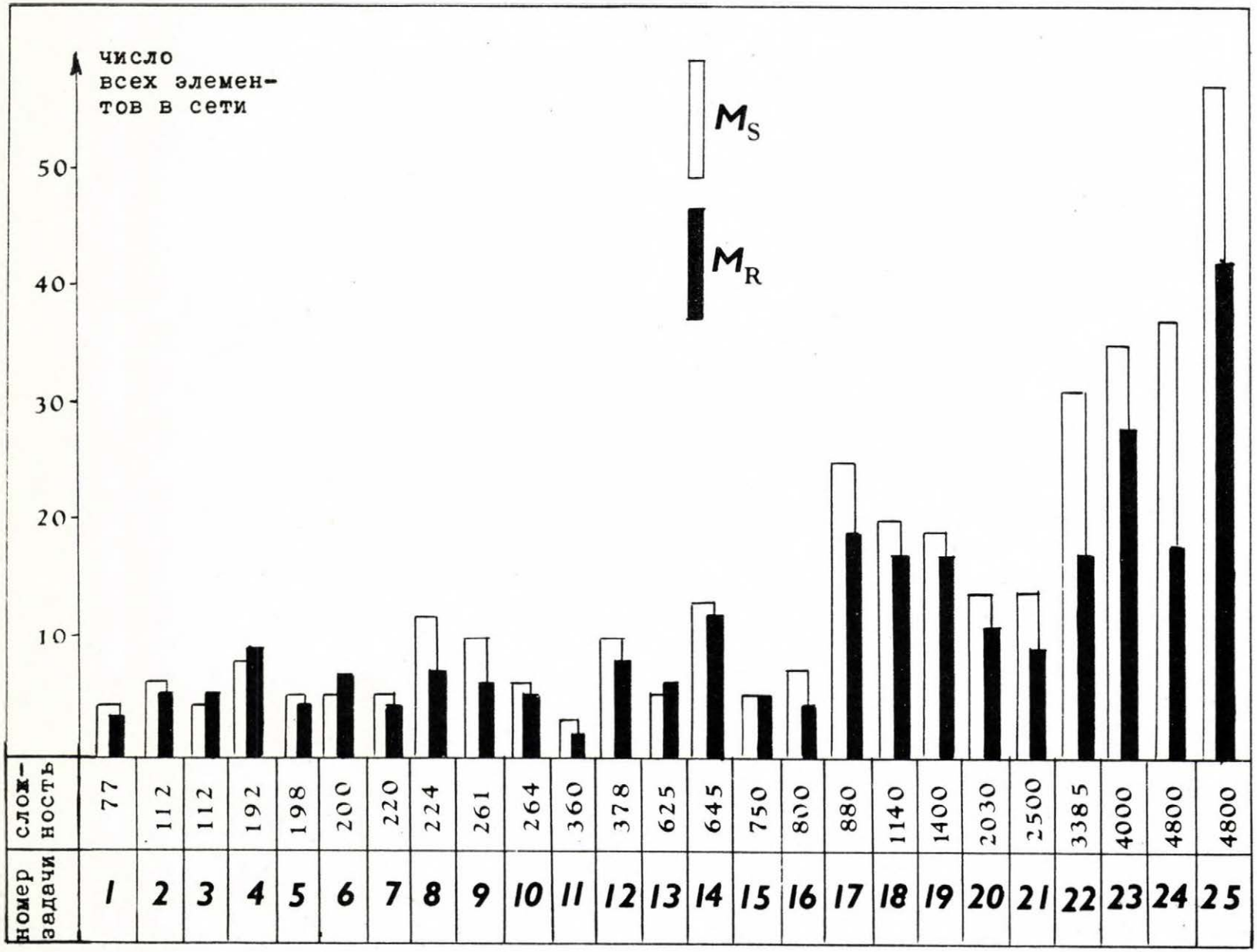
Табл.1

Для отдельных задач обозначим число всех элементов в сети, созданной при помощи критерия S, символом M_S и аналогично число всех элементов в сети, созданной при помощи критерия R, символом M_R. Потом мы можем вычислить отношение M_S/M_R для каждой задачи, которое приведено в табл.2. Как видно, среднее значение этого отношения равно 1,3 и свидетельствует о том, что при использовании критерия R мы поличим в большинстве случаев более простую логическую сеть.

2.3 Число логических ступеней

В табл.3 находится перечень чисел всех логических ступеней в конечной логической сети для обоих критериев. Число всех ступеней в сети созданной при помощи критерия S обозначено символом P_S и аналогично введен символ P_R.

Как видно, частное P_S/P_R = 1,288 говорит опять в пользу критерия R, который в большинстве случаев позволяет построить сеть с меньшим числом ступеней, следовательно тоже с меньшей задержкой.



номер задачи	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
сложность	77	112	112	192	198	200	220	224	261	264	360	378	625	645	750	800	880	1140	1400	2030	2500	3385	4000	4800	4800

Рис. 2

номер задачи	M_S / M_R
1	1,33
2	1,20
3	0,80
4	0,88
5	1,25
6	0,75
7	1,25
8	1,74
9	1,67
10	1,20
11	1,67
12	1,27
13	0,82
14	1.08
15	1
16	1,75
17	1,32
18	1,17
19	1,15
20	1,17
21	1,55
22	1,84
23	1,25
24	1,27
25	2,05
среднее значение	1,3

Табл. 2

номер задачи	P_S	P_R	P_S / P_R
1	2	2	1
2	3	2	1,5
3	3	2	1,5
4	4	4	1
5	3	2	1,5
6	3	4	0,75
7	3	2	1,5
8	5	4	1,25
9	3	3	1
10	3	2	1,5
11	4	2	2
12	4	3	1,33
13	3	3	1
14	4	4	1
15	5	5	1
16	3	2	1,5
17	6	5	1,2
18	5	5	1
19	4	4	1
20	4	3	1,33
21	5	3	1,66
22	6	4	1,5
23	6	5	1,2
24	6	4	1,5
25	8	6	1,33
среднее значение			1,288

Табл.3

2.4 Числа элементов на отдельных уровнях

Обозначим число элементов на 1 и 2 уровнях сети созданной при помощи критерия S символом $M_{1,2,S}$ и аналогично $M_{1,2,R}$. В табл.4 представлен перечень значений отношения $M_{1,2,S}/M_{1,2,R}$ среднее значение которого равно 0,970 и свидетельствует о том, что числа логических элементов на 1 и 2 уровнях в сетях созданных при помощи критериев S, R не намного отличаются. Но мы тоже видим, что от задачи 18 (т.е. от сложности около 1100) уже число элементов по критерию R постоянно больше на первых уровнях.

Обозначим аналогично числа элементов на 3. уровне (т.е. уровень реализации дополнительных функций) через символы $M_{3,S}$ и $M_{3,R}$. Как видно из табл.4, среднее значение частного

$M_{3,S}/M_{3,R}$ = 2,08 - следовательно, реализация дополнительных функций является - что касается числа элементов - в среднем приблизительно в два раза сложнее по критерию S. Начиная задачей 18 (т.е. сложностью 1100) уже среднее значение этого отношения равно 2,9 и мы можем предполагать, что с повышающейся сложностью задачи оно будет еще возрастать. Из этого вытекает, что на основе использования критерия R мы можем получить более простую реализацию дополнительных функций (на 3-ем уровне) и этот результат соответствует результату у числа логических ступеней (табл.3).

номер задачи	$M_{1,2,S}/M_{1,2,R}$	$M_{3,S}/M_{3,R}$
1	1,33	- неопред.
2	1,0	-
3	0,6	-
4	1,0	0,75
5	1,0	-
6	1,0	0,33
7	1,0	-
8	1,66	1,75
9	1,0	5,0
10	1,0	-
11	1,5	-
12	0,85	2,5
13	1,0	0,0
14	1,0	0,11
15	0,75	-
16	1,5	-
17	1,0	1,50
18	0,65	2,00
19	0,75	1,50
20	0,75	2,75
21	0,82	7,00
22	0,9	3,00
23	0,82	1,50
24	0,56	1,56
25	0,82	3,65
среднее значение	0,970	2,08

Табл. 4

2.5 Время вычисления

В табл. 5 находится перечень времени вычисления конечной формы заданных функций для отдельных задач при использовании ЦВМ САЛБ Д-21. Символ T_S обозначает машинное время при создании формы по критерию S и аналогично введен символ T_R.

	1	2	3	4	5	6
номер задачи	T_S [s]	T_R [s]	T_S / T_R	сложность	ΦT_S [s]	ΦT_R [s]
1	10	7	1,43	77	10	7
2	15	9	1,66	112	13	8,7
3	11	7	1,57	112		
4	14	10	1,40	192		
5	12	9	1,33	198		
6	34	56	0,61	200	22,1	17,7
7	18	11	1,64	220		
8	25	14	1,79	224		
9	20	11	1,82	261		
10	14	11	1,27	264		
11	14	11	1,27	360		
12	30	10	3,00	378		
13	14	11	1,27	625	34	27,4
14	18	21	0,86	645		
15	68	58	1,21	750		
16	40	25	1,60	800		
17	30	22	1,36	880		
18	118	145	0,81	1140	112	205
19	105	235	0,45	1400		
20	130	320	0,41	2030		
21	95	120	0,79	2500		
22	345	512	0,67	3385	331	610,5
23	250	600	0,42	4000		
24	455	880	0,52	4800		
25	275	450	0,61	4800		

Табл. 5

В 3-м столбце табл. 5 представлен перечень значений частного T_S/T_R и эти значения тоже изображены на рис.3. В 4-ом столбце таблицы 5 записан перечень значений сложности исходной таблицы состояний отдельных логических функций и задачи разделены в группы по сложности следующим образом:

В столбце 5 потом приведены средние значения времени вычисления для отдельных групп при применении критерия S и обозначены символом ΦT_S ; в столбце 6 аналогично находятся значения ΦT_R. Эти значения потом изображены на рис. 3 и соединены в кривую.

Из. рис.3 мы наглядно видим, что сложность около 1000 опять является значительной, потому что при больших значе-

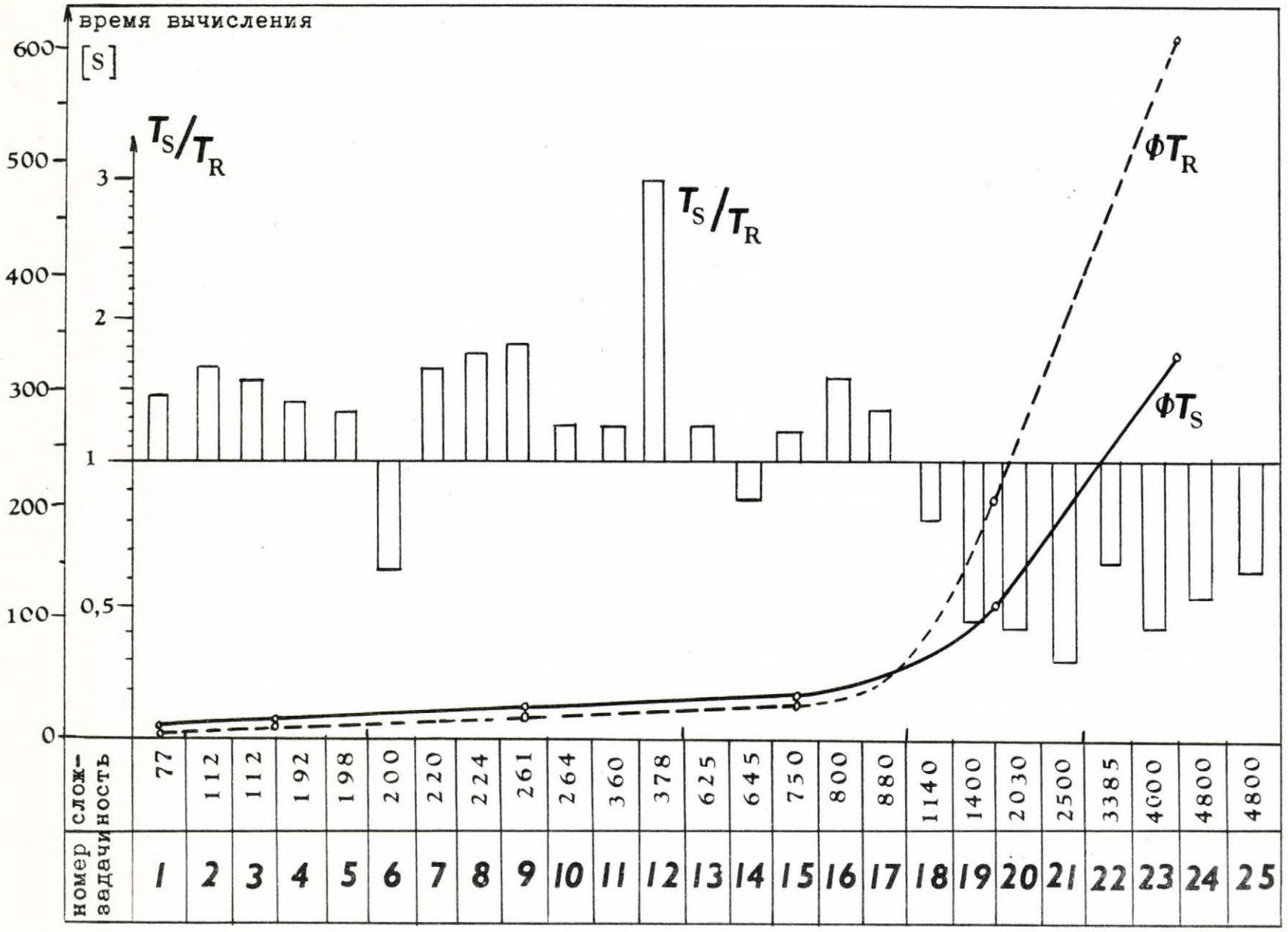

время вычисления
[S]
600
500
400
300
200
100
0
3
2
1
0,5
T_S/T_R
T_S/T_R
ΦT_R
ΦT_S
номер задачи
сложность
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
77 112 112 192 198 200 220 224 261 264 360 378 625 645 750 800 880 1140 1400 2030 2500 3385 4000 4800 4800

Рис. 3

ниях сложности среднее значение времени вычисления $ФТ_R$ уже больше чем $ФТ_S$ и оно тоже повышается быстрее.

3. Заключение

Из всего выше сказанного вытекает, что при использовании критерия S может быть в среднем:

1) число переменных в таблице T2 приблизительно на 25% больше чем при использовании критерия R ;
2) число всех логических элементов в конечной сети приблизительно на 30% больше чем по критерию R ;
3) число логических ступеней приблизительно на 30% больше чем по критерию R;
4) число логических элементов на 1-ом и 2-ом уровнях конечной сети может быть меньше чем по критерию R , но на 3-ем уровне оно уже в два раза и больше превосходит критерий R;
5) время вычисления до сложности около 1000 говорит, правда, в пользу критерия R , но различия не являются значительными; однако у сложности над 1000 время вычисления по критерию S уже в среднем на половину ниже времени, достигнутого при использовании критерия R.

Литература:

1) Гаврилов,М.А.; Копыленко,В.М.: Синтез структуры бесконтактных релейных устройств. Bull.math.de la Soc.Sci math. de la R.S.R. 10, N° 3, 1966

2) Вострова,З.И. Копыленко,В.М.: Критерий направленного поиска структур, приближающихся к монотонности, с минимальным числом инверторов на входах. Сборник "Элементы и методы синтеза дискретных информационно-логических систем", Фрунзе, 1969.

3) Коленичка, Ян: Алгоритмический синтез комбинационной логической схемы. Кандидатская диссертация, кафедра ЭВМ. Политехнический институт Брно, 1972.

УДК 62-507

СИНТЕЗ СТРУКТУР МНОГОФУНКЦИОНАЛЬНЫХ ЛОГИЧЕСКИХ МОДУЛЕЙ

Э.А. Якубайтис, А.Я. Калнберзинь, В.П.Чапенко

(СССР, Рига)

Технология изготовления больших интегральных схем (БИС) оказывает существенное влияние на принципы проектирования логических модулей для дискретных автоматов. Модули, выполненные в виде БИС, содержат большое количество компонент,функционально связанных между собой, и могут реализовать сложные логические функции. Вместе с тем технологические соображения и требования надежности накладывают ограничения на количество внешних выводов модуля. На первый план выдвигается требование минимизации числа внешних выводов, даже за счет введения некоторой структурной избыточности в схему модуля.Однако усложнение логической схемы ведет к "специализации" модуля,уменьшению их серийности и, следовательно, к увеличению стоимости БИС. Избежать этой отрицательной тенденции позволяет применение многофункциональных логических модулей (МЛМ), настраиваемых извне на выполнение той или иной требуемой логической функции [I,2].Такие модули должны обладать способностью реализации большого числа функций при возможно меньшем числе внешних вводов. Частным случаем МЛМ являются универсальные логические модули, настраиваемые на реализацию любой функции некоторого числа переменных.

На основе МЛМ возможно строить дискретные автоматы, обладающие следующими положительными свойствами: а) высокая повторяемость модулей в автомате; б) адаптивность логических автоматов, достигаемая настройкой или "обучением"; в) высокая "живучесть" автоматов, базирующаяся на том, что модули могут иметь структуру, обеспе-

чивающую при появлении неисправностей лишь некоторое уменьшение числа реализуемых функций; г) облегчается контроль и диагностика неисправностей.

Рассмотрим схему МЛМ, имеющую g входов, один выход и описываемую булевой функцией

$$Z = \varphi(B_1, B_2, \ldots, B_g). \qquad (1)$$

Назовем настройкой операцию замены каждого аргумента B_k, $k = 1, 2, \ldots, g$, функции (1) некоторым элементом из конечного множества:

$$\mathcal{F} = \{A_1, \bar{A}_1, A_2, \bar{A}_2, \ldots, A_n, \bar{A}_n, 0, 1\},$$

где A_i, $\bar{A}_i$ – двоичные переменные, $n < g$.

Пусть задан список v различных булевых функций:

$f_1(A_1, A_2, \ldots, A_n)$, $f_2(A_1, A_2, \ldots, A_n)$, $\ldots$, $f_v(A_1, A_2, \ldots, A_n)$,

где $1 < v \leq 2^{2^n}$. Предположим, что рассматриваемый МЛМ реализует любую из этих функций при соответствующей настройке. Это означает, что при соответствующей настройке для каждой функции f_i выполняется условие

$$\varphi(B_1, B_2, \ldots, B_g) = f_i(A_1, A_2, \ldots, A_n). \qquad (2)$$

Исходя из специфических требований к МЛМ, при синтезе структуры МЛМ желательно найти такую функцию φ, которая содержит минимальное число g аргументов и удовлетворяет при этом условию (2) для каждой функции f_i из списка. Однако для поиска структуры МЛМ с минимальным числом входов (то есть для точного решения задачи) требуется провести большое количество вычислений, трудно реализуемое даже на современных вычислительных машинах [4]. Поэтому ниже рассматривается приближенное решение этой задачи, состоящее в нахождении структуры МЛМ, реализующего любую функцию f_i из заданного списка, с числом входов, близким к минимальному.

Минимальное число входов МЛМ находится в пределах:

$$g_m = n +]\log_2 v[\geq g_{min} \geq g_k, \qquad (3)$$

где $g_к$ определяется как наименьшее значение g, при котором количество различных разрешенных настроек не меньше числа v [5].

Следует отметить, что МЛМ, реализующий заданное множество булевых функций, всегда можно реализовать с использованием двух внешних настроечных входов, внеся в структуру модуля элементы памяти, образующие, например, $(g_m - n)$ -разрядный регистр сдвига и запоминающие сигналы настройки 0 и 1. Этот метод обсужден в работе [3], мы же будем рассматривать схемы МЛМ без памяти.

Точное решение задачи синтеза МЛМ состоит [4,5] в последовательном рассмотрении значений $g = g_к, g_к + 1, \ldots$ При фиксированном значении g каждой функции f_i $(i = 1, \ldots, v)$ поставляется множество $M_i = \{\varphi^i_1, \varphi^i_2, \ldots, \varphi^i_{N(g,n)}\}$ недоопределенных функций $\varphi^i_j(B_1, \ldots, B_g)$, где $j = 1, 2, \ldots, N(g,n)$ - номер настройки. Любая функция φ^i_j не определена на $2^g - 2^n$ наборах аргументов и такова, что при j-й настройке выполняется условие $\varphi^i_j(B_1, \ldots, B_g) = f_i(A_1, \ldots, A_n)$. Две функции $\varphi^{i_1}_{j_1}$ и $\varphi^{i_2}_{j_2}$ $(i_1 \neq i_2;\ i_1, i_2 = 1, \ldots, v;\ j_1, j_2 = 1, \ldots, N(g,n))$ называются совместимыми, если множество всех наборов, на которых функция $\varphi^{i_1}_{j_1}$ равна I(0), не пересекается с множеством всех наборов, на которых функция $\varphi^{i_2}_{j_2}$ равна 0(I) соответственно. Синтез МЛМ сводится к поиску при каждом рассматриваемом значении g множества попарно совместимых функций φ^i_j, включающего в себя по одной функции из каждого множества M_i. Объединение найденных функций $\varphi^1_{j_1}, \ldots, \varphi^v_{j_v}$ в одну функцию $\varphi^{1,\ldots,v}_{j_1,\ldots,j_v} = Z(B_1, \ldots, B_g)$ дает структуру искомого МЛМ.

В работе [5] предложена последовательность синтеза структуры МЛМ с минимальным числом входов, состоящая при каждом фиксированном значении g в нахождении множества $\mathcal{M}$ всех максимально совместимых множеств функций φ^i_j и поиске кратчайшего покрытия множества $\{M_1, M_2, \ldots, M_v\}$ некоторым подмножеством множества $\mathcal{M}$.

Очевидно, при больших значениях величин g, $N(g,n)$ и v поиск всех максимально совместимых множеств связан с большим перебором, который не всегда осуществим даже при использовании вычислительной машины. Поэтому в ряде случаев целесообразно ограничить список настроек и искать приближенное решение задачи покрытия методами направленного перебора. Направленности перебора можно достичь за счет использования некоторых критериев выбора функций и настроек на каждом шаге решения.

Рассмотрим процедуру синтеза, основанную на поиске некоторого максимально совместимого множества $M_u' \in \mathcal{M}$, по возможности большей мощности. Предлагаемая процедура является пошаговой, и каждый k-й шаг заканчивается выбором некоторой функции $\varphi_{j_k}^{i_k}$, включаемой в искомое множество совместимых функций M_u'. Выбор каждой следующей функции $\varphi_{j_k}^{i_k}$ и объединение её со всеми выбранными на предыдущих шагах функциями $\varphi_{j_1}^{i_1}, \ldots, \varphi_{j_{k-1}}^{i_{k-1}}$ доопределяют значения функции $\varphi_{j_1,\ldots,j_{k-1}}^{i_1,\ldots,i_{k-1}}$ на некоторых наборах, разрешенных j_k-й настройкой. Критерием выбора функции $\varphi_{j_k}^{i_k}$, включаемой в искомое множество M_u', может служить минимальное число доопределяемых на k-м шаге значений функции $\varphi_{j_1,\ldots,j_{k-1}}^{i_1,\ldots,i_{k-1}}$. Это обусловлено тем, что функция $\varphi_{j_1,\ldots,j_{k-1}}^{i_1,\ldots,i_{k-1}}$, имеющая большее число неопределенных значений, дает больше возможностей для выбора совместимой с ней функции $\varphi_{j_k}^{i_k}$.

Рассмотрим таблицу, состоящую из 2^g строк и $N(g,n)$ столбцов. Строки таблицы соответствуют различным наборам значений переменных $B_1, \ldots, B_g$, а столбцы - рассматриваемым настройкам. Элемент таблицы, на пересечении строки l и столбца j, каким-либо образом отмечается, если l-й набор является разрешенным при j-й настройке. В каждом столбце j отмечены (обведены замкнутой линией) все элементы, соответствующие наборам, на которых может быть определена функция φ_j^i. Таблицу рас-

сматриваемого вида назовем таблицей совместимых функций и будем ее использовать при синтезе МЛМ для выбора и записи совместимых функций φ_j^i.

Процедура поиска множества $M_u' \in \mathcal{M}$, использующая указанный выше критерий, может выполняться следующим образом.

Принимается значение $g = g_n$ и строится таблица совместимых функций. Выбирается функция f_1 из списка и находится соответствующая ей функция φ_1'. Заданные значения функции φ_1' записываются в отмеченные клетки первого столбца таблицы. Таким образом, принимается, что функция f_1 реализуется при первой настройке, и функция φ_1' включается в искомое множество M_u'. Каждое значение 0 и 1, записанное в отмеченную клетку некоторой строки, переписывается во все остальные отмеченные клетки данной строки. Затем подсчитывается вес (число отмеченных клеток, заполненных значениями 0 и 1) каждого столбца. Вес столбца указывает число наборов, на которых определена функция, задаваемая значениями 0 и 1 этого столбца. Рассматривается j-й столбец с максимальным весом. Проверяется, существует ли такая функция φ_j^i $(i = 2,3,\ldots,\nu;\ j = 2,3,\ldots,N(g,n))$, которую можно задать путем дописывания значений 0 и 1 в оставшиеся незаполненными отмеченные клетки j-го столбца. Если такая функция не существует, то есть если в j-м столбце нельзя представить функцию φ_j^i, то j-й столбец исключается из последующего рассмотрения. Тогда выбирается первый из нерассмотренных столбцов с максимальным весом и вновь проводится вышеуказанная проверка. Если же такая функция существует, то в соответствующие строки j-го и остальных столбцов дописываются её значения. Функция φ_j^i включается в искомое множество M_u'. Следовательно, функция f_i реализуется при j-й настройке. Затем вновь подсчитываются веса столбцов таблицы, выбирается столбец с максимальным весом и осуществляется вышеописанная проверка. На этом шаге не рассматри-

ваются столбцы, задающие значения найденных совместимых функций φ_1' и φ_j^i . Процедура итеративно продолжается до тех пор, пока для каждой функции f_i из списка не будет найдена соответствующая функция φ_j^i или пока не будут исключены из рассмотрения все столбцы таблицы.

В последнем случае возможны два продолжения решения задачи.

I) Значение g увеличивается на I и процедура повторяется или сначала, или с использованием предыдущего результата до тех пор, пока для каждой функции f_i не будет найдена функция φ_j^i .

2) Функции f_i , которым были сопоставлены найденные совместимые функции φ_j^i , удаляются из списка. Для оставшихся в списке функций вновь выполняется процедура синтеза при том же значении g , и т.д. до тех пор, пока не будут найдены множества совместимых функций M_1', M_2',..., M_k' $(k>1)$ такие, что их объединение $M_1' \cup M_2' \cup \ldots \cup M_k'$ включает в себя функцию из каждого множества M_i .Затем осуществляется объединение функций, входящих в каждое найденное множество M_u' $(u=1,2,\ldots,k)$. В результате образуются функции $\varphi_1', \varphi_2', \ldots, \varphi_k'$, где $\varphi_u' = \varphi_u'(B_1,\ldots,B_{g_k+t})$ - функция, полученная объединением всех функций, принадлежащих множеству M_u', $0 \le t <]\log_2 v[$. Далее находится функция

$$Z = \bigvee_{u=0}^{k-1} \varphi'_{u+1}(B_1,\ldots,B_{g_k+t}) B_{g_k+t+1}^{u_1} B_{g_k+t+2}^{u_2} \ldots B_{g_k+t+r}^{u_r} \qquad (4)$$

где $r =]\log_2 k[$; $(u_1 u_2 \ldots u_r)$ – двоичная запись числа u. Функция (4) описывает структуру МЛМ с числом входов, находящимся в пределах (3) [5].

Предлагаемая процедура позволяет найти решение с малыми затратами времени. Однако качество решения, получаемого при этом, зависит от порядка следования функций в списке и настроек. В частности, при $g=n+1$ представляется целесообразным в качестве первой настройки рассматривать подстановки вида $B_g = 0$ или $B_g = 1$. Для выбора I-й настройки возможен предварительный анализ функций f_i . Для

функции f_1 находятся $2n$ подфункций вида $f_1(A_1,\dots,A_n)|_{A_\ell=0}$ и $f_1(A_1,\dots,A_n)|_{A_\ell=1}$, $\ell=1,2,\dots,n$, полученных разложением функции f_1 по переменной A_ℓ. Для остальных функций f_i $(i=2,3,\dots,v)$ находятся подфункции $f_i(A_1,\dots,A_{n-1},0)$ и $f_i(A_1,\dots,A_{n-1},1)$. Среди $v-1$ подфункций $f_i(A_1,\dots,A_{n-1},0)$ может оказаться N_1 подфункций, совпадающих с подфункциями функции f_1. Аналогично, среди $v-1$ подфункций $f_i(A_1,\dots,A_{n-1},1)$ может быть N_2 подфункций, совпадающих с подфункциями функции f_1. Если $N_1>N_2$ $(N_1<N_2)$, то в качестве I-й настройки выбирается подстановка $B_g=0$ $(B_g=1)$ соответственно. В случае $N_1=N_2\neq 0$ выбирается любая из двух настроек. И, наконец, если $N_1=N_2=0$, то целесообразно в качестве первой функции из списка выбрать другую функцию. По аналогии с вышеприведенными соображениями предварительный анализ может быть проведен в случае $g=n+\eta$, $\eta>1$. Следует лишь отметить, что полный анализ по объему требуемых вычислений может оказаться соизмеримым с поиском всех максимальных множеств совместимых функций.

Пример. Задано IO функций, являющихся представителями различных типов бесповторных функций 4-х переменных:
$f_1=A_1A_2A_3A_4$, $f_2=A_1A_2A_3\vee A_4$, $f_3=(A_1A_2\vee A_3)A_4$, $f_4=A_1A_2\vee A_3\vee A_4$,
$f_5=(A_1\vee A_2)A_3A_4$, $f_6=(A_1\vee A_2)A_3\vee A_4$, $f_7=(A_1\vee A_2\vee A_3)A_4$,
$f_8=A_1\vee A_2\vee A_3\vee A_4$, $f_9=A_1A_2\vee A_3A_4$, $f_{10}=(A_1\vee A_2)(A_3\vee A_4)$.
Эти функции представлены в табл. I. Требуется найти структуру МЛМ вышеописанным методом с использованием настроек вида $B_k=0$, $B_k=1$ при фиксированной перестановке переменных $A_1,\dots,A_n$ на входах модуля. Число настроек заданного вида $N(g,n)=C_g^4\,2^{g-4}$ [2]. Условие $N(g,n)\geq v$ выполняется при $g_k=n+1=5$. Принимаем $g=5$ и строим таблицу совместимых функций (табл. 2). Настройки приведены в верхней части табл. 2. Например, во 2-м столбце приведена настройка $B_5=1$, $B_4=A_4$, $B_3=A_3$, $B_2=A_2$, $B_1=A_1$.

Для выбора I-й настройки рассмотрим таблицу I. Определяем числа подфункций $f_2\div f_{10}$, совпадающих с подфунк-

циями функции f_1: $N_1 = 4$, $N_2 = 0$. Так как $N_1 > N_2$,то в качестве первой выбираем настройку $B_5 = 0$, $B_4 = A_4$, $B_3 = A_3$, $B_2 = A_2$, $B_1 = A_1$ (I-й столбец табл.2).

Значения функции f_1 из таблицы истинности записываются в отмеченные клетки I-го столбца табл. 2. Таким образом, I-й столбец задает функцию φ_1^1 . Каждый символ 0 или I, записанный в некоторой строке I-го столбца, записывается также во все остальные отмеченные клетки данной строки. Эту операцию далее будем называть заполнением строк. Под каждым столбцом табл. 2 приведен его вес. Среди множества столбцов с максимальным весом выбираем 3-й столбец. Ищем функцию f_i $(i = 2,3,\ldots,10)$, значения которой совпадают с соответствующими значениями в отмеченных клетках 3-го столбца. Находим f_3, f_5, f_7 и выбираем f_5 . Дописываем недостающие значения функции f_5 в свободные отмеченные клетки 3-го столбца,получая таким образом, на 2-м шаге функцию φ_3^5 . Затем выполняем операцию заполнения строк и подсчет весов столбцов.Результат 2-го шага приведен в табл. 3.

Среди столбцов с максимальным весом рассматриваем 5-й столбец. Проверка показывает, что не существует функции φ_5^i $(i \neq 1,5)$, которая может быть представлена в 5-м столбце. Поэтому 5-й столбец исключается из дальнейшего рассмотрения. По той же причине исключается из рассмотрения и 6-й столбец. Для 7-го столбца находим, что его элементы совпадают с соответствующими значениями функции f_3.Дописав недостающие значения в 7-й столбец, получаем на 3-м шаге функцию φ_7^3 . Затем выполняем операцию заполнения строк и подсчет весов столбцов.Результат выполнения 3-го шага приведен в табл. 4. На 4-м шаге находим функцию φ_2^6 , на 5-м шаге - функцию φ_4^2 (табл. 5). Таким образом, найдено множество совместимых функций $M_1' = \{\varphi_1^1, \varphi_3^5, \varphi_7^3, \varphi_2^6, \varphi_4^2\}$.

Функции f_i $(i = 1,2,3,5,6)$ исключаем из списка и повторяем процедуру синтеза. В результате находим множество

$M_2' = \{\varphi_2^4, \varphi_4^8, \varphi_3^9\}$. Следующее повторение процедуры дает множество $M_3' = \{\varphi_1^7, \varphi_5^{10}\}$. После объединения функций, входящих в множества M_1', M_2', M_3' , и минимизации получаем:

$$\varphi^{1,2,3,5,6}_{1,4,7,3,2} = B_1B_2B_3B_4 \vee ((B_1 \vee B_2)B_3 \vee B_4)B_5,$$

$$\varphi^{4,8,9}_{2,4,3} = (B_1 \vee B_2 \vee B_3)B_4 \vee B_1B_2 \vee (B_3 \vee B_4)B_5,$$

$$\varphi^{7,10}_{1,5} = (B_1 \vee B_2 \vee B_3)B_4 \vee (B_1 \vee B_2 \vee B_3)B_5.$$

Согласно выражению (4) находим функцию, описывающую МЛМ:

$$Z = (B_1B_2B_3B_4 \vee ((B_1 \vee B_2)B_3 \vee B_4)B_5)\bar{B}_6\bar{B}_7 \vee$$
$$\vee ((B_1 \vee B_2 \vee B_3)B_4 \vee B_1B_2 \vee (B_3 \vee B_4)B_5)\bar{B}_6B_7 \vee$$
$$\vee ((B_1 \vee B_2 \vee B_3)B_4 \vee (B_1 \vee B_2 \vee B_3)B_5)B_6\bar{B}_7.$$

Применение рассмотренного приближенного метода синтеза позволило получить МЛМ с числом $q = 7$, в то время как верхняя оценка равна $4 +]\log_2 10[= 8$. Заметим, что при возможности использования для настроек всех сигналов из множества $\mathcal{F}$ полученная структура представляет МЛМ, универсальный в классе бесповторных функций от 4-х переменных.

Таблица I

A_4	A_3	A_2	A_1	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	I	0	0	0	0	0	0	0	I	0	0
0	0	I	0	0	0	0	0	0	0	0	I	0	0
0	0	I	I	0	0	0	I	0	0	0	I	I	0
0	I	0	0	0	0	0	I	0	0	0	I	0	0
0	I	0	I	0	0	0	I	0	I	0	I	0	I
0	I	I	0	0	0	0	I	0	I	0	I	0	I
0	I	I	I	0	I	0	I	0	I	0	I	I	I
I	0	0	0	0	I	0	I	0	I	0	I	0	0
I	0	0	I	0	I	0	I	0	I	I	I	0	I
I	0	I	0	0	I	0	I	0	I	I	I	0	I
I	0	I	I	0	I	I	I	0	I	I	I	I	I
I	I	0	0	0	I	I	I	0	I	I	I	I	0
I	I	0	I	0	I	I	I	I	I	I	I	I	I
I	I	I	0	0	I	I	I	I	I	I	I	I	I
I	I	I	I	I	I	I	I	I	I	I	I	I	I

Таблица 2.

	B_5	B_4	B_3	B_2	B_1	I	2	3	4	5	6	7	8	9	I0
						B_5 0	1	A_4	A_4	A_4	A_4	A_4	A_4	A_4	A_4
						B_4 A_4	A_4	0	1	A_3	A_3	A_3	A_3	A_3	A_3
						B_3 A_3	A_3	A_3	A_3	0	1	A_2	A_2	A_2	A_2
						B_2 A_2	A_2	A_2	A_2	A_2	A_2	0	1	A_1	A_1
						B_1 A_1	A_1	A_1	A_1	A_1	A_1	A_1	A_1	0	1
I	0	0	0	0	0	0	–	0	–	0	–	0	–	0	–
2	0	0	0	0	I	0	–	0	–	0	–	0	–	–	0
3	0	0	0	I	0	0	–	0	–	0	–	–	0	0	–
4	0	0	0	I	I	0	–	0	–	0	–	–	0	–	0
5	0	0	I	0	0	0	–	0	–	–	0	0	–	0	–
6	0	0	I	0	I	0	–	0	–	–	0	0	–	–	0
7	0	0	I	I	0	0	–	0	–	–	0	–	0	0	–
8	0	0	I	I	I	0	–	0	–	–	0	–	0	–	0
9	0	I	0	0	0	0	–	–	0	0	–	0	–	0	–
I0	0	I	0	0	I	0	–	–	0	0	–	0	–	–	0
II	0	I	0	I	0	0	–	–	0	0	–	–	0	0	–
I2	0	I	0	I	I	0	–	–	0	0	–	–	0	–	0
I3	0	I	I	0	0	0	–	–	0	–	0	0	–	0	–
I4	0	I	I	0	I	0	–	–	0	–	0	0	–	–	0
I5	0	I	I	I	0	0	–	–	0	–	0	–	0	0	–
I6	0	I	I	I	I	I	–	–	I	–	I	–	I	–	I
I7	I	0	0	0	0	–			–		–		–		–
I8	I	0	0	0	I	–			–		–		–	–	
I9	I	0	0	I	0	–			–		–	–			–
20	I	0	0	I	I	–			–		–	–		–	
2I	I	0	I	0	0	–			–	–			–		–
22	I	0	I	0	I	–			–	–			–	–	
23	I	0	I	I	0	–			–	–		–			–
24	I	0	I	I	I	–			–	–		–		–	
25	I	I	0	0	0	–		–			–		–		–
26	I	I	0	0	I	–		–			–		–	–	
27	I	I	0	I	0	–		–			–	–			–
28	I	I	0	I	I	–		–			–	–		–	
29	I	I	I	0	0	–		–		–			–		–
30	I	I	I	0	I	–		–		–			–	–	
3I	I	I	I	I	0	–		–		–		–			–
32	I	I	I	I	I	–		–		–		–		–	
						φ_1^1	0	8	8	8	8	8	8	8	8

Таблица 3.

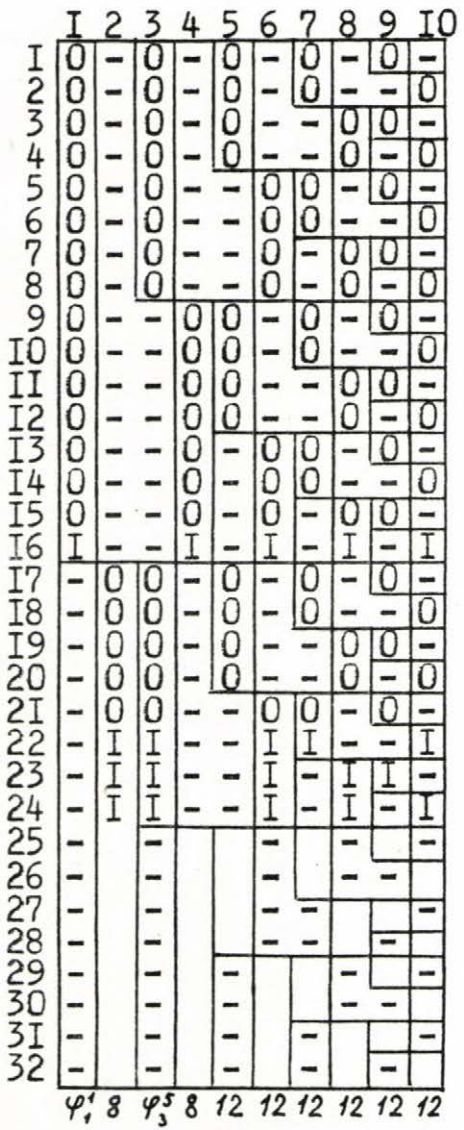


Таблица 4.

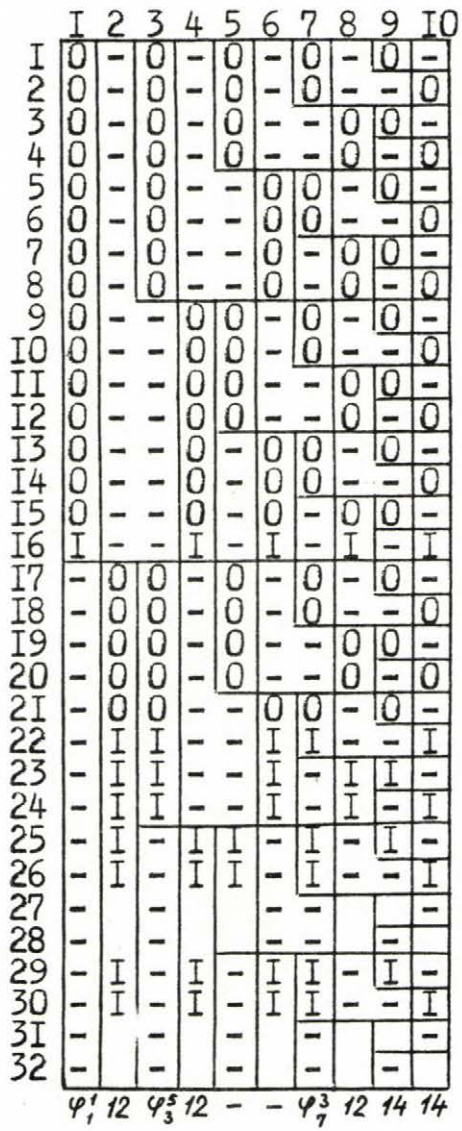
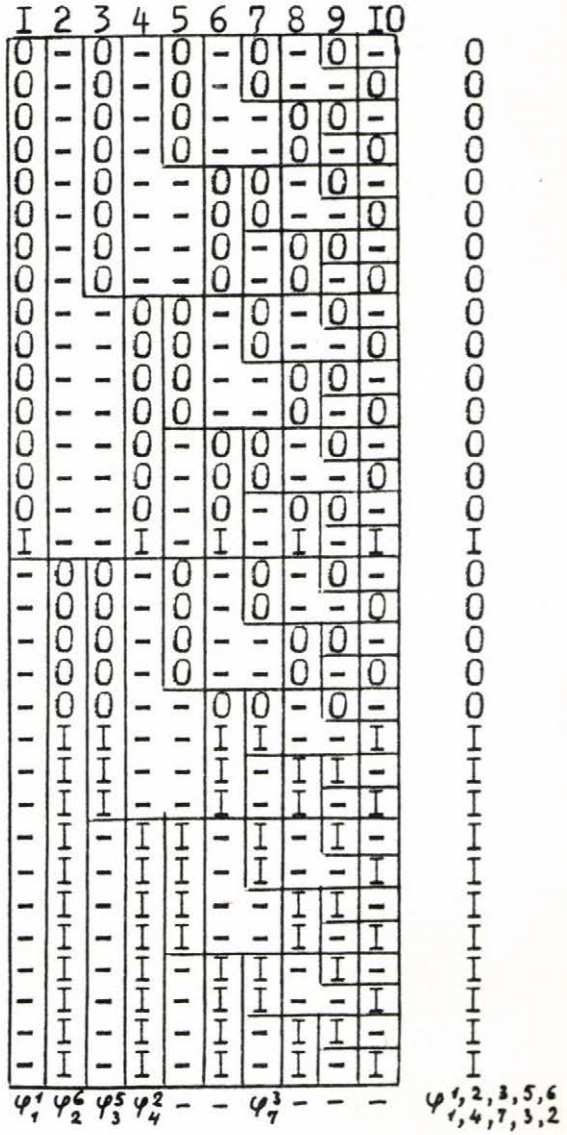


Таблица 5.



Литература

I. ЯКУБАЙТИС Э.А. и др. Теория автоматов.- "Теория вероятностей. Математическая статистика.Теоретическая кибернетика". (Итоги науки и техники.) Том I3. Москва, I976.
2. ЯКУБАЙТИС Э.А. Логические автоматы и микромодули. Рига, "Зинатне", I975.
3. ЯКУБАЙТИС Э.А. Универсальные логические элементы.- Автоматика и вычислительная техника,I973,5.
4. ЯКУБАЙТИС Э.А., ЧАПЕНКО В.П., КАЛНБЕРЗИНЬ А.Я. Метод синтеза структуры многофункционального логического элемента.-Автоматика и вычислительная техника, I973,6.
5. ЯКУБАЙТИС Э.А., КАЛНБЕРЗИНЬ А.Я., ЧАПЕНКО В.П. Синтез многофункциональных логических элементов.- Дискретные системы. Том I. Международный симпозиум. Рига, "Зинатне", I974.

Институт электроники и вычислительной
техники Академии наук Латвийской ССР,
ул. Академияс, I4, г. Рига, СССР.

ПОСТРОЕНИЕ ПРОВЕРЯЮЩИХ ТЕСТОВ МЕТОДОМ АНАЛИЗА АВТОМАТОВ С ВОЗМОЖНЫМИ НАРУШЕНИЯМИ

О.С.Денисенко, А.Н.Скляревич

(СССР)

Изготовление и техническое обслуживание больших интегральных схем предполагает использование эффективных проверяющих и диагностических тестов.

В настоящем докладе развивается структурно-аналитический метод, позволяющий аналитически решать большой круг задач, связанных с отысканием таких тестов. В основе метода — использование структурно-операторной модели конечного автомата с возможными нарушениями [I].

В соответствии с этой моделью в каждой линии e передачи сигнала допускается наличие логических неисправностей — K_{oe} („$\equiv 0$"), K_{1e} („$\equiv 1$"), $\bar{K}_e$ (дополнительное инвертирование сигнала в линии). Исправное состояние линии обозначается N_e.

Так как возможны четыре технических состояния линии, их можно закодировать двумя логическими символами L_{oe}, L_{1e}. Будем считать, что сочетания $\bar{L}_{oe}\bar{L}_{1e}$, $\bar{L}_{oe}L_{1e}$, $L_{oe}\bar{L}_{1e}$, $L_{oe}L_{1e}$ соответственно определяют исправность линии N_e, наличие в ней нарушения $K_{1e}, K_{oe}, \bar{K}_e$. Тогда функционирование линии с возможными нарушениями определяется формулой

$$u = \bar{L}_{oe}x \vee L_{1e}\bar{x} = R_e(x), \qquad (I)$$

где $R_e = \bar{L}_{oe} \vee L_{1e}\mathcal{D}$ – оператор преобразования входной переменной линии e в выходную переменную u ($\mathcal{D}$ – оператор инвертирования).

Вводя в каждую линию передачи сигнала в схеме автомата оператор R с соответствующим индексом, получим схему автомата с возможными нарушениями (АВН). Чтобы раскрыть особенности метода, последовательно рассмотрим случаи, когда автомат является комбинационным, элементом памяти, автоматом с элементом

памяти. Если $\{A_i\}$ - множество входных наборов комбинационного автомата M , то его исправное функционирование описывается формулой

$$z = \bigvee_{i=1}^{n_1} A_i \ ,$$

где суммирование производится по входным наборам, которым отвечает единичный выход автомата. Соответственно, функционирование АВН описывается формулой

$$u = \bigvee_{i=1}^{n} \varphi_i(M) A_i \ , \qquad (2)$$

где логическое суммирование проводится по всему множеству входных наборов и компонент $\varphi_i(M)$ описывает все технические состояния АВН, при которых выходной сигнал автомата на наборе A_i равен I.

Найдя функцию различимости сигналов z и u , получим

$$z\bar{u} \vee \bar{z}u = \bigvee_{i=1}^{n} d_i(M) A_i \ , \qquad (3)$$

где $d_i(M)$ определяет все нарушения, при которых выходные сигналы исправного и анализируемого автоматов отличаются на наборе A_i . Естественно составляющую $d_i(M)$ называть проверяющей возможностью входного набора A_i в автомате. Характеристики $\varphi_i(M)$ и $d_i(M)$ связаны зависимостями

$$d_i(M) = \begin{cases} \bar{\varphi}_i(M) , & 1 \le i \le n_1 \ , \\ \varphi_i(M) , & n_1+1 \le i \le n \ , \end{cases} \qquad (4)$$

согласно которым легко находятся их схемные представления. Так, схема, представляющая характеристику $d_i(M)$, находится, если ко входам схемы АВН приложить набор A_i и оставить его выходной канал неизменным, если набору A_i соответствует нулевой выход исправного автомата. Если же ему соответствует единичный выход, то в полученной схеме необходимо предусмотреть инвертирование выходного сигнала.

Характеристики для входных наборов автомата связаны зависимостями с аналогичными характеристиками наборов подавтоматов, подключенных к его выходному элементу. Такие зависимости проверяющих возможностей для двухвходовых элементов И, ИЛИ приведены в табл. I.

Столбец I этой таблицы содержит структурную схему автомата, столбцы 3, 4 — проверяющие характеристики в зависимости от сочетания сигналов z_1, z_2 (столбец 2) в исправном автомате на данном наборе.

Так как выход подавтомата может считаться выходом автомата, данные зависимости являются рекуррентными. Рекуррентная методика отыскания характеристик может использоваться, если они известны для подавтомата низшего уровня. Таким можно считать линию подачи входного набора. Согласно (I), для входного канала e $d_e = L_{0e}$, $d_{\bar{e}} = L_{1e}$.

По характеристикам отдельных наборов определяются характеристики множества T входных наборов:

$$Q(M) = \bigwedge_T \bar{d}_i(M), \qquad P(M) = \bigwedge_T d_i(M), \qquad (5)$$

описывающие соответственно: технические состояния, не обнаруживаемые данным множеством; неисправности, обнаруживаемые каждым из наборов этого множества.

Найдя характеристику $Q(M)$ для полного множества входных наборов автомата, можно ввести понятия избыточного и неизбыточного, достаточного и недостаточного множества входных наборов. Множество входных наборов будем считать достаточным, если его проверяющая возможность $D(M) = \bar{Q}(M)$ совпадает с проверяющей возможностью полной последовательности. В противоположном случае множество входных наборов не считается достаточным.

Множество входных наборов является избыточным, если в нем существует набор A_i, исключение которого не вызывает изменения соответствующей множеству функции $Q(M)$. Если такого набора нет, множество входных наборов неизбыточно. Достаточное неизбыточное множество входных наборов составляет тупиковый тест проверки автомата. Чтобы найти такое множество, нет необходимости анализировать все множество характеристик $d_i(M)$ и конъюнкцию $Q(M)$. Достаточно установить рекуррентные зависимости между характеристикой $Q(M)$ для автомата и аналогичными характеристиками подавтоматов M_1, M_2, подключенных к выходному элементу автомата M.

Во многих случаях такие зависимости представляются равенством

Таблица 1

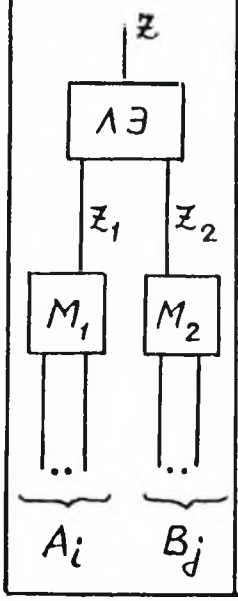
1	2	3	4
	$z_1 z_2$	$d_{ij}(и)$	$d_{ij}(и \wedge и)$
	1 1	$\bar{R}_z[\bar{d}_i(M_1)\bar{d}_j(M_2)]$	$R_z[\bar{d}_i(M_1) \vee \bar{d}_j(M_2)]$
	1 0	$R_z[\bar{d}_i(M_1)d_j(M_2)]$	$\bar{R}_z[\bar{d}_i(M_1) \vee d_j(M_2)]$
	0 1	$R_z[d_i(M_1)\bar{d}_j(M_2)]$	$\bar{R}_z[d_i(M_1) \vee \bar{d}_j(M_2)]$
	0 0	$R_z[d_i(M_1)d_j(M_2)]$	$\bar{R}_z[d_i(M_1) \vee d_j(M_2)]$

Таблица 2

u_{m-1}	y_m	
	1	0
1	$\bar{\varphi}_{iu}$	φ_{iu}
0	$\bar{\varphi}_{i\bar{u}}$	$\varphi_{i\bar{u}}$

$$Q(M) = N_{\Sigma}\, Q(M_1) Q(M_2) \quad , \qquad (6)$$

из которого следует, что тест проверки автомата M может быть образован путем рационального сочетания (склеивания) наборов теста проверки подавтомата M_1 с наборами теста проверки подавтомата M_2 . Правила склеивания следуют из развернутого представления функции $Q(M)$ через характеристики d_i автомата и подавтоматов. То же имеет место и в других случаях, когда характеристика $Q(M)$ выражается более сложным образом через характеристики входных наборов подавтоматов.

Из формул (6) могут быть получены правила склеивания теста проверки автомата из наборов тестов проверки подавтоматов для случаев, когда в автомате возможны только неисправности K_0, K_1 и когда в автомате возможна только одиночная неисправность. Для общего случая комбинационного автомата эти правила отличаются от правил склеивания в классе одиночных неисправностей.

Рассмотрим теперь элемент памяти (автомат с обратной связью) Задержку сигнала будем считать сосредоточенной в линии обратной связи. Если y – выход исправного элемента памяти (ЭП), то его функционирование на такте m описывается формулой

$$y_m = \bigvee_{i=1}^{n_1} A_i \delta \vee \bigvee_{i=n_1+1}^{n_2} A_i \bar{\delta} \vee \bigvee_{i=n_2+1}^{n_3} A_i y_{m-1} \vee \bigvee_{i=n_3+1}^{n} A_i \bar{y}_{m-1} \quad , \qquad (7)$$

где $\delta = 1$ и первая (вторая) сумма в правой части распространена по входным наборам A_i , которым независимо от выходного сигнала ЭП на предыдущем такте соответствует единичный (нулевой) выход, третья (четвертая) сумма – по входным наборам A_i , которым соответствует единичный выход при единичном (нулевом) выходном сигнале ЭП на предыдущем такте. Во многих случаях формула (7), соответствующая реальным автоматам, не содержит отдельных сумм.

Для того, чтобы получить схему элемента памяти с возможными нарушениями (ЭПВН), необходимо во все линии передачи сигналов схемы ЭП, в частности, и в линию обратной связи, ввести операторы R . Функционирование ЭПВН описывается равенством

$$u_m = \bigvee_{i=1}^{n} [\varphi_{iu}(M) u_{m-1} \vee \varphi_{i\bar{u}}(M) \bar{u}_{m-1}] A_i \quad , \qquad (8)$$

где суммирование осуществляется по всем возможным входным состояниям автомата. Функции $\varphi_{iu}(\varphi_{i\bar{u}})$ описывают технические состояния ЭП, при которых его выходной сигнал равен I, если этот сигнал на предыдущем такте был равен I(0). Подавая на входы ЭПВН набор A_i, на вход обратной связи сигнал $u_{m-1}(\bar{u}_{m-1})$, получаем схемное представление функции $\varphi_{iu}(\varphi_{i\bar{u}})$.

Найдя функцию различимости выходных сигналов исправного ЭП и ЭПВН, получим

$$u_m\bar{y}_m \vee \bar{u}_m y_m = \bigvee_{i=1}^{n} [d_i(u_{m-1}, y_m)u_{m-1} \vee d_i(\bar{u}_{m-1}, y_m)\bar{u}_{m-1}] A_i , \quad (9)$$

где $d_i(u_{m-1}, y_m)$, $d_i(\bar{u}_{m-1}, y_m)$ - условные проверяющие возможности набора A_i, определяющие технические состояния ЭП, при которых различны выходные сигналы исправного и анализируемого ЭП.

Термином "условные" подчеркивается, что они зависят не только от входного набора, но и от действующего совместно с набором сигнала обратной связи, определяемого выходным сигналом ЭП на предыдущем такте работы. Согласно (9), условные проверяющие возможности набора зависят от действующего вместе с набором сигнала обратной связи и от выходного сигнала исправного элемента памяти на такте m. Связи между характеристиками $d_i(u_{m-1}, y_m)$ и функциями $\varphi_{iu}, \varphi_{i\bar{u}}$ представляются табл.2.

Чтобы найти проверяющие возможности последовательности входных состояний ЭП, необходимо учесть зависимости между его полными состояниями. Для этого построим граф переходов ЭПВН при подаче наборов последовательности. Будем считать заданными условные проверяющие характеристики $d_i(u_{m-1}, y_m)$ наборов последовательности $A_1, \ldots A_\ell$, начальные установки исправного y_0 и анализируемого u_0 ЭП.

По значению y_0 и наборам A_i последовательности вычислим последовательные значения y_i выходного сигнала исправного ЭП. Тогда в верхней линии графа переходов ЭПВН (рис.I) расположим узел, отражающий начальное состояние (u_0, y_0), если $u_0 = y_0$, и узлы, соответствующие состояниям, последовательно получаемым при исправной работе ЭП, в нижней линии графа - узел начального состояния, если $u_0 = \bar{y}_0$, и узлы других состояний с выходным сигналом u_i, отличным от выходного сигнала y_i, соответствующего при исправности ЭП тому же набору. Входные наборы обозначены

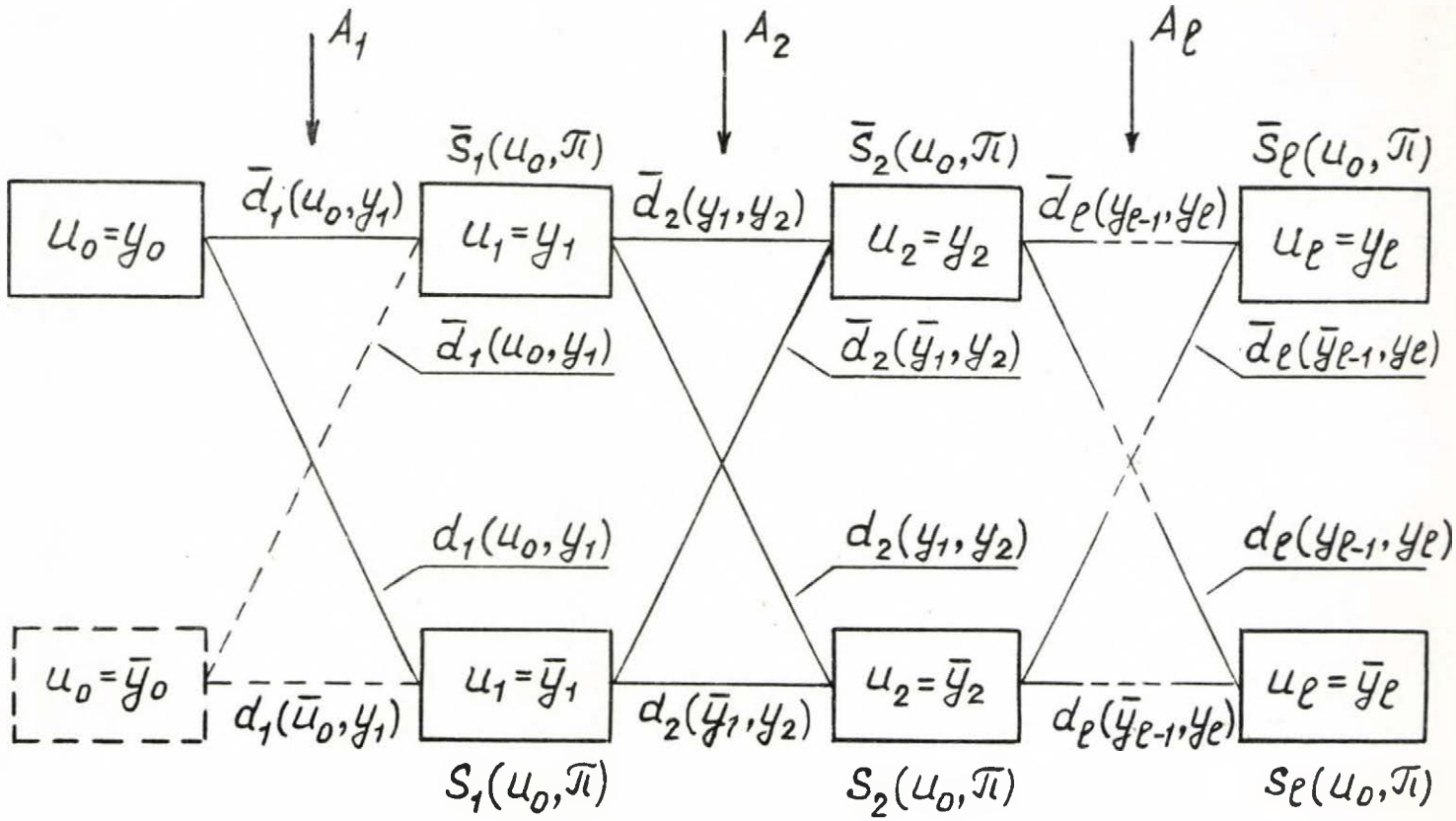


Рис. 1 Граф переходов ЭПВН

стрелками.У каждой линии перехода указана функция,описывающая технические состояния ЭП,при которых осуществляется переход. Если характеризуется переход в состояние нижней (верхней) линии,то такой функцией является условная проверяющая возможность $d_i(u_{i-1},y_i)$ $/\bar{d}_i(u_{i-1},y_i)$ /,при этом $u_{i-1}=y_{i-1}$ $(u_{i-1}=\bar{y}_{i-1})$, если переход осуществляется от узла верхней (нижней) линии.

Пусть $S_i(u_0,\pi)$- описание технических состояний ЭПВН,при которых значение его выходного сигнала на такте i отличается от значения выходного сигнала исправного ЭП.Тогда технические состояния ЭП,при которых на всех тактах подачи последовательности наблюдается правильное значение выходного сигнала,описывается формулой

$$Q(u_0,\pi)=\bigwedge_{i=1}^{\ell}\bar{S}_i(u_0,\pi) \quad . \qquad (10)$$

Характеристика,дополнительная к этой,является характеристикой проверяющих возможностей последовательности.Так как согласно рис.I

$$\begin{aligned}\bar{S}_i(u_0,\pi)&=\bar{S}_{i-1}(u_0,\pi)\bar{d}_i(y_{i-1},y_i)\vee S_{i-1}(u_0,\pi)\bar{d}_i(y_{i-1},y_i)\,;\\ S_i(u_0,\pi)&=\bar{S}_{i-1}(u_0,\pi)d_i(y_{i-1},y_i)\vee S_{i-1}(u_0,\pi)d_i(y_{i-1},y_i)\,,\end{aligned} \qquad (11)$$

то после преобразований получим

$$Q(u_0,\pi)=\bar{d}_1(u_0,y_i)\bigwedge_{i=2}^{\ell}\bar{d}_i(y_{i-1},y_i) \quad . \qquad (12)$$

Таким образом,технические состояния,не выявляемые последовательностью по отличию хотя бы на одном такте выходных сигналов исправного и анализируемого ЭП,описываются конъюнкцией дополнений условных проверяющих возможностей ее наборов при аргументах,отражающих правильное изменение состояний ЭП при подаче последовательности.

Но верхняя линия графа работы ЭПВН отражает работу соответствующего комбинационного устройства (СКУ) с возможными нарушениями,получаемого из ЭП путем разрыва линии обратной связи и имеющего дополнительный, по сравнению с ЭП,вход,если на входы этого устройства подается последовательность входных состояний,совпадающая с последовательностью входных и внутренних состояний исправного ЭП.Определив для этой последовательности невыявляемые технические состояния,получим,что они описываются той же формулой (12),что и для ЭП.

Так как полученный результат справедлив для любой последовательности,то он применим и к полной последовательности входных состояний.Отсюда следует,что входные состояния,избыточные для проверки СКУ,не являются необходимыми,если нет других оснований для их использования,при проверке ЭП.

Таким образом,чтобы найти состояния,которые должны быть включены в тест проверки ЭП,нужно найти входные состояния СКУ, составляющие тест его проверки.Задача отыскания теста проверки ЭП свелась в этой части к задаче отыскания теста проверки СКУ. Но этим задача отыскания теста проверки ЭП не исчерпалась.Состояния,составляющие этот тест,должны подаваться в такой последовательности,что выходной сигнал,отвечающий данному состоянию,определял бы значение сигнала на входе обратной связи при последующем состоянии.Если следование входных состояний теста проверки СКУ можно упорядочить так,чтобы это правило выполнялось,задача построения теста проверки ЭП будет решена.Если нет необходимо использовать дополнительные,устанавливающие требуемое соответствие состояний,или же предусматривать повторную установку анализируемого ЭП в известное исходное состояние.

Дополнительно проанализировав структуру тестов проверки ЭП при различных исходных состояниях,можно найти тест проверки ЭП,когда его исходное состояние неизвестно.

Аналогичное положение имеет место и для автоматов с ЭП. В этом случае граф его переходов представляется рис.2,где y,z - выходные сигналы исправного ЭП и автомата, u,r - выходные сигналы ЭПВН и АВН (элемента памяти с возможными нарушениями,автомата с возможными нарушениями).

Из рис.2 следуют формулы для определения проверяющих возможностей i-го набора последовательности,проверяющих возможностей всей последовательности,правила отыскания теста проверки автомата,которые и в этом случае заключаются в отыскании теста проверки СКУ и дальнейшем упорядочении следования его наборов,правила отыскания теста проверки автомата по тестам проверки составляющих его подавтоматов,а также и другие выводы,связанные с решением различных задач по анализу проверяющих и диагностических возможностей последовательностей при заданных условиях.

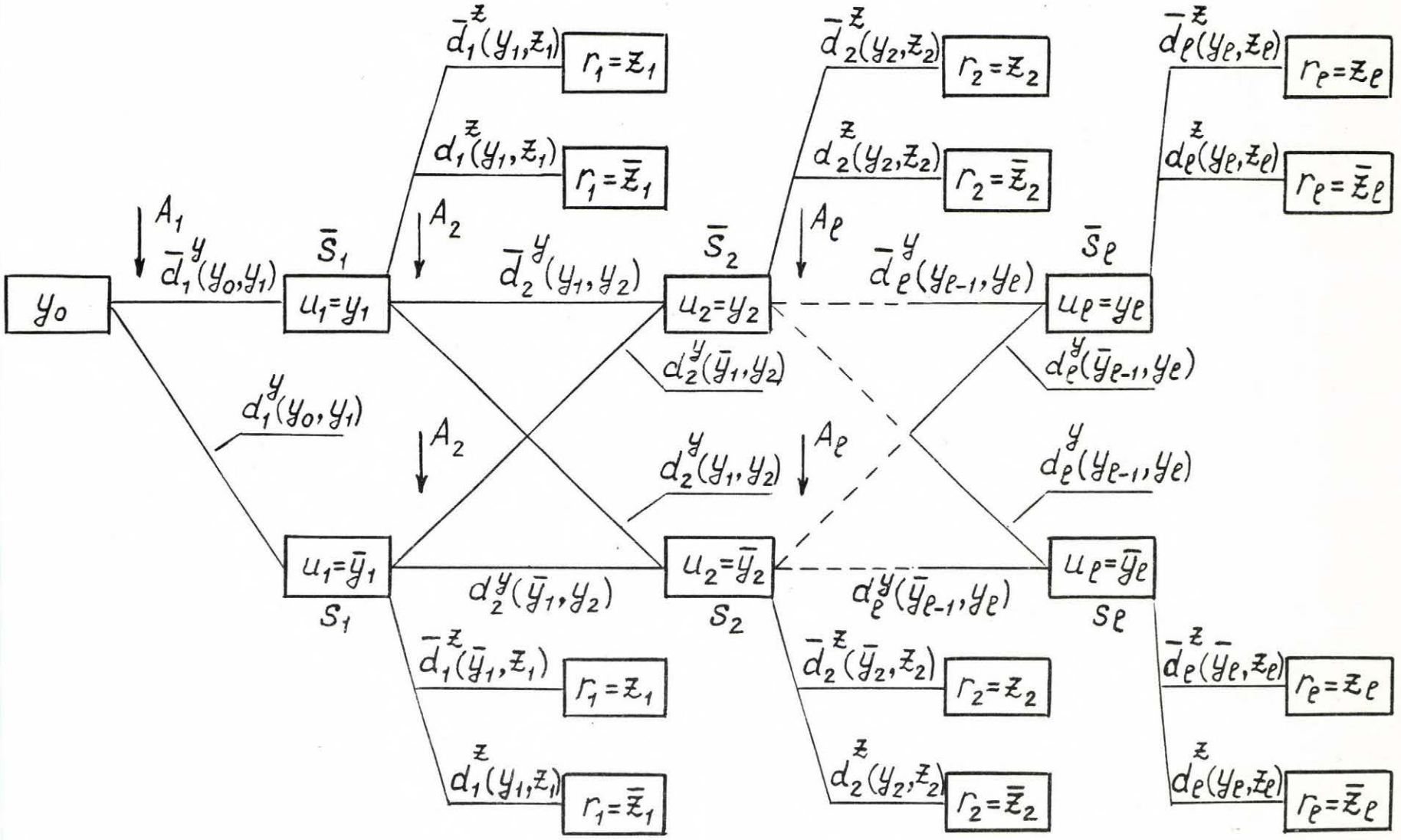


Рис. 2 Граф работы СКУВН при подаче условной последовательности.

СПИСОК ЛИТЕРАТУРЫ

I. Скляревич А.Н. Логические методы проверки комбинационных автоматов. "Зинатне",I975, I96 с.
2. Скляревич А.Н. Условные проверяющие возможности входных наборов элемента памяти. - Автоматика и вычислительная техника, I975, №5.
3. Скляревич А.Н. Характеристики последовательности входных наборов автомата с элементом памяти. - Автоматика и вычислительная техника, I976, №2.

Институт электроники и вычислительной техники
Академии наук Латвийской ССР.
СССР, г.Рига, ул.Академияс I4.

СТАНДАРТНАЯ РЕАЛИЗАЦИЯ АСИНХРОННЫХ АВТОМАТОВ ПРИ МАШИННОМ ПРОЕКТИРОВАНИИ

Амбарцумян А.А., Потехин А.И.

I. Постановка задачи.

С появлением и развитием систем машинного проектирования в значительной степени изменились требования к методам синтеза асинхронных дискретных управляющих устройств.

Проектировщик при "ручном" синтезе в соответствии с существующими методиками синтеза [I] последовательно получает отдельные блоки структуры будущего устройства (память, логический преобразователь, из последнего - блоки входной и выходной логики и т.д. до элементарной структуры). При этом проектировщик знает функцию каждого блока (элемента) в структуре и возможности ее, видоизменения. Кроме того, создавая макет устройства, проектировщик одновременно с логическим проектировщиком осуществляет временное моделирование, при этом в его распоряжении находятся как средства временного согласования различных цепей (задержки, фильтры) так и достаточное значение структуры и ее поведения, что позволяет ему обеспечить необходимую коррекцию в алгоритме функционирования и требуемую устойчивость устройства путем изменения логической структуры и/или использования фильтров и задержек.

Совсем по другому обстоит дело при машинном проектировании, когда невозможно не только изменить полученную структуру с целью коррекции алгоритма функционирования и/или обеспечения требуемой устойчивости, но также трудно разобраться в самой схеме, вследствие различных преобразований исходного описания, осуществляемых машиной.

В связи с этим при создании систем машинного проектирования возникла потребность в разработке новых алгоритмов синтеза, отличающихся простотой, высоким быстродействием, применение которых обеспечивало такие свойства структурам как хорошая наблюдательность: соответствие состояний устройства состояниям объекта управления (или, что то же элементам описывания поведения объекта управления), достаточная гибкость при изменении

по тем или иным причинам требуемого поведения объекта управления, логическая структура не должна целиком подвергаться изменению (заново проектироваться), достаточный запас устойчивости к реальному разбросу временных параметров логических элементов и их изменению во времени.

Свойство устойчивости связано с существующим разбросом временных параметров логических элементов. В работе [2] отмечается, чтр только вследствие технологического разброса параметров элементов временное рассогласование сигналов, следующих по двум параллельным цепям, каждая из которых содержит два и более логических элементов, может достигать критической величины и приводить к сбою устройства, если не принять специальных мер.

В работе [3], по-видимому, впервые был описан алгоритм построения логической структуры, обладающий в некоторой мере вышеописанными свойствами, устойчивость обеспечивается за счет использования естественных ненулевых задержек логических элементов путем специальной организации соединений элементов.

Попытки распространить идею алгоритма на случай когда задан граф переходов наталкнулись на значительные трудности.

В работе [4] предложен другой способ соединений между элементами логической структуры, обеспечивающий устойчивость устройства при любом наперед заданном разбросе временных параметров логических элементов. Однако изложенный в работе метод построения структуры применим лишь к таблице переходов устройства, точнее необходимое знание переходов между полными устойчивыми состояниями. Поэтому представляет практический и теоретический интерес развитие этого подхода, а именно, разработка метода построения структуры по заданному графу переходов.

В обеих указанных работах логическая структура проектируется путем сопоставления каждому элементу описания алгоритма функционирования устройства типовой ячейки, построенной определенным образом из логических элементов типа ИЛИ-НЕ, и организации соединений между ними регулярным образом. Такой способ построения структуры будем называть стандартной реализацией. Естественно, что стандартные реализации, обеспечивающие одновременно устойчивость устройства, могут быть с успехом исполь-

зованы в системах машинного проектирования.

В настоящей работе излагаются основы двух стандартных реализаций логической структуры устройства асинхронного типа, используемых в системе машинного проектирования, разрабатываемой под руководством член-корреспондента АН СССР М.А.Гаврилова.

2. Основные понятия и определения

Обычно математической моделью дискретного устройства управления асинхронного типа является асинхронный автомат [I]. Алгоритм функционирования автомата задан в виде графа переходов (B, V), где B – множество вершин графа, соответствующее множеству внутренних состояний, V – множество ребер графа. Ориентированное ребро (b_i, b_j) выходит из вершин b_i и входит в вершину b_j. Каждому ребру $(b_i, b_j) \in V$ сопоставлена реберная функция $F_{ij}(x_1, \ldots, x_m)$. Выходные состояния автомата сопоставлены вершинам графа (модель Мура).

Структуру для определенности будем строить на элементах типа ИЛИ-НЕ, однако излагаемые ниже реализации двойственным образом могут быть сформулированы в случае элементов типа И-НЕ.

Некоторый логический элемент структуры обозначим через e_α, выход z_α которого может принимать значение 0 и I. Входы этого элемента, соединенные с выходами z_β, z_γ элементов e_β, e_γ, обозначим через $x_{\alpha,\beta}$, $x_{\alpha,\gamma}$, где второй индекс указывает на тот единственный элемент, выход которого соединения с данным входом. Булеву функцию, реализуемую элементом e_α обозначим через z'_α, $z'_\alpha = \overline{x_{\alpha,\beta} \vee x_{\alpha,\gamma} \vee \ldots}$. Временную задержку элемента e_α обозначим через Δ_α, через $\Delta_{\beta,\alpha}$ – задержку соединительного проводника от выхода z_β до входа $x_{\alpha,\beta}$.

Рассмотрим кратко способ обеспечения устойчивости структуры (этот способ подробно изложен в работе [4]), используемый в нижеследующих реализациях. Обозначим через E множество логических элементов некоторой структуры, $E = \{e_1, \ldots, e_N\}$. Если $z_\alpha = z'_\alpha$ для всех $e_\alpha \in E$, то говорят, что структура находится в устойчивом состоянии, при этом совокуп-

ность значения $x_1, \ldots, x_m, z_1, \ldots, z_N$ определяет полное устойчивое состояние структуры, множество которых обозначим через $P = \{\mathcal{X}_1, \ldots \mathcal{X}_M\}$. Значение выхода z_α в состоянии $\mathcal{X}_i$ обозначим через $z_\alpha^{(i)}$. Рассмотрим переход $\mathcal{X}_i \to \mathcal{X}_j$. Пусть каждый элемент структуры в этом переходе должен (согласно заданного алгоритма функционирования структуры) либо сохранить свое состояние, либо изменить состояние, но не более одного раза. Тогда относительно данного перехода множество E можно разбить на два подмножества E_1 и E_2: I) $e_\alpha \in E_1$, если $z_\alpha^{(i)} = z_\alpha^{(j)}$; 2) $e_\alpha \in E_2$, если $z_\alpha^{(i)} \neq z_\alpha^{(j)}$. Обозначим через t – начало переходного процесса, $(t + T_n)$ – конец переходного процесса.

В идеальной структуре предполагается, что значение выхода каждого элемента из E_1 остается постоянным в течение времени T_n, то время как значение выхода каждого элемента из E_2 изменяется в точности один раз. Это можно записать в виде следующих двух свойств поведения элементов идеальной структуры. В любом заданном переходе $\mathcal{X}_i \to \mathcal{X}_j$ имеет место:

I) для каждого $e_\alpha \in E_1$ $z_\alpha(\tau) = z_\alpha(t-\varepsilon), \varepsilon > 0$ в интервале $t \le \tau \le t + T_n$;

2) для каждого $e_\alpha \in E_2$ существует $t^* \ge t + \Delta_\alpha$ такое, что $z_\alpha(t^*) \neq z_\alpha(t)$ и $z(\tau) = z_\alpha(t^*)$ в интервале $t^* \le \tau \le t + T_n$

В реальной структуре при наличии разброса временных параметров логических элементов возможно нарушение свойств (I) и (2). Так, например, значение выхода z_α, $e_\alpha \in E_1$, равное 0 в состояниях $\mathcal{X}_i$ и $\mathcal{X}_j$, может изменяться следующим образом: 0 → I → 0. Эта кратковременная "I" может изменить значение выхода структуры или перевести ее в состояние отличное от заданного. В этих случаях говорят о наличии в структуре опасных состязаний сигналов и соответственно об отсутствии устойчивости структуры.

В работе [4] было показано, что в структуре на элементах типа ИЛИ-НЕ источником ложных "I" являются такие элементы $e_\alpha \in E_1$, нулевое значение выхода которых в переходе $\mathcal{X}_i \to \mathcal{X}_j$ обеспечивается только за счет такой пары входов

$x_{\alpha,\beta}$ и $x_{\alpha,\gamma}$, что $z_\beta^{(i)}=1$, $z_\beta^{(i)}=0$, $z_\gamma^{(i)}=0$, $z_\gamma^{(i)}=1$. Действительно, неодновременное изменение значений z_β, z_γ (IO → 00 → 0I) приводит к ложной "I" на выходе z_α. Было показано, что обеспечив свойство (I) для указанных элементов, получаем структуру со свойствами (I) и (2). При этом естественно предполагается, что одновременно может изменяться значение только одного внешнего входа структуры. Предложенный там же способ устранения ложной "I" состоит в том, что значения $x_{\alpha,\beta}$ и $x_{\alpha,\gamma}$ принудительным образом изменяются в следующей последовательности: IO → II → 0I, причем промежуточное состояние (II) удерживается достаточное время. Это достигается применением специальной схемы соединения элементов структуры, показанной на рисунке I, которую в дальнейшем будем называть основной схемой соединения элементов. В приложении дан временной анализ основной схемы соединения. В том случае, когда основная схема соединения не обеспечивает надежного сохранения 0 на выходе z_α (при очень большом разбросе параметров элементов) можно применить модифицированную схему соединения включением четного числа элементов в линию ($z_\beta - x_{\alpha,\beta}$) (рис.I). В нижеследующих стандартных реализациях связи между элементами осуществляются таким образом, чтобы обеспечить, по крайней мере, основную схему соединения во всех переходах.

3. Описание стандартных реализаций[x]. Стандартная реализация № I.

Основным конструктивным модулем для построения логической структуры по заданному графу переходов устройства является ячейка, построенная из трех элементов типа ИЛИ-НЕ (рис.2). Элемент e' является выходным элементом ячейки, элементы e' и e'' и связь между ними образуют триггер типа R-S.

Каждой вершине b_i графа переходов сопоставим столько ячеек, сколько она имеет выходящих ребер, т.е. можно говорить о реализации типа "ребро-ячейка". Ячейку, сопоставленную ребру (b_i, b_j) обозначим через $e_{i,j}$, в соответствии с рисунком 2 можно написать, что

---

x) Реализация № I разработана Потехиным А.И., реализация № 2 - Амбарцумяном А.А.

$$z'_{i,j} = \overline{F_{i,j} \vee f_{i,j} \vee v_{i,j}},$$
$$v'_{i,j} = \overline{\varphi_{i,j} \vee w_{i,j}},$$
$$w'_{i,j} = \overline{\psi_{i,j} \vee v_{i,j}} \qquad (I)$$

где $f_{i,j}$, $\varphi_{i,j}$, $\psi_{i,j}$ – функции связи.

Функция связи элемента ячейки есть дизъюнкция выходных переменных ячеек, выходы которых соединены со входами данного элемента. Найти функции связи – это значит установить соединение между ячейками и наоборот.

Так как выходные состояния автомата сопоставлены только вершинам графа (модель Мура), то выходной преобразователь можно легко реализовать на триггерах типа $R-S$, управляемых выходными элементами ячеек, поэтому в дальнейшем его реализацией заниматься не будем. Реберные функции $F_{i,j}$, $i \neq j$, реализуем отдельным блоком, в котором устраним статистические и динамические состязания известными методами. Однако, как будет следовать из дальнейшего изложения, рассматриваемые стандартные реализации допускают наличие статического I – риска в блоке, реализующем реберные функции, поэтому его можно реализовать более экономно, например, представить реберные функции в виде ДНФ и реализовать известными методами в виде двухъярусной структуры.

В зависимости от значений функций связи и состояния триггера будем различать основные устойчивые состояния ячейки (табл.I), где знак " ~ " означает безразличное значение функции связи.

Таблица I.

F	f	φ	ψ	z	v	w	Состояние ячейки
0	0	~	0	I	0	I	Рабочее состояние
~	0	0	~	0	I	0	Нерабочее состояние
~	I	I	0	0	0	I	Подготовительное состояние
~	I	0	I	0	I	0	Нерабочее состояние с запретом

Рассмотрим связи, обеспечивающие переход ячейки в то или иное устойчивое состояние. Пусть автомат находится в вершине x_i графа переходов, в этом случае все ячейки, сопоставленные этой вершине (или просто ячейки вершины b_i), находятся в рабочем состоянии при этом имеет место единичное значение выходов этих ячеек. Пусть в графе существуют переходы $b_i \to b_j$ и $b_i \to b_c$, в этом случае ячейка $l_{i,j}$ устанавливает некоторую ячейку $l_{i,p}$ вершины b_j в подготовительное состояние, для чего выход z ячейки $l_{i,j}$ необходимо одновременно соединить со входами элементов e и e' ячейки $l_{j,p}$, при этом выход $z_{j,p}$ сохраняет нулевое значение, так как указанные связи образуют основную схему соединения элементов (рис.I). Аналогично, ячейка $l_{i,c}$ устанавливает некоторую ячейку $l_{c,k}$ вершины b_c в подготовительное состояние. Пусть осуществился переход $b_i \to b_j$. Это означает, что $F_{i,j} = I$, вследствие чего (согласно системе уравнений (I)) изменится из I в 0 выход z ячейки $l_{i,j}$. При этом создаются условия перехода ячейки $l_{j,p}$ в рабочее состояние, который произойдет только тогда, когда значение функции $F_{j,p}$ станет равным 0, таким образом исключается возможность ложного перехода через вершину b_j в другую вершину, присущая классическим автоматным реализациям и известная в литературе как возможность ложного перехода из-за наличия ангеровских состязаний [I] . После того как ячейка $l_{i,p}$ установилась в рабочем состоянии, она переводит другие ячейки вершины b_j в рабочее состояние, для этого между ячейками вершины b_j устанавливаются следующие связи: выход одной ячейки необходимо соединить со входом элемента e' другой ячейки, выход которой - соединить со входом элемента e' третьей ячейки и т.д., в результате чего образуется замкнутый контур ячеек и переход любой из ячеек в рабочее состояние обеспечивает последовательный переход в рабочее состояние всех остальных ячеек этой вершины. Возможны другие способы соединения ячеек вершины, например, каждый выход ячейки можно соединить со всеми элементами e' других ячеек, что позволит увеличить быстродействие автомата. Ячейка $l_{j,p}$, перейдя в рабочее состояние, устано-

вит ячейку $\ell_{i,j}$ в нерабочее состояние (при отсутствии перехода $b_j \to b_i$ в графе переходов), для чего ее выход необходимо соединить со входом элемента e'' ячейки $\ell_{i,j}$. Кроме того, ячейка $\ell_{j,p}$ устанавливает одновременно ячейку $\ell_{i,c}$ в нерабочее состояние и ячейку $\ell_{c,\kappa}$ в нерабочее состояние с запретом, для чего выход z ячейки $\ell_{i,p}$ необходимо одновременно соединить со входами элементов e и e'' ячейки $\ell_{c,\kappa}$ и со входом элемента e'' ячейки $\ell_{i,c}$, при этом выход ячейки $\ell_{c,\kappa}$ сохраняет нулевое значение, так как элементы $e''_{i,c}$, $e'_{i,c}$, $\ell_{i,c}$ и $\ell_{c,\kappa}$ образуют модифицированную схему соединения элементов.

В дальнейшем будем говорить, что "ячейка устанавливает другую ячейку в такое то состояние" имея ввиду при этом наличие соответствующих соединений между ними.

Рассмотрим правила взаимодействия ячеек, сопоставленных вершинам некоторого графа переходов. Естественно, что возможно отступление от нижеизлагаемых правил, но в любом случае необходимо обеспечивать надежное сохранение нулевого значения выходов ячеек в переходах между устойчивыми состояниями, которым сопоставлены $z=0$ (табл. I).

Правила соединения ячеек рассмотрим относительно пары переходов $b_i \to b_j$ и $b_i \to b_c$.

Правило I. Это правило устанавливает связи между ячейками, сопоставленными одной вершине: первая ячейка устанавливает в рабочее состояние вторую ячейку, которая в свою очередь устанавливает в рабочее состояние третью и т.д. последняя ячейка устанавливает в рабочее состояние первую ячейку.

Правило 2. Ячейка $\ell_{i,j}$ устанавливает некоторую ячейку $\ell_{j,p}$ в подготовительное состояние, аналогично, ячейка $\ell_{i,c}$ устанавливает некоторую ячейку $\ell_{c,\kappa}$ в подготовительное состояние, при этом если в графе переходов существует переход $b_j \to b_i$ $(b_c \to b_i)$, то в качестве ячейки $\ell_{j,p}$ $(\ell_{c,\kappa})$ необходимо использовать ячейку $\ell_{j,i}$ $(\ell_{c,i})$.

Правило 3. а) при отсутствии в графе перехода $b_j \to b_i$ ячейка $\ell_{j,p}$ устанавливает ячейку $\ell_{i,j}$ в нерабочее сос-

тояние; б) при отсутствии в графе перехода $b_j \to b_c$ или при наличии обоих переходов $b_c \to b_i$ и $b_c \to b_j$ ячейка $e_{j,p}$ устанавливает одновременно ячейку $e_{c,k}$ в нерабочее состояние с запретом и ячейку $e_{i,c}$ – в нерабочее состояние;
в) при наличии в графе перехода $b_j \to b_c$ и при отсутствии хотя бы одного перехода $b_c \to b_j$ или $b_c \to b_i$ ячейка $e_{j,p}$ ($p=c$) устанавливает одновременно ячейку $e_{c,k}$ в подготовительное состояние и ячейку $e_{i,c}$ – в нерабочее состояние.

В качестве примера рассмотрим граф переходов, изображенный на рисунке 3. В соответствии с вышеприведенными правилами на рисунке 4 показаны связи между ячейками, каждой связи сопоставлено номер соответствующего правила. С целью экономии числа входов логических элементов некоторые связи можно совместить, например, точку (а) можно совместить с точкой) (б) (см. рис.4).

В случае полных графов переходов, связи между ячейками достаточно просто можно выразить аналитически, определив функции связи. Пусть имеется полный граф переходов, содержащий вершины $b_1, \ldots, b_R$. Найдем функции связи $f_{i,j}$, $\varphi_{i,j}$, $\psi_{i,j}$ элементов ячейки $e_{i,j}$: $f_{i,j} = f_{i,j,1} \vee f_{i,j,2}$,

$$\varphi_{i,j} = f_{i,j,1}, \quad \psi_{i,j} = \psi_{i,j,1} \vee \psi_{i,j,2},$$

где

$f_{i,j,1} = \varphi_{i,j} = z_{j,i}$,

$f_{i,j,2} = \psi_{i,j,2} = \bigvee z_{k,j}$ по всем $k = 1,2,\ldots,R, k \neq i,j$,

$\psi_{i,j,1} = \bigvee z_{k,j}$ по всем $k = 1,2,\ldots,R, k \neq i,j$.

Поясним некоторые функции. Функция $f_{i,j,2}$ и равная ей функция $\psi_{i,j,2}$ обращаются в I на множестве связей, устанавливающих ячейку $e_{i,j}$ в нерабочее состояние с запретом, $\psi_{i,j,1}$ — на множестве связей, устанавливающих ячейку $e_{i,j}$ в нерабочее состояние. В выражении функции $\varphi_{i,j}$ не отражена связь ячейки $e_{i,i}$ с другими ячейками, сопоставленной вершине b_i (правило I), эту связь надо иметь в виду при построении структуры. В качестве примера найдем функции связи между ячейками структуры, реализующей полный граф переходов из трех вершин. Эти функции сведем в таб-

лицу 2.

Стандартная реализация № 2.

Задание автомата такое же как в предшествующей реализации. Каждую вершину b_i графа переходов автомата в зависимости от свойств предшественников (множества K_i) и последователей (множества L_i) будем относить к одному из 4 типов вершин: вершины типа А, вершины типа B , вершины типа C , вершины типа $Д$. Каждый тип вершин реализуем ячейкой соответствующего типа (A , B , C , $Д$).

Будем говорить, что вершина b_i относительно множества K_i обладает свойством (I), если, для каждого $b_j \in K_i$ вершина b_i является единственным последователем. Иными словами $L_j = \{b_i\}$ для всякого $b_j \in K_i$, в противном случае вершина b_i относительно множества K_i обладает свойством (2).

Относительно множества L_i вершина b_i также может обладать одним из двух свойств, а именно, вершина b_i обладает свойством (3), если множество L_i состоит из единственной вершины b_j , которая одновременно является единственным последователем для каждого своего предшественника. Иными словами $L_i = \{b_j\}$ и $L_k = \{b_j\}$ для каждого $b_k \in K_j$. В противном случае вершина b_i обладает свойством (4). Вершину со свойствами (I) и (3) назовем вершиной типа A , вершину со свойствами (I) и (4) - типа B , вершину со свойствами (2) и (4) - типа C , вершину со свойствами (2) и (3) - типа $Д$. Каждому типу вершины можно сопоставить типы вершин предшественников и последователей. Если вершина графа переходов автомата принадлежит к типу A , то по определению ее последователями могут быть только вершины типа A или B , в том время как ее предшественниками могут быть только вершины типа A или $Д$. Построим таблицу, в которой каждому типу вершин сопоставим типы вершины последователей и предшественников.

Таблица 3

Тип вершин	Тип вершин-последователей	Тип вершин-предшественников
A	A, B	A, Д
B	C, Д	A, Д
C	C, Д	B, C
D	A, B	B, C

В качестве ячейки типа A используем ячейку, применяемую в предшествующей реализации (рис.2). Ячейки остальных типов показаны на рис.5.

Сформулируем для каждого типа вершин и соответствующих ячеек правила определения функций связи.

Правило I. Пусть вершина b_i является вершиной типа A, тогда функции связи ячейки типа A определяются следующим образом: $f_i = \varphi_i = \vee z_j$ по всем j таким, что $b_j \in K_i$

$\psi_i = \vee z_j$ по всем j таким, что $b_j \in L_i \setminus K_i$

Правило 2. Пусть вершина b_i является вершиной типа B. Ячейка типа B отличается от ячейки типа A дополнительным триггером ($q'-q''$) (рис.5). Функции связи ячейки типа B определяются следующим образом: $f_i = f_{i,1} \vee \varphi_i$

$\varphi_i = \vee z_j$ по всем j таким, что $b_j \in K_i$

$f_{i,1} = \vee F_{i,j}$ по всем j таким, что $b_j \in L_i$

$\psi_i = \vee z_j$ по всем j таким, что $b_j \in L_i \setminus K_i$

$\psi_i^* = \vee z_j$ по всем j таким, что $b_j \in L_i$

Правило 3. Пусть вершина b_i является вершиной типа C. Ячейка типа C состоит из двух триггеров типа $R-S$; ($e-e'$) и ($q'-q''$) (рис.5). Функции управления триггерами (функции связи ячейки) определим следующим образом:

$f_i = f_{i,1} \vee f_{i,2}$

где $f_{i,1} = \vee F_{i,j}$ по всем j таким, что $b_j \in L_i$

$f_{i,2} = \vee z_j$ по всем j таким, что $b_j \in K_i$

$\varphi_i = \vee U_j \cdot F_{j,i}$ по всем j таким, что $b_j \in K_i$

где U_j - выход дополнительного триггера ($q_j' - q_j''$) ячейки b_j типа B или типа C.

$\varphi_j^* = \vee z_j$ по всем j таким, что $b_j \ni L_i$

Правило 4. Ячейка типа D (рис.5) является обычным триггером типа $R-S$, функции управления которого определим следующим образом $\varphi_i = \vee U_j \cdot F_{j,i}$ по всем j таким, что $b_j \in K_i$

$f_i = f_{i,1} \vee f_{i,2}$

где $f_{i,1} = \vee F_{i,j}$ по всем j таким, что $b_j \in L_i$,

$f_{i,2} = z_j$ по всем j таким, что $e_j \in K_i$.

Ячейки типа C и $Д$ существенно отличаются от ячеек типа A и B. Для построения ячеек этого типа требуются дополнительные двухвходовые конъюнкторы (для реализации функций φ_i). Число конъюнкторов равно числу вершин в множестве K_i, которые согласно таблицы № 3 являются вершинами типа B или C. Элементы e_j, q'_j, q'', e'_i и e_i образуют модифицированную схему соединения элементов, обеспечивающую сохранение значения 0 на выходе z_i элемента e_i в переходах в вершину b_j.

Предлагаемый метод реализации заключается в последовательном просмотре состояний, при котором для каждого состояния определяется его тип (по множествам L_i, K_i), сопоставляется соответствующая ячейка e_i^J (где J - тип ячейки) и определяются функции связи. Покажем работу метода на примере синтеза структуры памяти автомата, заданного графом переходов (см. рис.6а).

Классифицируем состояние автомата: типа $A - b_4$, тип $B - b_1$, тип $C - b_2, b_3$, тип $Д - b_5, b_6$.

Сопоставим состояниям соответствующие ячейки e_4^A, e_1^B, e_2^C, e_3^C, e_5^D, e_6^D и, руководствуясь сформулированными правилами, определим функции связи.

Для ячейки e_1^B: $K_1 = \{b_4\}$, $L_1 = \{b_2, b_3, b_6\}$; $f_{1,1} = F_{1,2} \vee F_{1,3} \vee F_{1,6}$

$\varphi_1 = z_4$; $f_1 = F_{1,2} \vee F_{1,3} \vee F_{1,6} \vee z_4$; $\psi_1 = \psi_1^* = z_2 \vee z_3 \vee z_6$.

Для ячейки e_4^A: $K_4 = \{b_5, b_6\}$; $L_4 = \{b_1\}$; $f_{4,1} = \varphi_4 = z_5 \vee z_6$; $\psi_4 = z_1$;

Для ячейки e_2^C: $K_2 = \{b_1\}$; $L_2 = \{b_3, b_5\}$; $f_{2,1} = F_{2,3} \vee F_{2,5}$; $f_{2,2} = z_1$

$f_2 = F_{2,3} \vee F_{2,5} \vee z_1$; $\varphi_2 = U_1 \cdot F_{1,2}$; $\psi_2^* = z_3 \vee z_5$

Для ячейки e_3^C: $K_3 = \{b_1, b_2\}$; $L_3 = \{b_5\}$; $f_{3,1} = F_{3,5}$; $f_{3,2} = z_1 \vee z_2$

$f_3 = F_{3,5} \vee z_1 \vee z_2$; $\varphi_3 = U_1 \cdot F_{1,3} \vee U_2 \cdot F_{2,3}$; $\psi_3^* = z_5$

Для ячейки $e_5^{D_1}$: $K_5 = \{b_2, b_3\}$; $L_5 = \{b_4\}$; $f_{5,1} = F_{5,4}$;

$f_{5,2} = z_2 \vee z_3$; $f_5 = F_{5,4} \vee z_2 \vee z_3$; $\varphi_5 = U_2 \cdot F_{2,5} \vee U_3 \cdot F_{3,5}$

Для ячейки ℓ_6^D : $K_6 = \{b_1\}$; $L_6 = \{b_4\}$; $f_{6,1} = F_{6,4}$; $f_{6,2} = z_1$ $f_6 = F_{6,4} \vee z_1$; $\varphi_6 = u_1 \cdot F_{1,6}$

Структура предоставлена на рис.6б.

Анализ устойчивости функционирования структур, получаемых при стандартной реализации 3, проведем на примере структуры, представленной на рис.6б.

Рассмотрим переход из состояния типа A в состояние типа B (переход типа $A \to B$).

В нашем примере это переход $b_4 \to b_1$. Ячейка ℓ_4^A, находясь в рабочем состоянии ($z_4 = I$, $v_4 = 4$, $w_4 = I$) устанавливает ячейку ℓ_1^B в подготовительное состояние ($z_1 = 0$, $v_1 = 0$, $w_1 = I$, $u_1 = 0$). При изменении из 0 в I значения функции $F_{4,1}$ значение выхода z_4 становится равным 0, что вызывает переход ячейки ℓ_1^B в рабочее состояние ($z_1 = I$, $v_1 = 0$, $w_1 = I$, $u_1 = I$), что, в свою очередь, вызывает переход ячейки ℓ_4^B в нерабочее состояние ($z_4 = 0$, $v_4 = I$, $w_4 = 0$).

Такой процесс будет происходить всегда при переходах типа $A \to B$, ($A \to A$), поскольку вершина типа B не может быть предшественником вершины типа A (см. табл. № 3).

Выход z_1 соединен со входами выходных элементов ячеек $\ell_k^J \in \mathcal{L}_4$ (где $\mathcal{L}_4$ множество ячеек последователей ℓ_4^B, в нашем примере ячейки $\ell_2^C, \ell_3^C, \ell_6^D$), при этом элементы ℓ_4, q'_4, q''_4, e'_k и e_k образуют модифицированную схему соединения, поэтому в течение времени переходного процесса обеспечивается значение 0 на выходе элементов e^k (в нашем примере на выходе элементов e_2, e_3, e_6).

Рассмотрим переход типа $B \to C$. Ячейка ℓ_1^B, находясь в рабочем состоянии, в отличие от предыдущего типа ячеек, не изменяет состояния ячеек $\ell_k^J \in \mathcal{L}_1$ (ℓ_2^C, ℓ_3^C, ℓ_6^D), которые при этом находятся в нерабочем состоянии ($z_k = 0$, $u_k = 0$). При изменении из 0 вIнекоторой $F_{1,k}$ (например, $F_{1,2}$) z_1 станет равным 0, что разрешает триггерам $e'_k - e_k$ изменить свое состояние, но это произойдет только для одной ячейки, а именно для ячейки ℓ_2^C, после того как $\varphi_2 = u_1 \cdot F_{1,2}$ изменит свое значение из 0 в I, в результате

чего ячейка e_2^C перейдет в рабочее состояние ($z_2 = 1$, $u_2 = 1$), что вызовет переход ячейки e_1^B в нерабочее состояние ($z_1 = 0$, $v_1 = 1$, $w_1 = 0$, $u_1 = 0$): так как $Y_1 = Y_1^* = z_2 \vee z_3 \vee z_6$ Выход z_a соединен со входами выходных элементов ячеек e_3^C, e_5^D, обеспечивая $z_3 = 0$ и $z_5 = 0$ во время переходного процесса за счет модифицированной схемы соединения элементов $e_1, q_1', q_2'', e_3'(e_4')$ и $e_3(e_4)$, которая будет иметь место при переходе из ячеек типа B всегда, поскольку как это видно из таблицы № 3, последователями ячейки типа B могут быть только ячейки типа C или D. Переходы типа $C \to C$, $C \to D$ аналогичны переходу $B \to C$, переходы типа $D \to A$, $D \to B$ аналогичны переходу типа $A \to B$ и поэтому переходные процессы в этих переходах происходят по основной либо модифицированной схеме. Таким образом при всевозможных переходах в структуре состояния на входах выходных элементов ячеек изменяются по основной либо модифицированной схеме, что является гарантией отсутствия опасных состязаний в получаемой структуре.

Практическое применение

На основе предложенных методов разработаны программы (общий объем $\sim$1000 операторов Фортрана), которые являются основными в подсистеме стандартной реализации Диалоговой автоматизированной системы проектирования дискретных устройств (ДАСП). Подсистема воспринимает описание устройства (в виде графа переходов либо системы функций возбуждения и выходов), директивы о методе (ст. № 1 либо ст. 2) и структуру базиса реализации (тип элементов И-НЕ,ИЛИ-НЕ, их размещение в корпусах и т.д.). В подсистеме осуществляется реализация: памяти устройства одним из описанных методов, реберных функций в виде двухъярусной ДНФ с предварительной алгебраической оптимизацией и выходных функций.

Выходной информацией подсистемы является структура устройства, описанная в виде перечня корпусов и таблиц их соединений. Подсистема отлажена и испытана на потоке из 9 реальных задач. Всего было синтезировано 32 блока, средняя сложность блоков 9 корпусов, самый сложный блок – 44 корпуса. Базис реа-

лизации - Логика-2.

Проведенный эксперимент хорошо подтвердил одно из основных требований к алгоритмам для машинного проектирования - линейность затрат машинного времени и памяти относительно сложности исходного описания. Процессорное время синтеза самого сложного блока около 5 сек.

В заключение следует отметить важное свойство структур, построенных по любой из рассмотренных стандартных реализаций. Помимо высокой устойчивости к разбросу временных параметров логических элементов структура обладает повышенной устойчивостью к внешним помехам. В самом деле, выходные воздействия, ячейки, находящиеся в рабочем состоянии, можно использовать для удержания триггеров и выходных элементов остальных ячеек в нужном состоянии, не нарушая алгоритма функционирования структуры.

ПРИЛОЖЕНИЕ

Дадим краткий временный анализ основной схемы соединения элементов (рис.I). Для элемента e_α (элемента типа ИЛИ-НЕ) требуется обеспечить отсутствие ложного "0" на выходе той части его структуры, где реализуется операция ИЛИ, это достигается при выполнении неравенства

$$\Delta_{\gamma,\beta} + \Delta_\beta + \Delta_{\beta,\alpha} + \tau_{\beta,\alpha} \geq \Delta_{\gamma,\alpha} + \tau_{\gamma,\alpha} \qquad (*)$$

где $\tau_{\beta,\alpha}$ и $\tau_{\gamma,\alpha}$ означают соответственно задержку от входов $x_{\alpha,\beta}$, $x_{\alpha,\gamma}$ элемента e_α до выхода указанной части структуры этого элемента. С другой стороны, величину задержки Δ элемента можно представить как

$$\Delta = \Delta\,(\text{ИЛИ}) + \Delta\,(\text{НЕ}),$$

где $\Delta\,(\text{ИЛИ})$, $\Delta\,(\text{НЕ})$ - величины задержки тех частей структуры элемента, где реализуются соответственно операции ИЛИ, НЕ. Можно написать, что

$$\Delta\,(\text{ИЛИ}) = \frac{1}{p+1}\cdot\Delta, \qquad \Delta\,(\text{НЕ}) = \frac{p}{p+1}\cdot\Delta, \; p \geq 0$$

Разброс величины Δ запишем как

$$\tau_1 \leq \Delta \leq \tau_2$$

Подставим в неравенство (*) самые неблагоприятные значения параметров элементов:

$\Delta_\beta = T_1$, $\tau_{\beta,\alpha} = \frac{1}{\rho+1} \cdot T_1$, $\tau_{\gamma,\alpha} = \frac{1}{\rho+1} \cdot T_2$; $\Delta_{\gamma\beta} + \Delta_{\beta,\alpha} = \Delta_{\gamma,\alpha}$

После ряда преобразований нетрудно получить, что неравенство (*) будет выполняться при $\frac{T_2}{T_1} \leq \rho+2$. На рис.7 зависимость $T_2/T_1 = f(\rho)$ обозначена цифрой I. Для основных схемотехнических реализаций функции ИЛИ-НЕ (диодно-транзисторных, резистивно-транзисторных, транзисторных, с эмиттерными связями между логическими элементами и т.д.) величина $\rho \geq 1$, поэтому такой способ соединения элементов, как видно из рисунка, будет "эффективным" при значительном разбросе временных параметров элементов.

Если величины T_1, T_2, ρ таковы, что неравенство (*) не выполняется, то включением z четного числа дополнительных элементов в линию ($z_\beta - x_{\alpha,\beta}$) (рис.I) всегда можно обеспечить его выполнение, в этом случае число определяется из следующего неравенства:

$$\frac{T_2}{T_1} \leq \rho + 2 + z(\rho+1).$$

Схему соединения элементов в этом случае будем называть <u>модифицированной</u>.

Для сравнения проведен аналогичный временный анализ схемы соединения, используемой в работе [3], получено, что неравенство (*) выполняется при

$$T_2/T_1 \leq 1 + \frac{\rho+1}{\rho+2}$$

На рисунке 7 зависимость $T_2/T_1 = 1 + \frac{\rho+1}{\rho+2}$ изображена кривой 2.

ЛИТЕРАТУРА

I. Р.Миллер. Теория переключательных схем, т.2, М., 1971.

2. Ю.С.Наумов, Н.А.Аваев, М.А.Бедрековский, Помехоустойчивость устройств на интегральных логических схемах. М., Советское радио, 1975.

3. R.J.David, C.Laurent, R.Ferret. Principle and realisation with MOS technology of an universal cell for asinchronous seguences. Труды международного семинара по прикладным аспектам теории автоматов. Варна, 1971, том I, № I.

4. А.И.Потехин. Об одном подходе к синтезу асинхронных автоматов.Труды международного симпозиума. "Дискретные системы", изд. "Зинатне", Рига, 1971, т.I.

Таблица 2.

	$\mathcal{F}_{i,j,1}=\phi_{i,j}$	$\mathcal{F}_{i,j,2}=\psi_{i,j,2}$	$\psi_{i,j,1}$
$\ell_{1,2}$	$z_{2,1}$	$z_{3,2}$	$z_{3,1}$
$\ell_{1,3}$	$z_{3,1}$	$z_{2,3}$	$z_{2,1}$
$\ell_{2,1}$	$z_{1,2}$	$z_{3,1}$	$z_{3,2}$
$\ell_{3,1}$	$z_{1,3}$	$z_{2,1}$	$z_{2,3}$
$\ell_{3,2}$	$z_{2,3}$	$z_{1,2}$	$z_{1,3}$

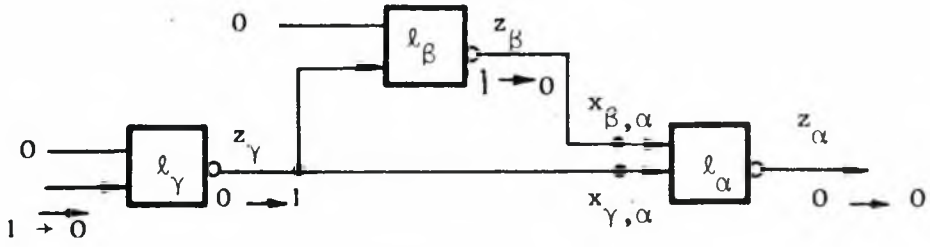


Рис. 1.

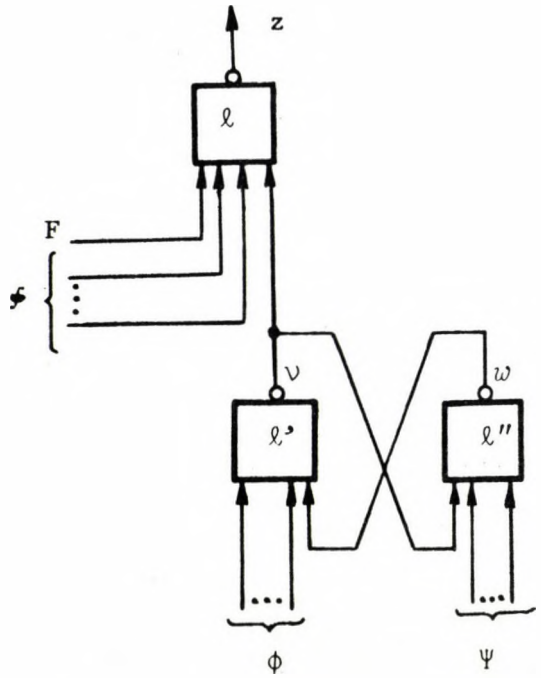


Рис. 2.

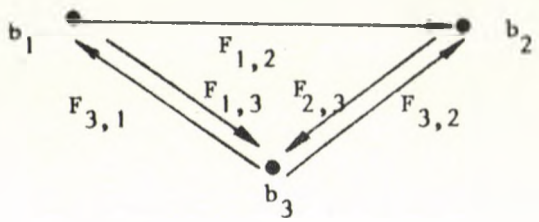


Рис. 3.

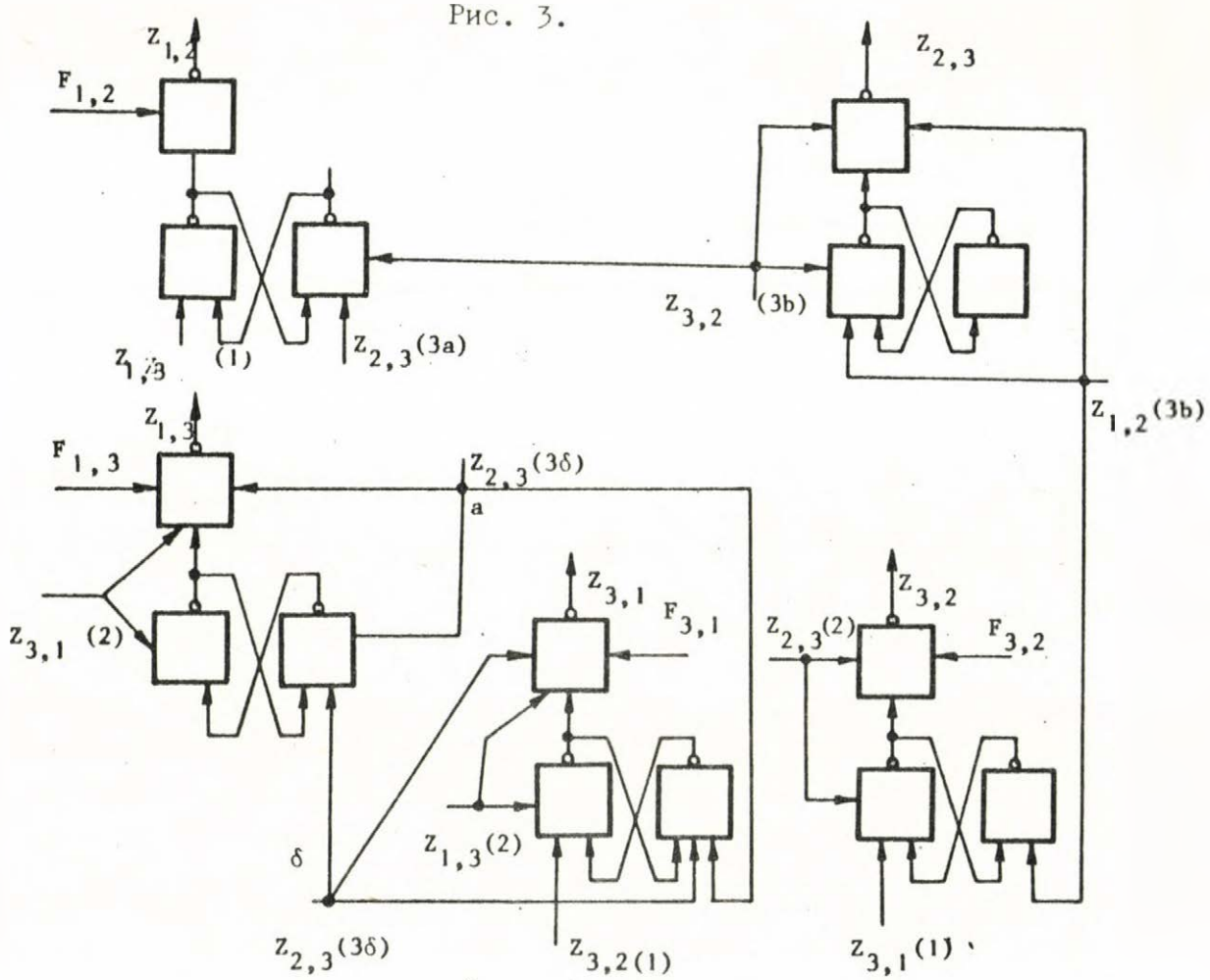


Рис.4.

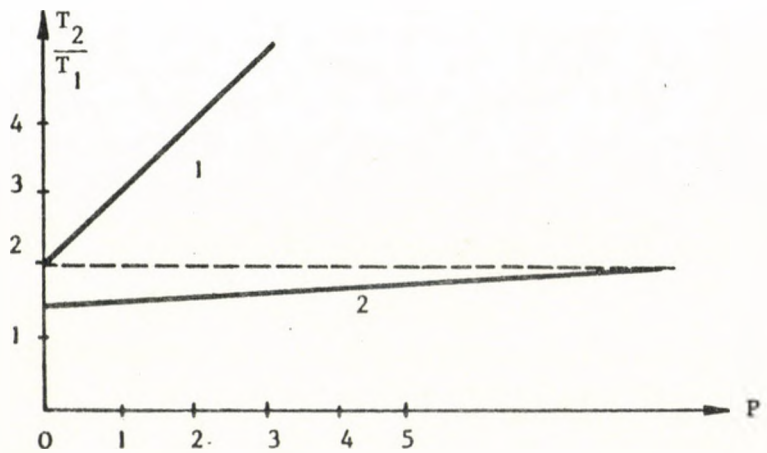
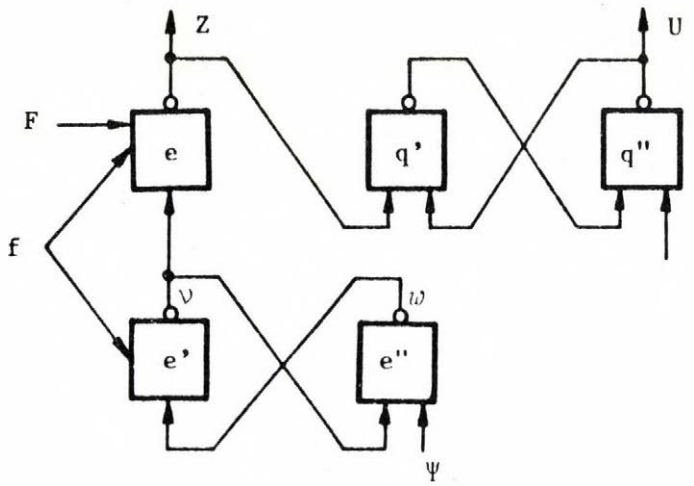
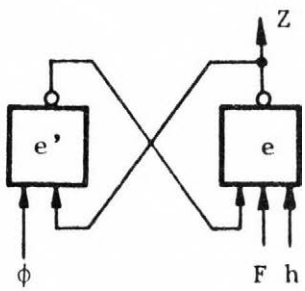


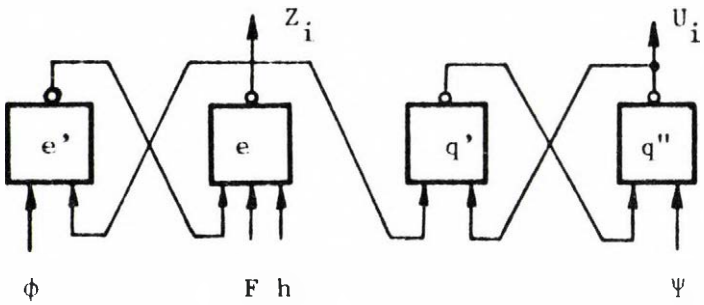
Рис. 7.



ячейка типа B

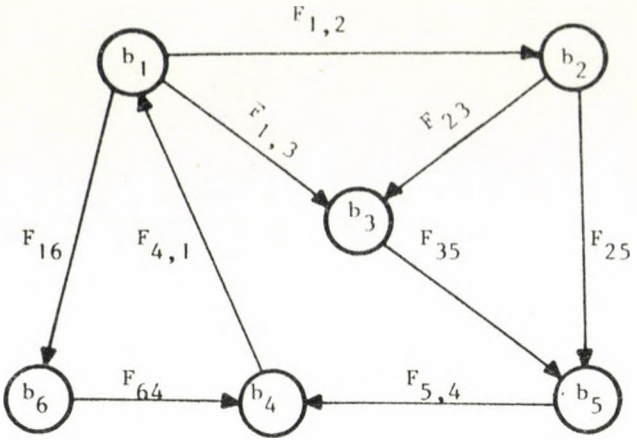


ячейка типа D

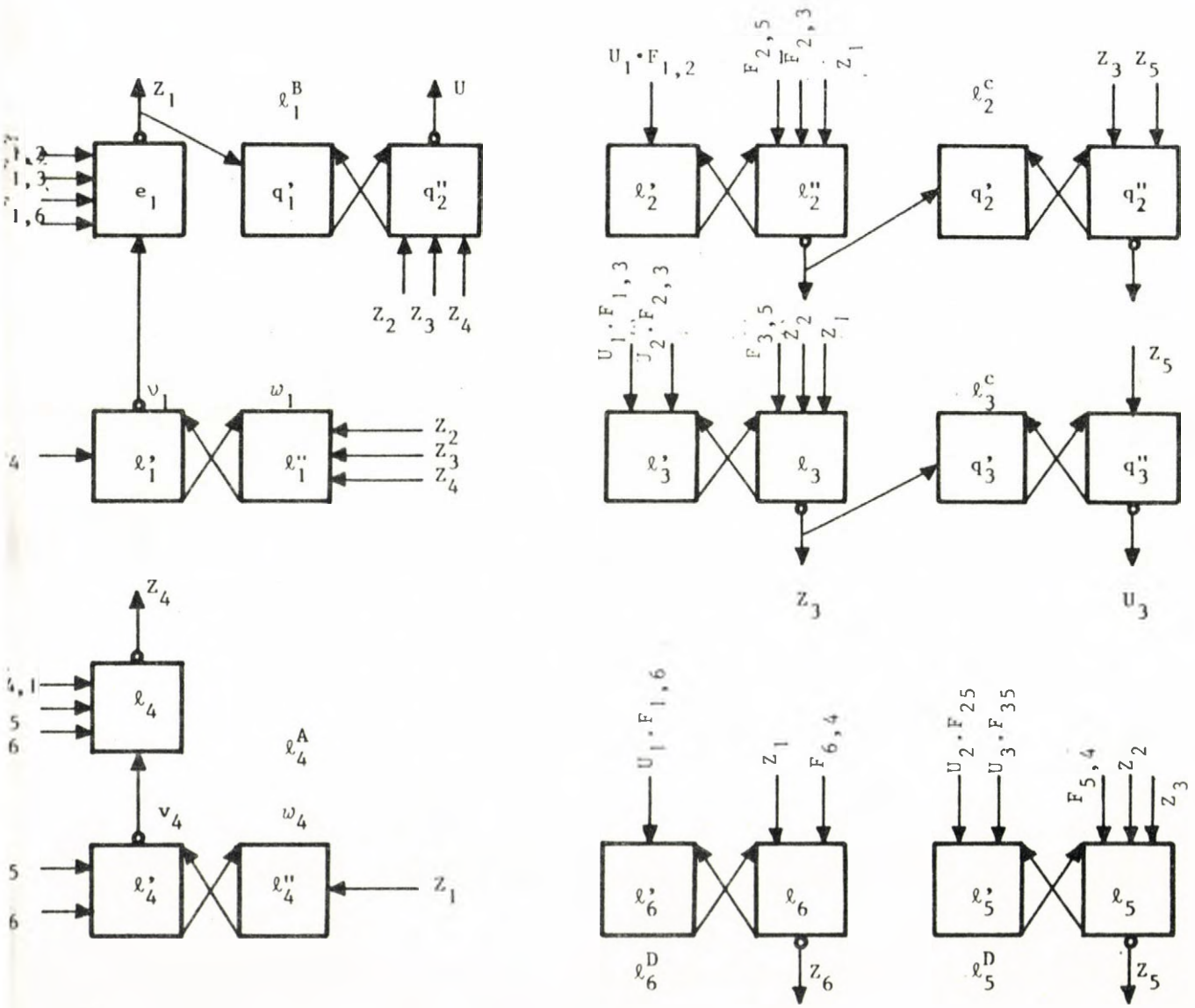


ячейка типа C

Рис. 5.


b1
b2
b3
b4
b5
b6
F1,2
F1,3
F23
F16
F4,1
F35
F25
F64
F5,4

а)


e1
q'1
q''2
Z1
U
Z2 Z3 Z4
ℓ'1
ℓ''1
ν1
ω1
ℓ'2
ℓ''2
q'2
q''2
U1·F1,2
F2,5
F2,3
Z1
Z3 Z5
U1·F1,3
U2·F2,3
F3,5
Z2
Z1
ℓ'3
ℓ3
q'3
q''3
Z5
Z3
U3
Z4
ℓ4
ν4
ω4
ℓ'4
ℓ''4
U1·F1,6
Z1
F6,4
ℓ'6
ℓ6
Z6
U2·F25
U3·F35
F5,4
Z2
Z3
ℓ'5
ℓ5
Z5

б)

Рис. 6.

ПОСТРОЕНИЕ КОНЕЧНОГО АВТОМАТА ПО ОПИСАНИЮ АЛГОРИТМА ФУНКЦИОНИРОВАНИЯ ДИСКРЕТНОГО УСТРОЙСТВА НА ЯЗЫКЕ ОСПАП.

В.В.Девятков

I. Введение.

Теория конечных автоматов накопила довольно много методов оптимизации и преобразований, которые могут быть успешно использованы при логическом проектировании дискретных устройств. Однако эти методы в своём подавляющем большинстве связаны с такими языками описания конечных автоматов (автоматными языками), как таблицы переходов, графы переходов, матрицы переходов и т.п. Описание реальных дискретных устройств большой размерности сразу на автоматных языках вызывает большие трудности. В последнее время, особенно в связи с развитием автоматизированного логического проектирования, появилось много языков описания дискретных устройств и систем, являющихся входными (первичными) языками для последующего логического проектирования обладающих широкими изобразительными возможностями и удобных для описания дискретных устройств. Однако вследствие их широты и "неавтоматности" методы теории автоматов становятся трудноприменимыми к описанию на этих языках. Для разрешения этой трудности идут различными путями.

Один из таких путей состоит в том, что непосредственно в первичном языке разрабатываются процедуры эквивалентных преобразований, используемые, в частности,для целей оптимизации проектируемого устройства. Эти процедуры иногда интерпретируются в автоматных языках [I].

Другой путь состоит в том, что разрабатываются эвристические процедуры перехода от описания в данном первичном языке к описанию в автоматном языке, на уровне последнего либо осуществляются преобразования по известным методам тео-

рии автоматов, либо вообще никаких преобразований не производится [2].

Основной недостаток первого пути состоит в том, что смена первичного языка требует разработки новых методов преобразований. Недостаток второго пути состоит в том, что преобразования, если они есть, носят далеко невсеобъемлющий и недостаточно объективный характер, вследствие эвристичности процедур перехода к автоматным языкам.

Один из подходов, исключающий указанные недостатки, состоит в следующем. Разрабатывается некоторый формальный базовый язык, достаточно гибкий, широкий, пригодный служить в качестве модели для интерпретации первичных языков различного назначения и возможностей.

Для базового языка разрабатывается процедура перевода его в автоматный, а для каждого нового первичного языка разрабатывается процедура перевода его в базовый, в то время как процедура перевода базового языка в автоматный остаётся всё время одной и той же.

При этом обе процедуры должны гарантировать сохранение всей информации, которая может быть использована при преобразованиях, производимых на уровне автоматного языка.

Описанный подход реализован в диалоговой автоматизированной системе проектирования (ДАСП) логики дискретных управляющих устройств [3]. Одним из первичных языков этой системы является близкий к естественному язык описания архитектуры и алгоритма функционирования УСЛОВИЕ [4]. Базовым языком системы является язык операторных схем параллельных алгоритмов с памятью (ОСПАП) [5].

В настоящей статье рассматриваются принципы перехода от базового языка ОСПАП к автоматному языку.

2. Краткие сведения о языке ОСПАП.

Описание на языке ОСПАП (или просто ОСПАП) может быть представлено в виде графа с пятью типами вершин-операторов (рис.I): стартёр, предикат, функтор, разветвитель и останов.

Вершины-операторы (или в дальнейшем просто операторы) могут соединяться друг с другом дугами с учётом следующих ограничений (входящие и выходящие дуги операторов также показаны на рис.I): стартёр не имеет входящих дуг и должен иметь только одну выходящую; останов не может иметь выходящих дуг и должен иметь одну или более входящих; предикат должен иметь не менее одной входящей и не менее одной выходящей дуги; функтор имеет одну входящую и одну выходящую дугу; разветвитель имеет одну входящую и не менее двух выходящих дуг.

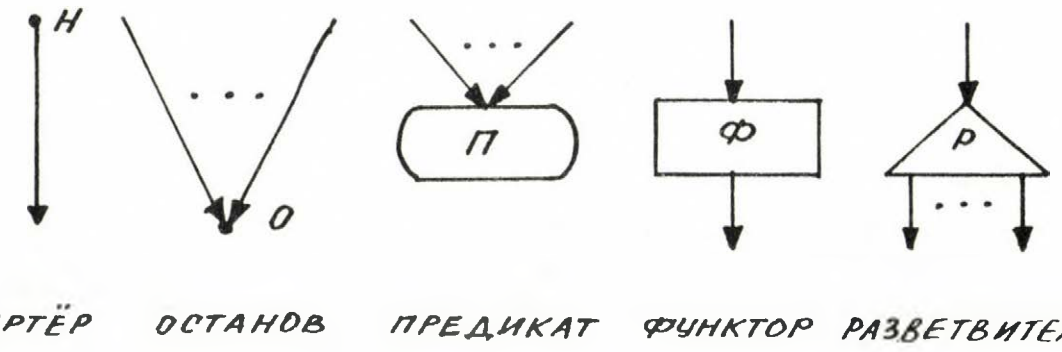


Рис.I.

Стартёр в ОСПАП может быть только один, разветвитель не может быть соединён с другим разветвителем или с самим собой, функтор не может соединяться непосредственно с другим функтором, с самим собой или только через разветвитель (т.е. между двумя функторами должен быть предикат).

Каждой ОСПАП соответствует множество входных переменных $X=\{x_1,\dots,x_m\}$, внутренних переменных $Y=\{y_1,\dots,y_n\}$ и множество выходных переменных $Z=\{z_1,\dots,z_p\}$. Каждой выходящей дуге предиката взаимнооднозначно соответствует булевая функция $f(x_1,\dots,x_m,y_1,\dots,y_n,z_1,\dots,z_p)$, а каждому функтору соответствует некоторое множество выражений вида $z:=f(x_1,\dots,x_m,y_1,\dots,y_n,z_1,\dots,z_p)$ или $y:=f(x_1,\dots,x_m,y_1,\dots,y_n,z_1,\dots,z_p)$ (1) где $x_i\in X, y_j\in Y, z_h\in Z$. Некоторые переменные в функциях могут быть несущественными.

Каждая ОСПАП задаёт алгоритм функционирования дискретного устройства, имеющего множество входов X и множество выхо-

дов Z . Считается, что этот алгоритм должен выполняться в дискретном времени, а его выполнение, в некоторый момент времени состоит в выполнении в течении этого момента одновременно некоторой группы операторов, после выполнения которых в следующий момент времени выполняется другая группа операторов (в которой один, несколько или даже все операторы могут совпадать с выполняемыми в предыдущий момент времени).

Выполнение каждого оператора S_i в некоторый момент времени состоит в следующем.

Если S_i -стартёр, то выполнение его в некоторый момент времени состоит в указании оператора (предиката, функтора или разделителя), который должен выполняться в следующий момент времени. Этим оператором является тот, для которого выходящая дуга стартёра является входящей.

Если S_i - функтор, то он по множеству значений входных и внутренних переменных вычисляет в один и тот же момент времени множество значений выходных и внутренних переменных и указывает оператор, к выполнению которого следует приступить в тот же момент времени. Этим оператором является тот, для которого выходящая дуга функтора является входящей.

Если S_i - предикат, то он по множеству значений входных и внутренних переменных вычисляет значения функций, соответствующих его выходящим дугам.

При этом только одна из этих функций должна принимать единичное значение (в противном случае описание на языке ОСПАП считается некорректным) и дуга, которой она соответствует, указывает на оператор, к выполнению которого следует перейти в следующий момент времени.

Этим оператором является тот, для которого указанная дуга является входящей.

Если S_i - разветвитель, то он указывает множество операторов, к одновременному выполнению которых следует перейти в тот же момент времени. Этими операторами являются те, для которых выходящие дуги разветвителя являются входящими.

Если S_i - останов, то он не указывает ни на какой опе-

ратор, который после него должен выполняться. Если в данный момент выполняются только остановы, то в последующие моменты либо вообще никакие операторы не будут выполняться, либо будут выполняться операторы, указываемые некоторым особым образом.

Алгоритм функционирования, задаваемый ОСПАП, указывает, каким образом в каждый данный момент времени должно вычисляться множество значений выходных переменных и, может быть, множество значений внутренних переменных, если известно множество значений входных переменных и может быть, множество значений внутренних переменных, вычисленное в предыдущий момент времени.

3. Постановка задачи.

ОСПАП как язык описания алгоритма функционирования дискретного устройства не использует понятия внутреннего состояния. В настоящее время имеется довольно много работ, исследующих вопросы построения абстрактного конечного автомата по описанию алгоритма функционирования автомата множеством входо-выходных последовательностей, которые он должен реализовать [6 ÷ 10]. Достоинством этих работ является то, что в них построены конструктивные алгоритмы перехода к таблицам или графам переходов абстрактных автоматов от задания на языке описания множеств входо-выходных последовательностей, а также то, что в них разработаны процедуры, позволяющие выявить непосредственно в этих языках те или иные свойства автомата.

Недостаток указанных языков как первичных тот же, что и таких автоматных языков как таблицы переходов: прежде всего это бедность их изобразительных средств и практическая невозможность описания поведения автоматов большой размерности. Последнее в значительной степени связано с тем, что эти языки имеют дело с абстрактными входными и выходными состояниями. Вследствие указанных недостатков такие языки не могут быть использованы в качестве базовых.

Нашей задачей является построение инициального детерминированного конечного автомата (ИДКА), который бы выполнял алгоритм функционирования, задаваемый ОСПАП. Это означает, что множеством входов этого автомата должно быть множество X,

множеством выходов - множество Y , и этот автомат, будучи установленным в некоторое начальное состояние в момент начала отсчёта времени, точно также должен перерабатывать множество значений входных переменных в множество значений выходных переменных, как это задается ОСПАП.

Конечной целью решения этой задачи должно быть построение алгоритма, который бы непосредственно по ОСПАП, не переходя от неё к какому-либо языку, оперирующему с понятиями абстрактных входных, выходных и внутренних состояний, позволял получать ИДКА, представляемый системой булевых функций.

Для того, чтобы более подробно изложить суть метода построения ИДКА, ограничимся ОСПАП без разветвителей, с функторами, допускающими только выражения вида $z := \theta, \theta \in \{0,1\}$. и с предикатами, имеющими функции, зависящие только от входных переменных, такие ОСПАП будем называть простыми.

4. Построение автомата по множеству функциональных вв - последовательностей.

На рис.2 показана простая ОСПАП. На дугах, выходящих из предикатов поставлены соответствующие им булевые функции, выражения вида (I) записаны внутри функторов.

Пусть функтору Φ_i соответствует множество выражений $z_a := \theta_a, \ldots, z_t := \theta_t$, где $\theta_i \in \{0,1\}$. Обозначим через φ_i функцию $z_a^{\theta_a} \& \ldots \& z_t^{\theta_t}$, где $z_i^{\theta_i} = z_i$, если $\theta_i = 1$, и $z_i^{\theta_i} = \bar{z}_i$, если $\theta_i = 0$.

Дугу, выходящую из стартёра, будем считать помеченной символом e , а дугу, выходящую из функтора Φ_i будем считать помеченной функцией φ_i , учитывая это, выпишем все пути, ведущие от стартёра к останову:

$$
\begin{array}{llllllll}
1. & e & f_{11} & \varphi_1 & & & & \\
2. & e & f_{12} & \varphi_2 & f_{21} & \varphi_3 & & \\
3. & e & f_{12} & \varphi_2 & f_{22} & \varphi_4 & f_{31} & \varphi_6 \\
4. & e & f_{12} & \varphi_2 & f_{22} & \varphi_4 & f_{32} & \varphi_7 \\
5. & e & f_{12} & \varphi_2 & f_{22} & \varphi_4 & f_{33} & \varphi_8 \\
6. & e & f_{12} & \varphi_2 & f_{23} & \varphi_5 & & \\
 & t_0 & & t_1 & & t_2 & & t_3
\end{array}
\qquad (2)
$$

По путям (2) могут быть получены все вв-последовательности, которые должен реализовать автомат, алгоритм функционирования которого задан ОСПАП на рис.2.

В (2) ниже всех последовательностей проставлены моменты времени, в которые происходит выполнение операторов, соответствующих функциям при выполнении алгоритма функционирования ОСПАП. Рассмотрим одну из этих последовательностей, например, вторую. В момент времени t_0 выполняется оператор H. Множество значений входных переменных, определяющее структурное входное состояние автомата в этот момент может считаться неопределенным, если на то нет специальных указаний. Точно также в этот момент выходное структурное состояние может считаться неопределенным. В момент времени t_1 входным структурным состоянием автомата может быть любое из состояний, на которых функция f_{12} принимает единичное значение и при любом из этих состояний автомат должен выдавать выходное состояние I, поскольку ОСПАП на рис.2 имеет только одну выходную переменную z и при выполнении функтора φ_1 осуществляется присвоение единичного значения этой переменной. В момент времени t_2 входным структурным состоянием автомата может быть любое из состояний, на которых функция f_{21} принимает единичное значение и при любом из этих состояний автомат должен выдавать выходное состояние I. В любой момент времени t_{2+k}, $k \geq 1$ входное и выходное состояния, продолжающие данную последовательность неопределены. Обозначим через $\mathcal{A}(f_e)$ - множество всех структурных входных состояний, на которых функция f_e принимает единичное значение, $\mathcal{C}(\varphi_e)$ - множество всех структурных выходных состояний, на которых функция φ_e принимает единичное значение, e и Δ-символы соответственно неопределенных входного и выходного состояний. Через $\mathcal{S}$ обозначим множество всех вв-последовательностей, полученных по данной ОСПАП. Функции f_e будем называть входными, φ - выходными, через $\mathcal{P}(\mathcal{S})$ или просто через $\mathcal{P}$ будем обозначать множество всех последовательностей $u^* = e f_a \dots f_e \dots f_t$, полученных по последовательностям $e \Delta f_a \varphi_a \dots f_e \varphi_e \dots f_t \varphi_t$ и их собственным началам. Последовательности u^* будем на-

зывать входными. Через $\varphi^*(\alpha^*)$ будем обозначать функцию φ_t. Будем говорить, что вв-последовательность $e\alpha_{f_a}\varphi_a \dots f_t\varphi_t$ реализуется в некотором автомате, если после подачи на вход этого автомата любой последовательности $e\alpha_a \dots \alpha_t$, где $\ell = a, \dots, t$, $\alpha_\ell \in \mathcal{A}(f_\ell)$ на выходе появляется выходное состояние $\gamma \in \mathcal{C}(\varphi_\ell) = \mathcal{C}(\varphi^*(e\alpha_a \dots \alpha_\ell))$.

Множество $\mathcal{S}$ реализуется некоторым автоматом, если каждая вв-последовательность этого множества реализуется в этом автомате.

Сформулируем теперь алгоритм построения автомата, реализующего множество $\mathcal{S}$, а затем покажем, что этот алгоритм действительно позволяет построить автомат, реализующий множество $\mathcal{S}$.

Алгоритм I.

а) Пронумеровать числами I,2,3,... входные функции всех входных последовательностей множества $\mathcal{P}$ таким образом, что каждой определенной функции взаимнооднозначно соответствует свой номер, а нумерация входных функций каждой последовательности идёт подряд слева направо. Функции e присвоить номер 0.

б) Каждой входной последовательности $ef_a \dots f_\ell \dots f_t$ множества $\mathcal{P}$ (ℓ - номер входной функции f_ℓ) сопоставить множество функций $Y_{\ell+1} = u_\ell y_\ell$, где

$$u_\ell = \begin{cases} f_\ell & \text{, если } \ell \geq 1, \\ 0 & \text{, если } \ell = 0. \end{cases}$$

в) Получить функции z_I, $I = 1,2,\dots,\rho$. Каждая функция z_I получается как дизъюнкция всех тех переменных y_ℓ, входящих в правые части функций $Y_{\ell+1} = u_\ell y_\ell$, для которых имеет место $\varphi^*(ef_a \dots f_\ell) = z_A^{\theta_A} \dots z_I^{\theta_I} \dots z_T^{\theta_T}$, $\theta_I = 1$.

г) Принять $\tau = 1$, $s = 0$, $z_I^{(s)} = z_I$.

д) Получить функцию $y_\tau^{(s)}$ как дизъюнкцию всех тех переменных y_ℓ, введенных в п.б), начиная с любой, не вошед-

шей ещё ни в одну функцию $y_r^{(s)}$, для которых наборы значений $z_1 y_e, z_2 y_e, \ldots, z_p y_e$ при y_e =I одинаковы (при условии, что $y_k \cdot y_e = 0$, если $k \neq e$).

е) Если имеются внутренние переменные y_e, не вошедшие ни в одну функцию $y_r^{(s)}$, то принять $r = r + I$ и перейти к п.д.). В противном случае перейти к следующему пункту.

ж) Получить функции $Y_i^{(s)}, i = 1, 2, \ldots, r$, как дизъюнкцию всех тех функций возбуждения Y_e, которым в функции $y_i^{(s)}$ соответствуют переменные (конъюнкции) y_e. Принять $j = 1$, $A_j^{(s)} = \varnothing$.

з) Поместить функцию $y_j^{(s)}$ в множество $A_j^{(s)}$.

и) Найти все функции $y_j^{(s)} \cdot Y_i^{(s)}$, $i = 1, 2, \ldots, r$ для каждой функции $y_j^{(s)}$.

к) Если в п.и) для некоторой функции $y_j^{(s)}$ найдены неравные нули функции $y_j^{(s)} \cdot Y_a^{(s)}, y_j^{(s)} \cdot Y_b^{(s)}, \ldots, y_j^{(s)} \cdot Y_t^{(s)}$, $\{a, b, \ldots t\} \subseteq \{1, 2, \ldots, r\}$, число которых не менее двух и существуют пары функций $y_j^{(s)} \cdot Y_g^{(s)} = \bigvee u_g \cdot y_g \cdot Y_g^{(s)}$, $y_j^{(s)} \cdot Y_h^{(s)} = \bigvee u_k \cdot y_k \cdot Y_h^{(s)}$ такие, что хотя бы для одной пары $g \neq h$ имеет место $u_g \cdot u_h \neq 0$, то разбить функцию $y_j^{(s)}$ на подфункции, каждая из которых является дизъюнкцией всех y_g, y_h, для которых $u_g \cdot u_h = 0$ для всех различных пар, а дизъюнкция этих подфункций даёт $y_j^{(s)}$. Заменить в множестве $A_j^{(s)}$ функцию $y_j^{(s)}$ полученными подфункциями.

В противном случае перейти к следующему пункту.

л) Принять $j = j + 1$. Если $j \leq r$, то перейти к п.з). В противном случае перейти к следующему пункту.

м) Если суммарное число подфункций в множествах $A_1^{(s)}$, $A_2^{(s)}, \ldots, A_r^{(s)}$ равно r, то перейти к п.н). В противном случае перенумеровать все подфункции множеств $A_1^{(s)}$, $A_2^{(s)}, \ldots, A_r^{(s)}$ числами I,2,..., r_1, принять $r = r_1$, присвоить им обозначения $y_j^{(s)}$, где $s = s + 1$, $j = 1, \ldots, r = r_1$, перейти к п.ж.).

н) Конец. Функции $Y_1^{(s)}, \ldots, Y_r^{(s)}$ задают детерминированный конечный автомат.

Пример I. $S = \{ e\varphi_1, f_2\varphi_1, f_3\varphi_2, f_3 f_1 \varphi_2, f_3 f_2 \varphi_1, f_3 f_3 \varphi_1, f_3 f_2 f_1 \varphi_2, f_3 f_2 f_2 \varphi_2, f_3 f_2 f_3 \varphi_1 \}$, (3)

$f_1 = \bar{x}_1 \bar{x}_2$, $f_2 = \bar{x}_1 x_2$, $f_3 = x_1$, $\varphi_1 = z_2$, $\varphi_2 = z_1$.

Согласно п.а) алгоритма I перенумеруем входные функции последовательностей множества $\mathcal{P}(S)$ следующим образом:

$f_2 = u_1$, $f_3 = u_2$, $f_3 f_1 = u_3 u_4$, $f_3 f_2 = u_5 u_6$, $f_3 f_3 = u_7 u_8$,

$f_3 f_2 f_3 = u_9 u_{10} u_{11}$, $f_3 f_2 f_2 = u_{12} u_{13} u_{14}$, $f_3 f_2 f_3 = u_{15} u_{16} u_{17}$, $e = u_0$.

Получим согласно п.б) множество функций Y_ℓ :

$Y_1 = 0$ $\quad Y_7 = u_6 y_6 = x_1 y_6$, $\quad Y_{13} = u_{12} y_1 = x_1 y_1$,

$Y_2 = u_1 y_1 = \bar{x}_1 x_2 y_1$, $\quad Y_8 = u_7 y_1 = x_1 y_1$, $\quad Y_{14} = u_{13} y_{13} = \bar{x}_1 x_2 y_{13}$,

$Y_3 = u_2 y_1 = x_1 y_1$, $\quad Y_9 = u_8 y_8 = x_2 y_8$, $\quad Y_{15} = u_{14} y_{14} = \bar{x}_1 x_2 y_{14}$,

$Y_4 = u_3 y_1 = x_1 y_1$, $\quad Y_{10} = u_9 y_1 = x_2 y_1$, $\quad Y_{16} = u_{15} y_1 = x_1 y_1$,

$Y_5 = u_4 y_4 = \bar{x}_1 \bar{x}_2 y_4$, $\quad Y_{11} = u_{10} y_{10} = \bar{x}_1 x_2 y_{10}$, $\quad Y_{17} = u_{16} y_{16} = \bar{x}_1 x_2 y_{16}$,

$Y_6 = u_5 y_1 = x_1 y_1$, $\quad Y_{12} = u_{11} y_{11} = \bar{x}_1 \bar{x}_2 y_{11}$, $\quad Y_{18} = u_{17} y_{17} = x_1 y_{17}$.

Поскольку в данном примере только два выхода, то согласно п.в) будем иметь:

$z_1 = y_3 \vee y_4 \vee y_5 \vee y_6 \vee y_8 \vee y_{10} \vee y_{12} \vee y_{13} \vee y_{15} \vee y_{16}$, (4)

$z_2 = y_1 \vee y_2 \vee y_7 \vee y_9 \vee y_{11} \vee y_{14} \vee y_{17} \vee y_{18}$.

Значения функций $z_I \cdot y_\ell$ при $y_\ell = 1$, требуемые согласно п.г) для всех z_I и y_ℓ будут следующими

z_I \ y_ℓ	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}	y_{16}	y_{17}	y_{18}
z_1	0	0	1	1	1	1	0	1	0	1	0	1	1	0	1	1	0	0
z_2	1	1	0	0	0	0	1	0	1	0	1	0	0	1	0	0	1	1

Согласно п.г), д), е) при $r = I$, $s = 0$ будем иметь

$y_r^{(s)} = y_1^{(0)} = y_1 \vee y_2 \vee y_7 \vee y_9 \vee y_{11} \vee y_{14} \vee y_{17} \vee y_{18}$,

а при $r = 2$, $s = 0$

$y_r^{(s)} = y_2^{(0)} = y_3 \vee y_4 \vee y_5 \vee y_6 \vee y_8 \vee y_{10} \vee y_{12} \vee y_{13} \vee y_{15} \vee y_{16}$.

По п.ж) получим

$Y_1^{(0)} = \bar{x}_1 x_2 y_1 \vee x_1 y_6 \vee x_1 y_8 \vee \bar{x}_1 x_2 y_{10} \vee \bar{x}_1 x_2 y_{13} \vee$

$\bar{x}_1 x_2 y_{10} \vee x_1 y_{17} = x_1 (y_6 \vee y_8 \vee y_{17}) \vee \bar{x}_1 x_2 (y_1 \vee y_{10} \vee y_{13} \vee y_{16})$.

$Y_2^{(0)} = x_1 y_1 \vee \bar{x}_1 \bar{x}_2 y_4 \vee \bar{x}_1 \bar{x}_2 y_{11} \vee \bar{x}_1 x_2 y_{14} = x_1 y_1 \vee$
$\bar{x}_1 \bar{x}_2 (y_4 \vee y_{11}) \vee \bar{x}_1 x_2 y_{14}$.

Согласно п.з) поместим функцию $y_1^{(0)}$ в множество $A_1^{(0)} = \{ y_1^{(0)} \}$.

По п.и) вычислим

$y_1^{(0)} \cdot Y_1^{(0)} = x_1 y_{17} \vee \bar{x}_1 x_2 y_1$,

$y_1^{(0)} \cdot Y_2^{(0)} = \bar{x}_1 \bar{x}_2 y_{11} \vee \bar{x}_1 x_2 y_{14} \vee x_1 y_1$.

Поскольку $y_1^{(0)} \cdot Y_1^{(0)} \cdot Y_2^{(0)} = x_1 y_1 y_{17} \vee \bar{x}_1 x_2 y_1 y_{14}$ то согласно п.и) переменные y_1 и y_{14}, y_1 и y_{17} должны входить в разные подфункции функции $y_1^{(0)}$. Заменим в множестве $A_1^{(0)}$ функцию $y_1^{(0)}$ на две подфункции

$$y_{1,1}^{(0)} = y_1 \vee y_2 \vee y_7 \vee y_9 \vee y_{18},$$
$$y_{1,2}^{(0)} = y_{11} \vee y_{14} \vee y_{17},$$

в результате чего получим множество $A_1^{(0)} = \{ y_{1,1}^{(0)}, y_{1,2}^{(0)} \}$. Переменная y_{11} помещена в функцию $y_{1,2}^{(0)}$, но точно также она могла быть помещена вместо этого в функцию $y_{1,1}^{(0)}$

Далее снова возвращаемся к п.з) по которому получим множество $A_2^{(0)} = \{ y_2^{(0)} \}$.

По п.и) вычислим

$$y_2^{(0)} \cdot Y_1^{(0)} = x_1 (y_6 \vee y_8) \vee \bar{x}_1 x_2 (y_{10} \vee y_{13} \vee y_{16}),$$
$$y_2^{(0)} \cdot Y_2^{(0)} = \bar{x}_1 \bar{x}_2 y_4.$$

Нетрудно видеть, что $y_2^{(0)} \cdot Y_1^{(0)} \cdot Y_2^{(0)} = 0$. Следовательно, согласно п.к) множество $A_2^{(0)}$ не должно расширяться. Поскольку все функции рассмотрены, то можно переходить к п.м)

Согласно этому пункту вследствие увеличения суммарного числа функций в множествах $A_1^{(0)}$, $A_2^{(0)}$ переобозначим функции этих множеств следующим образом

$$A_1^{(0)} = \{ y_1^{(1)} = y_{1,1}^{(0)}, y_3^{(1)} = y_{1,2}^{(0)} \}$$
$$A_2^{(0)} = \{ y_2^{(1)} = y_2^{(0)} \}, \text{ т.е.}$$
$$y_1^{(1)} = y_1 \vee y_2 \vee y_7 \vee y_9 \vee y_{18}, \quad y_3^{(1)} = y_{11} \vee y_{14} \vee y_{17},$$
$$y_2^{(1)} = y_2 \vee y_4 \vee y_5 \vee y_6 \vee y_8 \vee y_{10} \vee y_{12} \vee y_{13} \vee y_{15} \vee y_{16}.$$

Переходя снова к п.ж) вычисляем функции

$$Y_1^{(1)} = x_1(y_6 \vee y_8 \vee y_{17}) \vee \bar{x}_1 x_2 y_{11}, \quad Y_3^{(1)} = \bar{x}_1 x_2 (y_3 \vee y_{16} \vee y_{20}),$$
$$Y_2^{(1)} = x_1 y_{11} \vee \bar{x}_1 \bar{x}_2 (y_4 \vee y_{111}) \vee \bar{x}_1 x_2 y_{14}. \qquad (5)$$

Если провести ещё один цикл вычислений, то можно убедиться, что число функций $y_i^{(2)}$ равно числу функций $y_i^{(1)}$. Функции (4) и (5) задают детерминированный конечный автомат, реализующий множество вв-последовательностей

Функции (4) и (5) могут быть преобразованы к каноническому виду соответственно в функции (6) и (7) подстановкой вместо каждой переменной у той переменной $y^{(i)}$, в которую эта y входит

$$z_1 = y_2^{(1)}, \quad z_2 = y_1^{(1)} \vee y_2^{(1)}. \qquad (6)$$

$$Y_1^{(1)} = x_1(y_2^{(1)} \vee y_3^{(1)}) \vee \bar{x}_1 x_2 y_1^{(1)}, \quad Y_3^{(1)} = \bar{x}_1 x_2 y_2^{(1)}.$$
$$Y_2^{(1)} = x_1 y_1^{(1)} \vee \bar{x}_1 \bar{x}_2 (y_2^{(1)} \vee y_3^{(1)}) \vee \bar{x}_1 x_2 y_3^{(1)}. \qquad (7)$$

По функциям (6) и (7) может быть построен граф переходов конечного автомата $M^{(1)}$, если каждой переменной $y_i^{(1)}$, $i = 1,2,3$ взаимнооднозначно сопоставить вершину графа переходов (внутреннее состояние этого автомата).

Покажем теперь, что алгоритм I действительно позволяет по заданному множеству вв-последовательностей $\mathcal{S}$ построить инициальный детерминированный конечный автомат, реализующий множество $\mathcal{S}$. Для этого докажем ряд утверждений.

УТВЕРЖДЕНИЕ I. Алгоритм I позволяет построить по заданному множеству $\mathcal{S}$ автомат $M^{(i)}$ функция переходов которого может быть задана выражением

$$f^{(i)}(u_e, y_j^{(i)}) = \left\{ y_k^{(i)} \mid u_e \cdot y_j^{(i)} \cdot Y_k^{(i)} \neq 0 \right\}, \qquad (8)$$

где в общем случае

$$u_e = u_e(x_1, \ldots, x_m), \quad y_j^{(i)} = y_j^{(i)}(y_1, \ldots, y_n),$$
$$Y_k^{(i)} = Y_k^{(i)}(x_1, \ldots, x_m, y_1, \ldots, y_n).$$

Доказательство. Утверждение справедливо по построению функций $y_j^{(i)}$ и $Y_i^{(i)}$ с помощью алгоритма I. Действительно каждую переменную $y_j^{(i)}$, $j = 1,2,3\ldots,z$ отождествим

с внутренним состоянием автомата $M^{(i)}$. Будем считать, что автомат $M^{(i)}$ находится в состоянии $y_j^{(i)}$, если $y_j^{(i)}=1$. Поскольку $y_j^{(i)}=y_a \vee y_b \vee \dots \vee y_t$, где $\{a,\dots,t\}\subseteq\{1,2,\dots,n\}$, то для того, чтобы $y_j^{(i)}=1$, необходимо по крайней мере одна переменная y_l =I. Будем считать, что автомат переходит из некоторого состояния $y_j^{(i)}$ в состояние $y_k^{(i)}$, если $y_j^{(i)}\cdot Y_k^{(i)}$=I. Поскольку $Y_k^{(i)}=u_a y_a \vee \dots \vee u_t y_t$, то $Y_k^{(i)}$ =I, если по крайней мере для одного $l\in\{a,\dots,t\}$ имеет место $u_l y_l$ =I. Следовательно, если u_l =I и y_l =I, то $y_j^{(i)}$ =I, $Y_k^{(i)}$ =I, т.е. автомат из состояния $y_j^{(i)}$ переходит в состояние $y_k^{(i)}$ при u_l =I

Если не только одна переменная y_l, входящая в функцию $y_j^{(i)}$ принимает единичное значение, то может быть и не одна функция $Y_k^{(i)}$ =I, т.е. $y_j^{(i)}$ будет переходить во все $y_k^{(i)}$ такие, что $u_l\cdot y_j^{(i)}\cdot Y_k^{(i)}=1$,

$$f^{(i)}(u_l, y_j^{(i)}) = \left\{ y_k^{(i)} \,\middle|\, u_l y_j^{(i)} Y_k^{(i)}=1 \right\}.$$

СЛЕДСТВИЕ 1. Функция $f^{(i)}(u^*, y_j^{(i)})$, где $u^*=u_a u_b \dots u_t \in P$ определяется рекурсивно следующим образом

$$f^{(i)}(\epsilon, y_1^{(i)}) = \{ y_1^{(i)} \}, \qquad (9)$$

$$f^{(i)}(u_1^* u, y_1^{(i)}) = \bigcup_{y_e^{(i)}\in f^{(i)}(u_1^*, y_1^{(i)})} f^{(i)}(u, y_e^{(i)}),$$

где $y_1^{(i)}$ – функция, в которую входит переменная y_1.

УТВЕРЖДЕНИЕ 2. Функция выходов автомата Мура $M^{(i)}$, построенного по алгоритму I может быть задана выражением

$$\varphi^{(i)}(y_j^{(i)}) = z_1^{\theta_1}, \dots, z_h^{\theta_h}, \dots, z_p^{\theta_p} \,\middle|\, \theta_h = \begin{cases} 0, & y_j^{(i)}\cdot z_h = 0 \\ 1, & y_j^{(i)}\cdot z_h \neq 0. \end{cases} \qquad (10)$$

Доказательство. Согласно алгоритму I каждая функция $y^{(\omega)}=y^{(\omega)}(y_1,\dots,y_n)$ получается как дизъюнкция всех тех y_l, для которых наборы значений $z_1 y_l, \dots, z_p y_l$ при y_l =I одинаковы. Функция $y^{(\omega)}$ получается как дизъюнкция некоторых конъюнкций (переменных), входящих в д.н.ф. одной и той же функции $y^{(\omega)}$ и, следовательно, сохраняет свойство функции $y^{(\omega)}$ о равенстве наборов значений $z_1 y_l, \dots, z_p y_l$ для всех

переменных y , входящих в $y^{(i)}$. Следовательно, значение функции $\varphi^{(i)}(y_j^{(i)})$, вычисляемой по выражению (10) однозначно, что и требовалось доказать.

СЛЕДСТВИЕ 2. $\varphi^{(i)}(u^*)$, где $u^* = u_0 \dots u_t \in \mathcal{P}$, определяется рекурсивно следующим образом

$$\varphi^{(i)}(\epsilon) = \{\varphi^{(i)}(y_1^{(i)})\},$$

$$\varphi^{(i)}(u^*) = \bigcup_{y_j^{(i)} \in f^{(i)}(u^*, y_1^{(i)})} \varphi^{(i)}(y_j^{(i)}) \qquad (11)$$

Таким образом был определен своими функциями переходов $f^{(i)}$ выходов $\varphi^{(i)}$ автомат $M^{(i)}$. Как следует из (8) этот автомат необязательно детерминированный. Нашей задачей является построение ИДКА по множеству $\mathcal{S}$. Следующие утверждения гарантируют, что алгоритм I позволяет построить такой автомат.

УТВЕРЖДЕНИЕ 3. Если $u^* \in \mathcal{P}(\mathcal{S})$, то существует $y_j^{(i)} \in f^{(i)}(u^*, y_1^{(i)})$ такое, что $\varphi^*(u^*) \vee \varphi^{(i)}(y_j^{(i)}) = \varphi^{(i)}(y_j^{(i)})$

Доказательство. Согласно определению функция $\varphi^*(u^*) = z_1^{\theta_1} \vee \dots \vee z_I^{\theta_I} \vee \dots \vee z_T^{\theta_T}$, где $\theta_I \in \{0,1\}$. Пусть $u^* = u_0 \dots u_t$ Тогда функция z_I содержит конъюнкцию y_t , если $\theta_I = 1$ и не содержит её, если $\theta_I = 0$. Одна из функций $y_j^{(i)} \in f^{(i)}(u^*, y_1^{(i)})$ обязательно содержит конъюнкцию y_t , т.к. все такие функции $y_j^{(i)}$ получены в результате разбиения на подфункции некоторой функции $y^{(i)}$, содержащей конъюнкцию y_t . Это означает согласно (10), что в функции $\varphi^{(i)}(y_j^{(i)}) = z_1^{\theta_1} \vee \dots \vee z_h^{\theta_h} \vee \dots \vee z_p^{\theta_p}$, где $y_j^{(i)}$ содержит конъюнкцию y_t , если $h = 1$, то $\theta_h = \theta_I$, что и доказывает утверждение.

УТВЕРЖДЕНИЕ 4. Если ℓ больше или равно длине самой длинной последовательности $u^* \in \mathcal{P}(\mathcal{S})$, то $f^{(i)}(u^*, y_1^{(i)}) = \{y_j^{(i)}\}$.

Доказательство. Ограничимся идеей доказательства. Причина того, что функция $f^{(i)}(u^*, y_1^{(i)})$ может быть неоднозначна состоит в том, что могут существовать две и более пары последовательностей $u_1^*, u_2^* \in \mathcal{P}$, которым соответствуют конъюнкции y_{t_1} и y_{t_2} , входящие в одну и ту же функцию $y_i^{(i)}$ и

при этом существует функция такая, что $u_1^* u$, $u_2^* u \in$ $\in \mathcal{P}$, $\varphi^*(u_1^* u) \neq \varphi^*(u_2^* u)$, причем может быть, что $u_3^* = u_1^*$ или $u_3^* = u_2^*$. Чтобы устранить указанную неоднозначность функции $f^{(s)}$ нужно разместить переменные y_{u_1}, y_{u_2} в различные подфункции $y^{(s+1)}$, что и делается на каждом шаге алгоритма I. Поскольку длина любой последовательности не превосходит s , то достаточно s шагов работы алгоритма, чтобы выполнить все разнесения переменных.

5. Построение автомата непосредственно по ОСПАП.

Алгоритм I требует для своей работы описания алгоритма функционирования автомата в виде множества вв-последовательностей $\mathcal{S}$. Такое описание может быть чрезмерно громоздким, т.к. требует перечисления всех допустимых вв-последовательностей. Кроме того, последовательности могут быть бесконечными и алгоритм I не содержит указаний, как быть в этом случае.

В то же время ОСПАП позволяет компактно описывать множество вв-последовательностей, в том числе бесконечных. Спрашивается, нельзя ли модифицировать алгоритм I таким образом, чтобы он позволял получать ИДКА без перехода от ОСПАП к множеству вв-последовательностей?

Алгоритм I, начиная с п.б, имеет дело с функциями $y_i^{(s)}$ и $z_j^{(s)}$. Поэтому для решения поставленного вопроса получим непосредственно по ОСПАП функции аналогичные этим и будем идти по пути модификации алгоритма I таким образом, чтобы его можно было применять к этим новым функциям и в результате получался ИДКА, реализующий множество вв-последовательностей, задаваемое ОСПАП. При модификации алгоритма I будем говорить, что он инвариантен к некоторому его изменению, если результат его работы остаётся прежним.

Введём ещё одно обозначение: если u_1^* , $u_2^* \in \mathcal{P}$ и $u_1^* = u_{a_1} u_{b_1} \dots u_{t_1}$, $u_2^* = u_{a_2} u_{b_2} \dots u_{s_2}$, $t \leq s$ то $u_1^* \odot u_2^* = u_{a_1 \& a_2}\, u_{b_1 \& b_2} \dots u_{t_1 \& t_2}\, u_{(t+1)_2} \dots u_{s_2}$ Будем считать, что $u_1^* \odot u_2^* = 0$, если хотя бы для одной пары

u_{i_1}, u_{i_2}, где $i \le t$ имеет место $u_{i_1} \& u_{i_2} = 0$.

Докажем следующие утверждения.

УТВЕРЖДЕНИЕ 5. Если для любых двух последовательностей $u_1^*, u_2^* \in \mathcal{P}(S)$ таких, что $u_1^* = u_{H1}^* u_{m1} u_{k1}$, $u_2^* = u_{H2}^* u_{m2} u_{k2}^*$, $u_{H1}^* \odot u_{H2}^* \neq 0$, $u_{m1} \& u_{m2} = 0$, $u_{H1}^* = u_{a1} u_{a2} \dots u_{e1}$, $u_{H2}^* = u_{a2} u_{b2} \dots u_{e2}$, $\varphi^*(u_{a1} u_{b1} \dots u_{i1}) = \varphi^*(u_{a2} u_{b2} \dots u_{i2})$ для всех $i = a, \dots, e$, то алгоритм I инвариантен к замене каждой пары переменных y_{i1}, y_{i2}, соответствующих функциям u_{i1}, u_{i2}, одной переменной y_i.

Доказательство. Пусть входные функции последовательностей имеют соответственно различные номера $a_1, b_1, \dots, e_1$ и $a_2, b_2, \dots, e_2$. Тогда этим номерам по алгоритму I соответствуют переменные $y_{a1}, y_{b1}, \dots, y_{e1}$, $y_{a2}, y_{b2}, \dots, y_{e2}$. Поскольку $\varphi^*(u_{a1} u_{b1} \dots u_{i1}) = \varphi^*(u_{a2} u_{b2} \dots u_{i2})$ для любого $i \in \{a, b, \dots, e\}$ то по п.д алгоритма I каждая пара переменных y_{i1} и y_{i2} войдёт в одну и ту же функцию $y_i^{(0)}$. По условию утверждения $u_{m1} \& u_{m2} = 0$. Поэтому при выполнении п.к. алгоритма I не будет существовать ни одной пары $Y_{n1}^{(0)}$, $Y_{n2}^{(0)}$ таких, что $y_e^{(0)} \cdot Y_{n1}^{(0)}$ и $y^{(0)} \cdot Y_{n2}^{(0)}$ содержат соответственно $u_{m1} \cdot y_{m1}$ и $u_{m2} \cdot y_{m2}$ такие, что $u_{m1} \& u_{m2} \neq 0$. Следовательно переменные y_{m1} и y_{m2} могут быть оставлены в одной и той же функции $y_m^{(1)}$. Переменные y_{e1} и y_{e2} могут быть оставлены в одной и той же функции $y_e^{(1)}$, т.к. $u_{e1} \cdot y_{e2}$ и $u_{e2} \cdot y_{e2}$ сходят в одну и ту же $Y_m^{(0)}$, y_{k1} и y_{k2} могут быть оставлены в одной и той же $y_k^{(1)}$, т.к. $u_{k1} \cdot y_{k1}$ и $u_{k2} \cdot y_{k2}$ входят в одну и ту же $Y_e^{(1)}$ и т.д. до y_{a1}, y_{a2}. Индуктивно, если y_{i1}, y_{i2} ($i = a, b, \dots, m$) входят в одну и ту же функцию $y_i^{(s-1)}$, то $u_{i1} y_{i1}$, $u_{i2} y_{i2}$ входят в одну и ту же функцию $Y_j^{(s-1)}$ и, следовательно, y_{i1}, y_{i2} могут быть оставлены в одной и той же функции $y_i^{(s)}$ для любого s и заменены функцией одной переменной y_i.

УТВЕРЖДЕНИЕ 6. Если входная последовательность $u^* = u_H^* u_{ce}^* u_t^* \in \mathcal{P}(S)$ где u_{ce}^* означает входную периодическую последовательность с периодом $u_c^* = u_m u_n \dots u_t$, повторенным ε раз, и $\varphi^*(u_H^* u_t^*) = \varphi^*(u_H^* u_{ce}^* u_t^*)$,

то алгоритм I инвариантен к замене каждого множества переменных y_{qi}, $i = 1,2,\ldots,\varepsilon$, соответствующих входной функции u_q в последовательности $u^* = u_н^* u_{m1} \ldots u_{q1} \ldots u_{t1} u_{m2} \ldots u_{q2} \ldots u_{t2} \ldots \ldots u_{m\varepsilon} \ldots u_{q\varepsilon} \ldots u_{t\varepsilon} u_к^*$, где $u_{qi} = u_q$ для всех i, одной переменной y_q.

Доказательство. Пусть входные функции последовательности $u_{c\varepsilon}^* = u_{m1} \ldots u_{q1} \ldots u_{t1} u_{m2} \ldots u_{q2} \ldots u_{t2} \ldots u_{m\varepsilon} \ldots u_{q\varepsilon} \ldots u_{t\varepsilon}$ имеют соответственно номера $m_1,\ldots, q_1,\ldots, t_1, m_2,\ldots, q_2,\ldots,t_2,\ldots, m_\varepsilon,\ldots, q_\varepsilon,\ldots, t_\varepsilon$. Тогда этим номерам по алгоритму I соответствуют переменные y_{qi}, где $q \in \{m,\ldots,t\}$, $i \in \{1,2,\ldots,\varepsilon\}$. Поскольку $\varphi^*(u_н^* u_к^*) = \varphi^*(u_н^* u_{c\varepsilon}^* u_н^*)$, то по п.д. алгоритма I множество всех переменных y_{qi} для каждого фиксированного $q \in \{m,\ldots,t\}$ и всех i войдёт в одну и ту же функцию $y_q^{(0)}$. Тогда все $u_{qi} y_{qi}$ входят в одну и ту же функцию $y_2^{(0)}$, поскольку y_{2i} для всех i входят в одну и ту же $y_2^{(0)}$. Следовательно, все y_{qi} могут быть оставлены в одной и той же $y_q^{(1)}$. Индуктивно, если y_{qi} входят в одну и ту же $y_q^{(s-1)}$, а y_{2i} в одну и ту же $y_2^{(s-1)}$, то $u_{qi} y_{qi}$ входят в одну и ту же $y_2^{(s-1)}$ и, следовательно, y_{qi} для всех i могут быть оставлены в одной и той же $y_q^{(s)}$ для любого s и заменены одной переменной y_q.

Прежде, чем показать возможность применения алгоритма I непосредственно к системе функций $y^{(0)}$, $z^{(0)}$, получаемой по ОСПАП, покажем как эта система получается.

Для получения системы функций производится разметка ОСПАП, которая состоит в простановке пометок на дугах ОСПАП, входящих в предикат, причём дугам входящим в один и тот же предикат ставится в соответствие одна и та же пометка. Перед остановом также ставится пометка. Пример разметки показан на рис.2 (пометки обозначены крестиками и нумеруются подряд). Каждой пометке i ставится во взаимнооднозначное соответствие переменная y_i (пометке на дуге, выходящей из стартёра, для удобства сопоставлена переменная y_1).

В ОСПАП могут быть следующие случаи:

а) Входящая дуга предиката Π_j является выходящей дугой предиката Π_i или функтора Φ_i, имеющего входящую дугу, являющуюся выходящей для предиката Π_i.

б) Входящая дуга останова O_j является выходящей дугой предиката Π_i или функтора, имеющего входящую дугу, являющуюся выходящей для предиката Π_i.

в) Входящая дуга предиката Π_j является выходящей для стартёра H или для функтора, имеющего входящую дугу, являющуюся выходящей для стартёра.

В первых двух случаях будем говорить, что предикат Π_i предшествует пометке y_j дуги, входящей в предикат Π_j или останов O_j, а в третьем случае, что стартёр предшествует пометке y_j предиката Π_j.

Пусть пометке $y_j, j>1$, предшествуют предикаты $\Pi_{i1},\dots,\Pi_{id}$. Для каждой y_j получим функцию

$$Y_j = \bigvee_{\substack{\text{по всем}\\ k=1,\dots,d}} f_{ik}\, y_{ik}$$

где f_{ik} – функция предиката Π_{ik}, соответствующая дуге, помеченной переменной y_j.

Каждая функция z_I, $I=1,2,\dots,\rho$ получается как дизъюнкция всех тех переменных y_j, которые соответствуют дугам, выходящим из функторов с выражениями $z_I := I$.

Доказательство того, что функции Y_j, z_I, полученные непосредственно по ОСПАП описанным образом, задают тот же алгоритм функционирования, что и ОСПАП, опустим.

Докажем утверждение о том, что эти функции могут быть использованы для построения ИДКА без получения по ОСПАП множества вв-последовательностей.

УТВЕРЖДЕНИЕ 7. Алгоритм I инвариантен к замене функций Y, получаемых в п.а, б и функций z, получаемых в п.в. по множеству вв-последовательностей, задающих тот же алгоритм функционирования, что и ОСПАП, системой функций получаемой непосредственно по ОСПАП.

Доказательство. Пусть некоторая переменная y_i соответ-

ствует входящей дуге предиката $П_i$ с s выходящими дугами, которым соответствуют функции u_a, $a = m_1, \dots, m_s$. К предикату $П_i$ от стартёра ведёт один или несколько путей, которым соответствует входная последовательность $u_н^*$ (получение входных последовательностей для простых ОСПАП описано выше). От предиката $П_i$ к нему же может идти один или несколько путей, которым соответствуют последовательности $u_a u_c^*$ и от предиката $П_i$ остановам ведут один или несколько путей, которым соответствуют последовательности $u_a u_k^*$. Таким образом входные последовательности, которые задает ОСПАП и которые соответствуют путям, проходящим через предикат $П_i$ могут быть последовательностями $u_н^* u_a u_k^*$, $u_н^* (u_a u_c^*)_\varepsilon$, $u_н^* (u_a u_c^*) u_k^*$.

Если строить ИДКА по алгоритму I по вв-последовательностям, реализуемым данной ОСПАП, то входным функциям u_a следует сопоставить переменные y_a. Но каждая пара последовательностей $u_н^* u_{m_i} u_k^*$, $u_н^* u_{m_j} u_k^*$ удовлетворяет условию утверждения 5, а последовательность $u_н^* (u_a u_c^*)_\varepsilon u_k^*$ удовлетворяет условию утверждения 6. Кроме того $u_н^* (u_a u_c^*) u_k^* = u_н u_a u_c^* (u_a u_c^*)_{\varepsilon-1} u_k^*$, т.е последовательность $u_н^* (u_a u_c^*)_\varepsilon u_k^*$ можно представить как $u_н u_a u_{k1}^*$, где $u_{k1}^* = u_c^* (u_a u_c^*)_{\varepsilon-1} u_k^*$. Следовательно, согласно утверждениям 5 и 6 переменные y_a можно заменить одной переменной y_i. Очевидно, что эту переменную можно считать одинаковой с переменной, соответствующей пометке дуги, входящей в предикат $П_i$. Этим завершается доказательство первой части утверждения относительно возможности замены функций Y, получаемых в п.а,б алгоритма I, функциями Y, получаемыми непосредственно по ОСПАП.

Доказательство второй части относительно возможности замены функций выходов, получаемых в п.в алгоритма I, на функции выхода, получаемые непосредственно по ОСПАП, после доказательства первой части очевидно.

Утверждение 7 позволяет применять алгоритм I, начиная с п.г к функциям Y и z, полученным непосредственно по ОСПАП.

6. ЗАКЛЮЧЕНИЕ.

Алгоритм, изложенный в статье, рассчитан только на простые ОСПАП, а получающийся в результате работы алгоритма автомат является автоматом Мура. В описанном переходе от ОСПАП к функциям происходит некоторое доопределение этого автомата, которое нельзя считать наилучшим в смысле минимизации числа внутренних переменных. Однако подход, положенный в основу алгоритма, может быть с тем же успехом использован для ОСПАП другого типа.

Доопределение может быть сделано более эффективным, а вместо автомата Мура может получаться автомат Мили. Изменения алгоритма, которые требуется для этого провести, довольно незначительны.

Кроме уже отмеченных достоинств подхода можно также отметить возможность регулирования трудоёмкости построения ИДКА за счёт более "компактного" описания алгоритма функционирования на языке ОСПАП и минимизацию числа внутренних переменных в результирующих функциях по сравнению с исходными.

ЛИТЕРАТУРА

I. В.Ф.Дьяченко, В.Г.Лазарев, Г.Г.Саввин. Управление на сетях связи. Изд. "Наука", М., 1967.

2. T.C.Bartee, I.L.Lebow and I.S.Reed. Theory and Design of Digital Systems, New-York, 1962, McGrow Hill.

3. М.А.Гаврилов, В.В.Девятков. Особенности логического проектирования в системе ДАСП. Сб. "Абстрактная и структурная теория релейных устройств", Изд. "Наука", М.,1975.

4. В.В.Девятков, А.Б.Чичковский. УСЛОВИЕ - язык для описания условий работы дискретных управляющих устройств. Сб. "Абстрактная и структурная теория релейных устройств", Изд. "Наука", М., 1975.

5. М.А.Гаврилов, В.В.Девятков, А.Б.Чичковский. Язык операторных схем параллельных алгоритмов с памятью (язык ОСПАП). Сб. "Абстрактная и структурная теория релейных устройств", Изд. "Наука", М., 1975.

6. A.Nerode. Linear automation transtormations, Proc. Amer.Math. Soc., vol.9, 1958.

7. В.М.Глушков. Синтез цифровых автоматов. Физматгиз, М.,1962.

8. А.А.Таль. Анкетный язык и абстрактный синтез минимальных последовательностных машин. Автоматика и телемеханика, № 6, 1964.

9. S.Ginsburg. Sintnesis of minimal-state machihes. IRE Trans. Electron. Comput, vol. Ec-8, pp.441-449, Dec.1959.

IO. A.W.Biermann, Y.A.Feldman. On the Syntnesis of finite-state machines from samples of their behavior. IEEE Transactions on Computers, June, 1972.

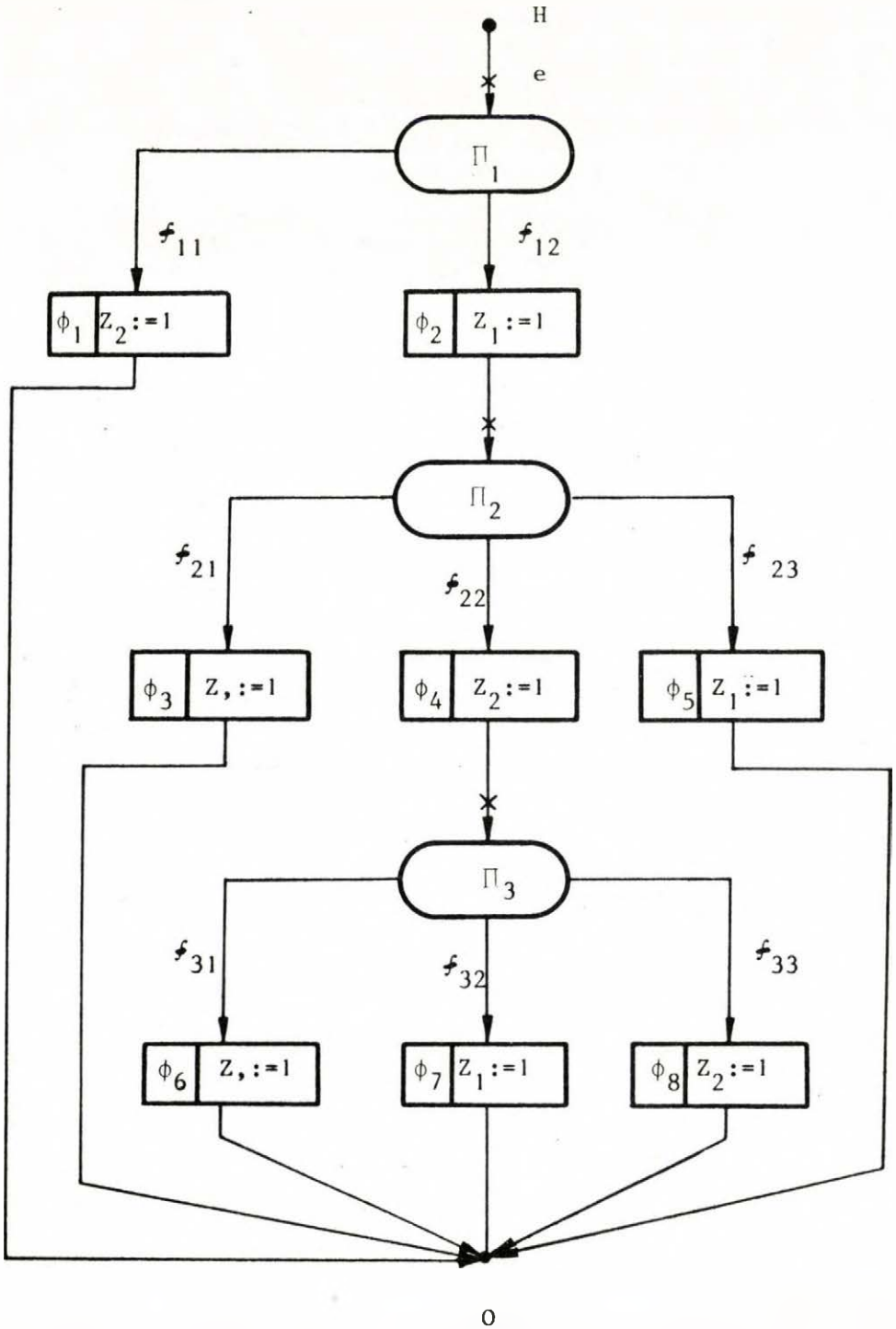

н
е
П1
f11
f12
φ1 Z2:=1
φ2 Z1:=1
П2
f21
f22
f23
φ3 Z,:=1
φ4 Z2:=1
φ5 Z1:=1
П3
f31
f32
f33
φ6 Z,:=1
φ7 Z1:=1
φ8 Z2:=1
о

Рис. 2.

Закревский А.Д.

ОПТИМИЗАЦИОННЫЕ ПРЕОБРАЗОВАНИЯ СЕКВЕНЦИАЛЬНЫХ АВТОМАТОВ

При проектировании дискретных устройств с большим числом переменных традиционные функциональные модели, связанные с рассмотрением множества входных, выходных и внутренних состояний, оказываются явно непригодными, поскольку эти множества могут содержать много элементов, перечислить которые практически невозможно. В этом случае может быть рекомендована модель, опирающаяся непосредственно на некоторое множество булевых переменных и называемая секвенциальным автоматом. Такая модель рассматривалась, например, в работах [1-3]. Настоящая статья посвящена уточнению этой модели и некоторым оптимизационным задачам на ней.

Секвенциальный автомат является средством описания функционирования дискретных технических систем, состоящих из входных датчиков и исполнительных механизмов (с цепями обратной связи), состояния которых можно представить соответственно значениями булевых переменных $x_1, x_2, \ldots, x_n$ и $w_1, w_2, \ldots, w_m$, образующих множества X и W. Такая система рассматривается как объект управления, секвенциальный автомат - как алгоритм управления, задающий причинно-следственную связь между событиями, происходящими в системе. События могут быть элементарными и сложными. В первом случае они задаются отдельными комбинациями значений всех переменных из множества $X \cup W$, т.е. соответствуют полным состояниям автомата. Во втором случае они представляются булевыми функциями некоторых переменных из множества $X \cup W$. Именно рассмотрение сложных событий и задание причинно-следственных связей между ними обеспечивают определенные преимущества секвенциального автомата перед другими функциональными моделями дискретных устройств.

Секвенциальный автомат - это некоторая система S, представляющая собой неупорядоченное множество выражений типа $f_i \vdash k_i$, называемых секвенциями. Через f_i обозначается булева функция переменных, выбираемых из множества $X \cup W$, а через k_i - элементарная конъюнкция переменных, выбираемых из множества W. При этом считается, что функция f_i не равна тождественно нулю, а ранг элементарной конъюнкции k_i (число символов в ней) не менее 1.

Секвенция $f_i \vdash k_i$ задает непосредственное следование двух событий: событие f_i должно сопровождаться событием k_i. Это означает, что если в некоторый момент времени переменные $x_1, x_2, \ldots, x_n, w_1, w_2, \ldots, w_m$ принимают значения, обращающие функцию f_i в единицу, то непосредственно вслед за этим должна принять значение 1 элементарная конъюнкция k_i и, следовательно, однозначно определяются значения всех переменных, входящих в состав этой конъюнкции. Каждую из секвенций можно рассматривать как определенное требование, предъявляемое к способу функционирования технической системы со стороны проектировщика. Система S в целом задает совокупность таких требований.

Система S должна быть непротиворечива, для чего необходимо, чтобы для любых ее секвенций s_i и s_j выполнялось условие $(k_i \wedge k_j = 0) \Rightarrow (f_i \wedge f_j = 0)$, где $\Rightarrow$ - символ формальной импликации. Ниже мы будем предполагать, если не оговаривается противное, что это условие выполняется всегда.

К системе S могут предъявляться и некоторые дополнительные требования, связанные, например, с устранением опасных состязаний, возникающих при одновременном изменении значений нескольких переменных. В данной статье они рассматриваться не будут.

Если требование, представленное секвенцией s_i, является расширением требования, представленного секвенцией s_j (т.е. последнее автоматически удовлетворяется при удовлетворении первого), будем говорить, что секвенция s_i реализует секвенцию s_j. Отношение реализации транзитивно: если s_i реализует s_j, и s_j реализует s_k, то s_i реализует s_k.

Теорема 1. Секвенция s_i реализует секвенцию s_j, если и только если $f_j \Rightarrow f_i$ и $k_i \Rightarrow k_j$.

Например, секвенция $x_1 \vee \omega_1(\bar{x}_1 \vee \omega_2) \vdash \omega_1\omega_3\omega_4$ реализует секвенцию $x_1 \vee \omega_1(\bar{x}_1 \vee x_2\,\omega_2) \vdash \omega_3\omega_4$, а последняя реализует секвенцию $\omega_1(\bar{x}_1 \vee x_2\,\omega_2) \vdash \omega_3$.

Две секвенции s_i и s_j назовем эквивалентными, если каждая из них реализует другую. Очевидно, что при этом $f_i = f_j$ и $k_i = k_j$, т.е. $s_i = s_j$.

Назовем элементарной такую секвенцию s_i, у которой функция f_i принимает значение 1 ровно на одном наборе значений всех переменных множества $X \cup W$ (такую функцию можно представить полной элементарной конъюнкцией, содержащей, как известно, символы всех переменных), а элементарная конъюнкция k_i имеет ранг 1, т.е. содержит символ только одной переменной.

Теорема 2. Если элементарная секвенция s_i реализует секвенцию s_j, то последняя эквивалентна ей и также является элементарной.

Перейдем к определению отношений между секвенциальными автоматами, рассматривая последние как неупорядоченные множества без повторений и обозначая верхним индексом ^{э} элементарные секвенциальные автоматы, как мы будем называть автоматы, представляемые системами элементарных секвенций.

Элементарные секвенциальные автоматы $S_i^э$ и $S_j^э$ эквивалентны, если и только если $S_i^э = S_j^э$ (т.е. соответствующие множества равны). Автомат $S_i^э$ реализует автомат $S_j^э$, если и только если $S_j^э \subseteq S_i^э$. Автомат S_i реализует автомат $S_j^э$, если и только если каждый элемент множества $S_j^э$ реализуется некоторым элементом множества S_i. Автомат $S_j^э$ реализует автомат S_i, если все элементарные секвенции, реализуемые элементами множества S_i, принадлежат множеству $S_j^э$. Автоматы S_i и $S_j^э$ эквивалентны, если и только если S_i реализует $S_j^э$ и $S_j^э$ реализует S_i.

Отсюда следует, что для каждого секвенционального автомата S_i существует единственный эквивалентный ему элементарный секвенциальный автомат $S_i^э$. Автоматы S_i и S_j эквивалентны, если и только если $S_i^э = S_j^э$. Автомат S_i реализует автомат S_j, если и только если $S_j^э \subseteq S_i^э$.

Элементарный секвенциальный автомат можно рассматривать как каноническую форму произвольного секвенционального автомата. Эта форма может быть получена путем объединения множеств элементарных секвенций, получаемых в результате разложения отдельных секвенций $f_i \vdash k_i$ множества S по правилам "секвенция $f_{i1} \vee f_{i2} \vdash k_i$ эквивалентна паре секвенций $(f_{i1} \vdash k_i, f_{i2} \vdash k_i)$" и "секвенция $f_i \vdash k_{i1} \wedge k_{i2}$ эквивалентна паре секвенций $(f_i \vdash k_{i1}, f_i \vdash k_{i2})$", позволяющим последовательно расщеплять секвенции преобразуемой системы, пока они не станут все элементарными. Множество элементарных секвенций можно получить и быстрее, если иметь в виду, что они образуются всевозможными парами, первый элемент которых выбирается из множества членов совершенной дизъюнктивной нормальной формы функции f_i, а второй - среди сомножителей элементарной конъюнкции k_i.

Элементарный секвенциальный автомат оказывается удобной формой для теоретических построений, для определения смысла вводимых отношений между секвенциальными автоматами. Однако для практических применений, связанных с рассмотрением реальных технических систем, эта форма слишком громоздка. В связи с этим представляет интерес следующие две формы секвенциальных автоматов.

Одна из них задается множеством секвенций, левыми частями которых служат полные элементарные конъюнкции переменных из множества $X \cup W$, причем все эти конъюнкции различны. Это множество назовем точечным секвенциальным автоматом (все требования задаются на отдельных точках булева пространства переменных множества $X \cup W$) и будем обозначать его с помощью верхнего индекса ^{т}.

Другая форма задается множеством секвенций с различными правыми частями, которыми служат символы переменных из множества W, взятые со знаком отрицания или без него. Это множество назовем функциональным секвенциальным автоматом (каждая переменная ω_i определяется в нем как частичная функция, задаваемая парой полностью определенных булевых функций) и будем обозначать его с помощью верхнего индекса ^{ф}

Точечный секвенциальный автомат $S^{т}$ может быть получен из элементарного автомата $S^{э}$ путем разбиения последнего на группы элементарных секвенций с общей левой частью и замены каждой из этих групп одной секвенцией с той же левой частью. Правая часть данной секвенции определяется как конъюнкция правых частей секвенций соответствующей группы. Автомат $S^{т}$ можно получить также путем разложения секвенций автомата S посредством дизъюнктивного расщепления их левых частей.

Функциональный секвенциальный автомат $S^{ф}$ также может быть получен из элементарного автомата $S^{э}$, путем группирования элементарных секвенций по правым частям и последующей замены групп секвенциями, левая часть которой представляется дизъюнкцией левых частей секвенций соответствующей группы. Нетрудно получить автомат $S^{ф}$ и из автомата S, разлагая секвенции последнего путем конъюнктивного расщепления правых частей.

Теорема 3. Секвенциальные автоматы $S^{т}$ и $S^{ф}$ являются каноническими.

Действительно, обозначая через $S_i^{т}$ и $S_i^{ф}$ точечный и функциональный автоматы, эквивалентные некоторому секвенциальному автомату S, можно утверждать, что если два автомата S_i и S_j эквивалентны, то $S_i^{т} = S_j^{т}$ и $S_i^{ф} = S_j^{ф}$, если же S_i и S_j не эквивалентны, то $S_i^{т} \neq S_j^{т}$ и $S_i^{ф} \neq S_j^{ф}$.

Рассматривая некоторый конкретный секвенциальный автомат S, разумно задаться вопросом о его полноте. Очевидно, что последняя будет обеспечена, если $\vee f_i = 1$ и каждая элементарная конъюнкция k_i содержит символы всех переменных из множества W. При рассмотрении более общего случая требуется уточнить интерпретацию множества секвенций S. Ниже предлагаются два способа такого уточнения.

Назовем инерционным секвенциальным автоматом систему секвенций, интерпретируемую следующим образом: если во всех элементарных конъюнкциях k_i, соответствующих функциям f_i, принимающим в текущий момент времени значение 1, отсутствует символ некоторой переменной ω_j, то считается, что эта переменная сохраняет свое значение. Очевидно, что функциональные свойства такого автомата определены тем самым полностью.

Преобразуем инерционный секвенциальный автомат к форме $S^{ф}$, представив ее в следующем виде:

$$f_1 \vdash \omega_1 , \quad f_1' \vdash \bar{\omega}_1 ,$$
$$f_2 \vdash \omega_2 , \quad f_2' \vdash \bar{\omega}_2 ,$$
$$\cdots\cdots\cdots\cdots \quad \cdots\cdots\cdots\cdots$$
$$f_m \vdash f_m , \quad f_m' \vdash \bar{\omega}_m$$

(пару функций f_i и f_i' удобно рассматривать как функции переключения значений переменной ω_i).

Теорема 4. Все автоматы, полученные из инерционного автомата $S^{ф}$ заменой функций f_i и f_i' соответственно на любые функции φ_i и φ_i', удовлетворяющие условиям

$$\bar{\omega}_i f_i \Rightarrow \varphi_i \Rightarrow \bar{\omega}_i f_i \vee \omega_i \bar{f}_i'$$
$$\omega_i f_i' \Rightarrow \varphi_i' \Rightarrow \omega_i f_i' \vee \bar{\omega}_i \bar{f}_i ,$$

взаимно эквивалентны.

Конкретный выбор может подчиняться задаче реализации системы $S^{ф}$ некоторой комбинационной логической схемой, на выходных полюсах которой представляется значение функций φ_1, φ_1', φ_2, φ_2', ..., φ_m, φ_m', управляющих переключением тех элементов технической системы, состояния которых задаются значениями переменных ω_1, ω_2, ..., ω_m (если φ_i и φ_i' одновременно принимают значение 0, состояние соответствующего элемента не изменяется).

В частности, при выборе конкретных функций φ_i и φ_i' можно стремиться к уменьшению числа существенных переменных в этих функциях.

Пусть функция f представлена формулой ϕ, в которой, в частности, фигурирует символ переменной x. Подставляя всюду в формуле вместо этого символа

константу 1, мы получим формулу, которую обозначим через $\Phi : x$ и которая будет представлять функцию, обозначаемую через $f : x$. Подстановка вместо x значения 0 приводит соответственно к формуле $\Phi : \bar{x}$ и $f : \bar{x}$.

Теорема 5. Если некоторые функции f_i и f_i' инерционного секвенциального функционального автомата S^{Φ} удовлетворяют условию $(f_i : \bar{w}_i) \wedge (f_i' : w) = 0$, то замена их соответственно на $f_i : \bar{w}_i$ и $f_i' : w$ приводит к получению автомата, эквивалентного исходному.

Частичным секвенциальным автоматом назовем систему S, интерпретируемую следующим образом: если во всех элементарных конъюнкциях k_i, соответствующих функциям f_i, принимающим в текущий момент времени значение 1, отсутствует символ некоторой переменной w_i, то последующее значение этой переменной считается неопределенным. Какое бы значение не приняла эта переменная, представленную системой S связь событий будем считать реализованной. В соответствии с этой интерпретацией будем считать, что система S задает функциональные свойства рассматриваемой технической системы (объекта управления) лишь частично, и их можно произвольно доопределять при построении устройства, реализующего алгоритм управления.

При технической реализации инерционного секвенциального автомата целесообразно сначала минимизировать его, то есть найти минимальный (с минимальным числом секвенций) среди всех автоматов, эквивалентных исходному. Если же автомат частичный, поиск решения ведется среди секвенциальных автоматов, реализующих его.

Обозначим через S^{O} ортогональный секвенциальный автомат, эквивалентный исходному автомату S и получаемый из S^{T} путем слияния (взятия дизъюнкции левых частей) секвенций с одинаковыми правыми частями. Все функции f_i^{o}, служащие левыми частями секвенций автомата S^{O}, оказываются взаимно ортогональными, т.е. конъюнкция любых двух из них тождественно равна нулю. Автомат S^{O} можно получить и непосредственно из исходного автомата S путем его ортогонализации, при которой строится булево пространство значений функций f_i и удаляются его пустые элементы. Результат такого построения однозначен - ортогональный автомат каноничен.

Совокупность правых частей секвенций автомата S^{O} удобно изобразить в виде троичной матрицы K^{O}, способ построения которой можно понять из рассмотрения следующего примера:

	w_1	w_2	w_3	w_4	w_5
f_1^{o}	0	1	-	1	-
f_i^{o}	1	-	1	-	1
f_3^{o}	-	1	0	0	-
f_4^{o}	-	1	0	0	1
f_5^{o}	0	1	0	0	-
f_6^{o}	0	-	-	1	-
f_7^{o}	1	-	1	0	1

Первая строка матрицы представляет элементарную конъюнкцию $w_1 w_2 w_4$ (из секвенции $f_i^{o} \longmapsto \bar{w}_1\ w_2\ w_4$), вторая - $w_1 w_3 w_5$ и т.д.

Минор матрицы K^{O}, все строки которого равны и содержат лишь нули и единицы, назовем правильным. Совокупность правильных миноров матрицы K^{O} назовем ее минорным покрытием, если каждый элемент этой матрицы, имеющий значение 0 или 1, принадлежит по крайней мере одному из миноров. Покрытие, составленное из минимального числа миноров, назовем кратчайшим.

Теорема 6. Минимизация секвенциального автомата в классе эквивалентных сводится к нахождению кратчайшего минорного покрытия матрицы K^{O}.

Например, можно указать на следующее кратчайшее минорное покрытие рассмотренной троичной матрицы: (1,5) (1,2) (минор образуемый пересечением строк с номерами 1,5 и столбцов с номерами 1,2), (3,4,5) (2,3,4), (1,6) (1,4), (2,7)

(1,3,5) и (4,7) (4,5). Представляя каждый из этих пяти миноров соответствующей секвенцией, мы получим секвенциальный автомат, эквивалентный исходному и являющийся при этом минимальным:

$$\begin{array}{c} \\ f_1' \\ f_2' \\ f_3' \\ f_4' \\ f_5' \end{array} \begin{array}{c} \begin{array}{ccccc} w_1 & w_2 & w_3 & w_4 & w_5 \end{array} \\ \begin{bmatrix} 0 & 1 & - & - & - \\ - & 1 & 0 & 0 & - \\ 0 & - & - & 1 & - \\ 1 & - & 1 & - & 1 \\ - & - & - & 0 & 1 \end{bmatrix} \end{array}$$

Функции f_i' определяются при этом как дизъюнкции левых частей секвенций, представленных строками соответствующих миноров: $f_i' = f_1^0 \vee f_5^0$, $f_2' = f_3^0 \vee f_4^0 \vee f_5^0$ и т.д.

Будем говорить, что троичная матрица K_i^0 является доопределением матрицы K_j^0, если она может быть получена из последней сменой значений некоторых элементов с "–" на 0 или 1.

Теорема 7. Минимизация секвенциального автомата в классе реализующих сводится к нахождению кратчайшего среди минорных покрытий всевозможных доопределений матрицы K^0.

Эта комбинаторная задача довольно сложна. Процесс ее решения можно существенно упростить, если отказаться от гарантий оптимальности получаемых решений и ограничиться рассмотрением строчных миноров (в образовании которых участвуют все столбцы матрицы). В этом случае задача сводится к разбиению матрицы K^0 на минимальное число строчных миноров, составленных из взаимно неортогональных строк. Очевидно, что задачу можно сформулировать как задачу о нахождении минимальной правильной раскраски неориентированного графа, вершины которого поставлены во взаимно-однозначное соответствие со строками матрицы K^0, а ребра отмечают пары взаимно-ортогональных строк.

Применение данного метода к приведенной выше конкретной матрице K^0 приводит к ее разбиению на три строчных минора, образованных группами строк (1,6), (2,7), (3,4,5) и получению следующей матрицы, представляющей один из минимальных автоматов, реализующих исходный:

$$\begin{array}{c} \\ f_1'' \\ f_2'' \\ f_3'' \end{array} \begin{array}{c} \begin{array}{ccccc} w_1 & w_2 & w_3 & w_4 & w_5 \end{array} \\ \begin{bmatrix} 0 & 1 & - & 1 & - \\ 1 & - & 1 & 0 & - \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Функции f_i'' определяются при этом следующим образом: $f_1'' = f_1^0 \vee f_6^0$, $f_2'' = f_2^0 \vee f_7^0$ и $f_3'' = f_3^0 \vee f_4^0 \vee f_5^0$.

Литература

1. Григорьев В. С., Закревский А. Д., Перчук В.Л. Секвенциальная модель дискретного автомата. – В сб. "Вычислительная техника в машиностроении". Минск, ИТК АН БССР, 1972, март.
2. Захаров В. Н. Секвенциальное описание управляющих автоматов. "Известия АН СССР. Техническая кибернетика", 1972, № 2.
3. Захаров В. Н. Автоматы с распределенной памятью. М., "Энергия", 1975.

НОВЫЕ НАПРАВЛЕНИЯ В ТЕОРИИ АВТОМАТОВ С ПЕРЕМЕННОЙ СТРУКТУРОЙ

З. МИАДОВИЧ

Познаньский Политехнический Институт

В этой статье будут реферированы работы, связанные с новыми направлениями в теории автоматов с переменной структурой, которые выполнены в Познаньском Политехническом Институте в 1974-1975 гг. Речь идет, именно, о теории периодических сумм конечных автоматов, теории расширений конечных автоматов и других.

В работе - **"On the periodic equivalents of finite automata"** - представлены результаты, относящиеся к (τ, t) - эквивалентности конечных автоматов. Представлены также алгоритм нахождения (τ, T) - эквивалентов конечных автоматов.

В работе - **"On the periodic sum and extensions of finite automata"** даны результаты о периодических суммах, связанных с изоморфизмами конечных автоматов и расширением конечных автоматов.

Как периодическая сумма конечных автоматов так и расширение конечного автомата есть автомат с переменной структурой. Рассмотрены также проблемы связности и алгебрагические свойства таких автоматов.

Некоторые свойства расширений, независимых от состояний автоматов, были найдены в работе - **"On the extensions of state independent automata"**.

В наших работах автомат - это тройка (S,Σ, M), где S - конечные, непустое множество состояний, Σ - конечное, непустое множество входов, $M : S\times\Sigma \to S$ - /прямое произведение/ функция переходов.

Пусть T будет положительным числом. Точно периодический авто-

мат W /дальше просто - периодический автомат/ - определяется тройкой (S^+, Σ, M^+), где S^+ есть последовательность $S_o^+, S_1^+, \ldots, S_{T-1}^+$ конечных, непустых множеств состояний, Σ - конечное, непустое множество входов, M^+ есть последовательность $M_o^+, M_1^+, \ldots, M_{T-1}^+$ функций переходов, где $M_t^+: S_t^+ x \Sigma \to S_{t+1 (\mathrm{mod}\ T)}^+$ а $t = 0,1,\ldots,T-1$. Число T будем называть периодом W.

Пусть q будет натуральным числом. Пусть $A^o = (S^o, \Sigma, M^o)$ будет автоматом и пусть $g^o, g^1, \ldots g^{q-1}$ соответственно изоморфизмы из A^o на A^o, $A^1 = (S^1, \Sigma, M^1), \ldots, A^{q-1} = (S^{q-1}, \Sigma, M^{q-1})$

Расширением A^o связанным с изоморфизмами $g^o, g^1 \ldots g^{q-1}$ есть периодический автомат $W = (S^+, \Sigma, M^+)$ с периодом q где S^+ есть последовательность $S^o, S^1, \ldots, S^{q-1}$, и M^+ есть последовательность $M_o^+, M_1^+, \ldots M_{q-1}^+$ и

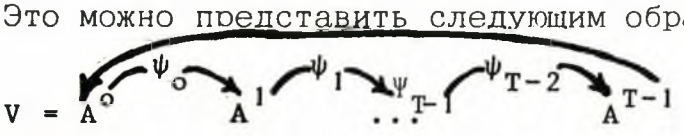
$$M_i^+ (s, \sigma) = g^{i-1 (\mathrm{mod}\ q)} (g^i)^{-1} M^\nu (s, \sigma)$$

где $s \in S^i$, $\sigma \in \Sigma$, а $i \in \{0,1,\ldots, q-1\}$

Периодический автомат W приводим к автомату A тогда и только тогда, если W является расширением A.

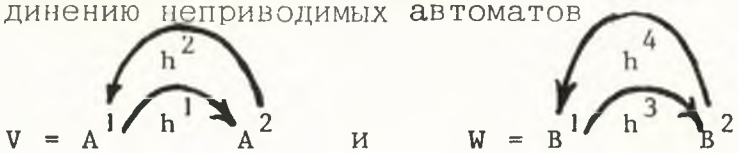
Пусть $A = (S^o, \Sigma, M^o)$, $A^1 = (S^1, \Sigma, M^1), \ldots, A^{T-1} = (S^{T-1}, \Sigma, M^{T-1})$ будут автоматами. Пусть $\psi_o: S^o \to S^1$, $\psi_1: S^1 \to S^2, \ldots, \psi_{T-1}: S^{T-1} \to S^o$ будут функциями, имеющие взаимно однозначное изображение. Периодическая сумма автоматов $A^o, A^1, \ldots, A^{T-1}$ связанная с функциями $\psi_o, \psi_i, \ldots, \psi_{T-1}$ есть периодический автомат $V = (S^+, \Sigma, M^+)$ с периодом T, где S^+ есть последовательность $S^o, S^1, \ldots S^{T-1}$, а M^+ есть последовательность $M_o^+, M_1^+, \ldots, M_{T-1}^+$, где для всех $s \in S^t$, $\sigma \in \Sigma$, а $t = 0,1,\ldots, T-1$ имеем $M_t^+(s, \sigma) = \psi_t M^t (s, \sigma)$

Это можно представить следующим образом



Некоторые результаты:

1. Пусть A^1, A^2, B^1, B^2, будут взаимно изоморфными автоматами, пусть $h^1 \in Iz\ (A^1 \to A^2)$ (принадлежит к множеству всех изоморфизмов из автомата A^1 на автомат A^2), $h^2 \in Iz\ (A^2 \to A^1)$, $h^3 \in Iz\ (B^1 \to B^2)$ и $h^4 \in Iz\ (B^2 \to B^1)$. Тогда автомат, равный объединению неприводимых автоматов



есть приводимый тогда и только тогда если существуют $\Phi \in Iz\ (A^1 \to B^1)$ и $\Phi^2 \in Iz\ (B^2 \to A^2)$ такие, что

$$h^2 f^2 h^3 f^{-1} = (f^1)^{-1}\ h^4 (f^2)^{-1}\ h^1 = id.$$

2. Пусть α будет множеством связанных неприводимых периодических автоматов с тем же периодом T. Объединение всех автоматов из α приводимое к обычному автомату, равному объединению связанных подавтоматов, для которых максимальны периодические представления имеют периоды D_1, D_2,...,D_n тогда и только тогда, если существует разбиение π на множестве α такое, что блоки разбиения π составлены с периодом (T-1), и которые адаптивны, взаимно изоморфные автоматы при чем мощность блоков соответственно равна наибольшим общим делителям (T,D_1), (T,D_2),..., (T,D_n).

3. Пусть W будет периодическим автоматом с периодом T а W^* /со звездочкой/ будет закрепленным аналогом автомата W. Если W^* есть независим от состояний, тогда все закрепленные компоненты W^*_o, W^*_1,..., W^*_{T-1} автомата W будут независимыми от состояний.

4. Пусть W будет расширением независимого от состояний автомата. Тогда закрепленный аналог W^* автомата W есть независимый от состояний.

Условия, при выполнении которых периодический автомат является

расширением некоторого конечного автомата, были даны в работе "On the reducibility of periodic automata".

В этой работе определены, между прочим, условия приводимости прямого произведения неприводимых периодических автоматов. Эти условия будут в точном соотношении с проблемами параллелной декомпозиций автоматов. Если данный периодический автомат есть приводимый, тогда существует параллельная декомпозиция такая, что один из компонентов есть автономный автомат.

Следующий вопрос: как найти для неприводимого периодического автомата $V^1 = (S, \Sigma, M)$ такой другой периодический автомат $V^2 = (U, \Sigma, N)$ что прямое произведение $V^1 \times V^2$ связанное и приводимое, был разрешен в цитированной работе.

Список цитированной литературы:

1. T. Gajewski: On the periodic equivalents of finite automata, IEEE Transactions on Computers, C-24, Oct.75, 991-994.

2. J.W. Gryzmala-Busse: On the periodic sum and extensions of finite automata. 3rd Symp. Math. Found. Computer Sci., Jadwisin, June 17-22, 1974. Lecture Notes in Computer Science. Vol. 28, 46-52. Springer Verlag, Berlin-Heidelberg-New York, 1975.

3. J.W. Gryzmala-Busse: On the estensions of state independent automata. Symp. 'Diskrete Mathematik und Anwendungen in der mathematischen Kybernetik", Rostock, 20-26 April, 1975.

3. Mikolajccak:"On the reducibility of periodic automata", Foundations, of Control Engineering 1, 1(1975), 15-36.

СПИСОК АВТОРОВ