ISSN 0555—2656

ПРОБЛЕМЫ БИОНИКИ

ВЫПУСК

26

УДК 007 : 573.6
32.818
П78

Проблемы бионики, вып. 26.— Респ. межвед. науч.-техн. сборник.— Харьков: Вища школа. Изд-во при Харьк. ун-те, 1981.— 132+4 с.

В сборнике рассматриваются вопросы математического и физического моделирования процессов переработки информации на различных уровнях нервной системы. Разрабатывается математический аппарат, ориентированный на нужды описания функций человеческого интеллекта. Предлагается методика моделирования, а также конкретные модели различных видов интеллектуальной деятельности человека. Обсуждаются вопросы построения многоцелевого банка лингвистических данных.

Для научных работников и специалистов в области бионики, кибернетики, вычислительной техники, инженерной психологии, биологии и медицины. Списки лит. в конце статей.

Адрес редакционной коллегии: 310218, Харьков-218, пр. Ленина, 14, институт радиоэлектроники, тел. 40-96-45

Редакция естественнонаучной литературы

П $\frac{30501—022}{M226(04)—81}$ 435—81 1502000000

УДК 510.62

Ю. П. ШАБАНОВ-КУШНАРЕНКО, д-р техн. наук

О МОДЕЛИРОВАНИИ АЛФАВИТНЫХ ОПЕРАТОРОВ СРЕДСТВАМИ ТЕОРИИ ИНТЕЛЛЕКТА

Идеи, заложенные в работах [1, 2], нашли дальнейшее развитие в настоящей статье. Опишем процесс дешифровки слов некоторыми алфавитными операторами неарифметического характера.

Пусть $\sigma_{11}\sigma_{21}\ldots\sigma_{m1}$, $\sigma_{12}\sigma_{22}\ldots\sigma_{m2}\ldots$, $\sigma_{1n}\sigma_{2n}\ldots\sigma_{mn}$ — n слов длины m, подлежащих дешифровке. В процессе дешифровки слову $\sigma_{1i}\sigma_{2i}\ldots\sigma_{mi}$ ставится в соответствие буква b_i, причем разным словам соответствуют различные буквы. Введем буквенные переменные $x_1 \in \{a_{11}, a_{12}, \ldots, a_{1k_1}\}$, $x_2 \in \{a_{21}, a_{22}, \ldots, a_{2k_2}\}, \ldots$, $x_m \in \{a_{m1}, a_{m2}, \ldots, a_{mk_m}\}$, $y \in \{b_1, b_2, \ldots, b_n\}$. Оператор дешифровки $y = D(x_1, x_2, \ldots, x_m)$ запишем

в неявной форме:

$$x_1^{\sigma_{11}}x_2^{\sigma_{21}}\ldots x_m^{\sigma_{m1}}y^{b_1} \vee x_1^{\sigma_{12}}x_2^{\sigma_{22}}\ldots x_m^{\sigma_{m2}}y^{b_2} \vee \ldots \vee x_1^{\sigma_{1n}}x_2^{\sigma_{2n}}\ldots$$
$$\ldots x_m^{\sigma_{mn}}y^{b_n} = 1, \tag{1}$$

в явной форме:

$$\begin{aligned} x_1^{\sigma_{11}}x_2^{\sigma_{21}}\ldots x_m^{\sigma_{m1}} &= y^{b_1},\\ x_1^{\sigma_{12}}x_2^{\sigma_{22}}\ldots x_m^{\sigma_{m2}} &= y^{b_2},\\ &\ldots\ldots\\ x_1^{\sigma_{1n}}x_2^{\sigma_{2n}}\ldots x_m^{\sigma_{mn}} &= y^{b_n}. \end{aligned} \tag{2}$$

В качестве примера построим дешифратор, нумерующий десятичными цифрами следующие окончания русских имен прилагательных: АЯ, ЯЯ, УЮ, ЮЮ, ОЮ, ЕЮ, ЫЕ, ИЕ, ОЕ, ЕЕ. Неявное описание этого дешифратора имеет вид

$$x_1^{\text{а}}x_2^{\text{я}}y^{0} \vee x_1^{\text{я}}x_2^{\text{я}}y^{1} \vee x_1^{\text{у}}x_2^{\text{ю}}y^{2} \vee x_1^{\text{ю}}x_2^{\text{ю}}y^{3} \vee x_1^{0}x_2^{\text{ю}}y^{4} \vee$$
$$\vee x_1^{\text{е}}x_2^{\text{ю}}y^{5} \vee x_1^{\text{ы}}x_2^{\text{е}}y^{6} \vee x_1^{\text{и}}x_2^{\text{е}}y^{7} \vee x_1^{0}x_2^{\text{е}}y^{8} \vee x_1^{\text{е}}x_2^{\text{е}}y^{9} = 1. \tag{a}$$

Здесь x_1x_2 — окончание; y — его номер. Введенные буквенные переменные имеют следующие области определения: $x_1 \in \{$а, я, у, ю, о, е, ы, и$\}$, $x_2 \in \{$я, ю, е$\}$, $y \in \{0, 1, \ldots, 9\}$.

По явному описанию

$$x_1^{а}x_2^{я} = y^0,\ x_1^{я}x_2^{я} = y^1,\ x_1^{у}x_2^{ю} = y^2,\ x_1^{ю}x_2^{ю} = y^3;\ x_1^{о}x_2^{ю} = y^4,$$
$$x_1^{е}x_2^{ю} = y^5,\ x_1^{ы}x_2^{е} = y^6,\ x_1^{и}x_2^{е} = y^7;\ x_1^{0}x_2^{е} = y^8,\ x_1^{е}x_2^{е} = y^9 \qquad (б)$$

строим переключательную цепь дешифратора (рис. 1). В ней указаны все каскады преобразования сигналов.

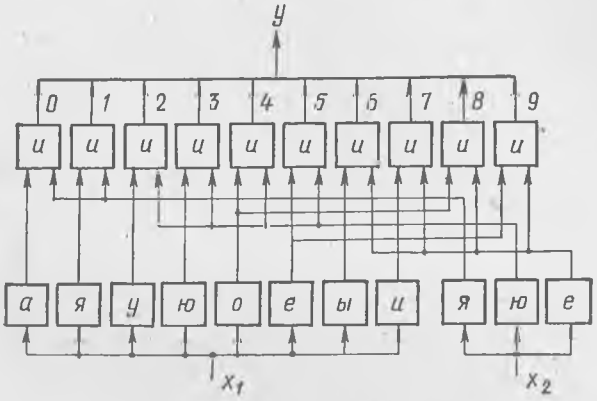


Рис. 1

Опишем теперь обратный процесс шифровки слов. Оператор шифровки $x_1x_2 \ldots x_m = s(y)$ ставит в соответствие букве b_i слово длины m $\sigma_{1i}\sigma_{2i} \ldots \sigma_{mi}$, где $1 \leqslant i \leqslant n$, причем разным буквам соответствуют различные слова. В неявной форме оператор шифровки описывается тем же уравнением (1), что и оператор дешифровки. В явной форме он может быть представлен следующей системой уравнений:

$$\begin{gathered}
\tau_{111}y^{b_1} \vee \tau_{112}y^{b_2} \vee \ldots \vee \tau_{11n}y^{b_n} = x_1^{a_{11}};\\
\tau_{121}y^{b_1} \vee \tau_{122}y^{b_2} \vee \ldots \vee \tau_{12n}y^{b_n} = x_1^{a_{12}};\\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
\tau_{1k_11}y^{b_1} \vee \tau_{1k_12}y^{b_2} \vee \ldots \vee \tau_{1k_1n}y^{b_n} = x_1^{a_{1k_1}};\\
\tau_{211}y^{b_1} \vee \tau_{212}y^{b_2} \vee \ldots \vee \tau_{21n}y^{b_n} = x_2^{a_{21}},\\
\tau_{221}y^{b_1} \vee \tau_{222}y^{b_2} \vee \ldots \vee \tau_{22n}y^{b_n} = x_2^{a_{22}};\\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
\tau_{2k_21}y^{b_1} \vee \tau_{2k_22}y^{b_2} \vee \ldots \vee \tau_{2k_2n}y^{b_n} = x_2^{a_2k_2},\\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
\tau_{m11}y^{b_1} \vee \tau_{m12}y^{b_2} \vee \ldots \vee \tau_{m1n}y^{b_n} = x_m^{a_{m1}};\\
\tau_{m21}y^{b_1} \vee \tau_{m22}y^{b_2} \vee \ldots \vee \tau_{m2n}y^{b_n} = x_m^{a_{m2}};\\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
\tau_{mk_11}y^{b_1} \vee \tau_{mk_m2}y^{b_2} \vee \ldots \vee \tau_{mk_mn}y^{b_n} = x_m^{a_{mk_m}}.
\end{gathered} \qquad (3)$$

Здесь τ_{ijk} — некоторые логические константы, определяемые по формулам

$$\tau_{ijk} = \sigma_{ik}^{a_{ij}}. \qquad (4)$$

Индексы i, j, k изменяются в пределах $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant k_i$, $1 \leqslant k \leqslant n$. В качестве примера построим шифратор, формирующий перечисленные выше окончания имен прилагательных по их номерам. В данном случае имеем

$m = 2$, $n = 10$, $k_1 = 8$,
$k_2 = 3$, $a_{11} = $ а, $a_{12} = $ Я,
$a_{13} = $ У, $a_{14} = $ Ю, $a_{15} = 0$,
$a_{16} = $ Е, $a_{17} = $ Ы, $a_{18} = $ И,
$a_{21} = $ Я, $a_{22} = $ Ю, $a_{23} = $ Е,
$\sigma_{11} = $ а, $\sigma_{21} = $ Я, $\sigma_{12} = $ Я,
$\sigma_{22} = $ Я, $\sigma_{13} = $ У, $\sigma_{23} = $ Ю,
$\sigma_{14} = $ Ю, $\sigma_{24} = $ Ю, $\sigma_{15} = 0$,
$\sigma_{25} = $ Ю, $\sigma_{16} = $ е, $\sigma_{26} = $ Ю,
$\sigma_{17} = $ ы, $\sigma_{27} = $ е, $\sigma_{18} = $ И,
$\sigma_{28} = $ е, $\sigma_{19} = 0$, $\sigma_{29} = $ е,
$\sigma_{1,10} = $ е, $\sigma_{2,10} = $ е, $b_1 = 0$,
$b_2 = 1$, $b_3 = 2$, $b_4 = 3$,
$b_5 = 4$, $b_6 = 5$, $b_7 = 6$,
$b_8 = 7$, $b_9 = 8$, $b_{10} = 9$.

i	j	k = 1	2	3	4	5	6	7	8	9	10
1	1	1									
	2		1								
	3			1							
	4				1						
	5					1				1	
	6						1				1
	7							1			
	8								1		
2	1	1	1								
	2			1	1	1	1				
	3							1	1	1	1

По формуле (4) определяем $\tau_{111} = \sigma_{11}^{a_{11}} = a^a = 1$, $\tau_{112} = \sigma_{12}^{a_{11}} = $ Я$^a = 0$. Аналогичным образом найденные константы τ_{ij} при всевозможных значениях i, j, k представлены в таблице. Нулевые значения в ней опущены. Теперь все подготовлено для записи уравнений (3):

$$y^0 = x_1^{\text{а}},\ y^1 = x_1^{\text{я}},\ y^2 = x_1^{\text{у}},\ y^3 = x_1^{\text{ю}},\ y^4 \vee y^8 = x_1^{0},\ y^5 \vee y^9 = x_1^{\text{е}},$$
$$y^6 = x_1^{\text{ы}},\ y^7 = x_1^{\text{и}},\ y^0 \vee y^1 = x_2^{\text{я}},\ y^2 \vee y^3 \vee y^4 \vee y^5 =$$
$$= x_2^{\text{ю}},\ y^6 \vee y^7 \vee y^8 \vee y^9 = x_2^{\text{е}}. \qquad (в)$$

Соответствующий этим уравнениям шифратор представлен на рис. 2.

Рассмотрим отношение равенства $x = y$ для двух буквенных переменных x, y, изменяющихся в одной и той же области $\{a,$

$a_2, \ldots, a_k\}$. Оно может быть описано в неявном виде следующим каноническим уравнением:

$$x^{a_1}y^{a_1} \vee x^{a_2}y^{a_2} \vee \ldots \vee x^{a_k}y^{a_k} = 1. \qquad (5)$$

В явном виде зависимость переменной y от переменной x выразится следующей системой равенств:

$$x^{a_1} = y^{a_1},\ x^{a_2} = y^{a_2},\ \ldots,\ x^{a_k} = y^{a_k}. \qquad (6)$$

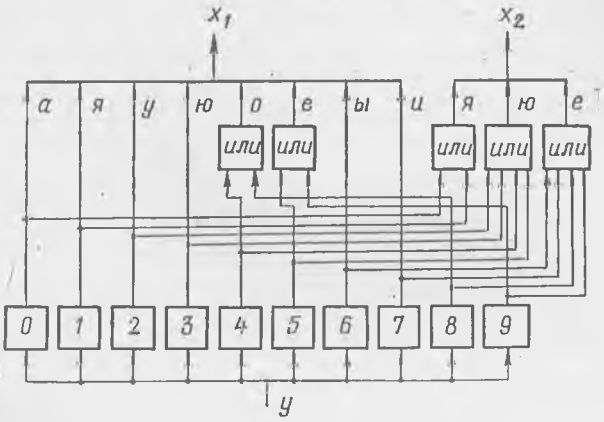


Рис. 2

В переключательной цепи для этой зависимости (рис. 3, *а*) второй каскад преобразования сигналов имеет вид простого пучка параллельных проводов.

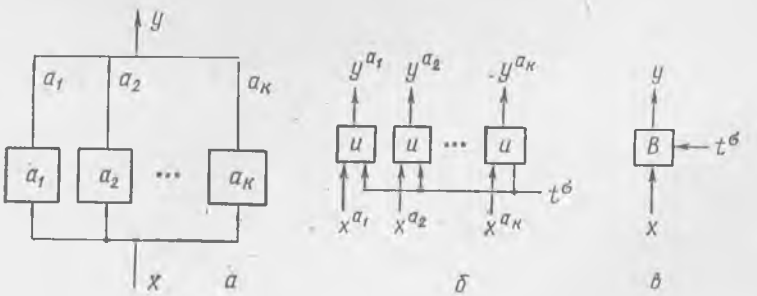


Рис. 3

Если переменные x и y заданы на различных областях A и B, то отношение равенства для них моделируется аналогично, с тем отличием, что в уравнениях и схеме теперь фигурируют лишь те буквы, которые составляют область $A \cap B$. В дальнейшем для краткости введем специальное обозначение

$$x \approx y = x^{a_1} y^{a_1} \vee x^{a_2} y^{a_2} \vee \ldots \vee x^{a_k} y^{a_k} \qquad (7)$$

для предиката, стоящего в левой части равенства (5). Предикат $x \approx y$ назовем предикатом равенства для области $\{a_1, a_2, \ldots, a_k\}$ и переменных x, y.

Рассмотрим отношение, описываемое уравнением

$$(x \approx y)\, t^{\sigma} = 1. \qquad (8)$$

Назовем его вентильным отношением. Здесь x, y — буквенные переменные, заданные на множестве $A = \{a_1, a_2, \ldots, a_k\}$, t — буквенная переменная, заданная на множестве $B = \{b_1, b_2, \ldots, b_e\}$, σ — некоторая фиксированная буква из множества B. Выражения зависимости y от x и t для (8) имеют вид:

$$x^{a_1} t^{\sigma} = y^{a_1},\ x^{a_2} t^{\sigma} = y^{a_2}, \ldots,\ x^{a_k} t^{\sigma} = y^{a_k}. \qquad (9)$$

Переключательную цепь (рис. 3, *б*), реализующую эти зависимости, назовем вентилем. Условное изображение вентиля показано на рис. 3, *в*. Вход x вентиля — рабочий, вход t — управляющий. Вентиль работает следующим образом: когда на управляющий вход вентиля поступает буква σ, сигнал с рабочего входа беспрепятственно проходит на выход вентиля; когда же $t \neq \sigma$, то на выходе вентиля выходной сигнал отсутствует.

Важно обратить внимание на то, что вентиль реализует частичную функцию: при $t \neq \sigma$ значение ее не существует. Все узнавания переменной y обращаются в нули. Интересно, что средствами комбинационной техники, обычно используемой для построения современных вычислительных машин, так работающий вентиль построить невозможно: с помощью комбинационных схем можно реализовать лишь всюду определенные функции, в данном же случае мы сталкиваемся с частичной функцией. Можно было бы, конечно, как это обычно и делается на практике, превратить частичную функцию во всюду определенную и построить комбинационную схему для нее. Но при этом мы неизбежно исказим принцип работы вентиля. Выходные сигналы теперь будут вырабатываться как при открытом, так и при закрытом вентиле, мы же первоначально хотели, чтобы закрытый вентиль не вырабатывал на своем выходе вообще никаких сигналов. Этим примером демонстрируется важное преимущество переключательных цепей теории интеллекта: в отличие от комбинационных схем, получаемых на базе формул алгебры логики, переключательные цепи способны реализовать настоящие частичные функции, а не их суррогат в виде функций, появляющихся в результате процедуры доопределения.

Рассмотрим уравнение

$$(x_1 \approx y) \vee (x_2 \approx y) \vee \ldots \vee (x_m \approx y) = 1, \qquad (10)$$

вводящее отношение сборки. Опишем зависимость переменной y от переменных $x_1, x_2, \ldots, x_m$, определяемую этим отношением:

$$\begin{aligned} x_1^{a_1} \vee x_2^{a_1} \vee \ldots \vee x_m^{a_1} &= y^{a_1}; \\ x_1^{a_2} \vee x_2^{a_2} \vee \ldots \vee x_m^{a_2} &= y^{a_2}; \\ \ldots\ldots\ldots\ldots\ldots\ldots & \\ x_1^{a_k} \vee x_2^{a_k} \vee \ldots \vee x_m^{a_k} &= y^{a_k}. \end{aligned} \tag{11}$$

Переключательную цепь (рис. 4, *а*), реализующую ее, назовем схемой сборки (рис. 4, *б*). Схема сборки предназначена для работы

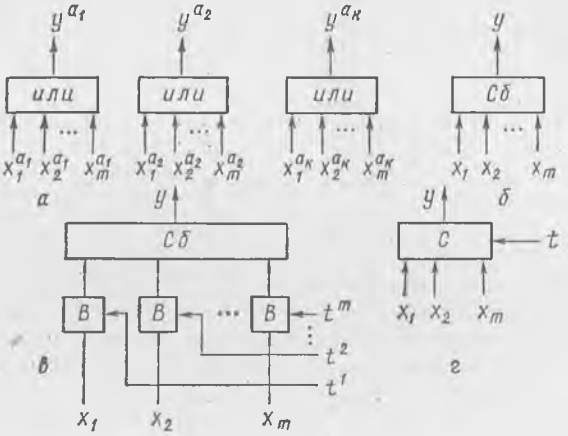


Рис. 4

в условиях, когда на все ее входы либо вообще не подается сигналов, либо входной сигнал поступает только на один из входов. В первом случае схема сборки не формирует никакого выходного сигнала, во втором — передает на выход сигнал, поступивший по одному из входов.

Используя схему сборки и вентили, можно построить переключательную цепь (рис. 4, *в*), выполняющую функции селектора. Условное изображение селектора представлено на рис. 4, *г*. Значениями управляющего сигнала t служат натуральные числа от 1 до m. Если $t = i$, то сигнал с входа x_i передается на выход y. Неявное описание работы селектора имеет вид

$$(x_1 \approx y)\, t^1 \vee (x_2 \approx y)\, t^2 \vee \ldots \vee (x_m \approx y)\, t^m = 1. \tag{12}$$

В явном виде схема селектора описывается уравнениями:

$$\begin{aligned} x_1^{a_1} t^1 \vee x_2^{a_1} t^2 \vee \ldots \vee x_m^{a_1} t^m &= y^{a_1}; \\ x_1^{a_2} t^1 \vee x_2^{a_2} t^2 \vee \ldots \vee x_m^{a_2} t^m &= y^{a_2}; \\ \ldots\ldots\ldots\ldots\ldots\ldots & \\ x_1^{a_k} t^1 \vee x_2^{a_k} t^2 \vee \ldots \vee x_m^{a_k} t^m &= y^{a_k}. \end{aligned} \tag{13}$$

Решая уравнение (12) относительно переменных $x_1, x_2, \ldots, x_m$, находим выражения для зависимости этих переменных от y и t:

$$y^{a_i} t^1 = x_1^{a_i},\ y^{a_i} t^2 = x_2^{a_i},\ \ldots,\ y^{a_i} t^m = x_m^{a_i}, \qquad (14)$$

где индекс i пробегает значения от 1 до k. Цепь (рис. 5, *а*), реализующую систему равенств (14), назовем переключателем. Обозначение переключателя приведено на рис. 5, *б*. Действие переключателя состоит в том, что он при подаче на управляю-

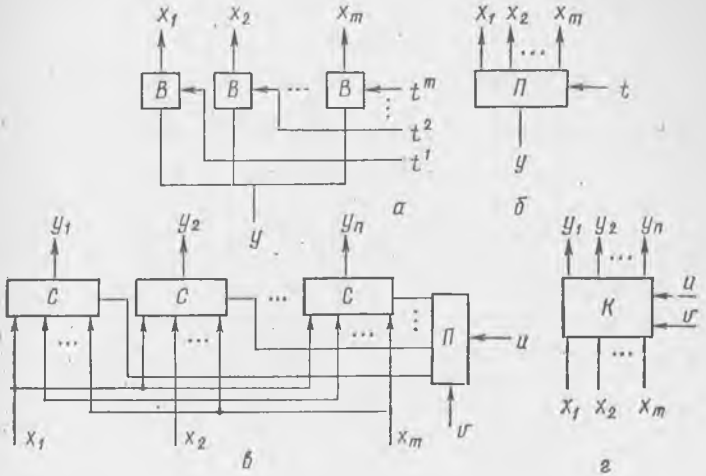


Рис. 5

щий вход t сигнала $i\,(1 \leqslant i \leqslant m)$ передает сигнал со входа y на i-й по счету выходной канал x_i.

На рис. 5, *в* показана переключательная цепь, которую мы назовем коммутатором на m входов и n выходов. В состав схемы коммутатора входит один переключатель на n входов и n селекторов на m входов каждый. Коммутатор имеет два управляющих входа u и v, на которые поступают номера i и j $(1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n)$ соответственно входа x_i и выхода y_i, соединяемых коммутатором. Когда $u = i$, $v = j$, коммутатор осуществляет передачу сигнала со входа x_i на выход y_j. На остальных своих выходах коммутатор не формирует при этом никаких сигналов. Условное обозначение коммутатора показано на рис. 5, *г*.

При неявном задании работа коммутатора опишется в виде уравнения

$$\bigvee_{(i,\ j)} (x_i \approx y_j)\, u^i v^j = 1. \qquad (15)$$

Здесь запись (i, j) под знаком дизъюнкции означает, что логическое суммирование ведется для всевозможных пар индексов i и j, где $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$. Явное задание коммутатора можно

получить описанными ранее приемами по уравнению (15), кроме того, его легко получить по схеме, поэтому здесь мы его не приводим.

Список литературы: 1. *Шабанов-Кушнаренко Ю. П.* О моделировании переключательных функций и алфавитных операторов.— АСУ и приборы автоматики, 1980, вып. 57, с. 63—70. 2. *Шабанов-Кушнаренко Ю. П.* О моделировании арифметических операций.— Проблемы бионики, 1980, вып. 25, с. 22—30.

Поступила 15 октября 1979 г.

УДК 62.506.2

Е. П. ПУТЯТИН, д-р техн. наук, *Т. Г. ДОЛЖЕНКОВА*

НОРМАЛИЗАЦИЯ НЕЛИНЕЙНЫХ ПРЕОБРАЗОВАНИЙ

СООБЩЕНИЕ 1

В статье описаны результаты исследований, начатых ранее [1, 2]. Предлагается новый более общий подход к отысканию параметров нелинейных преобразований, заключающийся в определении этих параметров непосредственно, а не путем использования обратных зависимостей.

Пусть изображение $B(x, y)$ получено из эталонного изображения $B_0(x, y)$ с помощью преобразования

$$B(x, y) = B_0(u(x, y), v(x, y)), \tag{1}$$

где $u(x, y)$, $v(x, y)$ представляются некоторыми сепарабельными соотношениями

$$u(x, y) = \sum_i a_i f_i(x, y); \quad v(x, y) = \sum_j b_j g_j(x, y), \tag{2}$$

a_i, b_j — неизвестные коэффициенты $(i, j = 1, 2, \ldots)$.

Для определения параметров a_i и b_j строим интегральные функционалы вида (заданы в пространстве L^2)

$$\iint_D B_0(u, v) K(u, v)\, du dv = \iint_D B(x, y) K(u(x, y), v(x, y)) \,|\, I(x, y) \,|\, dx dy. \tag{3}$$

Здесь якобиан $I(x, y)$ определяется следующим образом:

$$I(x, y) = \begin{vmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{vmatrix} = \begin{vmatrix} \sum_i a_i \frac{\partial f_i}{\partial x} & \sum_i a_i \frac{\partial f_i}{\partial y} \\ \sum_j b_j \frac{\partial g_j}{\partial x} & \sum_j b_j \frac{\partial g_j}{\partial y} \end{vmatrix} =$$

$$= \sum_{i,j} a_i b_j \left(\frac{\partial f_i}{\partial x} \frac{\partial g_j}{\partial y} - \frac{\partial g_j}{\partial x} \frac{\partial f_i}{\partial y} \right).$$

Обозначим $R_{ij}(x, y) = \frac{\partial f_i}{\partial x}\frac{\partial g_j}{\partial y} - \frac{\partial g_j}{\partial x}\frac{\partial f_i}{\partial y}$, тогда

$$I(x, y) = \sum_{i, j} a_i b_j R_{ij}(x, y). \tag{4}$$

Как и ранее в работах [1, 2], полагаем, что преобразования вида (1), (2) не выводят изображение $B(x, y)$ за пределы поля зрения D и $I(x, y) > 0$ на всей области интегрирования. Поэтому с учетом (2) и (4) выражение (3) примет вид

$$\iint_D B_0(u, v) K(u, v)\, du dv = \iint_D B(x, y) K\Big(\sum_i a_i f_i(x, y),$$

$$\sum_j b_j g_j(x, y)\Big)\Big(\sum_{i, j} a_i b_j R_{ij}(x, y)\Big)\, dxdy = \sum_{i, j} a_i b_j P_{ij}, \tag{5}$$

где $P_{ij} = \iint_D B(x, y) R_{ij}(x, y) K\Big(\sum_i a_i f_i(x, y), \sum_j b_j g_j(x, y)\Big)\, dxdy$.

Исследуем конкретное приложение изложенного метода на примерах различных преобразований.

Вначале рассмотрим нелинейные преобразования вида

$$u(x, y) = a_1 x^2 + a_2 y^2; \quad v(x, y) = b_1 x^2 + b_2 y^2. \tag{6}$$

Составляющие функции этого преобразования согласно (2) следующие: $f_1(x, y) = x^2$, $f_2(x, y) = y^2$, $g_1(x, y) = x^2$, $g_2(x, y) = y^2$, $I(x, y) = \Delta xy$, где $\Delta = 4(a_1 b_2 - a_2 b_1)$. Следовательно,

$$\iint_D B_0(u, v) K(u, v)\, du dv = \Delta \iint_D B(x, y) K(a_1 x^2 + a_2 y^2,$$

$$b_1 x^2 + b_2 y^2)\, xy dx dy. \tag{7}$$

В качестве интегральных функционалов введем двумерные моменты различных степеней i, j:

$$\Phi_{ij}(B_0) = \iint_D B_0(u, v)\, u^i v^j du dv, \quad \Phi_{ij}(B) = \iint_D B(x, y)\, x^i y^j\, dxdy. \tag{8}$$

Подставляя в (7) степенные значения подынтегральной функции $K(u, v)$, именно

$$1, x, y, x^2, y^2, \tag{9}$$

получаем

$$\Phi_{00}(B_0) = \Delta\Phi_{11}(B); \quad \Phi_{10}(B_0) = \Delta(a_1\Phi_{31}(B) + a_2\Phi_{13}(B));$$

$$\Phi_{01}(B_0) = \Delta(b_1\Phi_{31}(B) + b_2\Phi_{13}(B));$$

$$\Phi_{20}(B_0) = \Delta(a_1^2\Phi_{51}(B) + 2a_1 a_2\Phi_{33}(B) + a_2^2\Phi_{15}(B));$$

$$\Phi_{02}(B_0) = \Delta(b_1^2\Phi_{51}(B) + 2b_1 b_2\Phi_{33}(B) + b_2^2\Phi_{15}(B)).$$

Таким образом, получена система нелинейных уравнений того же типа, что и соответствующая система в работе [2], непосредственно относительно искомых параметров a_i, b_j (а не величин γ_i, связанных с ними обратными зависимостями). В данной

нелинейной системе, как видно из (10), использованы моменты входных изображений с ядрами xy, x^3y, xy^3, x^3y^3, x^5y, xy^5, а также моменты эталонных изображений с ядрами (9). Решая систему нелинейных уравнений (10), получаем параметры преобразования a_i, b_j $(i, j = 1, 2)$

$$a_1 = (-A \pm \sqrt{A^2 - 4FS})/(2F), \quad a_2 = T - Ea_1,$$
$$b_1 = (-Q \pm \sqrt{Q^2 - 4FC})/(2F), \quad b_2 = G - Eb_1; \qquad (11)$$

где

$$A = 2\Phi_{10}(B_0)\,\Phi_{11}(B)\,[\Phi_{33}(B)\,\Phi_{13}(B) - \Phi_{15}(B)\,\Phi_{31}(B)]/[\Phi_{00}(B_0)\,\Phi_{13}^2(B)];$$
$$F = \Phi_{51}(B) - 2\Phi_{33}(B)\,\Phi_{31}(B)/\Phi_{13}(B) + \Phi_{31}^2(B)\,\Phi_{15}(B)/\Phi_{13}^2(B);$$
$$S = [\Phi_{10}^2(B_0)\,\Phi_{11}^2(B)\,\Phi_{15}(B) - \Phi_{20}(B_0)\,\Phi_{11}(B)\,\Phi_{00}(B_0) \times$$
$$\times \Phi_{13}^2(B)]/[\Phi_{00}^2(B_0)\,\Phi_{13}^2(B)]; \quad T = \Phi_{10}(B_0)\,\Phi_{11}(B)/[\Phi_{00}(B_0)\,\Phi_{13}(B)];$$
$$E = \Phi_{31}(B)/\Phi_{13}(B); \quad Q = 2\Phi_{01}(B_0)\,\Phi_{11}(B)\,[\Phi_{33}(B)\,\Phi_{13}(B) -$$
$$- \Phi_{15}(B)\,\Phi_{31}(B)]/[\Phi_{00}(B_0)\,\Phi_{13}^2(B)]; \; C = [\Phi_{01}^2(B_0)\,\Phi_{11}^2(B)\,\Phi_{15}(B) -$$
$$- \Phi_{20}(B_0)\,\Phi_{11}(B)\,\Phi_{00}(B_0)\,\Phi_{13}^2(B)]/[\Phi_{00}^2(B_0)\,\Phi_{13}^2(B)];$$
$$G = \Phi_{01}(B_0)\,\Phi_{11}(B)/[\Phi_{00}(B_0)\,\Phi_{13}(B)].$$

В случае, когда $u(x, y)$ и $v(x, y)$ представляют собой более сложные зависимости, описанный подход в определении параметров преобразований, в отличие от подхода, изложенного в [1, 2], также позволяет свести задачу отыскания коэффициентов к системе нелинейных уравнений.

Пусть изображение $B(x, y)$ связано с эталонным изображением $B_0(x, y)$ зависимостью $B(x, y) = B_0(u(x, y), v(x, y))$, где

$$u = a_1x^2 + a_2xy + a_3y^2 + a_4, \quad v = b_1x^2 + b_2xy + b_3y^2 + b_4. \qquad (12)$$

Здесь $f_1(x, y) = x^2$; $f_2(x, y) = xy$; $f_3(x, y) = y^2$; $f_4(x, y) = 1$; $g_1(x, y) = x^2$; $g_2(x, y) = xy$; $g_3(x, y) = y^2$; $g_4(x, y) = 1$.

Якобиан преобразования (12) $I(x, y) = 2(a_1b_2 - b_1a_2)x^2 + 4(a_1b_3 - b_1a_3)xy + 2(a_2b_3 - b_2a_3)y^2$. Выражение (3) для рассмотренного случая примет вид

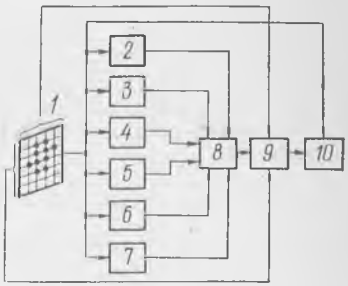
$$\iint_D B_0(u, v)\,K(u, v)\,du dv = \iint_D B(x, y)\,K(a_1x^2 + a_2xy + a_3y^2 +$$
$$+ a_4, \; b_1x^2 + b_2xy + b_3y^2 + b_4)\,|I(x, y)|\,dxdy. \qquad (13)$$

Положим $K(x, y) = 1$. Подставляя это значение в (13) и учитывая (8), имеем $\Phi_{00}(B_0) = 2[(a_1b_2 - b_1a_2)\Phi_{20}(B) + 2(a_1b_3 - b_1a_3) \times \times \Phi_{11}(B) + (a_2b_3 - b_2a_3)\Phi_{02}(B)]$. Аналогично для $K(x, y) = x$ получаем

$$\Phi_{10}(B_0) = 2\{a_1(a_1b_2 - b_1a_2)\Phi_{40}(B_0) + [a_2(a_1b_2 - b_1a_2) + 2a_1(a_1b_3 -$$
$$- b_1a_3)]\,\Phi_{31}(B) + [a_3(a_1b_2 - b_1a_2) + 2a_2(a_1b_3 - b_1a_3) + a_1(a_2b_3 -$$

$$- b_2a_3)] \Phi_{22}(B) + [2a_3(a_1b_3 - b_1a_3) + a_2(a_2b_3 - b_2a_3)] \Phi_{13}(B) +$$
$$+ a_3(a_2b_3 - b_2a_3) \Phi_{04}(B) + a_4(a_1b_2 - b_1a_2) \Phi_{20}(B) +$$
$$+ 2a_4(a_1b_3 - b_1a_3) \Phi_{11}(B) + a_4(a_2b_3 - b_2a_3) \Phi_{02}(B)\}.$$

Полагая далее, что $K(x, y)$ принимает значения y, x^2, xy, y^2 и т. д., получаем систему нелинейных алгебраических уравнений относительно неизвестных a_i и b_j (i, $j = 1, 2, 3, 4$). Как видим, система нелинейных уравнений представляется довольно сложной и для ее решения требуется хорошее математическое обеспечение.



Достоинством рассмотренного метода вычисления коэффициентов является его универсальность по отношению ко входным изображениям. Алгоритмы допускают техническую реализацию в специализированном вычислительном устройстве, структурная схема которого приведена на рисунке применительно к преобразованиям (6). Устройство включает блок преобразования изображений в электрические сигналы *1*, блоки определения признаков изображения *2—7*, вычислительный блок *8*, блок определения эталонных значений координат *9* и блок распознавания *10*.

Блок преобразования изображений в электрические сигналы *1* представляет собой телевизионную камеру. Каждый из блоков определения признаков изображения *2—7* предназначен для определения смешанного момента изображения из нечетных степеней координат x, y, начиная с младших, так, что ядрами моментов входного изображения являются функции xy, x^3y, xy^3, x^3y^3, x^5y, xy^5.

Устройство работает следующим образом. Изображение, характеризующееся некоторой функцией распределения яркости на плоскости и подвергающееся нелинейным преобразованиям координат типа квадратичных форм, проецируется в блок преобразования изображений в электрические сигналы *1*, на выходе которого вырабатываются сигналы освещенности, пропорциональные яркости изображения в соответствующих точках поля зрения.

Сигналы освещенности поступают в блоки определения признаков изображения *2—7*, где вычисляются смешанные моменты из нечетных степеней координат поля зрения, начиная с младших. По полученным значениям смешанных моментов второго, четвертого и шестого порядков вычислительный блок *8* определяет значения четырех неизвестных параметров a_i, b_j (i, $j = 1, 2$) преобразований, которым подвергается входное изображение по

отношению к эталонному, т. е. решает систему уравнений (10). Полученные параметры с выхода блока *8* поступают в блок определения эталонных координат *9*, где для каждого значения освещенности блока *1* вычисляются новые значения эталонных координат, при этом учитываются ядра эталонов (9). Значения освещенности с выхода блока *1* и соответствующие им эталонные координаты, снимаемые с блока *9*, подаются в блок распознавания *10*, где происходит процедура сравнения с эталонами с целью классификации входных изображений.

Список литературы: 1. *Путятин Е. П., Долженкова Т. Г.* Вопросы нормализации нелинейных преобразований. Сообщение 1.— Проблемы бионики, 1980, вып. 24, с. 111—115. 2. *Путятин Е. П., Долженкова Т. Г.* Вопросы нормализации нелинейных преобразований. Сообщение 2.—Проблемы бионики, 1980, вып. 24, с. 116—120. 3. *Путятин Е. П.* Теоретические предпосылки нормализации изображений. Сообщение 1. — Проблемы бионики, 1973, вып. 10, с. 82—89.

Поступила 16 октября 1979 г.

УДК 62.506.2

Е. П. ПУТЯТИН, д-р техн. наук, *Т. Г. ДОЛЖЕНКОВА*

НОРМАЛИЗАЦИЯ НЕЛИНЕЙНЫХ ПРЕОБРАЗОВАНИЙ

СООБЩЕНИЕ 2

В работах [1—3] рассмотрены вопросы нормализации некоторых видов нелинейных преобразований, предложены методы определения параметров этих преобразований. Один из методов [1, 2] предполагает нахождение обратных величин, связанных с искомыми параметрами, и только после этого — самих параметров преобразований. Второй метод [3] сводился к непосредственному определению коэффициентов нелинейных преобразований. Следует отметить, что оба метода приводили в конечном итоге к решению систем нелинейных уравнений. В случае сложных нелинейных преобразований не существует способа получения решения такого рода систем в виде формул, их приходится решать приближенно.

Нами продолжены исследования по отысканию коэффициентов нелинейных преобразований, в частности рассмотрен метод, позволяющий определить коэффициенты нелинейных искажений путем решения систем линейных уравнений.

Рассмотрим преобразование, когда между изображением $B(x, y)$ и эталонным изображением $B_0(x, y)$ существует следующая зависимость:

$$B(x, y) = B_0(a_1x^2 + a_2y^2, \quad b_1x^2 + b_2y^2). \qquad (1)$$

При вычислении якобиана преобразования (1) пользуемся рассуждениями, приведенными в [3], тогда

$$I(x,\ y) = \Delta xy, \tag{2}$$

где $\Delta = 4(a_1b_2 - a_2b_1)$.

Для определения параметров a_i и b_j $(i,\ j = 1, 2)$ рассмотрим наборы функционалов эталонов:

$$\Phi_{00}(B_0^k) = \iint\limits_D \varphi_k(B)\,dudv; \quad \Phi_{10}(B_0^k) = \iint\limits_D \varphi_k(B)\,ududv;$$

$$\Phi_{01}(B_0^k) = \iint\limits_D \varphi_k(B)\,vdudv, \tag{3}$$

здесь $\varphi_k(B)$ — произвольная, наперед заданная функция, $k = 1, 2, \ldots$ В качестве $\varphi_k(B)$ можно, например, выбрать степенную зависимость. Для определенности в дальнейшем будем считать $\varphi_k(B) = B^k$, $k = 1, 2, \ldots$

С учетом (2) функционалы изображений, в свою очередь, будут иметь вид

$$\Phi_{11}(B^k) = \iint\limits_D B^k(x,\ y)\,xydxdy; \quad \Phi_{31}(B^k) = \iint\limits_D B^k(x,\ y)\,x^3ydxdy;$$

$$\Phi_{13}(B^k) = \iint\limits_D B^k(x,\ y)\,xy^3dxdy. \tag{4}$$

Распишем выражение для $\Phi_{00}(B_0)$:

$$\Phi_{00}(B_0) = \iint\limits_D B_0(u,\ v)\,dudv = \Delta \iint\limits_D B(x,\ y)\,xydxdy = \Delta\Phi_{11}(B_0).$$

Отсюда

$$\Delta = \frac{\Phi_{00}(B_0)}{\Phi_{11}(B)}. \tag{5}$$

Преобразовав интегралы $\Phi_{10}(B^k)$ $(k = 1, 2)$ аналогичным образом, получим систему линейных уравнений $\Phi_{10}(B_0) = \Delta[a_1\Phi_{31}(B) + a_2\Phi_{13}(B)]$; $\Phi_{10}(B_0^2) = \Delta[a_1\Phi_{31}(B^2) + a_2\Phi_{13}(B^2)]$.

Учитывая (5), получаем

$$a_1\Phi_{31}(B) + a_2\Phi_{13}(B) = \Phi_{10}(B_0)\,\Phi_{11}(B)/\Phi_{00}(B_0);$$

$$a_1\Phi_{31}(B^2) + a_2\Phi_{13}(B^2) = \Phi_{10}(B_0^2)\,\Phi_{11}(B)/\Phi_{00}(B_0). \tag{6}$$

Полагаем, что множество M [1] плоских изображений удовлетворяет условию

$$\begin{vmatrix} \Phi_{31}(B) & \Phi_{13}(B) \\ \Phi_{31}(B^2) & \Phi_{13}(B^2) \end{vmatrix} = \begin{vmatrix} \iint\limits_D B(x,\ y)\,x^3ydxdy & \iint\limits_D B(x,\ y)\,xy^3dxdy \\ \iint\limits_D B^2(x,\ y)\,x^3ydxdy & \iint\limits_D B^2(x,\ y)\,xy^3dxdy \end{vmatrix} \neq 0. \tag{7}$$

Система (6) в этом случае, очевидно, имеет единственное решение.

Проделав аналогичной последовательности операции над функционалами $\Phi_{01}(B_0^k)$ $(k = 1, 2)$, получим подобную систему линейных уравнений

$$b_1\Phi_{31}(B) + b_2\Phi_{13}(B) = \Phi_{31}(B_0)\,\Phi_{13}(B)/\Phi_{00}(B); \quad b_1\Phi_{31}(B^2) + b_2\Phi_{13}(B^2) = \Phi_{31}(B_0^2)\,\Phi_{13}(B)/\Phi_{00}(B). \tag{8}$$

Решая систему (8) при аналогичных условиях (7), находим параметры b_j $(j = 1, 2)$.

Проиллюстрируем возможности изложенного способа определения параметров нелинейных преобразований. Рассмотрим реализацию его для более сложных зависимостей, когда $B(x, y) = B_0(u(x, y), v(x, y))$, где

$$u = a_1x + a_2y + a_3x^2 + a_4xy + a_5y^2 + a_6xy^2; \quad v = b_1x + b_2y + b_3x^2 + b_4xy + b_5y^2 + b_6xy^2. \tag{9}$$

К анализу таких преобразований сводятся задачи фотограмметрии [4, с. 43].

Пользуясь выкладками, представленными в [3], несложно определить якобиан преобразования (9) $I(x, y) = Ex + Py + Rxy + Zx^2 + Ly^2 + Nx^2y + Wxy^2 + Ky^3 + H$, где

$$E = a_1b_4 - b_1a_4 - 2(a_2b_3 - b_2a_3); \quad P = a_4b_2 - b_4a_2 - 2(a_5b_1 - b_5a_1);$$

$$R = 2[a_1b_6 - b_1a_6 - 2(a_5b_3 - b_5a_3)]; \quad Z = 2(a_3b_4 - b_3a_4),$$

$$L = a_6b_2 - b_6a_2 - 2(a_5b_4 - b_5a_4); \quad N = 4(a_3b_6 - b_3a_6);$$

$$W = a_4b_6 - b_4a_6; \quad K = 2(a_6b_5 - b_6a_5); \quad H = a_1b_2 - b_1a_2. \tag{10}$$

Воспользуемся набором функционалов эталонов (3). Функционалы изображений имеют вид

$$\Phi_{00}(B^k) = \iint_D B^k(x, y)\,dxdy; \quad \Phi_{10}(B^k) = \iint_D B^k(x, y)\,xdxdy;$$

$$\Phi_{01}(B^k) = \iint_D B^k(x, y)\,ydxdy; \quad \Phi_{11}(B^k) = \iint_D B^k(x, y)\,xydxdy;$$

$$\Phi_{20}(B^k) = \iint_D B^k(x, y)\,x^2dxdy; \quad \Phi_{02}(B^k) = \iint_D B^k(x, y)\,y^2dxdy;$$

$$\Phi_{21}(B^k) = \iint_D B^k(x, y)\,x^2ydxdy; \quad \Phi_{12}(B^k) = \iint_D B^k(x, y)\,xy^2dxdy;$$

$$\Phi_{22}(B^k) = \iint_D B^k(x, y)\,x^2y^2dxdy; \quad \Phi_{30}(B^k) = \iint_D B^k(x, y)\,x^3dxdy;$$

$$\Phi_{03}(B^k) = \iint_D B^k(x, y)\,y^3dxdy; \quad \Phi_{13}(B^k) = \iint_D B^k(x, y)\,xy^3dxdy;$$

$$\Phi_{31}(B^k) = \iint_D B^k(x, y)\,x^3ydxdy; \quad \Phi_{23}(B^k) = \iint_D B^k(x, y)\,x^2y^3dxdy;$$

$$\Phi_{32}(B^k) = \iint_D B^k(x, y)\,x^3y^2dxdy; \quad \Phi_{33}(B^k) = \iint_D B^k(x, y)\,x^3y^3dxdy;$$

$$\Phi_{40}(B^k) = \iint_D B^k(x, y)\,x^4dxdy; \quad \Phi_{04}(B^k) = \iint_D B^k(x, y)\,y^4dxdy;$$

$$\Phi_{14}(B^k) = \iint_D B^k(x, y)\,xy^4\,dxdy; \quad \Phi_{41}(B^k) = \iint_D B^k(x, y)\,x^4ydxdy;$$

$$\Phi_{24}(B^k) = \iint_D B^k(x, y)\,x^2y^4dxdy; \quad \Phi_{05}(B^k) = \iint_D B^k(x, y)\,y^5dxdy;$$

$$\Phi_{15}(B^k) = \iint_D B^k(x, y)\,xy^5dxdy. \tag{11}$$

Учитывая (11), преобразуем выражение для $\Phi_{00}(B_0^k)$:

$$\Phi_{00}(B_0^k) = \iint\limits_D B_0^k(u, v)\,dudv = \iint\limits_D B(x, y)\,|I(x, y)|\,dxdy =$$

$$= \iint\limits_D (Ex + Py + Rxy + Zx^2 + Ly^2 + Nx^2y^2 + Wxy^2 + Ky^3 +$$

$$+ H)\,dxdy = E\Phi_{10}(B^k) + P\Phi_{01}(B^k) + R\Phi_{11}(B^k) + Z\Phi_{20}(B^k) +$$

$$+ L\Phi_{02}(B^k) + N\Phi_{22}(B^k) + W\Phi_{12}(B^k) + K\Phi_{03}(B^k) + H\Phi_{00}(B^k).$$

Поскольку число неизвестных E, P, R, ..., H равно девяти, то индекс $k = 1, 2, \ldots, 9$. Неизвестные коэффициенты определяются путем решения следующей системы линейных уравнений:

$$\Phi_{00}(B_0) = E\Phi_{10}(B) + P\Phi_{01}(B) + R\Phi_{11}(B) + Z\Phi_{20}(B) + L\Phi_{02}(B) +$$
$$+ N\Phi_{22}(B) + W\Phi_{12}(B) + K\Phi_{03}(B) + H\Phi_{00}(B);$$

$$\Phi_{00}(B_0^2) = E\Phi_{10}(B^2) + P\Phi_{01}(B^2) + R\Phi_{11}(B^2) + Z\Phi_{20}(B^2) +$$
$$+ L\Phi_{02}(B^2) + N\Phi_{22}(B^2) + W\Phi_{12}(B^2) + K\Phi_{03}(B^2) + H\Phi_{00}(B^2); \quad (12)$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\Phi_{00}(B_0^9) = E\Phi_{10}(B^9) + P\Phi_{01}(B^9) + R\Phi_{11}(B^9) + Z\Phi_{20}(B^9) +$$
$$+ L\Phi_{02}(B^9) + N\Phi_{22}(B^9) + W\Phi_{12}(B^9) + K\Phi_{03}(B^9) + H\Phi_{00}(B^9).$$

Подобным образом распишем выражение для $\Phi_{10}(B_0^k)$ и получим систему, количество уравнений которой равно количеству искомых параметров a_i ($i = 1, 2, \ldots, 6$), т. е. шести ($k = 1, 2, \ldots, 6$). Каждое из уравнений системы имеет вид

$$\Phi_{10}(B_0^k) = [E\Phi_{20}(B^k) + P\Phi_{11}(B^k) + R\Phi_{21}(B^k) + Z\Phi_{30}(B^k) +$$
$$+ L\Phi_{12}(B^k) + N\Phi_{31}(B^k) + W\Phi_{22}(B^k) + K\Phi_{13}(B^k) + H\Phi_{10}(B^k)]\,a_1 +$$
$$+ [E\Phi_{02}(B^k) + P\Phi_{11}(B^k) + R\Phi_{12}(B^k) + Z\Phi_{21}(B^k) + L\Phi_{03}(B^k) +$$
$$+ N\Phi_{22}(B^k) + W\Phi_{13}(B^k) + K\Phi_{04}(B^k) + H\Phi_{01}(B^k)]\,a_2 + [E\Phi_{30}(B^k) +$$
$$+ P\Phi_{21}(B^k) + R\Phi_{31}(B^k) + Z\Phi_{40}(B^k) + L\Phi_{22}(B^k) + N\Phi_{41}(B^k) +$$
$$+ W\Phi_{32}(B^k) + K\Phi_{23}(B^k) + H\Phi_{20}(B^k)]\,a_3 + [E\Phi_{21}(B^k) + P\Phi_{12}(B^k) +$$
$$+ R\Phi_{22}(B^k) + Z\Phi_{31}(B^k) + L\Phi_{13}(B^k) + N\Phi_{32}(B^k) + W\Phi_{23}(B^k) +$$
$$+ K\Phi_{14}(B^k) + H\Phi_{11}(B^k)]\,a_4 + [E\Phi_{12}(B^k) + P\Phi_{03}(B^k) + R\Phi_{13}(B^k) +$$
$$+ Z\Phi_{22}(B^k) + L\Phi_{04}(B^k) + N\Phi_{23}(B^k) + W\Phi_{14}(B^k) + K\Phi_{05}(B^k) +$$
$$+ H\Phi_{02}(B^k)]\,a_5 + [E\Phi_{22}(B^k) + P\Phi_{13}(B^k) + R\Phi_{23}(B^k) + Z\Phi_{32}(B^k) +$$
$$+ L\Phi_{14}(B^k) + N\Phi_{33}(B^k) + W\Phi_{24}(B^k) + K\Phi_{15}(B^k) + H\Phi_{12}(B^k)]\,a_6.$$

Вместо переменных E, P, R, ..., H подставляем полученные ранее соответствующие им значения. Параметры a_i ($i = 1, 2, \ldots, 6$) преобразования (9) находим при решении системы линейных уравнений (13).

Аналогично, используя функционалы $\Phi_{01}(B^k)$, получаем систему линейных уравнений относительно неизвестных $b_j (j = 1, 2, \ldots, 6)$.

Таким образом, введение функционалов вида (3), (4) позволило определить неизвестные параметры заданных нелинейных преобразований путем решения соответствующих систем линейных алгебраических уравнений.

Список литературы: 1. *Путятин Е. П., Долженкова Т. Г.* Вопросы нормализации нелинейных преобразований. Сообщение 1.— Проблемы бионики, 1980, вып. 24, с. 111—115. 2. *Путятин Е. П, Долженкова Т. Г.* Вопросы нормализации нелинейных преобразований. Сообщение 2.— Проблемы бионики, 1980, вып. 24, с. 116—120. 3. *Путятин Е. П., Долженкова Т. Г.* Нормализация нелинейных преобразований. Сообщение 1.— См. статью в настоящем сборнике. 4. *Лобанов А. Н.* Аналитическая фотограмметрия.— М.: Недра, 1972.—180 с.

Поступила 16 октября 1979 г.

УДК 62.506.2.

В. А. ЛОВИЦКИЙ, канд. техн. наук, *Н. В. МОРОЗОВА, Е. И. ТИШИНА*

К ВОПРОСУ ОБ ИНЖЕНЕРНОЙ ПОЭТИКЕ

Введение в эксплуатацию мощных ЭВМ третьего поколения позволило подойти к созданию человеко-машинных систем, призванных быть «интеллектуальными» советчиками человека. «Советы» таких искусственных систем должны быть не только естественно-языковыми, но и, по возможности, нести как можно меньше синтаксической информации при сохранении семантики «совета». Известно [1], что рифмованные строчки несут меньше синтаксической информации, чем строчки прозы. Следовательно, для человеко-машинных систем, в которых человек вынужден воспринимать и перерабатывать большой объем информации, выдаваемых ЭВМ, проблема ее организации имеет немаловажное значение. Решением этой проблемы и занимается инженерная поэтика.

Существует несколько различных подходов к творческому процессу в искусстве. Наибольший интерес представляет так называемое пермутационное искусство [3], в котором постулируется существование атомов структуры, после чего машина порождает и исследует поле возможностей, определяемое заранее заданным алгоритмом. Под пермутацией понимается комбинирование простых, обладающих ограниченной изменчивостью элементов, открывающее колоссальное поле возможных вариаций. Таким образом, реализуется то разнообразие в единообразии, которое составляет основу свойств любого произведения искусства. При этом на первый план выступает иерархия порядков или уровней анализа. Машина, сочиняющая стихотворное произведение, может работать на уровне букв, слов, словосоче-

таний. В зависимости от этого получаются пермутационные произведения различных уровней сложности.

Исследование структуры языка естественно должно привести к учету частоты повторения целых слов. Учитывая дополнительно и то, что человек сначала учится оперировать словами, определяет их смысл, и только значительно позже он учит грамматику, изучает буквы, учится писать, приходим к выводу, что лучше всего за основу работы системы взять готовые слова с учетом связи этих слов между собой.

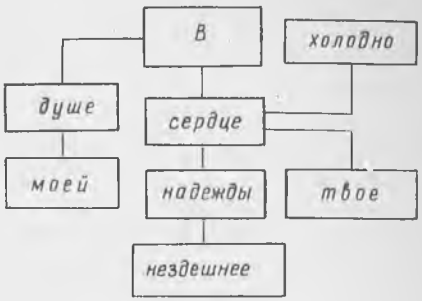


Рис. 1

Рассмотрим систему «Поэтесса», принцип работы которой основан на использовании структуры ближнего порядка [3]. Пример построения такой структуры, показывающий взаимосвязь слов в строке и возможность пересечения строк, приведен на рис. 1. Очевидно, что эта структура позволяет получать варианты строк, отличные от исходных. Общая схема алгоритма системы «Поэтесса» дана на рис. 2. Схема разбита на блоки, которые показывают поэтапную разработку системы по мере ее усложнения.

На первом этапе работы исследовались закономерности построения новой строки. Слова из вводимой строки MSMB (блок 1 (1)) последовательно заносятся в массив-словарь MVOCAB (блок 1 (2)). В результате каждое слово занумеровано. Повторяющиеся слова заносятся в MVOCAB только один раз. Каждой вводимой строке однозначно ставится в соответствие код этой строки MSTRIN (блок 1 (3)), состоящий из порядковых номеров этих слов в словаре MVOCAB. Большую роль играет взаимосвязь слов, т. е. необходимо запомнить, какое слово за каким следует или какое слово какому предшествует, определить, какие слова могут быть начальными, конечными, и конечными и начальными в строке одновременно. Для определения начального и конечного слова в строке, в MSTRIN, вводятся специальные метки. Далее, вводятся два массива MALPH и MSTRUC. Массив MALPH содержит перечень кодов всех слов по мере формирования кодов всех вводимых строк (блок 1 (4)). Повторяющиеся коды слов заносятся только один раз. Массив MSTRUC, заполнение которого происходит параллельно с записью кодов в MALPH, сохраняет последовательность слов в строке, в нем (массиве) определяется частота встречаемости слов в числовом материале. Массивы MALPH и MSTRUC симметричны, т. е. один из параметров, определяющих их размеры, совпадает. MSTRUC состоит из двух основных частей — «что

следует» и «что предшествует». В MSTRUC в столбце, соответствующем коду записываемого в MALPH слова, в части «что следует» записывается номер кода следующего за ним слова. В части «что предшествует» записывается номер кода пред-

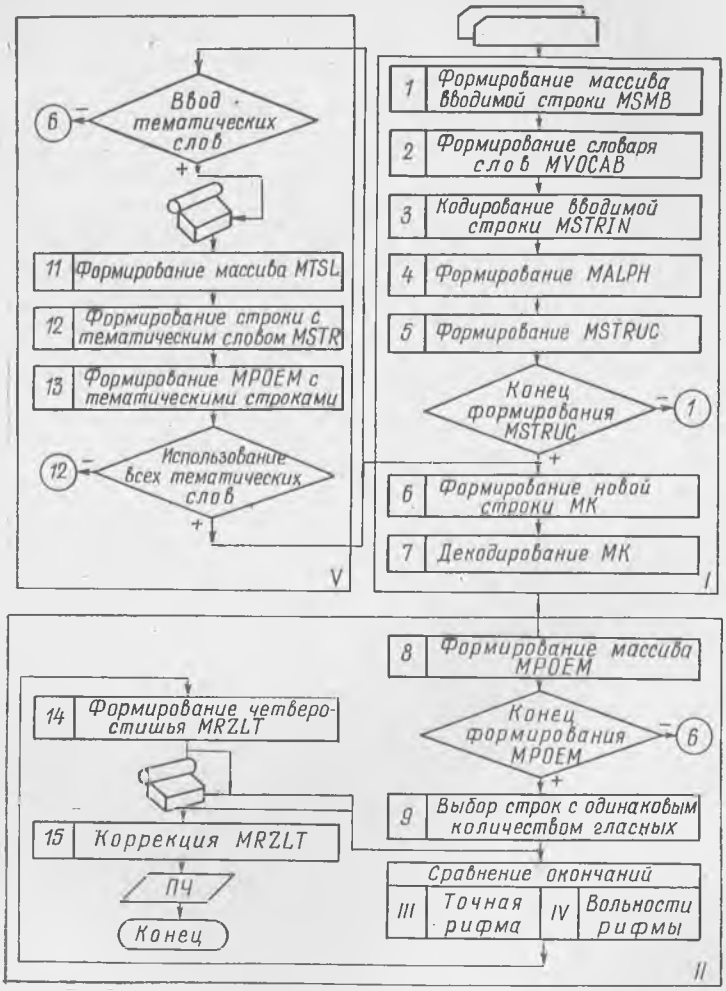


Рис. 2

шествующего слова. В процессе поступления числового материала массив MSTRUC заполняется полностью (блок 1 (5)). Построение новой строки осуществляется с помощью MSTRUC в режиме генератора и начинается с наиболее часто встречающего слова в MALPH. Если код этого слова несет одновременно и признак начала строки, то он записывается в MK. Выбор вто-

рого слова осуществляется с помощью MSTRUC. В столбце с номером, соответствующем первому слову, в части «что следует» по частоте встречаемости выбирается слово, следующее за первым. Процесс построения строки продолжается до тех пор, пока вновь выбранное слово не будет нести признака конца строки. Вновь созданная строка МК (блок 1 (6)) раскодируется (блок 1 (7)) и каждому коду в МК однозначно ставятся в соответствие слова из словаря MVOCAB. Формирование последующих строк производится аналогично. Как пример работы этого алгоритма можно привести следующие строки:

Глубокий снег в избушке отдохнуть
Повисла на жатве ночи
Сверкают в хрустальное ведро.

На этом промежуточном этапе результат представляет собой строку с определенным набором слов. Как видно из приведенного примера, совокупность этих и подобных им строк ничем не напоминает поэтическое произведение. Это объясняется тем, что поэтическая речь отличается от обыкновенной определенными приемами ритмизации речи и для получения поэтического текста необходимо учитывать внутреннюю организацию стиха. Школьная метрика различает три системы стихословия — метрическую, основанную на длительности; тоническую, построенную на ударении; силлабическую, исходящую из счета слогов [4]. Остановимся немного подробнее на силлабическом способе стихосложения, который задает первичную ритмическую структуру при построении текста в системе «Поэтесса». Размер стиха здесь определяется только количеством слогов, причем все слоги признаются равными между собой. Различий между слогами не делается никаких — ни по долготе и краткости, ни по ударности и неударности [4]. Только в школе учат, что такое ударение, а в быту ударение как бы не существует. Поскольку ударные и неударные слоги не различаются, остается только один признак — самое количество слогов. Но стих держится не только на размере, но и на выделении фразовых единиц. И в силлабическом стихосложении учитывается фразовое ударение, которое стоит в конце полустишья. Это единственное ударение, влияющее на силлабическое стихосложение.

Дополнительным приемом ритмизации речи служит расчленение произведения на строфы: если основным признаком стиха является последовательность равновеликих отрезков, создающих первичную ритмическую волну, то последовательность равновеликих групп таких отрезков, да еще упорядоченно размещенных в группе, углубляет речевую размерность [5]. Учитывая указанное выше, систему строили генерацией текстов в виде строф, каждая из которых содержит четыре стихотворных строки. Различные варианты строк, полученные на первом этапе обучения системы, содержатся в массиве МРОЕМ (блок II (8)).

Система подсчитывает и запоминает суммарное количество гласных букв в каждой строке, которые однозначно определяют число слогов в строке. Из массива МРОЕМ необходимо построить максимально возможное число четверостиший. Порядок строк с одинаковым количеством гласных (блок II (9)) выбирается по одному из трех возможных способов перестановки строк в данной строфе: (*aa*, *bb*), (*ab*, *ab*), (*ab*, *ba*), где *a* и *b* обозначают строки с различным числом гласных. Порядок строк задается заранее и при необходимости может быть изменен. Построение четверостишья (блок II (14)) по описанному алгоритму осуществляется в блоке II. Результатом работы алгоритма являются следующие два примера.

Пример 1. Блестит на охоту
Мороз и иней
Живее сладкого
Плоды дороги.

Пример 2. Ночью мороз не сдули
Который мне дороги
Мертвой дороги
Безлюдный простор.

Порядок строк в первом примере — (*ab*, *ab*), во втором — (*aa*, *bb*). Вероятно, какое-либо из двух четверостиший понравится читателю. Значит, и при данной, столь несовершенной, работе системы по моделированию сочинения стихотворных произведений может быть получен результат, в какой-то мере удовлетворяющий художественную потребность читателя [6].

Дальнейшее совершенствование системы связано с таким понятием, как рифма. Рифма — не обязательное условие русского стиха. Рифма — созвучие двух слов, стоящих в определенном месте ритмического построения стихотворения. В русском стихе рифма должна находиться на конце стиха. Именно концевые созвучия, дающие связь между двумя стихами, именуются рифмой. Следовательно, у рифмы есть два качества: ритмическая организация, потому что она (рифма) отличает концы стихов, и созвучие. Рифма — явление звуковое, а не графическое, поэтому не важно, как пишутся два разных слова, а важно, чтобы они звучали одинаково [7]. В русском языке различают несколько типов рифм, один из них так называемая точная рифма — совпадение всех звуков в окончании [7]. В системе «Поэтесса» при выделении окончаний считается достаточным взять для сравнения четыре последних символа в последнем слове каждой строки. Сравнение окончаний производится в строках, содержащих одинаковое количество слогов. Так как рифма — это всякий звуковой повтор, несущий организующую функцию в метрической композиции стихотворения, то различают три типа рифм, которые встречаются в строфе из четырех стихотворных строк [7]:

смежные (АА, ВВ), перекрестные (АВ, АВ), опоясывающие (АВ, ВА), где А и В строки с различными окончаниями последних слов.

Очевидно, что организация внутренней структуры строфы и типы рифмы в строфе совпадают. Определенное количество слогов в строке, величина и внутренняя структура строки, количество строк, рифмующиеся окончания придают стихотворной системе ритмичность. Результаты, полученные при работе системы с учетом точной рифмы (блок III), имеют уже большую близость с моделируемым объектом, чем те результаты, которые были приведены ранее:

Вдалеке я мечтала
Сухими жесткими словами
Осень солнцу отдала
Легла пушистыми коврами.

Но в русском языке существуют еще так называемые вольности рифмы, основанные на несоответствии правописания и произношения [7]. Подобными по произношению считаются согласные Д — Т, Б — П, Ж — Ш, Г — Х. Буквы А — Я, У — Ю, О — Ё обозначают одинаковый гласный звук. Исходя из этого, сравнение окончаний при формировании четверостишья осуществляется способом, несколько отличающимся от описанного ранее (блок IV). Выбор способа подбора окончаний выполняется в режиме диалога. В зависимости от полученного ответа на свой вопрос, система «Поэтесса» выбирает тот или иной способ сравнения окончаний. Следуя по второму пути выбора рифмующихся окончаний, получаем:

Летучей лаской снегопад
Сердце занято мечтами
Пусть снежинки свой заводят
Поздний вечер над прудами.

Академик А. Н. Колмогоров проанализировал соотношение запаса слов с рифмой. Оказалось, что запас в 100 слов дает тройную рифму, в 200 — четырехкратную, а в 500 слов — в изобилии дает поэтам десятикратные рифмы. Эти данные позволили определить минимальный словарный запас системы «Поэтесса».

Следующий этап работы по усовершенствованию системы — формирование четверостишья на любую заданную тему. Что позволяет человеку классифицировать литературные произведения по темам? По нашему мнению, это возможно благодаря определенному набору слов, наиболее характерных для той или иной темы. Система «Поэтесса» предусматривает генерацию четверостишья на определенную тему, которая задается набором тематических слов (блок V (11)). Ввод этих слов осуществляется в режиме диалога. Если в процессе работы какое-либо из тематических слов не было найдено в словарном запасе

MVOCAB системы, то в режиме диалога возможна замена этого слова другим, относящимся к данной теме словом (блок V). После ввода тематических слов свою работу система осуществляет следующим образом: генерируется новая строка, содержащая одно из тематических слов (блок V (12)). Эта строка заносится в массив MPOEM (блок V (13)). Процесс продолжается до тех пор, пока не будут использованы все тематические слова. Дальнейшая работа системы аналогична описанной. Как пример приведем четверостишие, составленное системой «Поэтесса» на тему «Зима»:

Мороз и иней.
Снег в любви моей
Живее сугробов таял город
Снежинки свой заводят хоровод.

По мере формирования строфы просто в режиме генератора либо в тематическом режиме осуществляется последовательный вывод их на печать. Далее, в режиме диалога возможна корректировка полученного текста при необходимости изменения какой-либо строки. В системе осуществляется подбор нескольких новых вариантов строк, согласующихся с изменяемой по числу гласных и окончанию (блок II (15)). После выбора одного из наиболее подходящих вариантов скорректированное четверостишие выдается на печать.

Полученные результаты нельзя считать окончательными. Дальнейшее совершенствование системы будет идти по пути еще большего усложнения внутренней структуры строфы, а именно: учет ударения, расстояния между ударными гласными, т. е. моделирование тонического стихосложения [7]. Это усложнение внутренней структуры позволит получать поэтические тексты любого размера — ямб, хорей, дактиль, амфибрахий, анапест. Огромные возможности улучшения работы системы «Поэтесса» открывает режим диалога. Он позволяет пополнить словарный запас системы, задать способ и размер стихосложения, контролировать систему непосредственно в процессе ее работы и при необходимости направить ее по пути, представляющему для нас наибольший интерес.

На оценку машинных результатов влияют многие факторы и среди них один из наиболее значительных — психологическая установка. Человек не может «объективно» оценивать стихи, написанные искусственной системой и сравнивать их с сочинениями поэтов, что вполне объяснимо психологически. Наибольшее различие возникает при оценке машинного творчества у представителей гуманитарного и инженерно-математического типов мышления.

Мы провели небольшой психологический эксперимент с участием учащихся студенческой группы специальности «Прикладная математика». Им было предложено два примера.

Пример 1. Я кровь ковал
Я Вас любил.
Любовь еще быть может...
Не Вас, не к вам.

Пример 2. Под бурями судьбы жестокой
Остались мне одни страданья
Плоды сердечной пустоты
Я разлюбил свои мечты.

Требовалось определить, какое из этих четверостиший написано человеком, а какое системой «Поэтесса».

В результате эксперимента 90% опрошенных отдали предпочтение второму примеру (система «Поэтесса»), а не первому (В. Сосюра [8]). Причем критерий оценки был один — «что хуже» с их точки зрения, то и написано ЭВМ. Нельзя, конечно, сказать, что это четверостишие лучше любого другого четверостишия, написанного человеком. Но ясно, что оно уже выиграло по сравнению с четверостишием среднего качества. Часто можно слышать, что машинные сочинения лишены смысла. Но что понимается под смыслом художественного произведения? Дать определение смысла, его психологическую оценку трудно. Анализируя художественное произведение, человек сопоставляет свой жизненный опыт, свои взгляды с описываемыми событиями. И если описываемым событиям человек не может найти объяснения, то он говорит, что это произведение бессмысленно. И наоборот, если предложить человеку какой-либо набор слов, то он подсознательно создает некоторый образ или ситуацию, подобную его жизненным идеалам, попытается найти какой-то смысл даже в бессмысленных на первый взгляд сочетаниях слов. Примером этого могут служить приведенные в [8] четверостишия.

При построении программного обеспечения системы «Поэтесса» использовался модульный принцип. Все модули написаны на языке Фортран-4 и Ассемблере и реализованы на ЭВМ ЕС-1050.

Программное обеспечение вместе с массивами занимает 240 К оперативной памяти ЭВМ.

Список литературы: 1. *Кондратов А. М.* Теория информации и поэтика (Энтропия ритма русской речи).— Проблемы кибернетики, 1963, вып. 9, с. 279—286. 2. *Грекова И.* Полемика и ее издержки (По поводу спора «Машина и творчество»).— Новый мир, 1973, № 7, с. 225—235. 3. *Моль А., Фукс В., Касслер М.* Искусство и ЭВМ.— М.: Мир, 1975.— 556 с. 4. *Томашевский Б. В.* Стилистика и стихосложение.— М.: Учпедгиз, Ленингр. отд-ние, 1959.—535 с. 5. *Шенгели Г. А.* Техника стиха.— М.: Госполитиздат, 1960.— 312 с. 6. *Лобанцев Ю. Л., Соколов В. В.* О влиянии художественной установки на восприятие поэзии.— В кн.: Эстетику в жизнь.— Свердловск, 1973, № 4, с. 45—50. 7. *Жирмунский В. М.* Теория стиха.— Л.: Сов. писатель, 1975.— 664 с. 8. *Соложенкина С.* У забавы кофта — ситчик.— Правда, 1979, 3 февраля, с. 3.

Поступила 22 февраля 1979 г.

УДК 62.506.2

Н. И. БОГДАНОВ, канд. техн. наук

ПРОБЛЕМНАЯ КОММУНИКАЦИЯ (ИНФОРМАЦИОННО-СЕМАНТИЧЕСКАЯ ТЕОРИЯ АНАЛИЗА СООБЩЕНИЯ)

Опыт педагогики, психологии и социологии свидетельствует о тесной связи процессов постановки и решения задач с различными сторонами человеческой деятельности.

Мы попытались наметить пути теоретического исследования этой связи на основе концепции проблемной коммуникации. Согласно этой концепции задача возникает в тех случаях, когда коммуникант не может в свернутом виде (симультанно) принять и проанализировать (или синтезировать и передать) сообщение.

Развитие теории проблемной коммуникации необходимо для совершенствования диалоговых систем [1], разработки направляющих принципов (доктрин) целесообразного функционирования интеллектуальных роботов [2] и может быть полезно для психолого-педагогических исследований.

Чтобы ввести исходные понятия, рассмотрим систему S, состоящую из соединенных каналами связи подсистем S_1, S_2, S_3, по меньшей мере одна из которых наделена интеллектом.

Система S_1, отображает S_3 и передает о ней сообщение S_2. Если система S_2 принимает сообщение и не ограничивается симультанным извлечением информации, она формирует образ сообщения и целей его развернутого анализа. Этот образ будем называть задачей, достижение описанных в ней целей — решением задачи, систему S — проблемной системой, S_1, S_2 — задающей и решающей системами, S_3 — объектом задачи. Взаимодействие подсистем S_1, S_2, S_3 при постановке и решении задач и представляет сущность проблемной коммуникации (возникающие при этом совокупности задач могут использоваться другими проблемными системами в качестве задачников).

Обозначив единицей наличие отношения тождественности, соответственно, между системами S_1 и S_2, S_1 и S_3, S_2 и S_3, а отрицание этого отношения нулем, получим пять основных классов проблемной коммуникации: 000, 001, 010, 100 и 111. Класс 000 порождает учебные и управленческие задачи, 001 — задачи рефлексивного управления и воспитания, 010 — задачи диагностики и обучения на личном примере, 100 — гносеологические и конструкторские задачи, класс III — задачи самоанализа и формирования целесообразного поведения.

Проблемная коммуникация классов 100 и 111 является автокоммуникацией, т. е. протекает как внутренний диалог у индивидуума, объединяющего в своем интеллекте задающую и решающую системы. Это позволяет ввести отношение рекурсии проблемных систем, т. е. описать ситуацию, при которой одна из подсистем проблемной системы сама является проблемной системой. В частности, если в проблемной системе с коммуникацией класса 000 решающая система сама является проблемной системой с коммуникацией класса 100, то она может ставить перед собой задачи относительно любых, в том числе и непроблемных сообщений системы S_1.

Таким образом, как целенаправленная проблемная коммуникация, основанная на сознательно формируемых потоках заданий (задачниках), являющихся средством рефлексивного управления, так и ситуационная проблемная коммуникация, возникающая из-за неопределенности ситуации или несоответствия знаний и иных характеристик коммуникантов, должны изучаться на общей теоретико-информационной основе.

Возможности обобщенного теоретико-информационного подхода рассмотрим на примере информационно-семантического анализа сообщения.

В соответствии с гомеостатическими процессами задачи, решаемые коммуникантами, в этом случае можно разделить на задачи извлечения из сообщения содержащейся в нем информации об объекте (познавательные, понимания); задачи получения на основе сообщения новой информации об объекте, не содержащейся в сообщении (творческие).

Чтобы описать решения задач, необходимо ввести некоторую меру знаний решающей системы. Ниже постулируется ее существование в виде, предложенном в [3].

Согласно [3] семантическая информация измеряется относительно запаса знаний решающей системы о внешнем мире. Этот запас описывается допускающей расширение нежесткой моделью, называемой тезаурусом.

Это понятие не будем связывать процедурой построения тезауруса, принятой в [3], так как достаточно допущения.

Д1. Существует мера T запаса знаний решающей системы. (Для удобства ссылок постулируемые допущения обозначаются буквой Д, утверждения, которые из них можно получить — буквой У, замечания — буквами Зм. После этих букв ставятся порядковые номера).

Вследствие индивидуальных различий в способностях и умениях две решающие системы, приняв одно и то же сообщение, пополнят свои (вначале одинаковые) запасы знаний в различной степени.

Д2. Существует мера k эффективности накопления знаний решающей системы на основе информации, извлеченной из сообщения, при этом $k \in C$, где множество $C = (k : 0 < k < \infty)$.

Д3. Существует мера I семантической информации, содержащейся в сообщении, причем

$$I(k,\ T) = \begin{cases} 0 & \text{при } T < T_{\text{н}}, \\ f(k,\ T) & \text{при } T_{\text{н}} \leqslant T \leqslant T_{\text{к}}, \\ 0 & \text{при } T_{\text{к}} < T, \end{cases} \tag{1}$$

где функция $f(k,\ T)$ является ограниченной и финитной, а $T_{\text{н}}$, $T_{\text{к}}$ — начальное и конечное значение тезауруса. Прием сообщения происходит в том случае, если тезаурус решающей системы T принадлежит множеству $B = (T : T_{\text{н}} \leqslant T \leqslant T_{\text{к}})$. Если $T < T_{\text{н}}$ решающая система не в состоянии понять сообщение, если $T > T_{\text{к}}$ она «насыщена» знаниями и сообщение не может изменить ее тезаурус (в дальнейшем $T_{\text{н}}$, $T_{\text{к}}$ будем называть начальным и конечным тезаурусом, соответственно, а множество B — базой сообщения).

Д4. Если решающая система имеет исходный тезаурус $T_0 \in B$, то после приема сообщения приращение тезауруса равно

$$T - T_0 = f(k,\ T_0). \tag{2}$$

Допущения Д1 и Д3, Д4 вытекают из [3], а Д2 предполагает возможность измерения «интеллектуальности» решающей системы.

У1. Если выполняются Д1 ÷ Д4, то повторное обращение к сообщению приводит к итерационному процессу накопления знаний решающей системой:

$$T_n = \varphi(k_{n-1},\ T_{n-1}), \tag{3}$$

где $\varphi(\ \ )$ — функция, определяемая мерой семантической информации сообщения и характером извлечения информации

Зм1. Итерационные модели обучения широко известны [4], однако У1 показывает, что процесс (3) введен не эмпирически, а вытекает из принятых допущений.

Зм2. Проблемная коммуникация всегда является процессом обучения.

Зм3. Повторное обращение к сообщению — достаточный отличительный признак проблемной коммуникации: решающая система принимает сообщение как задачу (проблемирует сообщение). При непроблемной коммуникации решающая система удовлетворяется информацией, полученной при приеме сообщения, возможно, не исчерпав содержание сообщения.

Рассмотрим итерационный процесс

$$T_n = T_{n-1} + f(k,\ T_{n-1}), \tag{4}$$

согласно Д3 и Д4 зависимость (4) справедлива при исходном тезаурусе решающей системы T_0:

$$T_1 = T_0 + f(k,\ T_0) \tag{5}$$

Предположим далее, что справедливо Д5. Решающая система с некоторым исходным тезаурусом T_{n-1}, полученным в результате n-кратного обращения к данному сообщению, приобретает при следующем обращении к этому сообщению такой же объем знаний, как и в том случае, если бы она с исходным тезаурусом $T_0 = T_{n-1}$ обратилась к сообщению впервые, т. е.

$$T_{n-1} + f(k,\ T_{n-1}) = T_0 + f(k,\ T_0) \tag{6}$$

(«марковское» свойство сообщения).

Тогда зависимость (4) справедлива для любого $T \in B$ и, следовательно, по индукции формула (4) определяет итерационный процесс — частный случай процесса (3).

Однако не менее обосновано и допущение, обратное Д5. В этом случае можно полагать, что при повторном обращении к сообщению извлекаются знания, характеризуемые приращением меры семантической информации:

$$T_n = T_{n-1} + f(k,\ T_{n-1}) - f(k,\ T_{n-2}). \tag{7}$$

Тогда итерационный процесс накопления знаний решающей системой имеет вид

$$T_n = \begin{cases} T_0 + f(k,\ T_{n-1} & \text{при } T_0 + f(k,\ T_{n-1}) > T_{n-1}, \\ T_{n-1} & \text{при } T_0 + f(k,\ T_{n-1}) \leqslant T_{n-1}. \end{cases} \tag{8}$$

Этот итерационный процесс также является частным случаем процесса (3).

Таким образом У1 справедливо не только при выполнении Д5, имплицитно содержащемся в Д1 ÷ Д4, но и при его нарушении.

Зм4. Использование дополнительных сообщений (например, ответов на вопросы) может быть описано специальным тезаурусным графом (тетрафом), отображающим не только переходы между сообщениями и их анализ, но и процесс пополнения знаний. Вводя критерии эквивалентности тетрафов, можно показать, что У1 охватывает и этот случай, если ввести переменный коэффициент эффективности накопления знаний:

$$T_n = \varphi(k_{n-1},\ T_{n-1}). \tag{9}$$

Зм5. Зависимость (9), как и зависимости (4) и (8), опирается на Д1 ÷ Д4. Учет изменений мотивации, памяти, способности прогнозировать и поведения в условиях сложного (например, конфликтного) взаимодействия потребовал бы допущений за рамками информационно-семантической теории и здесь не рассматривается.

Зм6. Для выделения общих закономерностей далее используется детерминированное описание на основе некоторых понятий функционального анализа [5].

От разрешимости задач анализа сообщения зависит не только объем приобретаемых знаний, но и устойчивость коммуника-

ции, что особенно важно, если для понимания последующих сообщений надо заполнять пробелы в знаниях, решая творческие задачи относительно предыдущих.

В связи с этим рассмотрим общие условия разрешимости задач анализа сообщений. Они даются утверждениями У2÷У7.

У2. Пусть а) на ограниченном замкнутом выпуклом множестве $G \in B$ определен оператор $\varphi(k, T)$, допускающий представление $\varphi(k, T) = \psi_1(k, T) + \psi_2(k, T)$, где оператор $\psi_1(k, T)$ вполне непрерывен, а оператор $\psi_2(k, T)$ удовлетворяет условию $\forall T_i, T_j \in G (D(\varphi(k_i, T_i), \varphi(k_j, T_j)) < LD(T_i, T_j); 0 < L < 1)$, в котором D метрика: $\forall T_i, T_j, T_m \in G ((D(T_i, T_j) \geqslant 0); (D(T_i, T_j) = 0, T_i = T_j); (D(T_i, T_j) \leqslant D(T_i, T_m) + D(T_m, T_j)))$; б) выполнено условие $\psi_1(k_i, T_i) + \psi_1(k_j, T_j) \in G, T_i, T_j \in G$.

Тогда для неразрешимости творческой задачи достаточно, чтобы хотя бы однажды итерационный процесс анализа сообщения $T_n = \varphi(k_{n-1}, T_{n-1})$, $T_0 \in B$ привел к значению $T \in G$.

Зм7. Этот результат следует из обобщения принципа Шаудера и сжатых отображений [5]: так как $G \in B$, то при $T_n \in G$ итерации сходятся к тезаурусу $T_R < T_к$, т. е. знания решающей системы не выходит за пределы сообщения. У2 дает достаточные условия неразрешимости творческих задач. В частности, при деградации тезауруса (от забывания, дезинформации, заблуждений) итерационные процессы могут зацикливаться или расходиться завершаясь при $T_R < T_0$. Однако их рассмотрение требует дополнительных допущений.

У3. Пусть на множестве B определен оператор $\varphi(k, T)$ и $\forall T \in B, k \in C(0 < \varphi(k, T) < \infty)$; $\forall T \in B, k \in C(\varphi(k, T) > T)$. Тогда итерационный процесс анализа сообщения $T_n = \varphi(k_{n-1}, T_{n-1})$, $T_0 \in B$ заканчивается при $T_R > T_k$, т. е. может привести к решению творческой задачи.

Зм8. В У3 приведены требования к характеристикам коммуниканта и сообщения, выполнения которых достаточно для решения познавательных и некоторых творческих задач.

Зм9. Известно, что при решении задач этапы накопления информации чередуются с этапами осознания нового знания (инсайт). Из У1 следует, что такой характер этого процесса естественно вытекает из Д1÷Д4.

У4. Решение творческих задач возможно только при дискретном (квантовом) процессе накопления знаний.

У5. Пусть а) определено множество G, на котором $\varphi(k, T)$ удовлетворяет условиям У2; введена мера решения познавательной задачи $T_R(1 - \varepsilon) T_к$ ($(1 - \varepsilon)$ — заданная степень извлечения информации); в) определено множество $R_1 = (T : (1 - \varepsilon) T_к \leqslant T \leqslant T_к$; г) определено множество $R_2 = (T : T = \varphi(k, T))$. Тогда для решения познавательной задачи достаточно, чтобы $R_2 \in R_1 \in G$ и хотя бы одно значение T, определяемое итерационным процессом $T_n = \varphi(k, T_{n-1})$, $T_0 \in B$ принадлежало множеству G.

Зм9. Аналогично учету степени извлечения информации в У5 могут быть учтены и другие требования к процессу анализа, сообщения, например ограничения на ресурсы решающей системы (времени, мотивации, затрат и т. п.). Поэтому соответствующие утверждения, развивающие У1—У4, здесь не приводятся.

Общий характер У1 ÷ У5 скрывает конкретные особенности анализа сообщений, важные для их практического использования и оценки адекватности эмпирическим закономерностям. Чтобы создать опору для интерпретации, были исследованы некоторые простейшие информационные характеристики сообщений. Результаты исследования одной из них («треугольной») приведены в У6 и У7.

У6. Пусть а) выполнены Д1 ÷ Д5 и мера семантической информации имеет вид

$$I(k,\ T) = \begin{cases} 0 & \text{при } T < T_н,\ T > T_к, \\ k\beta_1 (T - T_н) & \text{при } T_н \leqslant T \leqslant T_м, \\ k\,(I_м - \beta_2 (T - T_м) & \text{при } T_м < T \leqslant T_k, \end{cases} \qquad (10)$$

где $I_м$, $T_м$ — максимальное значение функции $I(k,\ T)$ и соответствующее ему значение T;

б) процесс анализа сообщения описывается формулой (4). Тогда разрешимость творческих и познавательных задач определяется условиями, приведенными в табл. 1.

Зм10. Необходимым условием решения творческой задачи является $T_R > T_к$.

Зм11. При условии решения 3 табл. 1 тезаурус решения T_R может иметь минимум по k, а увеличение T_0 уменьшает T. Этим в частности подтверждается бытующее мнение о «вредности» чрезмерной эрудиции. Одновременно становится очевидной относительность этого мнения.

Зм12. Сопоставление условий 4, 5, 6, 7 с условиями 1, 2, 3 табл. 1 свидетельствует, что снижение интеллектуальности приводит к качественному скачку: творческие задачи становятся неразрешимыми, а полное решение познавательных задач требует бесконечного числа шагов.

У7. Пусть выполнены Д1 ÷ Д4, мера семантической информации описывается зависимостью (12), а процесс анализа сообщения формулой (10). Тогда разрешимость творческих и познавательных задач определяется условиями, приведенными в табл. 2.

Зм13. В У7 описываются сообщения, не имеющие «марковского» свойства. При условиях 1 и 2 табл. 2 тезаурус решения T_R может быть как больше, так и меньше $T_к$, а при условии 3— только меньше.

Сопоставление табл. 1 и 2 приводит к

Таблица 1

№	Условия решения	Оценка тезауруса решения T_R	Число шагов решения n
1	$k\beta_1 > t_k$ $T_н < T_0 < T_м$ $t_k = \frac{T_k - T_н}{T_0 - T_н} - 1$	$T_k < T_R < (T_м - T_k) + \beta_1 k (T_м - T_н) - (T_k - T_н)$	1
2	$k\beta_2 > 1$ $T_м < T_0 < T_k$	$T_k < T_R < (T_k - T_0) \times (\beta_2 k - 1) + T_k$	1
3	$t_k > k\beta_1 > t_м$ $k\beta_2 > 1$ $T_н < T_0 < T_м$ $t_м = \frac{T_м - T_н}{T_0 - T_н} - 1$	$T_k < T_R < ((T_k - T_н) - (T_0 - T_н)(1 - \beta_1 k)) \times (\beta_2 k - 1) + T_k$	2
4	$k\beta_1 < t_м$ $k\beta_2 > 1$ $T_н < T_0 < T_м$	$T_k < T_R < (T_м - T_k) \times (1 - \beta_2 k) + T_k$	$[a] + \operatorname{sgn}(a - [a]) + 1$ $\left(a = \frac{\ln(T_м - T_н) - \ln(T_0 - T_k)}{\ln(1 - \beta_1 k)}\right.$, $\left.[a] — целая часть\ a\right)$
5	$t_k > k\beta_1 > t_м$ $k\beta_2 < 1$ $T_н < T_0 < T_м$	$T_R = (1 - \varepsilon) T_k < T_k$ $T_R = T_k$	$[a_1] + \operatorname{sgn}(a_1 - [a_1]) + 1$ $a_1 = \frac{\ln(1-\varepsilon) T_k - \ln(T_k - T_0)}{\ln(1 - \beta_2 k)}$ ∞
6	$k\beta_1 < t_м$ $k\beta_2 < 1$ $T_н < T_0 < T_м$	$T_R = (1 - \varepsilon) T_k < T_k$ $T_R = T_k$	$[a] + [a_1] + \operatorname{sgn}(a - [a]) + \operatorname{sgn}(a_1 - [a_1])$ ∞
7	$k\beta_2 < 1$ $T_м < T_0 < T_k$	$T_R = (1 - \varepsilon) T_k < T_k$ $T_R = T_k$	$[a_1] + \operatorname{sgn}(a_1 - [a_1])$ ∞
8	$\beta_2 = \infty$ $k\beta_1 < t_k$ $T_k = T_м$ $T_н < T_0 < T_k$	$T_k < T_R < (T_0 - T_н)(1 - \beta_1 k)^n + T_н$	$[a_2] + \operatorname{sgn}(a_2 - [a_2])$ $\left(a_2 = \frac{\ln(T_k - T_н) - \ln(T_0 - T_н)}{\ln(1 - \beta_1 k)}\right.$,

Таблица 2

№	Условия решения	Оценка тезауруса решения T_R	Число шагов решения n
1	$k\beta_1 > t_0$ $T_н < T_0 < T_м$ $t_0 \dfrac{T_k - T_0}{T_0 - T_н}$	$T_k \leqq T_R \leqslant T_0 + k\beta_1 \times$ $\times (T_0 - T_н)$	1
2	$T_н < T_0 < T_м$	$T_k \leqq T_R \leqslant T_0 +$ $+ k\,(J_м - \beta_2(T_0 - T_м))$	1
3	$k\beta_1 < t_0$ $T_н < T_0 < T_м$	$T_R < T_0 + (T_0 - T_н) \times$ $\times \dfrac{(\beta_1 k)^n - 1}{\beta_1 k + 1} < T_k$	$[a_3] + \operatorname{sgn}(a_3 - [a_3])$ $\left(a_3 = \dfrac{\ln (T_м - T_0)\,(\beta_1 k - 1)}{\ln \beta_1 k} - \right.$ $\left. - \dfrac{\ln (T_0 - T_н)}{\ln \beta_1 k}\right)$
4	$\beta_2 = \infty$ $T_н < T_0 < T_k$ $T_k = T_м$	$T_k < T_R < T_0 +$ $+ (T_0 + T_н) \dfrac{(\beta_1 k)^n - 1}{\beta_1 k + 1}$	$[a_4] + \operatorname{sgn}(a_4 - [a_4])$ $\left(a_4 = \dfrac{\ln (T_k - T_0)\,(\beta_1 k - 1)}{\ln \beta_1 k} - \right.$ $\left. - \dfrac{\ln (T_0 - T_н)}{\ln \beta_1 k}\right)$

У8. Сообщение можно сформировать так, что разрешимость творческих задач не будет зависеть от значения коэффициента k («интеллектуальности») решающей системы.

У9. Сообщение можно сформировать так, что процесс решения познавательных и творческих задач будет одношаговым.

Зм14. Одношаговый процесс анализа сообщения обеспечивает идеальную (непроблемную) коммуникацию — из сообщения извлекается сразу вся информация (непроблемная коммуникация второго рода). Этим она отличается от описанной в Зм3 непроблемной коммуникации первого рода. Условия одношаговой коммуникации дается в У9.

У9. Пусть на множестве B определены операторы $\varphi(k, T)$, обладающие свойствами $\forall T \in B$, $\forall k \in C\,(0 < \varphi(k, T) < \infty)$; $\forall T \in B$, $\forall k \in C\,(\varphi(k, T) > T_к)$. Тогда итерационный процесс анализа сообщения $T_n = \varphi(k_{n-1}, T_{n-1})$, $T_0 \in B$ может привести к одношаговому решению творческой задачи.

Зм15. Для выполнения У9 требуется, чтобы задающая система имела весьма полные знания о решающей системе. При неопределенности более полезно У10, обобщающее условия 8 табл. 1 и условия 4 табл. 2.

У10. Если выполнены Д1—Д4, отсутствуют ограничения на ресурсы решающей системы и мера семантической информации непрерывна и строго возрастает всюду на множестве B, то при $T_0 \in B$ итерационные процессы (4) и (10) могут привести к решению творческих задач за конечное число шагов.

Зм16. По У10 можно сформулировать доктрину проблемной коммуникации: сообщение следует формировать так, чтобы с ростом знаний решающей системы количество извлекаемой из него информации увеличивалось, иначе: чем больше знаешь — тем больше извлекаешь.

Таким образом, можно полагать, что предлагаемая концепция проблемной коммуникации позволила дедуктивно получить общие закономерности и конструктивные рекомендации по организации эффективного диалога и постановки задач в системах с искусственным интеллектом, а также создать предпосылки для исследования проблемных ситуаций в различных сферах человеческого общения.

Список литературы: 1. Человек и вычислительная техника / В. М. Глушков, А. М. Довгялло, З. Л. Рабинович и др.— Киев: Наук. думка, 1971.—295 с. 2. *Богданов Н. И.* Динамика целенаправленного функционирования землеройного робота.— В кн.: VI Всесоюз. симпозиум по теории и принципам устройства роботов и манипуляторов. Секция II, Тольятти, 1976, с. 78—80. 3. *Шрейдер Ю. А.* О семантических аспектах теории информации.— В кн.: Информация и кибернетика.— М.: Сов. радио, 1967.— 201 с. 4. *Цыпкин Я. З.* Основы теории обучающихся систем.— М.: Наука, 1970.— 252 с. 5. *Цыпкин Я. З.* Функциональный анализ.— М.: Наука, 1964.—470 с.

Поступила 28 марта 1979 г.

УДК 62.506.2

Л. Н. БЕЛЯЕВА, Е. М. ЛУКЬЯНОВА

ПРОБЛЕМЫ РАЗРАБОТКИ МНОГОЦЕЛЕВОГО АВТОМАТИЧЕСКОГО РУССКОГО СЛОВАРЯ

СООБЩЕНИЕ 1

Необходимость в разработке и широком промышленном внедрении лингвистических автоматов — систем автоматической переработки текста уже ни у кого не вызывает сомнений, однако высокому уровню развития кибернетики и информационной промышленности мы не можем до сих пор поставить в соответствие лингвистические разработки, реализующие все те богатства, которые были накоплены традиционной лингвистикой за многие века ее существования.

Опыт промышленной реализации МП у нас и за рубежом [1, 2] показал, что для создания первых практических систем «грубого» МП нужна мощная информационная база, которая содержит систему автоматических словарей с необходимой лингвистической информацией и управляется комплексом обслуживающих программ. Эта база даст наибольший эффект тогда, когда будет иметь универсальное применение, т. е. использоваться как для систем МП, так и вообще для нужд систем автоматизированной переработки текста в качестве основы лингвистического обеспечения. Такая комплексная система позволит оптимальным образом обрабатывать всю поступающую информацию и уже накопленные фонды и станет важным орудием в развитии ведущих областей науки в эпоху научно-технической революции.

Поскольку общей и наиболее важной частью любых систем переработки информации является словарь, то его и следует выбрать в качестве основной структурной единицы.

Как известно, основная часть семантической информации, передаваемой русским текстом, заложена в его лексике. Поэтому для решения задач, связанных с лингвистическим обеспечением (библиографический поиск, атрибуция, аннотирование, реферирование и машинный перевод документов), с лингвостатистическими и лексикографическими исследованиями (составление словарей по конкретным тематикам, получение частотных словарей канонических форм слов и т. д.) и более широко — с созданием алгоритмов искусственного интеллекта (диалоговые системы человек — машина), необходим универсальный машинный словарь, охватывающий всю современную лексику русского языка, включая и его специальную терминологию. Будем называть такой словарь многоцелевым автоматическим русским словарем (МАРС).

Идея универсальности, закладываемая в МАРС, определяется тем, что при всех видах переработки текста — вероятностном,

детерминистском и тезаурусном распознавании смысла документа, машинном переводе, а также при таких видах лексикографического анализа, как составление тезаурусов и идеографических словарей, в процессе преподавания иностранных языков и русского языка требуется обычно одна и та же лексико-семантическая, грамматическая и лексикографическая (отсылочная) информация.

Таким образом, современная инженерная лингвистика должна предоставить в распоряжение информатики, вычислительной техники, лингвистики и дидактики такой автоматический словарь, который был бы кладезем всех необходимых сведений о русском языке. Рассмотрим, какой должна быть структура такого словаря и каков тот объем информации, который он должен содержать.

Основным элементом МАРС является словарная статья, содержащая в себе всю информацию, характеризующую данную лексическую единицу и прогнозирующую ее поведение в связном тексте.

Поскольку МАРС — универсальный словарь, он описывает все многообразие слов русского языка, поэтому каждая лексическая единица должна характеризоваться всей совокупностью структурных признаков, а словарь в целом должен задавать всю систему лексики языка. Набор задач, решению которых должен служить МАРС, определяет минимум информации, включенный в каждую словарную статью.

При разработке систем характеристик лексических единиц следует разделять универсальные и идиоэтнические функции каждой языковой формы [3]. Универсальность МАРС предполагает возможность его использования при переводе на русский язык текста любого языка. Разделение универсальных и идиоэтических функций исключает дублирование универсальных характеристик для единиц входного языка и позволяет ограничить описание этих лексических единиц их идиоэтическими характеристиками. В соответствии с этим словарная статья должна содержать морфолого-синтаксическую и фразеологическую характеристику лексической единицы, код того подъязыка (подъязыков), в которых используется данное слово, указание на его вхождение в эталонные списки, коды родо-видовых и ассоциативных связей, а также семантические коды (см. ниже).

Естественно, что комбинаторика указанных характеристик ограничена, поэтому нецелесообразно повторять в каждой статье полный их перечень. Разумнее свести их в списочные структуры и определить наиболее рациональный способ их размещения, обеспечивающий быстроту выборки нужной словоформы и минимальный объем памяти ЭВМ.

Морфология в МАРС'е. Исходя из указанных выше требований, выберем основной элемент словаря. Возможности выбора у нас ограничены: словоформа, каноническая форма или основа.

Сразу оговорим, что использование традиционных канонических форм или традиционных основ связано с созданием очень сложных и поэтому ненадежных морфологических алгоритмов преобразования цепочек на стыках морф, перегруппировки букв, обработки чередований, элизий и стяжения. То есть русский язык как язык флективный для представления в автоматическом словаре очень громоздок. Поэтому приходится прибегнуть к искусственному преобразованию морфологии: к использованию изолирующей или агглютинативной машинной морфологии. В первом случае единицами словаря являются конкретные словоформы, т. е. в словаре должны быть словарные статьи отдельно для единиц **машина, машины, машине, машину, машиной** и т. д. Этот прием удобен благодаря простоте «изолирующей» морфологии, но он приводит к неоправданному расширению словаря и к увеличению времени поиска (большинство слов в русском языке являются изменяемыми, а количество форм колеблется от 12 у существительных до 188 — у глаголов в случае собственно видовой корреляции).

Чтобы избежать перегрузки памяти вычислительной машины, а также чтобы последовательно выдержать предъявленное требование универсальности, будем использовать в нашем словаре агглютинативную морфологию. При таком подходе в каждом слове должны быть выделены машинная основа и машинное окончание.

Разбиение словоформы на машинную основу и агглютинирование к ней окончания (точнее — постфикс или цепочку постфиксов) осуществляется формально: машинной основой считается последовательность букв от начала словоформы, которая является общей для всех словоформ, входящих в данную парадигму. Цепочка букв, оставшаяся после выделения машинной основы, рассматривается как окончание. Такая машинная основа не всегда совпадает с традиционной.

Например, машинная основа существительного **суффикс** равна самому слову в его канонической форме и совпадает с традиционной, а машинная основа глагола **выписать** не совпадает с традиционной **выпис-**, и равна последовательности букв **выпи** (ср. словоформу **выпишет**).

Порождение каждой словоформы осуществляется путем агглютинации машинных постфиксов, которые также могут не соответствовать традиционным окончаниям. При таком подходе наше определение машинного окончания как форма особого вида не противоречит традиционному представлению о морфе как о минимальном линейно выделимом и регулярно повторяющемся (или оставшимся после повторяющейся части) сегменте слова, обладающем структурной функцией и имеющем постоянный графический облик [4—6].

Возможные последовательности окончаний и типы машинных основ определены на основе анализа всей лексики, приведенной

в «Обратном словаре русского языка» [7]. В результате этого анализа создан набор типовых окончаний. Исходя из этого, в зависимости от типа словоизменений для каждой машинной основы мы определяем ее морфологический код: номер типовой парадигмы, характеризующей все словоформы, которые могут быть образованы от данной основы (см. таблицу).

Для обеспечения работы систем информационного поиска и машинного перевода каждая статья должна кроме машинной основы содержать лексико-грамматическую информацию, определяющую грамматический класс слова и все категории, которыми слово характеризуется внутри каждого класса.

К грамматическим категориям русского языка как языка синтетического строя, в противоположность аналитическому, относятся синтетико-морфологические категории падежа, числа, грамматического класса или рода, вида и т. д.

Значения всех этих категорий для каждой лексической единицы отражены в ее словарной статье в виде специального кода. В соответствии с этим каждая машинная основа получает лексико-грамматическую характеристику, в основном соответствующую стандартной схеме, разработанной в группе «Статистика речи» для русских лексических единиц [8]. Эта схема включает 9 грамматических классов — существительные, прилагательные, числительные, местоимения, глаголы (вместе с деепричастиями и причастиями, которые склоняются по моделям, предусмотренным для прилагательных), наречия, предлоги, союзы, частицы, среди которых первые пять дают изменяемые формы, а последние четыре включают неизменяемые слова. Поскольку количественные числительные и обобщенно-предметные местоимения изменяются как существительные, а порядковые числительные, обобщенно-качественные и обобщенно-количественные местоимения склоняются как прилагательные [9], то все машинные окончания целесообразно сгруппировать в три множества: множество окончаний прилагательных, существительных, глаголов. Чтобы унифицировать описание лексемы, в МАРС введены два дополнительных грамматических понятия: нулевая парадигма и нулевая основа [10].

Нулевая парадигма приписывается несклоняемым существительным, а также наречиям, предлогам, союзам и частицам. В этих случаях машинная основа равна самому слову. Понятие нулевой основы используется при представлении слов с супплетивной парадигмой (ср. **он, его, ему, им...; идти, иду..., шел...**). В этом случае считается, что словарная статья содержит нулевую машинную основу, к которой присоединяются машинные окончания, причем в качестве последних выступают супплетивные формы типа **он, его, ..., идти, шел** и т. д.

Пользуясь лексико-грамматическими характеристиками, можно приводить словоформу к каноническому виду и получать конкретную **словоформу**. Выбор окончаний к машинной основе

в рамках МП производится в соответствии с лексико-грамматической характеристикой, уточняемой в результате анализа иноязычного текста. Номер типовой парадигмы, приписанный основе, дает нам номер записи в соответствующем массиве окончаний, по значению кода определяется номер поля в парадигме (см. схему). Сочетание этих двух параметров определяет конкретную машинную флексию, которая присоединяется к основе. При информационном поиске происходит обратный процесс: сличение машинной основы из словарной статьи и словоформы, которая приводится к канонической форме. При совпадении основы с частью слова и совпадении выделившейся части словоформы с одним из машинных окончаний, характеризующих данную основу, словоформа считается отождествленной. Порождение канонической формы происходит путем присоединения к основе первого машинного окончания из записи в массиве окончаний.

Однако, проблемы автоматической переработки текста не ограничиваются задачами лексического перевода для системы МП. Поэтому лексико-грамматические характеристики, приписываемые каждой единице нашего словаря, не характеризуют полностью лексическую единицу в общей системе лексики языка. Каждая лексическая единица входит в сложную систему семантических парадигматических рядов, которые образуются многозначными и синонимическими языковыми формами.

Семантика в МАРС'е. Семантическая теория может описывать отношения на трех уровнях: на уровне слов, групп слов, представляющих одну синтаксическую структуру и на уровне расширенного контекста, объем которого в каждом конкретном случае определяется условиями использования. На уровне слов — значение последних определяется в виде формального описания. При этом в качестве семантических кодов могут использоваться лингвистические и экстралингвистические признаки.

Семантическая классификация предусматривается в словаре, являющемся в системе МП выходным, так как на уровне анализа предложения грамматические (синтаксические) и семантические характеристики могут быть разделены. В системе, предполагающей двуязычный словарь, семантические характеристики входных единиц задаются семантическими характеристиками их эквивалентов на входном языке. За счет этого достигается максимальная универсальность системы МП по отношению к входному языку. Семантический код, приписываемый слову в рамках определенного подъязыка, должен характеризовать значение этого слова.

Количество семантических кодов, которые по своей сути являются кодами семантических классов, значительно меньше числа разных слов в языке, поскольку самым различным терминам могут быть приписаны одинаковые коды. Например,

Фрагмент массива типовых парадигм

Фрагмент массива МО

02284	S002	КАПИТАЛОВЛОЖЕНИ
02286	S004	РАСШИФРОВ
02288	S005	ПАУЗ
02290	V115	ВЫР 554
02292	S001	ОТСЧЕТ
02294	S035	УРОВ
02296	S002	НАЗНАЧЕНИ
02298	S011	РАЗЛИЧЕНИ
02300	V247	ДЕЛ 672
02302	S001	АГЕНТ
02304	S001	АНАЛИЗАТОР
02312	S007	КОРПОРАЦИ
02314	V247	ХРАН 907
02316	A002	ЯЗЫКОВ
02318	V239	ЛЕЖ 000
02320	S001	ПРОСМОТР
02322	A014	КОМПАКТ
02324	A029	ЛУЧШ

Фрагмент	массива ЛГК
2801	030022200000000004
2915	030022242000000004
2827	030022200000000003

S001	3000		А	У		ОМ
S002	3050	Е	Я	Ю	Е	ЕМ
S003	3000		А	У		ОМ
S004	3025	КА	КИ	КЕ	КУ	КОЙ
S3005	3025	А	Ы	Е	У	ОЙ
S006	3088		А	У		ОМ
S007	3025	Я	И	И	Ю	ЕЙ
S034	3000	Й	Я	Ю	Й	ЕМ
S055	3000	ЕНЬ	НЯ	НЮ	ЕНЬ	НЕМ
A002						
	2826	ЫЙ	ОГО	ОМУ	ЫЙ	
A003	2800	ИЙ	ОГО	ОМУ	ИЙ	ОГО
A014	2883	НЫЙ	НОГО	НОМУ	НЫЙ	
			ЕГО	ЕМУ	ИЙ	ЕГО
A029	2914	ИЙ	ЕГО	ЕМУ	ИЙ	
A030	2940	ИЙ				
V115	0000	АВНИВА	ТЬ	ЕТ	ЕМ	ЕТЕ
V116	0000	БЬЁА	ТЬ	ЕТ	ЕМ	ЕТЕ
V117	0000	ЕВЬЕВА	ТЬ	ЕТ	ЕМ	ЕТЕ
V259	0188		АТЬ	ИТ	ИМ	ИТЕ
V240	0188		АТЬ	ЛЕТ	ЛЕМ	ЛИТЕ
V241	0188		ЕТЬ	ИТ	ИМ	ИТЕ
V242	0188		ЕТЬ	ИТ	ИМ	ИТЕ
V243	0188		ЕТЬ	ЕТ	ЕМ	ЕТЕ
V244	0188		ЕТЬ	ОЕТ	ОЕМ	СЕТЕ
V245	0188		ИТЬ	ИТ	ИМ	ИТЕ
V246	0188		ИТЬ	ИТ	ИМ	ИТЕ
V247	0188		ИТЬ	ИТ	ИМ	ИТЕ
V248	0188		ИТЬ	ЬЕТ	ЬЕМ	ЬЕТЕ

такие семантические признаки, как **конкретное, дискретное, тело, артефакт** могут быть одновременно приписаны словам **стол, стул, полка, кольцо, шарнир, стакан** и множеству других. Поскольку число таких кодов ограничено, то ограничена и их комбинаторика. Следовательно, если согласиться с тем, что синтаксис есть некоторая система ограничений, налагаемых на сочетаемость семантических единиц, то комбинаторика совместной встречаемости определенных семантических классов одновременно в рамках ограниченного контекста помогает снять и синтаксическую, и лексическую многозначность анализируемых фрагментов иноязычного и русского текстов. Задание возможности совместной встречаемости семантических кодов также является средством конкретизации значения слова.

Код терминологичности используется при решении задач распознавания общего смысла текста и при снятии многозначности отдельных лексических единиц в ходе машинного перевода.

В первом случае этот код служит указанием на то, что слово является ключевым или полиключевым в данной предметной области и может входить в эталоны, использующиеся при автоматической атрибуции, аннотировании и реферировании текста.

Во втором случае код терминологичности, относя слово к определенному подъязыку, одновременно указывает и на его точное значение. В качестве примера укажем на слово *катаракт*, которое будучи соотнесенным с подъязыком географии получает значение «*крупный водопад, где большая масса воды низвергается широким фронтом с относительно небольшой высоты*». Напротив, перенесенное в подъязык деталей машин, оно имеет значение «*специальный масляный тормоз, применяемый в системах автоматического регулирования*».

Нельзя забывать также о том, что существует большое число терминов, употребляющихся в одном и том же, или близких значениях в разных подъязыках.

Например, слова типа **таблица, матрица** имеют постоянное значение независимо от подъязыка, в котором они используются.

Обычно при описании значения многозначного термина выделяется некоторый семантический центр, определяющий его принадлежность к основному подъязыку, а также дополнительные семантические характеристики. Эти характеристики указывают на принадлежность данного слова к другим предметным областям, для которых контрольными являются иные слова. Таким образом, основа семантических кодов — это классификация подъязыков и разбиение последних на более узкие семантические области.

Семантический код слова предусматривает отнесение его к определенной лексико-грамматической группе. Этот код вырабатывается относительно каждого подъязыка с помощью

классификационных древовидных графов, построенных на использовании лингвистических и экстралингвистических (логических, психологических и натурно-физических признаков). В тех случаях, когда неоднозначность слова не может быть устранена с помощью словаря оборотов или кода терминологичности, эта задача решается с помощью специальных алгоритмов, исследующих сочетаемость семантических кодов.

Извлечение с помощью ЭВМ смыслового инварианта документа и определение его ремы (новизны), а также организация диалога человек-машина становится возможным только тогда, когда в машину вводится структурированное «тезаурусное» описание всего семантического пространства или его отдельных областей. В нашем случае в роли тезауруса выступает парадигматическая модель плана содержания, представляющая собой множество лексических единиц, между которыми заданы смысловые связи как иерархического (родо-видового), так и неиерархического (синонимия, антонимия, ассоциация) типов. В словаре в соответствии с этим фиксируется принадлежность слова к тезаурусной модели, а также его связи. Каждое слово и каждое сочетание слов может вступать в множество различных связей. При этом в каждом конкретном контексте количество этих связей ограничивается и определяется экстралингвистическими факторами, например, важностью определенных связей для передачи сообщения, для точного выражения мысли. Таким образом, многоцелевой автоматический словарь русского языка должен представлять собой возможно более полную модель ассоциативных связей, существующих между отдельными элементами лексики в нашем мозгу. Поскольку отражение этой системы связей для каждого слова конкретизируется и определяется рамками подъязыков, то именно последние и определяют для каждого слова тот контекст, в рамках которого множество его ассоциативных связей сужается до достаточно узкого набора.

Итак, автоматический словарь должен стать настоящей сокровищницей лексики современного русского языка. Каким же образом подобный словарь может быть размещен в памяти современных ЭВМ и как он будет функционировать в вычислительной системе?

Список литературы: 1. *Bruderer H. E.* Vorführung eines automatischen Übersetzungvarfahrens im Rechnenzentrumol der Universität Zürich — Association for Literary and Linguistic Computing Bulletin, 1975, 3, № 3, S. 197—200. 2. *Loh S.— C.* Machine Franslation: Past, Present and Future. — Association for Literary and Linguistic Computing Bulletin, 1976, 4, № 2, p. 105—109. 3. *Кацнельсон С. Д.* Типология языка и речевое мышление. — Л.: Наука, 1972. — 262 с. 4. *Вахек Й.* Лингвистический словарь пражской школы.— М.: Изд-во Моск. ун-та, 1964.—184 с. 5. *Волоцкая З. М., Николаева Т. М., Молошная Т. В.* Опыт описания русского языка в его письменной форме. — М.: Изд-во Моск. ун-та, 1964.—310 с. 6. *Маслов Ю. С.* О некоторых расхождениях в понимании термина морфема.— В кн.: Проблемы языкознания. — Л.: Изд-во

Ленингр. ун-та, 1961, с. 25—27. 7. Обратный словарь русского языка. — М.: Сов. энциклопедия, 1974.—420 с. 8. Кодирование грамматической информации в машинном словаре /А. Д. Борисевич, В. В. Гончаренко, В. А. Гречишина и др. — В кн.: Статистика текста. — Минск: Изд-во Белорус. ун-та, 1970, с. 42—46. 9. Русский язык. Грамматика. — М.: Изд. АН СССР, 1970.—352 с. 10. Зализняк А. А. Русское именное словоизменение. — М.: Наука, 1967.— 242 с.

Поступила 19 марта 1979 г.

УДК 62.506.2.

Л. Н. БЕЛЯЕВА, Е. М. ЛУКЬЯНОВА

ПРОБЛЕМЫ РАЗРАБОТКИ МНОГОЦЕЛЕВОГО АВТОМАТИЧЕСКОГО РУССКОГО СЛОВАРЯ

СООБЩЕНИЕ 2

Машинная реализация МАРС. Чтобы удовлетворять всем предъявленным к нему требованиям, МАРС должен представлять собой систему наборов лингвистических данных, а также алгоритмических, языковых и программных средств, обеспечивающих порождение лексических единиц (морфем, словоформ, канонических форм, оборотов-сегментов), которые задаются соответствующим алгоритмом.

Универсальная система МАРС в настоящее время наиболее оптимально может быть реализована в виде так называемого банка данных. Банк данных представляет собой хранилище больших массивов информации сложной структуры, расположенное на устройствах прямого доступа. Последнее объясняется тем, что большой объем МАРС, исчисляемый миллионами символов [1], не может быть размещен в основной памяти в ЭВМ.

Техническая база ЭВМ третьего поколения, включающая в качестве памяти устройства прямого доступа, обеспечивает необходимые «емкости» для хранения МАРС и чрезвычайно высокую скорость передачи информации во внутреннюю память ЭВМ (156 000 символов в секунду), где эта информация и может обрабатываться по конкретным лингвистическим алгоритмам.

Известно, что одним из главных направлений в развитии сложных автоматизированных систем обработки данных является создание условий, в которых основной контингент пользователя работал бы наиболее эффективно. Этому служит выделение процессов, связанных с математическим, информационным, лингвистическим, техническим, организационным обеспечением решения задач из сферы деятельности специалистов различных областей с последующей автоматизацией выполнения этих процессов в форме автономных функций. Именно поэтому информационное обеспечение ЭВМ третьего поколения содержит автоматические системы управления банками данных, которые выполня-

ют функции создания, ведения, пополнения, корректировки банков данных, одним словом, функции поддержания информации банка в актуальном состоянии.

Реализация МАРС в структуре банка лингвистических данных позволяет лингвисту работать в «удобной вычислительной обстановке» [2], не заботясь ни о машинной реализации связей между компонентами информации, хранимой в МАРС, ни о проблеме пополнения и корректировки этой информации. Последнее имеет для лингвиста чрезвычайно большое значение, поскольку язык является открытой системой, и лексика его постоянно изменяется и обогащается.

Преимуществами банковской структуры является также отсутствие многократного дублирования данных, обеспечение коллективного доступа к лингвистической информации со стороны абонентов, решающих задачи, защита информации от разрушения. При этом, так как логическая структура любого банка данных должна строиться на основе информационных моделей конкретных задач, использующих этот банк, в нашем случае при построении базы основных лингвистических данных (БОЛИД'а) необходимо учитывать информационные модели таких конкретных лингвистических задач, как: — атрибуция, аннотирование и реферирование русских документов;— атрибуция, аннотирование и реферирование иностранных документов средствами русского языка; — перевод иностранных текстов; — решение различных лексикографических, грамматических, лингвистических задач; — поиск библиографической информации; —поиск фактографической информации в русском тексте.

Очевидно, создание банка данных, отвечающего, с одной стороны, требованиям общей технологии разработки банков данных в нашей стране и за рубежом, а с другой стороны,— конкретным требованиям, необходимым для достижения лучшего качества и повышения эффективности работы различных автоматизированных систем научно-технической информации (в том числе и практических систем МП) является делом очень трудоемким. Поэтому, весьма ценен факт наличия в нашей стране готовых к использованию пакетов прикладных программ (нескольких типов), реализующих некоторый базисный набор функций, позволяющий удовлетворить основные требования лингвиста к банкам данных. Специальные и дополнительные требования должны обеспечиваться с помощью комплекса программ, разрабатываемого в соответствии с требованиями лингвистических алгоритмов, с одной стороны, и возможностями базового и специализированного математического и информационного обеспечения — с другой.

В качестве стандартной СУБД для базы лингвистических данных МАРС используется система математического обеспечения «Банк данных универсальной структуры — БАНК» [3], которая реализована для двух вариантов операционных систем

ЕС ЭВМ — ДОС и ОС. Программное обеспечение СУБД БАНК дополнено специально разработанным комплексом программ, расширяющим возможности функционального и сервисного обслуживания лингвистического банка данных.

Подготовка информации АС для загрузки в банк. Использование МАРС для задач МП потребовало расширения лингвистической структуры систем автоматических словарей за счет включения в ее состав АС входных языков. Для загрузки лингвистической информации в банк требуется представить ее в виде интегрированного загрузочного файла, содержащего записи следующих типов: развернутых лексико-грамматических кодов; типовых парадигм русских окончаний; русских машинных основ; английских словоформ.

Кроме того, для ускорения поиска информации в базе целесообразно ввести четыре дополнительных типа записей: букв русского и латинского алфавитов, длин русских машинных основ и английских словоформ.

Введение этих записей увеличивает объем базы для отраслевого словаря в среднем на 1%, но повышает скорость поиска информации в базе в несколько раз (имеется в виду поиск русских машинных основ и английских словоформ) [4].

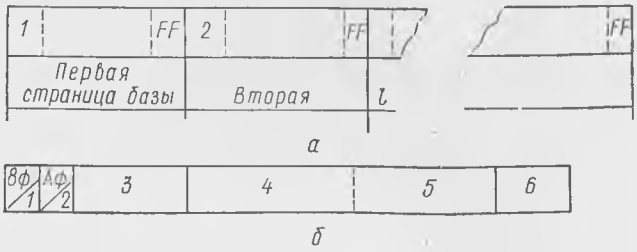
Перед загрузкой в банк интегрированный загрузочный файл обычно подготавливается на магнитной ленте (МЛ) или магнитных дисках (МД) и специальным образом сортируется с целью повышения скорости последующей загрузки информации в базу, вследствие чего такая загрузка и получила название структурно-уровневой. Для ее реализации записи всех типов должны иметь в качестве дополнительных или основных полей так называемые структурные ключи сортировки (которые в записях базы могут отсутствовать). Это налагает специальные условия на форматы подготовки данных на первичных перфоносителях.

Особенности хранения лингвистической информации в банке данных. В системе математического обеспечения каждый файл состоит из блоков фиксированной длины, которые носят название страниц. СУБД БАНК независимо от числа файлов в информационной базе (ИБ) производит сквозную нумерацию страниц. Номер страницы является числом, постоянно закрепленным за страницей, по которому СУБД определяет действительный адрес страницы на МД. Максимальное число страниц в ИБ не должно превышать 655535.

Размер страницы определяется пользователем, но при этом должны соблюдаться два условия:— размер страницы не должен превышать длину дорожки (3625 байт);— на дорожке должно размещаться целое число страниц.

При машинной реализации информационной базы БОЛИД выбран размер страницы, равный 1131 байту. Системой БАНК

определены следующие рекомендуемые размеры страниц: 256, 532, 1131, 3625 байт. Страничная структура базы указана на рисунке (*а*), из которого видно, что каждая страница начинается с 10-байтового заголовка и заканчивается байтом, содержащим $'FF'_{16}$. Структура заголовка страницы представлена на рисун-



ке (*б*) (заголовок страницы СУБД БАНК оформляет как обычную запись, номер типа которой постоянен).

Размер страницы связан с таким параметром программной реализации и использования банка лингвистических данных, функционирующего под управлением СУБД БАНК, как буфер основной памяти, в которой считываются страницы базы.

Размер буфера однозначно равен размеру страницы. Число буферов в ОП для каждой программной реализации лингвистической задачи определяется следующим образом.

1. Определяется объем раздела, в котором выполняется программа, $V_{\text{разд}}$.

2. Определяется объем программных компонентов, присутствие которых в ОП необходимо для выполнения рассматриваемой лингвистической задачи, $V_{\text{Сз}}$.

3. Вычисляется общий объем под буферную память $V_{\text{т}} = V_{\text{разд}} - V_{\text{Сз}}$.

4. Число буферов для подсхемы базы данной задачи определяется по формуле $n_{\text{т}} = \frac{V_{\text{т}}}{P_{\text{стр}}}$, где $P_{\text{стр}}$ — размер страницы информационной базы.

В банке лингвистических данных количество буферов определяется равным 9.

Система математического обеспечения БАНК полностью автоматизирует функции размещения информации на внешнем носителе (магнитных дисках), в то же время предоставляя пользователю следующие возможности:

1) определение размера физической записи, т. е. страницы, с учетом условий, определенных выше;

2) для ускорения поиска допускается при необходимости определять записи любого типа, как вычисляемые, тем самым обеспечивая непосредственный доступ к ним по ключу вычисления;

3) для ускорения поиска экземпляры записей различных типов можно располагать на МД рядом.

Представим страницу банка, загруженную следующей информацией:— записью русской машинной основы — **УСТРОЙСТВ;** — записями длин русских машинных основ — $\varnothing\varnothing\div 1D_{16}$;— записью начальной буквы этой основы — **У** (см. таблицу). При этом поиск русской машинной основы в одноязычной ситуации осуществляется по контуру: начальная буква — длина — основа и сокращается по времени в 3 раза по сравнению с тем случаем, когда все три записи располагаются на различных страницах.

Некоторые функциональные характеристики банка лингвистических данных. Основными функциональными характеристиками лингвистического банка являются:— его объем;— объем обслуживающих и сервисных программ;— время загрузки лингвистической информации в базу;— время реализации запроса.

Под временем реализации запроса подразумевается среднее время, необходимое для порождения русской словоформы в одно- и двуязычной ситуации.

Для подъязыка вычислительной техники объем банка лингвистических данных составляет 4,5 млн. байт и содержит: русских машинных основ — 12 006; типовых парадигм русских машинных окончаний — 665; букв русского алфавита — 30; длин русских машинных основ — 35; английских словоформ — 11 000; английских фразеологических оборотов — 11 000; букв латинского алфавита — 25; длин английских словоформ и фразеологических оборотов — 55; лексико-грамматических кодов — 916.

База лингвистических данных содержит 4000 страниц размером 1131 байт каждая и занимает 140 цилиндров на накопителях на магнитных дисках объемом 7,25 млн. байт.

Обслуживающие и сервисные программы системы представляют собой как стандартное обеспечение системы БАНК, так и специально разработанный пакет прикладных программ БОЛИД (Банк Основных Лингвистических Данных). Объем системы БАНК — 38 килобайт, объем ППП БОЛИД — 45 килобайт.

Загрузка отраслевого словаря в базу лингвистических данных на примере АС по вычислительной технике занимает 62 часа; включая время машинной подготовки описанного выше интегрированного загрузочного файла и время устранения ошибок в подготовке лингвистической информации на первичных перфоносителях.

Время реализации запроса в двуязычной ситуации (пословно-пооборотный машинный перевод) составляет порядка 0,5 с на одну лексему, в то время как аналогичное время в ныне действующих системах промышленного машинного перевода равно 1,32 с.

11(000B)	000B000A	46	10(000A)	000B195	000B0000
		47	28(001C)	000B001C	000B000A
		47	41(0029)	000B0029	000B001C
		47	54(0036)	000B0036	000B0029
		47	67(0043)	000B0043	000B0036
		47	B0(0050)	000B0050	000B0043
		47	93(005)	000B0050	000B0050
		47	106(006A)	000B006A	000B0050
		47	119(0077)	000B0077	000B006A
		47	132(0084)	000B0084	000B0077
		47	145(0091)	000B0091	000B0084
		47	158(009E)	000B009E	000B0091
		47	171(00AB)	000B00AB	000B009E
		47	184(00B8)	000B00B8	000B00AB
		47	197(00C5)	000Б0075	000B00B8
		47	210(0002)	000B0002	000B00C5
		47	223(00ДГ)	000B000Г	000B0002
		47	236(00EC)	000B00EC	000B000Г
		47	249(00Г9)	000B00Г9	000B00EC
		47	262(0106)	000B0106	000B00Г9
		47	275(0113)	000B0113	000B0106
		47	288(0120)	000C0108	000B0113
		47	301(0120)	000B012D	000B0120
		47	314(013A)	000C019E	00B012D
		47	327(0147)	00190108	000B013A
		47	340(0154)	000B0154	000B0147
		47	353(0161)	000B0161	000B0154
		47	366(016Г)	000B016E	000B0161
		47	379(017B)	000B017B	000B016E
		47	392(0188)	000B0188	000B017B
		47	405(0195)	000B0195	000B0188
		48	418(01A2)	003Г01F5	000A000A
				00000725	EBC3E3D7
				40404040	40404040
		OA	475(01DB)	00110108	0007010B
				00000000	000040

БАЙТ НАЧАЛА СВОБОДНОГО УЧАСТКА: 506(01ГА) ДЛИНА: 25

ПРОЦЕНТ ЗАПОЛНЕНИЯ РАВЕ

```
0007000A      EBFF
1D
1C
1B
1A
19
18
17
16
15
14
13
12
11
10
0F
0E
0D
0C
0B
0A
09
08
07
06
05
04
03
02
01
00
000A000A      000B0120      000B120       5
Д6СССЗЕЗ      С2404040      40404040      УСТРОЙСТВ
40404040      40
00000B25      03000202      02020200
```

Разработка системы МАРС производилась и производится при тесном сотрудничестве лингвистов и программистов. Несомненно, что только такое содружество может дать плодотворные результаты по созданию практических систем лингвистического обеспечения.

Список литературы: 1. *Беляева Л. Н., Крисевич В. С., Липницкий С. Ф., Пиотровский Р. Г.* О многоцелевом автоматическом словаре русского языка (МАРС). — Вопросы общей и прикладной лингвистики. — Минск: Изд-во Белорус. ун-та, 1975, с. 152—153. 2. Информационные системы общего назначения /Под ред. Е. Л. Ющенко. — М.: Наука, 1975.—215 с. 3. Банк данных универсальной структуры. Система математического обеспечения. Техническая документация. — Калинин, НПО «Центрпрограммсистем», 1975—ДОС, 1978—ОС. 4. *Заикин О. А., Лукьянова Е. М.* Использование частотного критерия для увеличения скорости поиска информации в базе лингвистических данных.— Изв. ЛЭТИ, 1979, вып. 221, с. 77—81.

Поступила 19 марта 1979 г.

УДК 510.62

Ю. П. ШАБАНОВ-КУШНАРЕНКО, д-р техн. наук

О МОДЕЛИРОВАНИИ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ

В статье развиваются некоторые идеи, изложенные в работах [1, 2]. На примерах арифметических операций рассматривается вопрос о моделировании конечных алфавитных операторов средствами теории интеллекта. Рассмотрим работу полного одноразрядного двоичного сумматора. Схематически такой сумматор изображен на рис. 1. На входы сумматора подаются значения i-х разрядов слагаемых x_i и y_i и переноса t_{i-1} из младшего i—1-го разряда. На выходах сумматора получаем значение i-го разряда суммы z_i и переноса t_i в старший $i+1$-й разряд. Закон соответствия между сигналами x_i, y_i, t_{i-1}, t_i, z_i указан в табл. 1.

Таблица 1

X_i	0	0	0	0	1	1	1	1
y_i	0	0	1	1	0	0	1	1
t_{i-1}	0	1	0	1	0	1	0	1
t_i	0	0	0	1	0	1	1	1
Z_i	0	1	1	0	1	0	0	1

В явной форме работа одноразрядного сумматора запишется согласно этой таблице в виде:

$$\begin{gathered} x_i^0 y_i^0 \vee (x_i^0 y_i^1 \vee x_i^1 y_i^0)\, t_{i-1}^0 = t_i^0; \\ x_i^1 y_i^1 \vee (x_i^0 y_i^1 \vee x_i^1 y_i^0)\, t_{i-1}^1 = t_i^1; \\ (x_i^0 y_i^0 \vee x_i^1 y_i^1)\, t_{i-1}^0 \vee (x_i^0 y_i^1 \vee x_i^1 y_i^0)\, t_{i-1}^1 = z_i^0; \\ (x_i^0 y_i^0 \vee x_i^1 y_i^1)\, t_{i-1}^1 \vee (x_i^0 y_i^1 \vee x_i^1 y_i^0)\, t_{i-1}^0 = z_i^1. \end{gathered} \qquad (1)$$

По формулам (1) может быть построен полный одноразрядный сумматор в виде некоторой переключательной цепи. Соединяя в

цепочку n одноразрядных сумматоров (рис. 2), получаем многоразрядный сумматор, формирующий n-разрядную сумму $z_n z_{n-1} \ldots z_1$ по n-разрядным слагаемым $x_n x_{n-1} \ldots x_1$ и $y_n y_{n-1} \ldots y_1$.

Работа многоразрядного сумматора описывается множеством систем уравнений (1), у которых индекс i пробегает значения

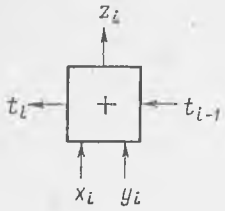


Рис. 1

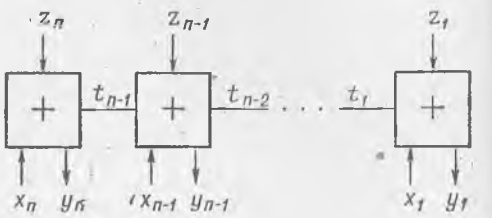


Рис. 2

от 2 до $n-1$. К этим уравнениям следует добавить еще два для одноразрядных сумматоров первого и n-го разрядов. Эти сумматоры отличаются от остальных и друг от друга. В первом отсутствует вход t_0 для переноса из младшего разряда, в последнем — выход t_n для переноса в старший разряд. Законы соответствия между сигналами x_1, y_1, t_1, z_1 и сигналами x_n, y_n, t_{n-1}, z_n указаны в табл. 2, 3.

Таблица 2

X	0	0	1	1
y	0	1	0	1
t	0	0	0	1
Z_i	0	1	1	0

Таблица 3

X_n	0	0	0	1
Y_n	0	0	1	0
t_{n-1}	0	1	0	0
Z_n	0	1	1	1

По таблицам составляем явные описания работы сумматоров первого разряда

$$x_1^0 \vee y_1^0 = t_1^0, \quad x_1^1 y_1^1 = t_1^1,$$
$$x_1^0 y_1^0 \vee x_1^1 y_1^1 = z_1^0;$$
$$x_1^0 y_1^1 \vee x_1^1 y_1^0 = z_1^1; \qquad (2)$$

n-го разряда

$$x_n^0 y_n^0 t_{n-1}^0 = z_n^0,$$
$$x_n^0 (y_n^0 t_{n-1}^1 \vee y_n^1 t_{n-1}^0) \vee x_n^1 y_n^0 t_{n-1}^0 = z_n^1. \qquad (3)$$

Рассмотрим, например, действие трехразрядного ($n = 3$) сумматора. Этот сумматор описывается уравнениями (2) и следующими двумя системами уравнений, которые получаются из (1) подстановкой $i = 2$:

$$x_2^0 y_2^0 \vee (x_2^0 y_2^1 \vee x_2^1 y_2^0)\, t_1^0 = t_2^0, \quad x_2^1 y_2^1 \vee (x_2^0 y_2^1 \vee x_2^1 y_2^0)\, t_1^1 = t_2^1,$$
$$(x_2^0 y_2^0 \vee x_2^1 y_2^1)\, t_1^0 \vee (x_2^0 y_2^1 \vee x_2^1 y_2^0)\, t_1^1 = z_2^0,$$
$$(x_2^0 y_2^0 \vee x_2^1 y_2^1)\, t_1^1 \vee (x_2^0 y_2^1 \vee x_2^1 y_2^0)\, t_1^0 = z_2^1 \qquad (а)$$

из (3) подстановкой $i = 3$:

$$x_3^0 y_3^0 t_2^0 = z_3^0, \quad (x_3^0 y_3^1 \vee x_3^1 y_3^0)\, t_2^0 \vee x_3^0 y_3^0 t_2^1 = z_3^1. \qquad (б)$$

Пусть $x_3x_2x_1 = 010$, $y_3y_2y_1 = 011$. Подставляя в (2) $x_1^0 = 1$, $x_1^1 = 0$, $y_1^0 = 0$, $y_1^1 = 1$, находим $t_1^0 = 1$, $t_1^1 = 0$, $z_1^0 = 0$, $z_1^1 = 1$. Далее, подставляем найденные значения t_1^0, t_1^1, вместе с заданными $x_2^0 = 0$, $x_2^1 = 1$, $y_2^0 = 0$, $y_2^1 = 1$, в (а) и вычисляем: $t_2^0 = 0$, $t_2^1 = 1$, $z_2^0 = 1$, $z_2^1 = 0$. Наконец, подставляем значения t_2^0, t_2^1 и $x_3^0 = 1$, $x_3^1 = 0$, $y_3^0 = 1$, $y_3^1 = 1$ в (б) и находим: $z_3^0 = 0$, $z_3^1 = 1$. Таким образом переключательная цепь формирует значение суммы $z_3z_2z_1 = 101$. Производя подобные вычисления для слагаемых $x_3x_2x_1 = 101$ и $y_3y_2y_1 = 110$, находим, что $z_3^0 = 0$, $z_3^1 = 0$. Это означает, что значение z_3 не существует. Вместе с ним не существует и вся сумма $z_3z_2z_1$. Мы видим, что сумматор, построенный методами теории интеллекта, сигнализирует о переполнении регистра суммы иным способом, чем это делает сумматор обычного типа (в качестве сигнала переполнения в нем обычно используется единичное значение сигнала переноса t_n в несуществующий $n+1$-й разряд).

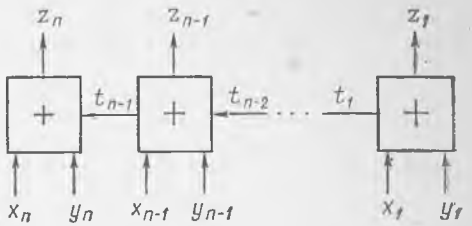


Рис. 3

Аналогичным образом можно построить многоразрядный вычитатель, формирующий разность кодов $y_ny_{n-1} \ldots y_1 = z_nz_{n-1} \ldots z_1 - x_nx_{n-1} \ldots x_1$. Его можно построить в виде цепочки одноразрядных вычитателей (рис. 3). Теперь сигнал t_i интерпретируем как заем из $i+1$-го в i-й разряд. В вычитателе реализуется то же самое отношение сложения $x_nx_{n-1} \ldots x_1 + y_ny_{n-1} \ldots y_1 = z_nz_{n-1} \ldots z_1$ между тремя n-разрядными кодами, что и в сумматоре. Однако теперь требуется по известным кодам $x_nx_{n-1} \ldots x_1$ и $z_nz_{n-1} \ldots z_1$ отыскать код $y_ny_{n-1} \ldots y_1$. По табл. 2 находим явное описание для вычитателя первого разряда:

$$x_1^0z_1^0 \vee x_1^1z_1^1 = y_1^0, \quad x_1^0z_1^1 \vee x_1^1z_1^0 = y_1^1, \quad x_1^0 \vee z_1^1 = t_1^0, \quad x_1^1z_1^0 = t_1^1. \quad (4)$$

По табл. 1 строим формулы для вычитателей с номерами $i = 2 \div n-1$:

$$\begin{gathered}(x_i^0t_{i-1}^0 \vee x_i^1t_{i-1}^1)\, z_i^0 \vee (x_i^0t_{i-1}^1 \vee x_i^1t_{i-1}^0)\, z_i^1 = y_i^0; \quad (5)\\ (x_i^0t_{i-1}^0 \vee x_i^1t_{i-1}^1)\, z_i^1 \vee (x_i^0t_{i-1}^1 \vee x_i^1t_{i-1}^0)\, z_i^0 = y_i^1;\\ x_i^0t_{i-1}^0z_i^0 \vee (x_i^0 \vee t_{i-1}^0)\, z_i^1 = t_i^0; \quad x_i^1t_{i-1}^1z_i^1 \vee (x_i^1 \vee t_{i-1}^1)\, z_i^0 = t_i^1.\end{gathered}$$

По табл. 3 строим формулы для вычитателя n-го разряда

$$x_n^0(t_{n-1}^0 \vee z_n^1) = y_n^0; \quad x_n^0t_{n-1}^0z_n^1 = y_n^1. \quad (6)$$

В случае, когда вычитаемое больше уменьшаемого, вычитатель сигнализирует о несуществовании решения.

Тем же способом могут быть построены сумматоры и вычитатели для кодов с произвольным основанием, а также смешанных кодов. Ограничимся неявным описанием отношения сложения k-х n-разрядных кодов $X + Y = Z$, где $X = x_n x_{n-1} \ldots x_1$, $Y = y_n y_{n-1} \ldots y_1$, $Z = z_n z_{n-1} \ldots z_1$. Явные описания операций сложения и вычитания и переключательные цепи для этих операций могут быть легко построены по неявному описанию методами, описанными в [2]. Переменные x_i, y_i, z_i в данном случае принимают k-е значения, переносы же t_{i-1}, t_i — двоичные. Введем предикат сложения $S(X, Y, Z)$, связанный с отношением сложения таким образом:

$$S(X, Y, Z) = \begin{cases} 0, & \text{если } X + Y \neq Z, \\ 1, & \text{если } X + Y = Z. \end{cases} \tag{7}$$

Этот предикат может быть записан системой уравнений:

$$\bigvee_{(\alpha, \beta, \gamma, \delta, \varepsilon)} x_i^{\alpha} y_i^{\beta} t_{i-1}^{\gamma} t_i^{\delta} z_i^{\varepsilon} = 1, \ (1 \leqslant i \leqslant n) \tag{8}$$

$$t_0^0 = t_n^0 = 1, \tag{9}$$

$$\bigvee_{\alpha=0}^{k-1} x_i^{\alpha} = \bigvee_{\beta=0}^{k-1} y_i^{\beta} = t_{i-1}^0 \vee t_{i-1}^1 = t_i^0 \vee t_i^1 = \bigvee_{\varepsilon=0}^{k-1} z_i^{\varepsilon} = 1 \ (1 \leqslant i \leqslant n) \tag{10}$$

В левой части уравнения (8) суммирование ведется по всем наборам $(\alpha, \beta, \gamma, \delta, \varepsilon)$ показателей узнаваний, удовлетворяющим условиям:

$$\delta = \begin{cases} 0, & \text{если } \alpha + \beta + \gamma < k, \\ 1, & \text{если } \alpha + \beta + \gamma \geqslant k; \end{cases} \tag{11}$$

$$\varepsilon = \alpha + \beta + \gamma \pmod{k}. \tag{12}$$

Рассмотрим пример построения переключательной цепи на основе приведенных зависимостей. Построим двухразрядный троичный вычитатель. Полагая $n = 2$, $k = 3$, запишем неявное описание этого вычитателя в виде системы двух уравнений:

$$x_1^0 y_1^0 t_1^0 z_1^0 \vee x_1^0 y_1^1 t_1^0 z_1^1 \vee x_1^0 y_1^2 t_1^0 z_1^2 \vee x_1^1 y_1^0 t_1^0 z_1^1 \vee x_1^1 y_1^1 t_1^0 z_1^2 \vee x_1^1 y_1^2 t_1^1 z_1^0 \vee$$
$$\vee x_2^2 y_1^0 t_1^0 z_1^2 \vee x_1^2 y_1^1 t_1^1 z_1^0 \vee x_1^2 y_1^2 t_1^1 z_1^1 = 1, \tag{в}$$

$$x_2^0 y_2^0 t_1^0 z_2^0 \vee x_2^0 y_2^1 t_1^0 z_2^1 \vee x_2^0 y_2^0 t_1^0 z_2^2 \vee x_2^1 y_2^0 t_1^0 z_2^1 \vee x_2^1 y_2^1 t_1^0 z_2^2 \vee x_2^2 y_2^0 t_1^0 z_2^2 \vee$$
$$\vee x_2^0 y_2^0 t_1^1 z_2^1 \vee x_2^0 y_2^1 t_1^1 z_2^2 \vee x_2^1 y_2^0 t_1^1 z_2^2 = 1. \tag{г}$$

Находим явные выражения для переменных y_1, t_1 в зависимости от x_1, z_1 — из уравнения (в) и для переменной y_2 в зависимости от x_2, t_1, z_2 — из уравнения (г). В левую часть выражений вводим узнавания лишь тех переменных, от которых устанавливается зависимость искомых переменных. Значения

остальных переменных оставляем неопределенными, приравнивая к единице все их узнавания. Имеем:

$$x_1^0z_1^0 \vee x_1^1z_1^1 \vee x_1^2z_1^2 = y_1^0;\ x_1^0z_1^1 \vee x_1^1z_1^2 \vee x_1^2z_1^0 = y_1^1;$$
$$x_1^0z_1^2 \vee x_1^1z_1^0 \vee x_1^2z_1^1 = y_1^2;\ \vee x_1^1z_1^1 \vee x_1^1z_1^2 \vee x_1^2z_1^2 = t_1^0;$$
$$x_1^1z_1^0 \vee x_1^2z_1^0 \vee x_1^2z_1^1 = t_1^1;\ x_2^0t_1^0z_2^0 \vee x_2^1t_1^0z_2^1 \vee x_2^2t_1^0z_2^2 \vee x_2^0t_1^1z_2^1 \vee$$
$$\vee x_2^1t_1^1z_2^2 = y_2^0; \qquad (д)$$
$$x_2^0t_1^0z_2^1 \vee x_2^1t_1^0z_2^2 \vee x_2^0t_1^1z_2^2 = y_2^1,\ x_2^0t_1^0z_2^2 = y_2^2.$$

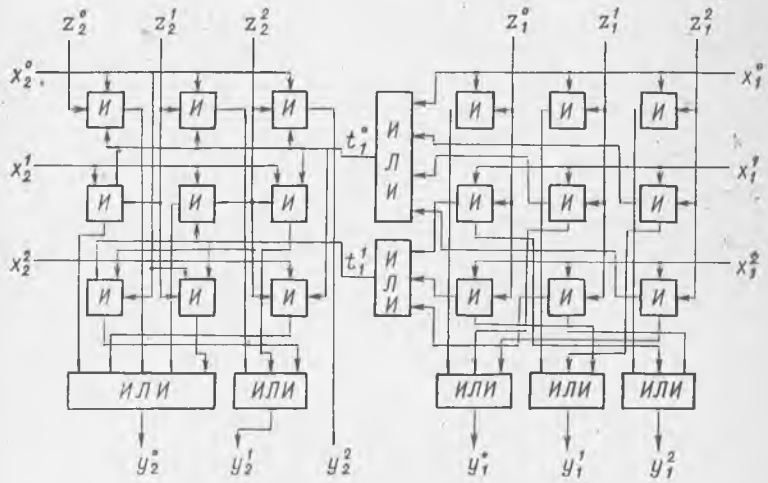


Рис. 4

На рис. 4 изображен двухразрядный троичный вычитатель, построенный по формулам (д), образующий разность чисел от 0 до 8. Для построения такого вычитателя требуются 69 диодов. Двоичный вычитатель, способный выполнять ту же работу, должен был бы иметь четыре разряда. Для его построения по формулам (8)—(10) требуется 101 диод. Как видим, сложность цепи существенно зависит от выбора способа кодирования исходных данных и результата.

Список литературы: 1. *Мурашко А. Г., Четвериков Г. Г., Шабанова-Кушнаренко З. Ю.* Математическое описание арифметических отношений двоичных кодов. — АСУ и приборы автоматики, 1979, вып. 50, с. 29—32. 2. *Шабанов-Кушнаренко Ю. П.* О моделировании переключательных функций и алфавитных операторов. — АСУ и приборы автоматики, 1980, вып. 57, с. 63—70.

Поступила 21 октября 1979 г.

УДК 519.76 : 681.3

А. Я. ДРЮЧЕНКО

НЕКОТОРЫЕ ВОПРОСЫ АВТОМАТИЧЕСКОГО РАСПОЗНАВАНИЯ РЕЧИ

Исключительная важность научно-поисковых работ в области автоматического распознавания и синтеза речи в настоящее время является общепризнанной. Достаточно сказать, что внедрение результатов этих работ только в сфере вычислительной техники, автоматизированных информационных систем, а также в технике узкополосной цифровой связи и в криминалистике позволило бы получить материальный и социальный эффект, истинные размеры которого сегодня еще трудно предсказать. Неслучайно существует мнение о том, что успешное решение задачи автоматического распознавания и синтеза речи по своему значению для развития прогресса сравнимо с тем, которое имело впоследствии открытие книгопечатания [1].

Интерес к вопросу автоматического распознавания и синтеза речи возник очень давно. Так, имеется упоминание [2] о том, что в XVIII веке часовые мастера отец и сын Дроц построили механического писца, который мог написать «... все называемые ему тексты — до 60 слов...». Известно также, что в 1791 г. В. Ф. Кемпелен [3] построил механический говорящий автомат.

Однако проблема автоматического распознавания и синтеза речи возникла где-то в конце 50-х годов нашего столетия. Огромная важность речевой коммуникации непосредственно между человеком и автоматом, а также все возрастающие возможности ее практической реализации привели к тому, что, начиная уже с 1965 г., в ряде промышленно высокоразвитых стран масштабы научного поиска в этой области резко увеличиваются. Вскоре были получены и первые весьма обнадеживающие результаты. В ФРГ, например, в 1965 г. был разработан простой и экономичный метод [4] автоматического распознавания цифр от нуля до девяти, позволивший получить для 74 дикторов (37 мужчин и 37 женщин) надежность распознавания не ниже 87% без нормализации уровня исходного речевого сигнала и 93% с нормализацией. К 1967 г. Цвиккер [5] разработал и исследовал аналогичное устройство для распознавания десяти числительных (от 0 до 9). В нем учитывались процессы преобразования сигналов в слуховой системе человека. Надежность распознавания составила 95,4%.

Последующие годы можно охарактеризовать, как период интенсивного накапливания знаний о природе и особенностях речевых сигналов, механизма речепорождения и психофизического восприятия, с одной стороны, и совершенствования технической базы, а также методики обработки речевых сигналов, с другой.

К 1974 г. во всем мире в целом было разработано около 50 методов [3] автоматического распознавания речи, часть из которых прошла серьезную лабораторную проверку. Надежность распознавания слов-команд в методах, использовавших приемы динамической оптимизации процесса обработки метрических и структурных характеристик речевых сигналов, превысила значение 90% для нескольких дикторов и словаря объемом до 300 слов.

На сегодняшний день во всем мире уже разработано около 80 методов. Некоторые из них впервые позволили получить в лабораторных условиях надежность распознавания, близкую к 99% для нескольких дикторов с хорошо отработанным произношением и словаря в пределах нескольких сотен слов. Эти методы предполагают использование современных ЭВМ и отличаются весьма сложной, связанной с большими непроизводительными затратами времени процедурой вычислений. Однако в целом надежность распознавания, как свидетельствует ряд новейших работ, продолжает оставаться довольно низкой (см. таблицу).

Количество дикторов	Количество слов	Надежность распознавания, %	Быстродействие, слов в минуту	Примечания
8	54	85	—	США, 1978 г.
20	120	90	60	Япония, 1978 г.
25	40	90,3	—	США, 1977 г.
105	40	89	7	США, 1978 г.
8	1984	85	—	Япония, 1978 г.
8	112	95,7	—	Япония, 1978 г.

Означает ли это, что все методы достаточно плохи или хороши? Какие критерии являются определяющими при оценке каждого метода? Совершенно очевидно, что здесь возможны по меньшей мере два подхода. Первый связан с проблемой узнавания в широком смысле и предполагает в конечном итоге разработку «высокоинтеллектуальных» программ, обеспечивающих в каждом конкретном случае нахождение таких алгоритмов обработки исходной информации, которые позволят оптимально и с максимальной надежностью решать поставленные задачи. Подобные программы целесообразно разрабатывать на модельных задачах и здесь, на этапе поиска, по-видимому, оправдана любая дороговизна метода даже при самых скромных результатах.

Другой подход — прикладного характера, где целью является, естественно, не изучение проблемы узнавания вообще, а получение хороших результатов при решении некоторых частных задач в конкретной области [6]. Понятно, что критерии оценки в этом случае должны помочь установить, насколько эффектив-

ней данный метод позволяет уже сегодня решать практические задачи по сравнению с другими методами.

Излишне доказывать, что проблема автоматического распознавания далеко выходит за рамки только познания, что она порождена нуждами сегодняшнего дня, и результаты научного поиска в этой области — это прежде всего действенные рычаги повышения эффективности производства.

За последние 15 лет, а это, как известно, был период интенсивнейших работ, проводившихся широким фронтом в области автоматического распознавания речи, показатель роста дороговизны процедуры распознавания нередко в десятки, а то и в сотни раз превосходит показатель роста надежности распознавания. Это объясняется главным образом непроизводительными затратами времени в процедуре анализа и классификации речевого сигнала с использованием современных универсальных ЭВМ. Если при этом учесть довольно скромные результаты, достигнутые в этой области на сегодняшний день, то едва ли можно говорить о высокой эффективности таких методов распознавания речи.

В 1967 г., например, сообщалось [7] о так называемом TASI-методе (сокращенно Tame Assignment Speech Interpolation). Принцип распознавания здесь был основан на выявлении только одного определенного физического параметра речевого сигнала, что давало возможность распознавать в общем ограниченный алфавит образов, но это был на то время единственный наиболее интересный с точки зрения экономичности и надежности распознавания метод, применение которого для такого дорогостоящего канала связи, как трансатлантический телефонный кабель, позволило повысить коэффициент его использования в 2—3 раза. Поэтому, нисколько не уменьшая важности проводимых научно-поисковых работ по решению целого ряда задач автоматического распознавания и синтеза речи, требующих применения одной или нескольких ЭВМ одновременно, необходимо, однако, указать на возрастающую перспективность разработки существенно упрощенных и в то же время обеспечивающих высокую надежность распознавания, методов [8]. Важным шагом на пути решения этой задачи, впрочем, как и глобальной задачи автоматического распознавания речи, является создание надежно работающего детектора звуков. Такое устройство должно отвечать ряду весьма жестких требований, чтобы оно могло быть эффективно использовано для практических целей, наиболее важными из которых являются высокая надежность распознавания, большое быстродействие и экономичность. При этом нежелательно использование ЭВМ, так как оно неизбежно связано с исключительно большой дороговизной системы. Кроме того, в основу принципа работы таких устройств не может быть положен принцип переходов через нуль, метод автокорреляции, линейного предсказания и т. п.

Задача настоящей статьи — разработать метод автоматического распознавания речи, отвечающий по основным своим характеристикам этим требованиям. В основу метода положен принцип структурного анализа речевого сигнала, дискретизированного с учетом эффекта психофизического сглаживания в слухе [9]. Такой метод представляется перспективным по ряду причин. Назовем некоторые из них. Речевой сигнал, дискретизированный с учетом эффекта психофизического сглаживания в слухе, полностью сохраняет исходную семантическую и эктосемантическую информацию. Единственным информативным элементом при этом является только временной интервал [10], что значительно упрощает процедуру нахождения признаков описания распознаваемого объекта. Дискретизированный сигнал отличается высокой помехоустойчивостью, кроме того, он может быть легко восстановлен в исходный речевой сигнал, что весьма важно для решения задач по синтезу речи.

В процессе эксперимента выявлено еще одно важное свойство сигнала. Оно заключается в том, что элементы двоичного кода не только численно выражают соответствующий анализируемый параметр речевого сигнала, но и сами представляют удобный объект анализа. Записанные в виде двумерного массива данных, они образуют «мозаику» с явно выраженной структурой отношений базовых блоков. В результате статистической обработки 1728 матриц массивов для 12 дикторов и шести стационарных гласных звуков русского языка авторы пришли к выводу о том, что существует однозначная связь между типом структурной композиции «мозаики» и фонетическим качеством звука независимо от индивидуальных особенностей голоса диктора.

Важность этого свойства становится понятной, если учесть, что в речевом сигнале «...признаки совершенно неочевидны. Они доступны объективному наблюдению и могут быть зафиксированы в виде осциллограмм звукового давления, представляющих абсолютное описание звуков речи. Эти осциллограммы имеют настолько сложную структуру, что их рассмотрение не дает никаких прямых указаний на существенные признаки тех или иных звуков. Между тем, такие признаки существуют, они просты и необыкновенно стойки. Это подтверждается тем, что человек уверенно распознает звуки речи в самых неблагоприятных условиях и при самом различном их произнесении» [11]. Как он это делает — во многом остается еще неясным. Не случайно процесс выбора признаков описания хоть и направляется какими-либо общими соображениями, все же в значительной мере является произвольным и чрезвычайно трудоемким.

Далее авторами установлено, что базовые блоки подразделяются на три типа: блоки нулей, блоки единиц и смешанные блоки. В свою очередь блоки нулей и блоки единиц могут быть стационарными и прерванными. Базовые блоки образуют

признаки двух видов—разграничительные маркерные и классификационные. Маркерные признаки позволяют надежно устанавливать границы сегментов аллофонов на квазистационарных участках речевого сигнала, в то время как признаки второго вида определяют тип структуры сегмента. В процессе речевого акта аллофоны гласных звуков реализуются последовательностью, состоящей по меньшей мере из 10 сегментов, каждый из которых несет полную информацию о фонетической принадлежности звука. Это обстоятельство использовано для повышения надежности распознавания путем накопления полезных признаков сигнала и отфильтровывания случайных. Указанная операция осуществляется с учетом свойств приспособляемости, характерных для процессов преобразования информации в биологических системах.

Анализируя закономерности распределения двоичных элементов дискретизированных речевых сигналов, авторы пришли к выводу о том, что тип структуры сегмента определяется наличием и взаиморасположением определенных базовых блоков с учетом некоторых временных характеристик.

Для базовых блоков, в зависимости от их типа и места расположения, введена система условных обозначений: Y, Z_1, Z_2, N_1, N_2. С учетом того, что структура может быть симметричной и асимметричной, вводится признак A, кроме того, для обозначения нестационарных блоков единиц и блоков нулей вводится признак β. T-период сигнала на интервале, ограниченном маркерными признаками.

На рисунке представлена диаграмма структурных признаков описания шести стационарных гласных звуков русского языка. Их проекции на оси времени отражают дисперсию фиксированных значений временных характеристик, наблюдавшихся в эксперименте на материале 1728 выборок для 6 звуков и 12 случайных дикторов. Одноименные признаки, проекции которых не перекрываются, обладают разделительными свойствами.

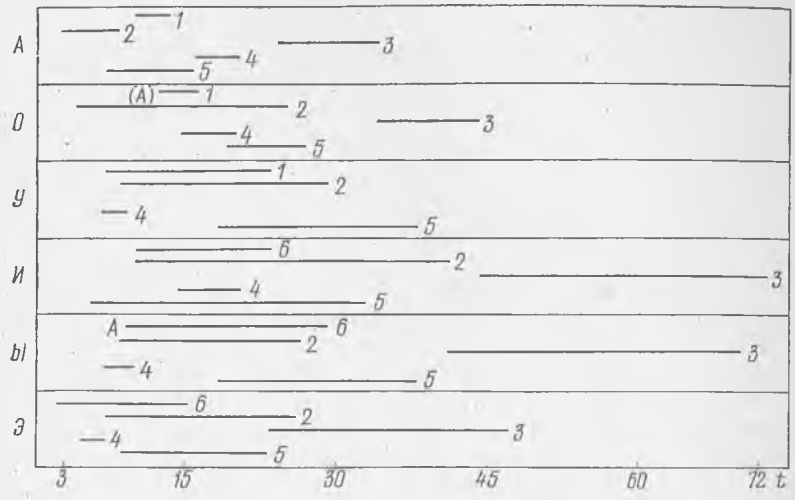
Как видно из диаграммы, все представленные на ней звуки могут быть разделены на два класса объектов по признаку β, а именно: I кл.— А, О, У; II кл.— И, Э, Ы. После этого легко сформулировать решающее правило для классификации объектов внутри каждого класса. Звук У характеризуется, например, признаком Z_2, A и О отличаются значением параметра N.

Среди объектов II класса признак А является разделительным для звука Э, в то время как И и Ы отличаются значением T.

Изучение совокупных свойств рассмотренных структурных параметров речевого сигнала, дискретизированного с учетом эффекта психофизического сглаживания в слухе, позволяет сделать вывод о целесообразности применения для его обработки на этапе анализа и классификации не вычислительных методов, а чисто логических. При этом процедура сегментации и нахо-

ждения признаков описания, организованная на аппаратурном уровне, становится тривиальной.

Для решения задачи классификации объектов в данном случае представляется удобным применение метода короткого кодирования. Представим перечисленные выше структурные пара-



метры с их признаками и значениями дисперсий временных характеристик через

$X_1, X_2, \ldots X_9$, где $Y = X_1$, $Y_\beta = X_2$, $(N < 16) = X_3$, $(N > 19) = X_4$, $Z_1 = X_5$, $Z_2 = X_6$, $(T > 140) = X_7$, $(70 < T < 100) = X_8$, $(T < 70) = A = X_9$.

Запишем разделяющие функции объектов в виде логических сумм конъюнкций, содержащих в качестве сомножителей разные переменные или их отрицание. Подстановка в конъюнкцию конкретных характеристик некоторого объекта обращает ее в 1 или 0:

$$А = X_1 \wedge \overline{X}_2 \wedge X_3 \wedge \overline{X}_4 \wedge X_5 \wedge X_6 \wedge X_7 \wedge \overline{X}_8 \wedge \overline{X}_9;$$

$$О = X_1 \wedge \overline{X}_2 \wedge \overline{X}_3 \wedge X_4 \wedge X_5 \wedge X_6 \wedge \overline{X}_7 \wedge \overline{X}_8 \wedge (X_9 \vee \overline{X}_9);$$

$$У = X_1 \wedge \overline{X}_2 \wedge \overline{X}_3 \wedge X_4 \wedge X_5 \wedge \overline{X}_6 \wedge \overline{X}_7 \wedge X_8 \wedge \overline{X}_9;$$

$$И = X_1 \wedge X_2 \wedge \overline{X}_3 \wedge X_4 \wedge X_5 \wedge (X_6 \vee \overline{X}_6) \wedge X_7 \wedge \overline{X}_8 \wedge \overline{X}_9;$$

$$Ы = X_1 \wedge X_2 \wedge \overline{X}_3 \wedge X_4 \wedge X_5 \wedge (X_6 \vee \overline{X}_6) \wedge \overline{X}_7 \wedge X_8 \wedge \overline{X}_9;$$

$$Э = X_1 \wedge X_2 \wedge \overline{X}_3 \wedge X_4 \wedge X_5 \wedge X_6 \wedge \overline{X}_7 \wedge \overline{X}_8 \wedge X_9.$$

При этом для звуков О и Ы должно выполняться условие

$$X_9 \Rightarrow (\overline{X} \wedge \overline{X}_8) \vee (\overline{X}_7 \wedge X_8) \vee (X_7 \wedge \overline{X}_8) \vee (X_7 \wedge X_8),$$

$$\overline{X}_9 \Rightarrow X_7 \vee X .$$

Разработанные алгоритмы классификации звуков в процессе эксперимента были смоделированы на программном уровне и опробованы на ЭВМ «Наири-К». «Наири-К» выбрали с целью продемонстрировать быстродействие метода даже при использовании ЭВМ такого типа. Благодаря тому, что в результате изучения структурных свойств речевого сигнала, дискретизированного с учетом эффекта психофизического сглаживания в слухе, отобран сравнительно небольшой набор параметров, отличающихся высокой стабильностью и инвариантных по отношению к диктору, от программы не требуется построения сложных функций от исходного описания объекта и сокращения перебора признаков.

Распознавание стационарных гласных звуков в эксперименте осуществлялось в процессе обработки только одной выборки длительностью 10 мс. Для 12 случайных дикторов надежность распознавания была не менее 95%. При фонемном распознавании слов метод работает весьма эффективно с использованием псевдонатурального словаря. В этом случае алфавит распознаваемых образов может превышать 800 слов. В экспериментах надежность распознавания практически не снижалась при изменении уровня акустических шумов в пределах от 18 до 70 децибел. Точность нормирования абсолютного уровня исходного речевого сигнала составляла $\pm 2\%$.

Список литературы: 1. *Tscheschner W.* Automatische Sprachverarbeitung. — Radio Fernsehen Elektronik, 1978, 27, № 12, p. 755. 2. *Mickeleit D.* — E. Magische Spielereien. — Urania — Verlag, Leipzig — Jena — Berlin, 1971. — 161 p. 3. *Tscheschner W.* Probleme der automatischen Sprachverarbeitung aus heutiger Sicht. — Nachrichtentechnik Elektronik, 1979, 29, № 1, p. 29. 4. *Kusch H.* Automatische Erkennung gesprochener Zahlen. — NTZ, 1965, Z18, p. 57—62. 5. *Zwicker E., Hess W., Terhardt E.* Erkennung gesprochener Ziffern mit Funtionmodell und elektronischer Ruchenanlage. — Kybernetik, 1967, Bd. 3, Hf. 6, p. 267—272. 6. *Бонгард М. М.* Проблема узнавания. — М.: Наука, 1967. — 245 с. 7. *Rothauser E.* Automatische Spracherkennung. — NTZ, № 7, 1967, p. 381. 8. *Fawe A. L.* Reconnaissance de la parole a partir des passages par zero. — Rev. H. F., 1978, 10, № 11—12, p. 296—310. 9. *Дрюченко А. Я.* Об одном методе автоматического распознавания в реальном масштабе времени. — Проблемы бионики, 1979, вып. 24, с. 27—31. 10. *Усенко С. А.* К вопросу о дискретности слуховой информации. — Проблемы бионики, 1979, вып. 23, с. 27—32. 11. *Харкевич А. А.* О выборе признаков при машинном опознании. — Изв. АН СССР. Техническая кибернетика, 1963, № 2, с. 48—51.

Поступила 2 октября 1979 г.

УДК 62.506.2

В. А. ЛОВИЦКИЙ, канд. техн. наук, *В. Я. ТЕРЗИЯН*

КОДИРОВАНИЕ СЛОВ В ТВ-СТРУКТУРЕ

Для решения задач морфологического анализа слов диалоговой естественноязыковой системой (ДЕСТА), разрабатываемой в Харьковском институте радиоэлектроники, была предло-

жена ТВ-структура [1]. Идея построения такой структуры основана на том факте, что слова ряда естественных языков состоят как бы из двух частей. Первая — вещественная имеет предметную отнесенность, вторая — формальная часть выражает грамматическое значение слова [2]. Например, слово *стекло* состоит из вещественной части *стекл* и формальной — *О*. Первая часть позволяет сравнивать это слово со словами *стекла*, *стеклам*,

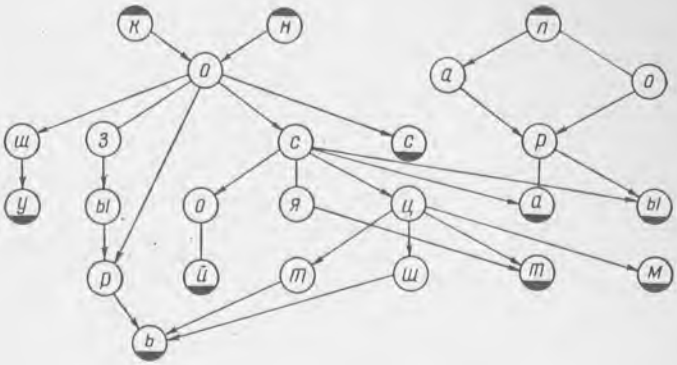


Рис. 1

стеклянный, *стекать* и т. д. Вторая говорит о грамматическом сходстве с такими словами, как *весло*, *колесо*, *бежало* и т. д. ТВ-структуру удобно представлять в виде ориентированного графа, вершины которого представляют собой буквы слов, а дуги указывают на связи этих букв в словах.

Рассмотрим принцип кодирования слов, хранящихся в ТВ-структуре. Основное требование, предъявляемое к кодам слов,— слова с одинаковой вещественной частью должны иметь одинаковые коды. Трудности, возникающие при кодировании слов, видны на конкретном примере. Пусть задано конечное множество слов *X*-{*косить*, *кошу*, *косишь*, *косит*, *косим*, *косят*, *коса*, *кос*, *косы*, *косой*, *пара*, *пора поры*, *нос*, *козырь*, *корь*}. Последовательно воспринимая слова из заданного множества, ДЕСТА строит ТВ-структуру по алгоритму, приведенному в [1]. Полученная структура представлена на рис. 1. У элементов, в которые занесены начальные буквы слова, заштрихована верхняя часть кружка, у элементов с конечными буквами слова — нижняя часть кружка. Эти элементы будем называть соответственно **начальными** и **конечными** вершинами ТВ-структуры. Элемент ТВ-структуры, в котором записана буква a_i, — a_i-элемент или элемент a_i. С каждым конечным элементом связана морфологическая информация, характеризующая класс слов с одинаковой формальной частью. Ограничим пока морфологическую информацию только вопросами, на которые отвечают слова. Так, для

нашего примера каждому конечному элементу можно поставить в соответствие следующее множество вопросов: **у** — {*что делаю*}, **ь** — {*что, что делать, что делаешь*}, **й** — {*чем, какой*}, **с** — {*чего, что*}, **а** — {*что*}, **ы** — {*что, чего*}, **т** — {*что делают, что делает*}, **м** — {*что делаем*}.

Под кодом слова будем понимать минимальный набор букв, который совместно с морфологической информацией позволяет однозначно восстановить путь в ТВ-структуре, соответствующий определенному слову. Так, для слов *кошу*, *ношу* кодами будут служить соответственно буквы **к** и **н**, так как легко видеть, что по этим кодам однозначно восстанавливаются исходные слова с помощью вопроса *что делаю*. Для слов пара и пора кодами слов будут соответственно па и по, а для слов *козырь* и *корь* — кз и кр, т. е. длина кода будет определяться числом параллельных ветвей. Вводимые правила кодирования слов с помощью ТВ-структуры должны обеспечивать систему ДЕСТА морфологической информацией. Как видно из рассматриваемого примера (рис. 1), с конечным элементом ь кроме перечисленных вопросов, на которые отвечают слова *козырь*, *корь*, *косить* и *косишь*, должна быть связана информация о роде, лице, числе и падеже. Она может попасть в ТВ-структуру только в процессе ее формирования, т. е. при встрече системой ДЕСТА незнакомого слова следует вводить вместе со словом всю необходимую морфологическую информацию. Выясним, какую морфологическую информацию считать необходимой. Для настоящего уровня развития системы ДЕСТА необходимая морфологическая информация, соответствующая каждому слову, выбирается из следующего перечня: 1) вопрос, на который отвечает слово; 2) род; 3) лицо; 4) число. Все множество вопросов можно разбить на три подмножества. В первое подмножество входят вопросы, которые однозначно определяют морфологическую информацию, не заданную в явном виде. Например, вопрос *какой*? указывает на мужской род и единственное число, а *что делают*? — на третье лицо множественного числа.

Ко второму подмножеству относятся вопросы, определяющие слова, для которых морфологическая информация должна быть определена, но не непосредственно из вопроса. Например, на вопрос *что* могут отвечать слова различного рода и числа. Для слов, отвечающих на вопросы данного класса, пользователь должен в явном виде задать недостающую морфологическую информацию. Для этого достаточно ограничиться использованием одного из четырех местоимений *он*, *она*, *оно*, *они*.

Третье подмножество включает в себя вопросы к словам, которые не требуют иной морфологической информации, кроме самого вопроса. Например, когда, как и т. д.

Если работа с омонимами не вызывает затруднений (например, косой — {*чем, какой*}), то при кодировании и декодировании омографов возникают определенные трудности. Прежде все-

го слова, значения которых зависят от ударения, должны иметь различные коды. В противном случае невозможно по коду правильно восстановить слово. Например, слова *косим* (*косим траву*) и *косим* (*косим глаза*) имеют одинаковый код к (рис. 1) по той причине, что в ТВ-структуре не учтены ударения. В дальнейшем ударение будем обозначать апострофом, который ставится перед ударной гласной буквой. Например, *к'осим*, *кос'им*.

Таким образом, при формировании ТВ-структуры в словах должны быть указаны ударения и каждое слово должно сопровождаться морфологической информацией. После предварительного рассмотрения идеи кодирования слов с помощью ТВ-структуры перейдем к формальному описанию правил кодирования. Для этого введем ряд определений.

Для формального описания входных объектов, которые определяют вид ТВ-структуры, используем металингвистическую Бэкусову нормальную форму (БНФ). В БНФ нетерминальные символы записываются как имена, заключенные в угловые скобки <>. Символ ::= читается *заменяется на*, а символ | соответствует ИЛИ и служит для разделения альтернатив замены.

Приведем формальное описание входных объектов для русского языка. Аналогичным образом можно описать входные объекты, например, для английского языка:

<входной объект> ::= <объект 1> | <объект 2>

<объект 1> ::= <строка> <разделитель> <вопрос>
<ограничитель>

<объект 2> ::= <строка> <разделитель>
<местоимение> <разделитель> <вопрос>
<ограничитель>

<вопрос> ::= ЧТО — ДЕЛАТЬ | ЧТО — СДЕЛАТЬ | ...
| ЧТО — СДЕЛАЛИ | КТО | ЧТО | КОГО | ... | О ЧЕМ |
| КАКОЙ | | ЧЕЙ | ... | КАКИЕ | ЧЬИ | КОГДА | ... | ГДЕ |

<местоимение> ::= ОН | ОНА | ОНО | ОНИ

<ограничитель> ::= Ь (где Ь — обозначает пробел)

<разделитель> ::= *

<строка> ::= <русская буква> | [<строка>
<русская буква>]$_1^{30}$

<русская буква> ::= А | Б | В ... | Я

Для обеспечения «понимания» системой ДЕСТА все вопросы, на которые отвечают глаголы, записываются через черточку. Это позволяет избежать неоднозначности в ответе на вопрос типа: Что делает Петя? Возможные ответы: самолет или читает.

Теперь же, для получения второго ответа, вопрос должен быть записан так: Что — делает Петя?

Согласно введенному описанию преобразуем конечное множество входных объектов X следующим образом:

*X = {кос'ить * что-делать, кош'у * что-делаю, к'осишь * что-делаешь, кос'ишь * что делаешь, к'осит * что-делает, кос'ит * что-делает, кос'им * что-делаем, к'осим * что-делаем, к'осят * что-делают, кос'ят*что-делают, кос'а * она * что, кос'ы * она * чего, к'осы * они * что, к'ос * они * чего, кос'ой * какой, кос'ой * она * чем, п'ара * она * * что, п'ора * она * что, пор'а * она * что, п'оры * они *что, н'ос* он * что, к'озырь * он * что, к'орь * она * что}*

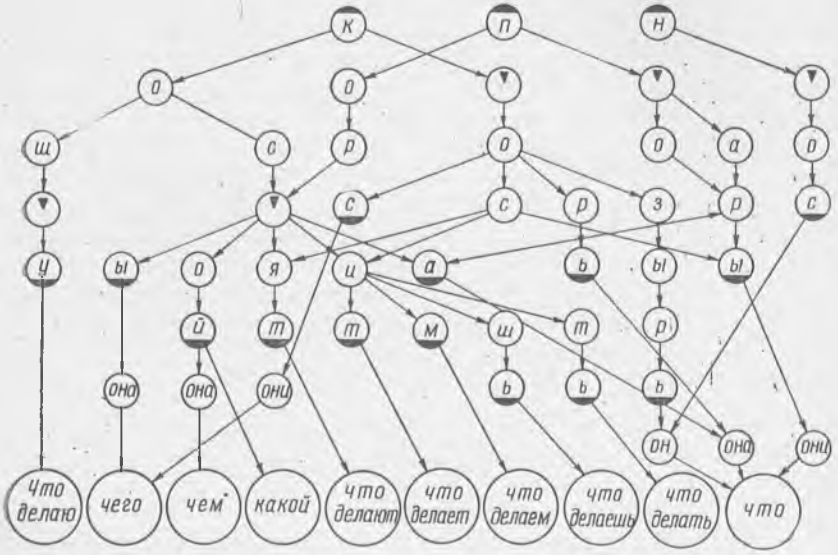


Рис. 2

Последовательно воспринимая входные объекты, системы ДЕСТА сформирует ТВ-структуру, представленную на рис. 2.

В общем случае будем считать, что входные объекты состоят из символов a_i, составляющих алфавит $\Gamma = \{a_1, a_2, \ldots, a_n\}$. Всякую конечную последовательность символов алфавита Γ будем называть **строкой (или цепочкой). Длина** строки равна числу символов в строке. Строка, которая не содержит ни одного символа, **называется пустой** строкой и обозначается символом λ.

Тот факт, что в строке x символ a_j следует за a_i, будем обозначать через $a_i \underset{x}{\rightarrow} a_j$, где символ $\rightarrow$ читается следует за, или предшествует. Непосредственное следование символа a_j за a_i

обозначим через $a_i \overset{0}{\underset{x}{\rightarrow}} a_j$, или $a_i a_j$. Если в строке $a_2 a_5 a_1 a_4 a_7$ важным является не просто сам факт следования a_7 за a_2 (в этом случае достаточно записать $a_2 \rightarrow a_7$), а то, что символ a_7 следует за a_2 через три символа, то в этом случае используется обозначение $a_2 \overset{3}{\underset{x}{\rightarrow}} a_7$. Если a_i является начальным символом строки x, а a_j — конечным, то это будем обозначать соответственно через $в \overset{0}{\underset{x}{\rightarrow}} a_i$ и $a_j \overset{0}{\underset{x}{\rightarrow}} e$, где e обозначает пустой символ.

Определение 1. Формально **строки в алфавите** Г определяются следующим образом: 1) λ строка в Г; 2) если x — строка в Г и $a_i \in$ Г, то xa_i — строка в Г; 3) x_j — строка в Г тогда и только тогда, когда она является таковой в силу 1) или 2).

Определение 2. Пусть x_i и x_j две произвольные строки. Будем говорить, что x_j **входит** в x_i, если имеет место равенство $x_i = v_1 x_j v_2$, где v_1 и v_2 — некоторые строки символов, возможно, равные λ.

Включение x_j в x_i обозначим через $x_j \subset x_i$; где символ $\subset$ читается как *входит в*.

Определение 3. Множество элементов ТВ-структуры, к которым на графе имеются пути от a_i элемента, называется **субмножеством** a_i элемента и обозначается как $SBS a_i$.

Определение 4. Множество элементов ТВ-структуры, от которых на графе имеются пути к a_i элементу, называется **супермножеством** a_i элемента и обозначается как $SPS a_i$.

Определение 5. Элементы из $SBS a_i$, связанные с a_i элементом непосредственно, образуют **0 — субмножество элемента** a_i, или $SBS^0_{a_i}$.

Определение 6. Элементы из $SPS a_i$, связанные с a_i элементом непосредственно, образуют **0 — супермножество** элемента a_i, или $SPS^0 a_i$.

Правило кодирования строки символов с помощью ТВ-структуры вводится следующим определением.

Определение 7. Кодом строки символов x будет служить также строка символов, которая составляется согласно правилам:

1) первым символом кода строки символов x является такой символ a_i, что

$$\left((a_i \subset x) \,\&\, \left(e \overset{0}{\underset{x}{\rightarrow}} a_i\right)\right) \vdash (SPS^0 a_i = \varnothing),$$

где & — обозначает операцию конъюнкций, а $\vdash$ читается как *дает* или *приводит к*;

2) в качестве последующих символов кода последовательно берутся такие символы a_j, что

$$\exists a_i \subset x \left(\left(a_i \overset{0}{\underset{x}{\rightarrow}} a_j \right) \& (|\mathrm{SBS}^0 a_i| > 1) \&,\right.$$

$$\left. \& \exists a_\nu \in (\mathrm{SBS}^0 a_i - a_j) \exists a_\mu \in \mathrm{SBS} a_\nu (a_\mu \subset x) \right),$$

где $|\mathrm{SBS}^0 a_i|$ — мощность множества $\mathrm{SBS}^0 a_i$, $\exists$ — квантор существования, а $\mathrm{SBS}^0 a_i - a_j$ — разность двух множеств: $\mathrm{SBS}^0 a_i$ и множества, представленного единичным элементом a_j.

Легко видеть, что длина кода строки символов x определяется числом символов исходной строки, для которых выполняется правило 2), и плюс начальный символ.

Пользуясь данными правилами кодирования и ТВ-структурой (рис. 2), получим для слов, заданных конечным множеством X, следующие коды. Множеству слов {*кос'ить, кос'ишь, кос'ит, ко'сим, кос'ят, кос'а, кос'ы, кос'ой*} соответствует код, КО, множеству {*к'осишь, к'осит, к'оси, к'осят, к'осы, к'ос*} — код К, слову: *кош'у*: — код К, слову *п'ара* — код П'А, словам {*п'ора, п'оры*} — код П'О, слову *пор'а* — код ПО, слову *н'ос* — код Н, слову *к'озырь* — код К'З и слову *к'орь* — код К'Р.

Непосредственное использование кодов слов при переходе от поверхностной структуры предложения к глубинной невозможно по той причине, что коды слов будут изменяться при расширении ТВ-структуры. Покажем это на примере. Пусть задано конечное множество слов X = {*приносить* * *что-делать, просить* * *что-делать, приходить* * что-делать, пилить * что-делать, придавать * что-делать, носить * что-делать, давать * что-делать. Ударения в словах не указаны, поскольку они не влияют в данном случае на коды слов. Проследим, как изменяется код слова *приносить* по мере добавления в ТВ-структуру других слов из заданного множества X. Соответствующая этим словам ТВ-структура отражена на рис. 3, а результаты изменения кодов под воздействием первых пяти входных объектов даны ниже. Входные объекты: приносить, просить, приходить, пилить, придавать

Коды слова *приносить*: П ПИ ПИН ПРИН ПРИН

Как видно, код слова *приносить* меняется с каждым новым словом, не считая слова *придавать*. Для того чтобы коды, используемые в глубинной структуре, не изменялись, введем множество C, элементами которого будут коды слов, полученные с помощью ТВ-структуры. В глубинной структуре предложения кодами слов служат их номера в множестве C. Перед номером в качестве разделителя ставится символ *. Код слова *приносить* обозначается *1, *просить* — *2 и т. д. После ввода в ТВ-структуру слова *приходить*, множество C имеет вид: C = {ПИН, ПО, ПИХ}, после добавления слова *придавать* —

$C = \{$ПРИН, ПРО, ПРИХ, ПИ, ПРИД$\}$, коды этих слов в глубинной структуре записываются соответственно $*1$, $*2$, $*3$, $*4$, $*5$.

Итак, добавление новых слов в ТВ-структуру может привести к изменению кодов слов, ранее включенных в ТВ-структуру. Но при этом имеет место интересная закономерность, чем больше слов содержится в ТВ-структуре, тем меньше кодов требуют коррекции после ввода в ТВ-структуру нового слова.

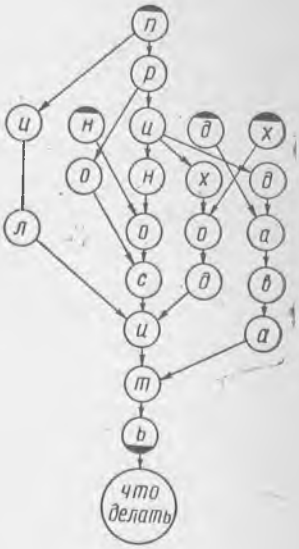


Рис. 3

Определение 8. Пусть код новой строки символов x, введенной в ТВ-структуру, будет $\mathrm{cod}(x)$. Тогда эта строка символов не приведет к коррекции кодов строк, уже записанных в ТВ-структуре, если имеет место следующее утверждение:

$$((a_i \leftarrow \mathrm{cod}(x)) \,\&\, (a_l \overset{0}{\underset{\mathrm{cod}(x)}{\rightarrow}} e)) \vdash (\exists a_j \leftarrow x \times$$

$$\times ((a_j \overset{0}{\underset{x}{\rightarrow}} a_i) \,\&\, (|\mathrm{SBS}^0 a_j| > 2)).$$

Включение в ТВ-структуру слова *придавать* не привело к коррекции кодов уже записанных слов по той причине, что $SBS^0_{\text{И}} = \{$Н, Х, Д$\}$, т. е. мощность этого множества больше двух.

Еще одну особенность кодирования слов продемонстрируем на следующем примере. Пусть задано множество $X = \{$*колбаса* $*$ $*$ *она* $*$ *что* $*$ *колба* $*$ *она* $*$ *что*$\}$. Легко представить соответствующую элементам множества X ТВ-структуру и коды этих слов: КА и КА, т. е. cod (*колбаса*) = КА и cod (*колба*) = КА. Буква А в коде слова *колба* отличается от буквы А в коде слова *колбаса* тем, что она является конечной. Отмечая эту букву как А', мы тем самым отличим код слова *колбаса* — КА' от кода первого слова — КА, т. е. в первом случае будем писать cod (*колбаса*) = КА, а во втором — cod (*колба*) = КА'. В общем случае для строки x это записывается следующим образом:

$$\left((a_i \leftarrow \mathrm{cod}(x)) \,\&\, \left(a_i \overset{0}{\underset{x}{\rightarrow}} e\right)\right) \vdash \mathrm{cod}'(x).$$

Рассмотренная система кодирования слов позволяет сохранять и фиксировать семантическую информацию, которая вносится в значение слова его приставкой. Характерно, что по мере расширения словаря системы ДЕСТА, а следовательно, и росте ТВ-структуры, происходит коррекция кодов, что в конце концов приведет к включению приставок в коды слов. Коды слов *при-*

носить, *приходить* и *придавать* включают в себя, в качестве начальных букв, приставку *при* (рис. 3). Умение системы ДЕСТА автоматически выделять приставку и связывать с ней семантическую информацию позволит приблизиться к решению задачи образования с помощью приставок новых слов. Для формального определения понятия приставка введем следующие определения.

Определение 9. Если x_1, x_2 и x_1x_2 — строки символов то x_2 называется *левым частным от деления* строки x_1x_2 на x и обозначается как $x_1 \setminus x_1x_2$, а x_1 называется *правым частным от деления* строки x_1x_2 на x_2 и обозначается через x_1x_2/x_2.

Определение 10. Пусть x_1 и x_2 строки символов, сопровождаемые одинаковой морфологической информацией, которые включены в ТВ-структуру. Тогда *приставку* определим как x_1/x_2, если выполняется следующее условие:

$$\exists a_i \subset x_1 \exists a_j \subset x_2 \left((a_i = a_j) \,\&\, \left(a_j \overset{0}{\underset{x_2}{\rightarrow}} e\right) \,\&\, (\mathrm{SBS}^0 a_j = \mathrm{SBS}^0 a_i) \,\&\, \right.$$

$$\left. \&\, ((a_j \setminus x_2) \subset x_1)\right).$$

Легко видеть, что $(x_1/x_2) \subset \mathrm{cod}(x_1)$ будем иметь место в том случае, когда $\forall a_i \subset (x_1/x_2)\,(|SBS^0 a_i)| > 1)$.

ТВ-структура позволяет также автоматически выделять суффиксы и использовать их при словообразовании.

Таким образом, кодирование слов в ТВ-структуре позволяет не только получать одинаковые коды различных словоформ, но и предоставляет возможность выделить приставки и суффиксы для использования их при словообразовании.

Список литературы: 1. *Ловицкий В. А.* Структурный подход к решению морфологических задач. — Проблемы бионики, 1980, вып. 25, с. 39—43. 2. *Глозман Ж. М.* Исследование нарушения, лингвистического отношения к слову при афазии.— Психологические исследования, 1974, вып. 6, с. 77—87.

Поступила 22 февраля 1979 г.

УДК 681.3.06

С. К. КОЛУБАЙ

ГРАФ-СХЕМЫ ПРОГРАММ НА АСИНХРОННОМ ЯЗЫКЕ ТИПА $<i, k, j, l>$

Рассматривается графическое представление параллельных программ на асинхронном языке типа $<i, k, j, l>$ [7] и их выполнение в графовой форме.

В практике последовательного программирования широко используется графическое представление программ в виде блок-схем алгоритмов. Соответствующую графовую форму имеют различные типы программ (схем программ) последовательных [9]

и параллельных (асинхронных), например, схемы потока данных [1], параллельные операторные схемы [4], граф-схемы параллельных алгоритмов [10] и т. д.

Достоинством такого представления является наглядность связей между операторами (командами) программы по управлению или по данным, а также возможность получения достаточно точных выводов относительно свойств программ при меньших затратах труда, чем в случае использования других форм представления, например, линейной.

Аналогичное графическое представление умеют асинхронные параллельные программы на языке типа $<i, k, j, l>$. Программа представляется в виде графа с помеченными дугами. Вершинами графа служат инструкции программы, а дуги указывают связи между ними по номерам j и l, причем каждая дуга помечается соответствующим номером i (или $-i$).

Более точно, пусть программа $P = \{a_0, a_1, \ldots, a_n\}$ **(1)** на $<i, k, j, l>$-языке [7] содержит $n+1$ инструкцию, причем каждая инструкция $a_r (r = 0, 1, \ldots, n)$ — это набор $a_r = (a_r^i, a_r^k, a_r^j, a_r^l)$, где a_r^i — номер, a_r^k — команда инструкции, а a_r^j, a_r^l — номера, задающие ее связи по данным и управлению. В программу P введена *начальная инструкция* $a_0 = (z, \boldsymbol{start}, 0, 0)$, в которой z — номер параллельных точек входа, а $\boldsymbol{start}$ — специальный символ языка. Номера $a_1^i, \ldots, a_n^i$ — натуральные числа, а номера $a_1^j, a_1^l, \ldots, a_n^j, a_n^l$ — целые числа, не равные нулю. Командой a_r^k инструкции a_r $(r = 1, \ldots, n)$ может быть либо команда обработки или ввода — вывода (a_r — инструкция обработки), либо команда, вычисляющая значение некоторого предиката (a_r — инструкция управления), причем a_r^k может быть нульместным предикатом, задаваемым специальным символом языка: $\boldsymbol{true}$ или $\boldsymbol{false}$.

Та же программа в графовой форме (или *граф-схема программы*) — это ориентированный граф $G = (P, H)$ **(2)** со множеством P (1) вершин и множеством H дуг. Из вершины $a = (a^i, a^k, a^j, a^l)$ в вершину $b = (b^i, b^k, b^j, b^l)$ ведет дуга, т. е. $(a, b) \in H$, в том и только том случае, если: 1) $a^i = b^j$ (или $a^i = b^l$) и $a^k \neq \boldsymbol{false}$ (эта дуга помечается номером a^i); 2) $-a^i = b^j$ (или $-a^i = b^l$) и a — инструкция управления, причем $a^k \neq \boldsymbol{true}$. Эта дуга помечается номером $-a^i$.

Дуга $(a, b) \in H$ называется *выходной дугой* вершины a или *входной дугой* вершины b.

Таким образом, в *начальную вершину* a_0 графа не заходит ни одна дуга, так как $a_0^j = a_0^l = 0$, а в программе P нет ни одной инструкции с номером, равным нулю. В отличие от граф-схем последовательных программ число выходных дуг всякой вершины граф-схемы G (2) заранее не ограничивается, т. е. вершина может иметь любое их количество, либо не иметь вовсе.

Это зависит только от программы P. Все выходные дуги *вершины-преобразователя* a, являющейся инструкцией обработки, помечены номерами a^i, а выходные дуги *вершины-распознавателя* a, являющейся инструкцией управления, помечены номерами a^i (или $+a^i$), и $-a^i$. Исключение составляют вершины-распознаватели с командами константами ***true*** и ***false***: вершина-распознаватель a с командой $a^k = \boldsymbol{true}$ имеет выходные дуги, помеченные только номерами $+a^i$, а с командой $a^k = \boldsymbol{false}$ — только номером $-a^i$.

Нами используются традиционные названия вершин графа: вершина-преобразователь и вершина-распознаватель, впервые введенные в работах [2, 3]. При изображении граф-схемы программы на рисунках начальная вершина и вершины-преобразователи показываются прямоугольниками, а вершины-распознаватели — овалами. Внутри фигуры, изображающей вершину, пишется команда инструкции, являющейся этой вершиной. Если вершина не имеет выходных дуг, то также указывается номер инструкции. Дуги, как обычно, изображаются стрелками, причем номер, помечающий дугу, выписывается рядом с ней.

На рис. 1 показана граф-схема программы решения следующего примера: решить n квадратных уравнений $a_i x^2 + b_i x + c = 0$ $(i = 1, 2, \ldots, n)$ для общего случая. Комплексное решение представить в виде модуля и аргумента комплексно сопряженных корней. Корни $k_{1,i}$ и $k_{2,i}$ каждого i-го уравнения находятся по формулам: $k_{1,i} = p + q$ и $k_{2,i} = p - q$, если $b_i^2 - 4a_i c_i \geqslant 0$; $k_{1,i} = \sqrt{p^2 + q^2}$ и $k_{2,i} = \arcsin \frac{q}{\sqrt{p^2 + q^2}}$, если $b_i^2 - 4a_i c_i < 0$;

здесь $p = -\frac{b_i}{2a_i}$ и $q \frac{\sqrt{|b_i^2 - 4a_i c_i|}}{2a_i}$.

Пример выбран в связи с тем, что его решение описывается $< i, k, j, l >$ — программой достаточно общего вида, т. е. содержащей цикл, в теле которого имеются логические и параллельные ветви.

Программа $P = \{a_0, a_1, \ldots, a_{26}\}$ решения уравнений имеет вид

$P = \{$(25, *start*; 0)$_0$,
(1, ввод (n, a, b, c); 25)$_1$,
(2, i: = 1; 25)$_2$,
(3, *true*; 1,2)$_3$,
(4, $r1$: = 2 × a [i]; 3)$_4$,
(5, $r2$: = 2 × c [i]; 3)$_5$,
(6, $r2$: = $r1$ × $r2$, 4,5)$_6$,
(7, $r3$: = b [i] ↑ 2; 3)$_7$,
(8, p: = b [i]/$r1$; 4)$_8$,
(9, d: = $r3$ − $r2$; 6,7)$_9$,
(10, $r2$: = *abs* (d); 9)$_{10}$,
(13, d < 0; 8,12)$_{13}$,
(14, $r1$: = p ↑ 2; + 13)$_{14}$,
(15, $r2$: = q ↑ 2; + 13)$_{15}$,
(16, $r1$: = $r1$ + $r2$; 14, 15)$_{16}$,
(17, $k1$ [i]: = *sqrt* ($r1$); 16)$_{17}$,
(18, $r1$: = q/$k1$ [i]; 17)$_{18}$,
(19, $k2$ [i]: = *arcsin* ($r1$); 18)$_{19}$,
(20, $k1$ [i]: = p + q; − 13)$_{20}$,
(21, $k2$ [i]: = p − q; − 13)$_{21}$,
(19, *true*; 20, 21)$_{22}$,
(22, s [i]: = *sign* (d); 9)$_{23}$,

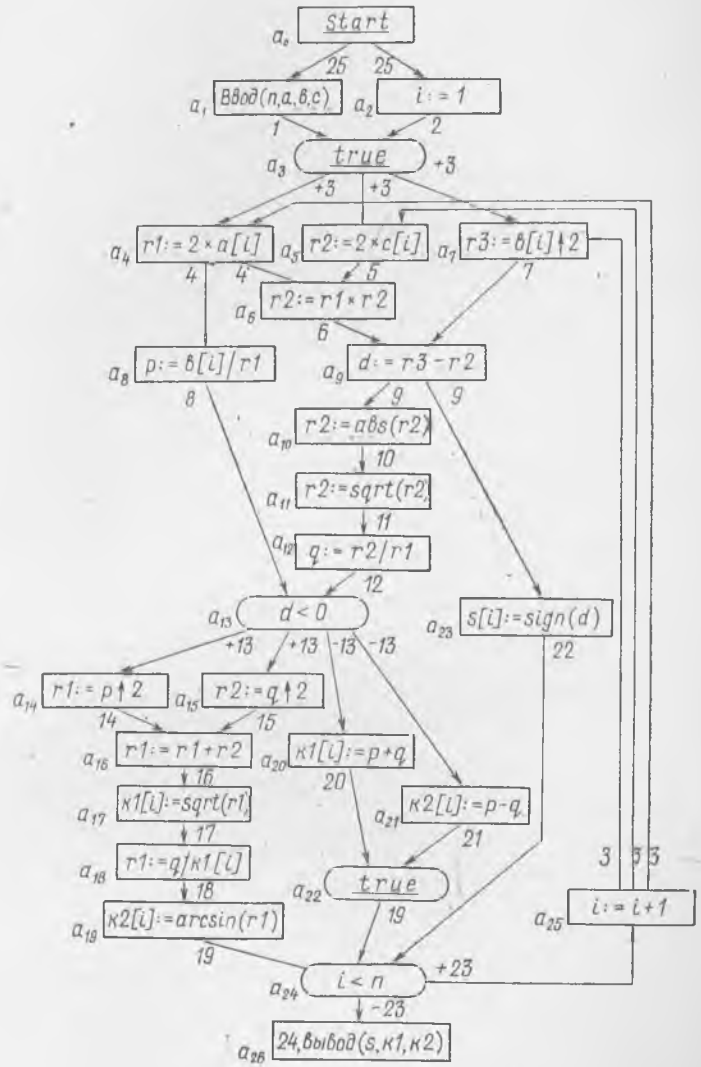

a0 Start
25
25
a1 Ввод(n,a,b,c)
a2 i := 1
1
2
a3 true
+3
+3
+3
a4 r1 := 2 × a[i]
a5 r2 := 2 × c[i]
a7 r3 := b[i] ↑ 2
4
4
5
7
a6 r2 := r1 × r2
6
a8 p := b[i] / r1
a9 d := r3 - r2
8
9
9
a10 r2 := abs(r2)
10
a11 r2 := sqrt(r2)
11
a12 q := r2 / r1
12
a13 d < 0
a23 s[i] := sign(d)
+13
+13
-13
-13
22
a14 r1 := p ↑ 2
a15 r2 := q ↑ 2
14
15
a16 r1 := r1 + r2
a20 к1[i] := p + q
16
20
a17 к1[i] := sqrt(r1)
a21 к2[i] := p - q
17
21
a18 r1 := q / к1[i]
3
3
3
a22 true
a25 i := i + 1
18
19
a19 к2[i] := arcsin(r1)
19
a24 i < n
+23
-23
a26 24, Вывод(s, к1, к2)

Рис. 1

$(11,\ r2:=sqrt(r2);\ 10)_{11}$,
$(12, q:=r2/r1;\ 11)_{12}$,
$(23,\ i<n;\ 19,\ 22)_{21}$,
$(3,\ i:=i+1;+23)_{25}$,
$(24,$ вывод $(s,\ k1,\ k2);-23)_{26}\}$.

В программе вводятся массивы a, b и c коэффициентов уравнений и число n компонент в этих массивах. Выводятся массивы $k1$, $k2$ и s по n компонент каждый, где $k1$ и $k2$ — массивы корней, а s — массив признаков, характеризующих типы корней, причем $s_i=1$, если корни i-го уравнения являются действительными, и $s_i=-1$, если корни — комплексные.

Каждая инструкция a_r (r=0,1, ..., 26) программы записана в виде: $a_r=(a_r^i,\ a_r^k;\ a_r^j,\ a_r^l)_r$, если $a_r^j\neq a_r^l$, или $a_r=(a_r^i,\ a_r^k;\ a_r^j)_r$, если $a_r^j=a_r^l$. Команды инструкций даны в алголоподобной нотации.

В программе P две параллельные точки входа — это инструкции a_1 и a_2. Номер параллельных точек входа равен 25.

На рис. 1 хорошо видно, что инструкции a_1 и a_2 находятся в параллельных ветвях, а точнее — в совместных, следуя терминологии языка Алгол-68 [11]. Программа не предписывает обязательное параллельное их исполнение, а указывает, что эти ветви могут выполняться в любом порядке и параллельно, т. е. асинхронно.

Более точно, пусть S_1, S_2, ... , S_n — ячейки процессоро-памяти [7], в которые записаны инструкции программы (1), причем в ячейке S_r записана инструкция a_r $(r=1,\ \ldots,\ n)$. Тогда будем говорить, что инструкции a_r и a_q $(1\leqslant r,\ q\leqslant n)$ являются *совместными* (находятся в *совместных ветвях*), если во время выполнения программы возможна ситуация, при которой ячейки S_r и S_q одновременно находятся в состоянии «готова».

Инструкции a_1 и a_2 программы примера демонстрируют самый простой случай совместных ветвей: они одновременно получают разрешение на выполнение (при поступлении на процессоро-память номера 25 ячейки, хранящие эти инструкции, перейдут в состояние «готова») и следующая инструкция a_3 может начать вычисляться только после того, как обе инструкции a_1 и a_2 реализуются.

Более сложный случай совместных ветвей демонстрируют инструкции $a_4 \div a_{23}$ тела цикла (рис. 1). Так после срабатывания a_4, инструкция a_8 может выполняться в любом порядке или параллельно с любой из инструкций $a_5 \div a_7$, $a_9 \div a_{10}$ и a_{23}.

После реализации инструкции a_{13} разрешается выполнение только одной из взаимно исключающих (логических) ветвей, состоящих из инструкций $a_{14} \div a_{19}$ и $a_{20} \div a_{22}$, соответственно, причем каждая из ветвей содержит совместные инструкции. В первой ветви — это a_{14}, a_{15}, а во второй — a_{20}, a_{21}.

При входе в тело цикла, то есть после срабатывания инструкции a_3 (или a_{25}), разрешается выполнение сразу трех совместных инструкций a_4, a_5 и a_7, так как при поступлении но-

мера 3 на процессоро-память ячейки, содержащие эти инструкции перейдут в состояние «готова».

Таким образом, на рис. 1, изображающем граф-схему программы примера, гораздо лучше видны логические и совместные (параллельные) ветви, чем в записи той же программы в виде множества.

При рассмотрении $<i, k, j, l>$-программ в графовой форме полезно иметь правила их выполнения, не связанные с такими элементами реализации в параллельных вычислительных системах [5] как процессоро-память, ее ячейки и т. д. Эти правила должны определять те же вычислительные процессы аналогично тому, как имеются правила выполнения последовательных программ в их графовой форме (см., например, [2]), которые не зависят ни от счетчика адреса команд, ни от оперативной памяти с линейной организацией ячеек и тому подобных средств реализации этих программ в последовательных вычислительных машинах. Наличие таких правил упрощает разработку и анализ программ, а также позволяет программировать без знания средств аппаратной реализации.

Важным свойством $<i, k, j, l>$-программ, как, впрочем, и других асинхронных программ (например, А-программ [8]), является то, что выполнять их можно различными способами. При соответствующих алгоритмах реализации программ могут быть получены вычислительные процессы, являющиеся параллельными или последовательными, двумерными или одномерными и так далее (классификация вычислительных процессов по работе [12]). Например, в статье [6] описана модель вычислений, основанная на $<i, k, j, l>$-программах, в которой допускаются не более, чем последовательные недетерминированные вычислительные процессы. В то же время наиболее общий случай процессов, которые позволяют реализовать эти программы — это неприведенные двумерные недетерминированные процессы без автопараллельности.

Определим начальное состояние граф-схемы, условие готовности вершин и правило выключения вершин.

Начальное состояние граф-схемы удовлетворяет требованиям: (1) ни одна из вершин графа G не выполняется, и (2) зажжены выходные дуги *начальной вершины* a_0, а все остальные дуги граф-схемы погашены.

Условие готовности: вершина $a=(a^i, a^k, a^j, a^l)$ является готовой в том и только в том случае, если она не выполняется и среди ее входных дуг зажжена хотя бы одна дуга, помеченная номером a^j и хотя бы одна дуга, помеченная номером a^l, если $a^j \neq a^l$.

Правило выключения: при выключении любой выполняющейся вершины $a=(a^i, a^k, a^j, a^l)$ гасятся все ее входные дуги и если: a — вершина-преобразователь, то зажигаются все ее выходные дуги; a — вершина-распознаватель, то зажигаются выходные

дуги, помеченные номером $+a^i$ или $-a^i$, причем первый случай имеет место, если $a^k = \mathbf{true}$ или результат выполнения a^k равен **true**, а второй, если $a^k = \mathbf{false}$ или результат выполнения команды a^k равен **false**.

Реализация граф-схемы описывается без уточнения понятия выполнения команд инструкций, так как мы предполагаем, что оно достаточно ясно интуитивно. При необходимости можно основываться на одном из известных (см., например, в [9]).

Неприведенные двумерные недетерминированные вычислительные процессы удобно рассматривать протекающими во времени t, которое принимает значения в области неотрицательных действительных чисел. Свойство неприведенности обусловливает то, что готовая вершина, включаемая в момент времени $\bar{t} > 0$, выполняется (работает) некоторый конечный отрезок времени $\tau > 0$ и в момент времени $\bar{\bar{t}} = \bar{t} + \tau$ выключается. При этом считается, что ее команда в момент $\bar{\bar{t}}$ выполнилась со значениями аргументов, которые могли быть взяты в любой момент в полуоткрытом интервале $[\bar{t}, \bar{\bar{t}})$. Свойство двумерности определяет возможность одновременного включения и выключения непустых множеств вершин во всякий момент времени. Свойство недетерминированности позволяет включать в каждый момент времени любые вершины из множества готовых вершин.

Реализация неприведенных двумерных недетерминированных вычислительных процессов без автопараллельности при выполнении граф-схемы определяется следующим образом: 1) в момент времени $t = 0$ граф-схема имеет начальное состояние; 2) в любой момент времени $t > 0$, если множество готовых вершин не пусто, включается его произвольное подмножество вершин, и, если множество работающих вершин не пусто, выключаются все вершины, у которых к этому моменту времени закончился период выполнения, т. е. $\bar{\bar{t}} \leqslant t$; 3) повторяется п. 2) до тех пор, пока непусто множество готовых или работающих вершин граф-схемы.

Процесс выполнения граф-схемы, т. е. динамику зажигания и гашения дуг, включения и выключения вершин и их текущие свойства (готовая, выполняющаяся), можно представлять и анализировать при помощи графика вычислительного процесса. График строится в прямоугольных координатах: на непрерывной горизонтальной оси времени и дискретной вертикальной оси, являющейся объединением множества H дуг со множеством $P \setminus \{a_0\}$ вершин без начальной. Элементы вертикальной оси упорядочиваются снизу вверх в следующем порядке: входные дуги вершины a_1, a_1, ..., входные дуги вершины a_n, a_n. Каждая вершина вертикальной оси обозначается на ней маленьким кружком, а дуга — стрелкой с номером, помечающем дугу.

На рис. 2, *а* показан фрагмент графика вычислительного процесса, который может быть получен при выполнении граф-

схемы примера (рис. 1). Горизонтальным отрезком сплошной прямой линии на уровне вершины обозначается интервал времени, в течение которого она является готовой, а отрезком волнистой линии — работающей. Интервал времени от момента зажигания дуги до момента ее гашения изображается отрезком прямой штриховой линии на уровне дуги.

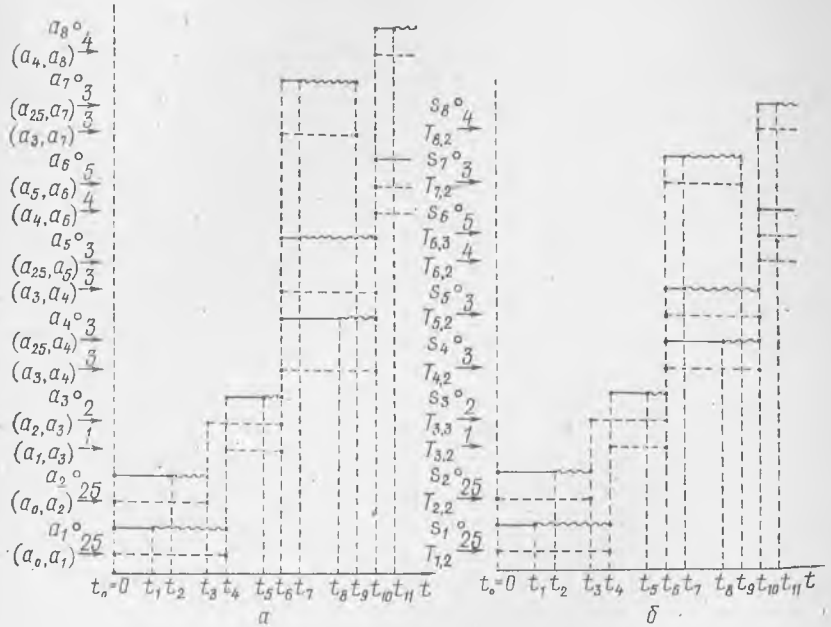


Рис. 2

Рассмотрим соответствие между реализацией граф-схемы G (2) и выполнением ее программы P (1) в параллельной вычислительной системе [5] при помощи процессоро-памяти с ячейками типа [7], чтобы убедиться, что графики вычислительных процессов, получаемых в обоих случаях, эквивалентны.

Пусть вершины $a_1, a_2, \ldots, a_r, \ldots, a_n$ — граф-схемы записаны в ячейках $s_1, s_2, \ldots, s_r, \ldots, s_n$ процессоро-памяти соответственно. Напомним, что начальная вершина a_0 в процессоро-память не записывается, а определяет номер z параллельных точек входа программы. Зажиганию выходных дуг вершины a_r поставим в соответствие инициирование программы P путем передачи номера z на процессоро-память; включению вершины a_r — выработку управляющего сигнала «чтение» на ячейку s_r, а выключению вершины a_r — передачу на процессоро-память номера x (либо $+ a_r^i$, либо $- a_r^i$), равного номеру, помечающему выходные дуги вершины a_r, которые были зажжены при этом выключении a_r. Тогда имеют место следующие утверждения.

1. Вершина a_r является готовой тогда и только тогда, когда ячейка s_r находится в состоянии «готова».

2. Вершина a_r выполняется тогда и только тогда, когда ячейка s_r находится в состоянии «выполняюсь».

Доказательство этих утверждений базируется на следующих двух вспомогательных утверждениях:

3. Триггер T_2 (T_3) ячейки s_r устанавливается в единичное состояние при инициировании программы P тогда и только тогда, когда входная дуга вершины a_r, помеченная номером a_r^i (a_r^l), зажжена в начальном состоянии граф-схемы G.

4. Триггер $T_2(T_3)$ ячейки s_r находится в единичном состоянии в том и только в том случае, если хотя бы одна входная дуга вершины a_r, помеченная номером $a_r^i(a_r^l)$, зажжена.

Доказательство утверждения (3). Действительно, если при инициировании программы P триггер $T_2(T_3)$ ячейки s_r в результате поступления на нее номера z установится в единичное состояние, то $z = a_r^l$ ($z = a_r^l$). Учитывая, что $a_0^i = z$, имеем $(a_0, a_r) \in H$, помечена номером $a_r^i(a_r^l)$ и в начальном состоянии граф-схемы зажжена, так как является выходной дугой начальной вершины a_0. Обратно, если в начальном состоянии граф-схемы входная дуга вершины a_r, помеченная номером $a_r^i(a_r^l)$, зажжена, то $(a_0, a_r) \in H$ и $a_r^i = z$ ($a_r^l = z$). Отсюда непосредственно следует, что передача номера z на процессоро-память в результате инициирования программы P приводит к установке в единичное состояние триггера $T_2(T_3)$ ячейки s_r. Утверждение (3) доказано.

Доказательство утверждения (4). Предположим, что триггер $T_2(T_3)$ ячейки s_r находится в единичном состоянии. Следовательно, на процессоро-память был передан номер $x = a_r^i$ ($x = a_r^l$). Если $x = z$, то дуга (a_0, a_r), помеченная номером $a_r^i(a_r^l)$, зажжена (см. доказательство утверждения (3)). Если же $x \neq z$, то имело место выключение некоторой вершины a_q $(1 \leqslant q \leqslant n)$, в результате которого зажглись все ее выходные дуги, помеченные номерами x. Поэтому, либо $a_q^i = x$, либо $-a_q^i = x$, и из $x = a_r^l$ ($x = a_r^l$) следует, что $(a_q, a_r) \in H$, является входной дугой вершины a_r, помечена номером $a_r^i(a_r^l)$ и зажжена. Обратно, предположим, что зажжена хотя бы одна входная дуга вершины a_r, помеченная номером $a_r^i(a_r^l)$. Если эта дуга ведет от начальной вершины a_0, т. е. $(a_0, a_r) \in H$, помечена номером $a_r^i(a_r^l)$ и зажжена, то триггер $T_2(T_3)$ установлен в единичное состояние (см. доказательство утверждения (3)). Если же эта дуга ведет от некоторой вершины a_q $(1 \leqslant q \leqslant n)$, то имело место выключение вершины a_q с зажиганием выходных дуг, помеченных номерами $x = a_r^i$ ($x = a_r^l$). Выключению вершины a_q соответствует передача на процессоро-память номера, которым помечены зажженные вы-

-ходные дуги. Следовательно, в результате передачи номера $x = a_r^j (x = a_r^l)$ на процессоро-память, триггер $T_2(T_3)$ установится в единичное состояние. Утверждение (4) доказано.

Доказательство утверждения (1). По условию готовности вершина a_r является готовой тогда и только тогда, когда среди ее входных дуг зажжена хотя бы одна дуга, помеченная номером a^j, и хотя бы одна дуга, помеченная номером a^l, если $a^j \neq a^l$. Отсюда и из утверждения (4) имеем, что триггеры T_2 и T_3 находятся при этом в единичном состоянии. Поэтому ячейка s_r находится в состоянии «готова». Утверждение (1) доказано.

Доказательство утверждения (2). Из исходных соответствий следует, что при включении готовой вершины a_r на ячейку s_r, находящуюся в состоянии «готова» (см. утверждение (1)) передается управляющий сигнал «чтение». В результате ячейка переходит в состояние «выполняюсь». При выключении вершины a_r на процессоро-память передается номер $+a^i$ (либо $-a^i$). Этот номер поступает на вершину «номер x^i» ячейки s_r и в результате она переходит из состояния «выполняюсь» в состояние «не готова». Утверждение (2) доказано.

Процесс выполнения программы P (2) в параллельной вычислительной системе можно представлять и анализировать при помощи графика вычислительного процесса, аналогичного описанному выше. График строится с той же горизонтальной осью времени t, но в качестве вертикальной оси используется объединение множества $\{s_1, \ldots, s_n\}$ ячеек, на которые записаны инструкции программы P, со множеством $\{T_{1,2}, T_{1,3}, \ldots, T_{n,2}, T_{n,3}\}$ их триггеров T_2 и T_3, причем если триггера $T_{r,2}$ и $T_{r,3}$ ячейки s_r могут только синхронно переходить в единичное и нулевое состояние, т. е, $a_r^j = a_r^l$, то на графике показывается только триггер $T_{r,2}$. Ячейка на графике изображается маленьким кружком, а триггер — стрелкой с номером, при поступлении которого на процессоро-память этот триггер переходит в единичное состояние.

На рис. 2, *б* показан фрагмент графика вычислительного процесса в параллельной вычислительной системе, соответствующий фрагменту, изображенному на рис. 2, *а*. Отрезком сплошной линии на уровне ячейки изображается интервал времени, в течение которого ячейка находится в состоянии «готова», а отрезком волнистой линии — в состоянии «выполняюсь»; отрезком штриховой линии на уровне триггера обозначен интервал времени, в течение которого этот триггер находится в единичном состоянии.

Таким образом, проведенное сравнение процесса выполнения граф-схемы с процессом реализации ее программы в параллельной вычислительной системе позволяет сделать вывод об их эквивалентности (например, в смысле графиков вычислительных

процессов), и поэтому при разработке и анализе $<i, k, j, l>$-программ можно основываться на любом из них.

Список литературы: 1. *Деннис Дж. Б., Фоссин Дж. Б., Линдерман Дж. П.* Схемы потока данных. — В кн.: Теория программирования. Ч. 2. Новосибирск, 1972, с. 7—41. 2. *Ершов А. П.* Операторные алгорифмы. I. — Проблемы кибернетики, 1960, вып. 3, с. 5—48. 3. *Калужнин Л. А.* Об алгорифмизации математических задач. — Проблемы кибернетики, 1959, вып. 2, с. 55—78. 4. *Карп Р. М., Миллер Р. Е.* Параллельные схемы программ. — Кибернетич. сб. Новая серия, 1976, вып. 13, с. 5—61. 5. *Колубай С. К.* Параллельная вычислительная система для МК-программ. — Проблемы бионики, 1979, вып. 23, с. 85—92. 6. *Колубай С. К.* О степени общности МК-программ. Сообщ. 2. Проблемы бионики, 1979, вып. 24, с. 76—83. 7. *Колубай С. К.* Программирование и реализация циклических процессов на асинхронном языке типа $< i, k, j, l >$. См. статью в настоящем выпуске. 8. *Котов В. Е., Нарыньяни А. С.* Асинхронные вычислительные процессы над памятью. — Кибернетика, 1966, № 3, с. 64—71. 9. *Котов В. Е.* Введение в теорию схем программ.— Новосибирск: Наука, 1978.—258 с. 10. *Кутепов В. П., Кораблин Ю. П.* Язык граф-схем параллельных алгоритмов. — Программирование, 1978, № 1, с. 3—11. 11. Сообщение об алгоритмическом языке АЛГОЛ-68 /Под ред. А. Ван Вейгаарден.) — Кибернетика, 1969, № 6, с. 17—144. 12. *Нариньяни А. С.* Теория параллельного программирования. Формальные модели.— Кибернетика, 1974, № 3, с. 1—15.

Поступила 12 ноября 1979 г.

УДК 62.506.2

А. В. НАПАЛКОВ

ИЗУЧЕНИЕ АЛГОРИТМОПОРОЖДАЮЩИХ СИСТЕМ РАБОТЫ МОЗГА

Творческий характер мышления человека в значительной степени определяется способностью к самостоятельному формированию новых алгоритмов. Вместе с тем анализ механизмов, лежащих в основе этого явления, связан с большими трудностями. Непосредственное экспериментальное изучение не приводит к выявлению алгоритмов и тем более систем, порождающих алгоритмы. Методы, используемые в области психологии и физиологии, оказываются недостаточно эффективными [1]. Вместе с тем использование существующих средств формального описания также не решает проблемы. Известно, что каждый из отделов математики абстрагирует некоторый тип отношений и применяется для изучения объектов, которые реализуют отношения этого же типа [2]. Трудности использования математики определяются тем, что при формировании механизмов работы мозга человека в процессе эволюции использовались все существующие вовне типы структурных отношений, в то время, как в области математики выделены и использованы только некоторые их категории [3]. Для решения проблемы алгоритмопорождающих систем необходимо выявление новых типов отношений, объективно существующих в окружающей действительности и построение на этой основе новых формальных систем (идеализаций

нового типа). Следует отметить, что в наши дни основные тенденции развития математики, связанные с тенденцией построения аксиоматических замкнутых систем, не создают нужных предпосылок для решения этой проблемы [4].

Возникает также проблема еще более общего типа — проблема формального описания тех систем отношений, на которых базируется интеллектуальная деятельность мозга человека. Очевидно, что причинно-следственные отношения окружающей действительности не представляют собой простую сумму. Следует говорить об объективном существовании определенных типов информационно-структурных организаций [1], создающих ту основу, на которой возможна работа мозга. Очевидно, что механизмы работы мозга, приводящие к формированию новых алгоритмов, основаны на преобразовании абстрактных структур общего типа, не связанных ограничениями, свойственными тому или иному отделу математики [1].

Пути решения этой проблемы оказались связанными с развитием учения И. П. Павлова. Попытки применения разработанной им теории и принципов исследования к анализу сложных форм мозга привели к изучению систем условных рефлексов [5—7].

При этом возникла необходимость формального описания различного типа «информационных сред», как основы изучения работы мозга [6—8]. На путях преодоления трудностей возникли новые проблемы. В частности стала очевидной необходимость классификации информационных сред [1]. Далее оказалось полезным использование представления об абстрактных структурах особого типа — информационных структурах [1] или системах специальных операторов [9]. Специфической особенностью информационных структур является отсутствие присущих отделам математики ограничений на типы абстрагируемых систем отношений. В таких структурах имело место исключение элементов и введение «локусов». Свойство «локуса» определялось только его местом в структуре. В результате осуществления специальных процедур локус может быть заполнен тем или иным сигналом и в этом случае имеет место построение конкретизированной частной структуры [1].

Одной из основных задач, решаемых в ходе исследования алгоритмопорождающих систем, являлась задача изучения тех типов структур описанного типа, которые актуальны как основа работы механизмов мышления человека, законов взаимодействия структур и тех новых в качественном отношении явлений, возникающих при функционировании систем этого типа. Показано, что все основные типы психической деятельности могут быть представлены как результат взаимодействия структур этого типа [3, 10]. Сделан вывод о том, что структуры описанного типа могут являться основой порождения новых алгоритмов. Также показано, что на основе изучения структур этого типа

может быть дано формальное описание перехода от анализа реальной действительности к построению новых средств формального описания. Процесс интуитивного мышления математика, ведущий к формулированию системы аксиом, постановке и поиску доказательства теорем, может быть представлен в виде системы структурных формул, а для каждого отдела математики определена его информационно-структурная основа.

В соответствии с концепцией, согласно которой передача ЭВМ функций эффективного общения с внешним миром, с человеком, функций самостоятельной постановки задач и выработки новых алгоритмов может быть достигнута только на основе использования информационных структур этого типа [1], определены новые перспективные направления в решении задач автоматизации в проектировании и строительстве.

Описанное направление работ, как нам кажется, намечает пути для решения противоречий, возникших в свое время между развитием психологии, с одной стороны, и эвристическим программированием с другой [11, 12]. В то же время оно опирается на предпосылки, созданные в работах ряда ведущих специалистов в области биологической кибернетики [13, 14]. Это направление работ существенно отличается от направления развития искусственного интеллекта, определенного в работах Г. С. Поспелова.

На секции «Алгоритмопорождающие системы работы мозга» сделан ряд докладов, в которых осуществлено теоретическое развитие, критическое обсуждение и использование описанных концепций. В сообщении Н. В. Целковой анализ алгоритмопорождающих систем дан на основе изучения игры в 15. Развита концепция, согласно которой возможность творческой работы мозга основывается на существовании в окружающей действительности скрытых от непосредственного восприятия информационно-структурных организаций различных типов. Это положение было иллюстрировано на основе анализа игры в 15. При внешнем наблюдении за ходом игры создается картина большого многообразия ситуаций и тактик, используемых испытуемым. Однако в действительности основные задачи формулируются на абстрактном языке в совершенно ином виде (скрытые информационные структуры).

Трудности решения задач, возникающих перед испытуемым, определялись тем, что было необходимо сформировать новую систему рефлексов (информационную структуру), включающую рефлекс на комплексный раздражитель. При этом каждый компонент комплекса мог быть получен в результате осуществления специальной цепи рефлексов. Но при осуществлении одной цепи возникал тормозной сигнал, препятствующий осуществлению другой, а следовательно и получение элемента комплекса.

Таким образом, было доказано существование двух различных систем понятий и языка, используемых в процессе мышле-

ния. Один из них опирался на использование понятий фишка, позиция и т. д. Другой использовал только язык информационных структур, отражающих отношения. Элементы обоих этих языков не имели однозначного соответствия и для перевода слов использовались специальные алгоритмы. Показано, что человек в процессе мышления использует языки обоих типов. При этом на языке информационных структур формируются алгоритмы игры, которые включают предметные понятия и используются в различных конкретных ситуациях. Раскрыта природа механизмов работы мозга человека, которые позволяют, анализируя различные возникающие в игре ситуации, осуществить:

а) постановку задачи на языке информационных структур; б) реализовать такое преобразование информационных структур, которое ведет к построению новых абстрактных понятий и новых алгоритмов; в) использовать алгоритмы в игре; г) осуществить анализ ошибок и провести доработку абстрактной постановки задачи и построенных алгоритмов. Показана природа механизмов формирования новых алгоритмов и разработан метод изучения сложных биологических систем, реализующих преобразование информационных структур, лежащих в основе работы алгоритмопорождающих систем.

В работе М. Ю. Володина, выполненной совместно с Н. В. Целковой, были продемонстрированы результаты применения метода анализа алгоритмопорождающих систем на примере изучения игры человека в шахматы. Показано, что анализ процессов преобразования «скрытых» абстрактных информационных структур позволяет дать расшифровку таких явлений, как построение концептуальных замыслов, используемых в игре, построение гипотез и версий о стратегиях, используемых противником, и их доказательств, их правдоподобия и т. д.

Э. П. Григорьев исследовал пути построения и использования систем повышения эффективности творческой деятельности человека в результате создания системы «инвариотрон». Он показал, что, используя теорию информационных структур, можно обеспечить способность ЭВМ самостоятельно анализировать события внешнего мира, ставить новые задачи, привлекать для их решения математические методы, создавать тактики поиска нужной информации. На этой основе возможно адекватное творческое общение человека и машины, которое позволяет рационально использовать возможности ЭВМ для повышения эффективности творческой деятельности.

М. З. Левина проанализировала работу алгоритмопорождающих систем на примере проектной деятельности человека, используя схемы, отражающие системы связей, возникающих у человека при проектировании. Анализ алгоритмопорождающих систем позволил выявить генезис и причинные связи, которые лежали в основе формирования определенных понятий и установления связей на различных этапах проектирования. Эти

результаты использовались для оптимизации взаимодействия ЭВМ и специалиста в процессе осуществления ими творческой деятельности.

Доказательство существования алгоритмопорождающих систем в живых организмах и раскрытие механизмов их организации ставит ряд новых вопросов в различных областях биологии. Очевидна неправомерность попыток изучения функций отделов мозга на основе использования понятий, заимствованных при изучении поведения человека и животных. Видимо, морфофизиологические системы реализуют информационные механизмы и именно в связи с задачами этого типа определяются функции отделов и принципы интеграции их работы в целостную деятельность [1].

При изучении проблем молекулярной биологии, эмбриологии, генетики следует учитывать, что роль биохимических систем часто сводится к реализации работы информационных механизмов. В связи с этим для решения актуальных проблем биологии важно не только расшифровка биохимических процессов, но и раскрытие природы информационных механизмов. Для этого необходимо использование специальных методов, основанных на знании природы алгоритмопорождающих систем.

Список литературы. 1. *Напалков А. В., Целкова Н. В.* Элементы теории структурно-информационных многоуровневых организаций (теория СИМО).— В кн. Комплексные проблемы охраны окружающей природой сферы и рационального использования природных ресурсов. — М., 1976, с. 52—58. 2. *Рубинштейн С. Л.* Бытие и сознание. — М.: Изд. АН СССР, 1957. — 252 с. 3. *Напалков А. В., Целкова Н. В., Моисеев И. Ф.* Эвристический анализ информационных структур — М. : Энергия, 1975. — 250 с. 4. *Тростников В. Н.* Конструктивные процессы в математике. — М.: Наука, 1975.—190 с. 5. *Воронин Л. Г., Никольская К. А.* Роль эффективной генерализации в формировании сложных форм двигательных рефлексов.— Журн. высшей нервной деятельности, 1977, 27, вып. 1, с. 34. 6. *Напалков А. В.* Физиологические механизмы, лежащие в основе формирований цепей двигательных условных рефлексов.— Науч. докл. высш. школы, Биолог. науки, 1958, № 2, с. 17—19. 7. *Напалков А. Ф.* Физиологический анализ некоторых сложных форм поведения. — Вопросы психологии, 1961, № 6, с. 25—29. 8. *Новиков П. П., Ситковский А. И.* Формальное описание экспериментальных сред. — Проблемы нейрокибернетики. Изд-во Ростовск. ун-та, 1969. Т. III, с. 187. 9. *Новиков П. П.* Информационное моделирование процесса принятия решений. — Проблемы нейрокибернетики. Изд-во Ростовск. у-та, 1969. Т. III, с. 289. 10. *Новиков П. П.* Влияние размерности среды на алгоритм принятия решений в дискретных средах. — Материалы симпозиума: Основные подходы к моделированию психики и эвристическому программированию. Тбилиси, 5—7. XII. 1968 г., с. 102—103. 11. *Тихомиров О. К.* Структура мыслительной деятельности человека. Изд-во Моск. ун-та, 1969.—210 с. 12. *Зинченко В. П., Гордон В. М.* Методологические проблемы психологического анализа деятельности. — В кн. : Системное исследование. — М. : Наука, 1976, с. 87—99. 13. *Амосов Н. Н.* Моделирование мышления и психики. — Киев : Наук. думка, 1965.—230 с. 14. *Анохин П. К.* Философский смысл кибернетических закономерностей. — В кн. : Философские вопросы биокибернетики. — М., 1969, с. 73—76.

Поступила 2 октября 1979 г.

УДК 62.506.2

Г. А. КРЫЖАНОВСКИЙ, Ю. Ф. ЦЕПЛЯЕВ

ВОПРОСЫ ОПТИМИЗАЦИИ ПРОЦЕССОВ ПРОФЕССИОНАЛЬНОЙ ПОДГОТОВКИ ДИСПЕТЧЕРОВ НА ТРЕНАЖЕРАХ

Одной из центральных проблем, возникающих на этапе организации систем управления воздушным движением (УВД), является проблема подготовки и переподготовки диспетчеров службы движения гражданской авиации. Актуальность данной проблемы вытекает из характера функционирования системы диспетчер — экипаж — воздушное судно и процессов, протекающих в этом контуре. Специфичность задач, возникающих при непосредственном управлении воздушным движением, состоящая в наличии «пиковых» нагрузок, когда на управлении у одного диспетчера может находиться более 10 воздушных судов одновременно, сменность в работе, наличие значительной неопределенности и неполной информации, предъявляют очень высокие требования к уровню профессиональной подготовки и психофизиологическим качествам данной категории операторов.

В связи с постоянным ростом плотности воздушного движения, вводом в эксплуатацию новых воздушных судов, автоматизацией процессов УВД и ряда других организационно-технических факторов, проблема совершенствования подготовки диспетчеров приобретает еще большую актуальность. Вполне естественно, что данный процесс должен базироваться на новейших достижениях в области педагогики, психологии, кибернетики и ряда других наук, т. е. на системном подходе к решению этой задачи. В основу нашей работы положены методы математического моделирования и параметрического оптимального управления процессом профессиональной подготовки диспетчеров, с учетом некоторых последовательных достижений педагогики и нейропсихологии [1—6].

В настоящее время известно значительное количество работ, посвященных математическому описанию процессов профессиональной подготовки операторов. Подробные обзоры различных математических моделей подготовки операторов радиолокационного контроля приведены в [1, 2]. В работе [3] дана модель обучения, согласно которой на каждом этапе обучаемому выдается один из N символов. При этом фиксируется результат опознавания обучаемым данного символа $b_i (i = \overline{1, N})$ и затрачиваемое при этом время τ_i. Формируется «функция успеха» в виде $a_i = b_i e^{-\alpha(\tau_i - \tau_{i\,\text{доп}})}$, где

$$b_i = \begin{cases} 1 \text{ при правильных действиях по } i\text{-у символу} \\ 0 \text{ в обратном случае;} \end{cases}$$

α — численный коэффициент, характеризующий интенсивность (динамику) процесса обучения.

Значения a_i по каждому символу позволяют подсчитать оценку качества работы обучаемого по всем символам: $A = \sum_{i=1}^{N} a_i$.

Задача оптимального управления процессом обучения сводится к отысканию такой последовательности символов, при которой к концу обучения A максимально. Как видно, в основу данной модели положен критерий достижения максимального уровня обучения за заданное время. Очевидно, модель наиболее пригодна на первоначальных этапах обучения при отработке простейших навыков с известными предельными характеристиками $\tau_{i\,доп}$, но не является эффективной при обучении сложнейшим навыкам и умениям по УВД, где требуется учет не только временных параметров отдельных операций, а также оценка эффективности актов принятия решения, последовательности, приоритичности решения задач в ходе выполнения сложных упражнений, например, по разведению потенциально-конфликтных ситуаций.

В работе [4] предлагается модель, описывающая вопросы взаимодействия учитель — ученик и разбиения учебного материала:
$p(t) = И(t) A_{\partial}[p(t)]$, где $p(t)$ — состояние знаний ученика; A — способность ученика; $U(t)$ — интенсивность воздействия при обучении; $g(p)$ — характеристическая функция обучения.

Находим оптимальное воздействие $U(t)$ учителя путем решения вариационной задачи, при котором обеспечивается максимум критерия поведения I на отрезке обучения T: $I = p(T) - \int_0^T l[U(t)]\,dt$, где $l(U)$ — функция потерь, $l \geqslant 0$; и решается задача оптимального разбиения учебного материала на блоки с целью максимизации отношения, где τ — время, необходимое на изучение одного блока учебного материала.

В результате решения задачи находятся оптимальные в среднем функции, определяющие разбиение учебного материала в зависимости от интенсивности обучения и способностей учащегося.

Как видно, модель описывает широкий класс отрабатываемых навыков и умений, однако в качестве управляющих функций рассматриваются только интенсивность обучения и разбиения учебного материала на блоки. Известен ряд других работ, в которых, как правило, рассматриваются только процессы приобретения профессиональных навыков и умений и не учитывается обратный процесс — разрушения навыков, взаимное влияние одного навыка на другой, такие качества оператора как ограничения по объему и скорости усвоения учебной программы и др.

В предлагаемой здесь модели делается попытка учета некоторых перечисленных свойств оператора и наиболее важных

управляющих функций на процесс обучения профессиональным навыкам и решений задач в ходе выполнения неэлементарных упражнений (на примере оператора УВД). Путем проведенных многочисленных экспериментов со слушателями-диспетчерами на тренажерах (при допущении о непрерывности во времени процесса обучения) была получена модель профессиональной подготовки [5]:

$$F(U, Q)\dot{X}(t) + AX(t) = Z, \qquad (1)$$

где $F(U, Q)$ — матрица управления процессом обучения; $X(t)$ — вектор состояния процесса обучения, количественно описывающий k-й навык в любой момент времени; A — матрица постоянных параметров, характеризующая степень подобия моделируемых (тренажерных) средств реальным; Z — вектор целей обучения.

Наиболее совершенным будет такой процесс обучения, при котором комплекс непосредственных прямых (U) и параметрических (Q) управлений обеспечивает оптимальный переход состояния $X(t)$ из начального $X^{(0)}$ в конечное $X^{(T)} = Z \pm \varepsilon$ смысле удовлетворения критерия минимума времени обучения T_Σ при заданном уровне обученности:

$$\min_U \min_Q \{T_\Sigma \text{ при } X^{(T)} = Z \pm \varepsilon,\ U \in Z_U^{(1)},\ Q \in L_U^{(2)}\}, \qquad (2)$$

где U — функция непосредственного управления в ходе учебного процесса; Q — параметрические управления учебным процессом; $L_U^{(1)}$ и $L_U^{(2)}$ — ограничения, задаваемые свойствами человека-оператора и учебным процессом.

В такой постановке задача (1)—(2) относится к классу задач параметрической оптимизации, так как управляющими воздействиями служат коэффициенты в матрице $F(U, Q)$ [6]. Для анализа столь сложной вариационной задачи рассмотрим более подробно структуру системы обучения и функции управления.

Известно, что процесс обучения на нейронном уровне можно рассматривать как процесс организации паттернов активности, относительно стабильных во времени форм пространственно-временной организации нейронных популяций, характеризующих устойчивость навыков (сенсорно-моторных, сенсорно-речевых), вырабатываемых в ходе тренировок [7]. При воздействии на мозг обучаемой последовательностью, например, заложенной в алгоритме упражнения, требующей выполнения тех или иных операций и актов принятия решения обучаемым (f_{ti}), происходит перестройка статистических характеристик нейронов и формирование матрицы долгосрочной памяти. Причем, в первом приближении формирование определенного навыка и его закрепление идет тем успешнее, чем чаще происходит выполнение операций, необходимых для его формирования, а также, чем проще этот комплекс операций при прочих равных условиях. Таким образом, мозг, используя

информацию, запасенную на предыдущих этапах обучения обеспечивает активную минимизацию времени восприятия активизирующего сигнала, декодирования, принятия решения и выполнения определенного действия до определенного устойчивого уровня.

Следовательно, чем чаще происходит воздействие на обучаемого путем выполнения предписанного упражнением алгоритма и воздействий инструктора при пропусках операций или неправильном их выполнении, тем быстрее происходит формирование требуемых навыков, умений. При этом следует и учитывать ограничения по скорости восприятия оператором информации, т. е. частота воздействий не должна нарушать эти ограничения.

Структурная схема процесса обучения операторов на тренажере в соответствии с приведенными рассуждениями приведена на рис. 1. На процесс приобретения профессиональных навыков и умений влияет еще ряд управляемых факторов.

Рассмотрим основные управляющие функции учебного процесса, влияющие на скорость приобретения и прочность сохранения профессиональных навыков и умений операторами, подчеркнув принципиальное отличие функций управления U_i и параметров $Q_j^{(N)}$, заключающейся в различной физической их природе. Управления U_i участвуют в формировании моментов управляющих воздействий инструктора и являются произвольной функцией времени: параметрические уравнения $Q_j^{(N)}$ определяются системой подготовки и ограничениями оператора по восприятию и усвоению новых стимулов и могут быть кусочно-постоянными или кусочно-медленноменяющимися на каждом из этапов подготовки. Тогда управляющими функциями будут: U_1 — частота выполняемых функций, заложенных в алгоритме отрабатываемого упражнения. Формирование данного навыка X_i происходит путем прямого воздействия i-й операции и косвенного воздействия при выполнении j-й операции на управляющий нервный код коры головного мозга, ответственного за данный навык, $U_1^{(i)} = \sum_{i=1}^{n} f_{ij}$; U_2 — количество воздействий инструктора в форме разъяснения, подкрепления, наказания, на те же структуры коры, но в другой форме. В первом приближении

$$U_2^{(i)} = \sum_{i=1}^{n} f_{ij} - f_{ij}^{(p)},$$

где $f_{ij}^{(p)}$ — количество правильно выполненных учащимся операций; U_3 — количество повторений одинаковых упражнений за одно занятие, связанных с переводом в долговременную память формируемых навыков, $U_3 = n_1(t)$; U_4 — количество вариаций отдельных операций, связанных с развитием творческого активного применения формируемых навыков в нестандартных вариан-

тах деятельности, $U_4 = W(t)$, где W — число операций, подвергающихся вариантному предъявлению; U_5 — скорость предъявления учебной последовательности, определяемая индивидуальными способностями учащихся по ее усвоению, $U_5 = k_V V$, где k_V — коэффициент скорости обучения. В большинстве практических задач $k = 0{,}5 \div 1{,}2$; Q_1 — время перерывов между занятиями, характеризующее следствие процесса памяти, $Q_1 = \tau(t)$, где τ — время между моментами предъявлениями предыдущей и последующей учебными последовательностями; Q_2 — количество отрабатываемых в данном упражнении различных навыков, умений, $Q_2 = n_2(t)$, где $n_2(t)$ — число различных f_{ij}.

В соответствии с высказанными предположениями, структура матрицы $F(U, Q)$ имеет следующую форму:

$$F(U, Q) = \begin{vmatrix} \frac{1}{f_{11}} & \cdots & \frac{1}{f_{1n}} \\ & \frac{1}{f_{ij}} & \\ \frac{1}{f_{n1}} & \cdots & \frac{1}{f_{nn}} \end{vmatrix}, \qquad (3)$$

где по главной диагонали стоят операции учебного алгоритма прямого воздействия на формируемый навык, а вне ее — косвенного воздействия. Вариационный смысл задачи состоит в неявной зависимости функции критерия качества (2) от параметров U, Q через решение системы дифференциальных уравнений (1) при учете ограничений $L_U^{(1,2)}$. Вычисляя матрицу, обратную (3), и умножая слева на $F^{-1}(U, Q)$ (1), получим систему дифференциальных нелинейных уравнений

$$\dot{X} = \tilde{A}X + DZ. \qquad (4)$$

Параметрический характер задачи оптимизации выражается в зависимости коэффициентов матриц $\tilde{A}$, D от параметрических управлений Q_1, Q_2. Используя каноническое преобразование Лурье [8, 9], произведем переход от матричной записи (4) операции к канонической записи, определяющей вектор состояния процесса навыков и умений

$$\dot{S} = -\Lambda S + BZ. \qquad (5)$$

Решение (5) ищем в виде

$$S(t) = Z\Lambda^{-1}B + [\exp -\Lambda(t - t_0)](S_0 - Z\Lambda^{-1}B), \qquad (6)$$

где $S_0 = S(t_0)$.

Сложность процессов профессиональной подготовки заставляет каждый шаг моделирования, формализации и исследования оптимальных решений сопоставлять с результатами экспериментов.

Проведенные исследования показали, что в реальном процессе обучения существует связь между $\lambda^{(l)}$ и действительным поведением обучаемого, определяющимся скоростью обучения $\alpha^{(l)}$, однако между $\lambda^{(l)}$ и $\alpha^{(l)}$ нет жесткой связи из-за явной нелинейности и стохастичности данных характеристик оператора. Учиты-

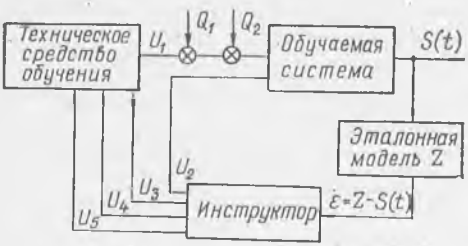


Рис. 1

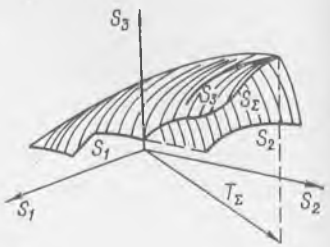


Рис. 2

вая нестабильность во времени поведения сформированных нейронных структур из-за отсутствия тренировок, интерференции навыков, необходимо ввести в модель показатель скорости их разрушения $\gamma^{(k)} = f(a)$, где a — параметр, характеризующий следовые процессы памяти обучаемого.

Кроме того, в модели необходимо учитывать также процессы восстановления разрушенных навыков после возобновления их тренировки со своей скоростью $\alpha^{(k)}$. Переходя к детерминированным оценкам процесса S на уровне математического ожидания $m[S^{(i)}]$, решение (6) запишем в виде

$$m[S^{(l)}(t_N)] = m[S_{np}^{(1)}] + \{m[S_0^{(l)}] - m[S_{np}^{(l)}]\} e^{-\alpha^{(l)} t_N}$$

для отрабатываемых навыков на t_N отрезке времени обучения;

$$m[S^{(k)}(t_N)] = m[S_{np}^{(k)}] + \{m[S^{(k)}] - m[S_{np}^{(k)}]\} e^{\gamma^{(k)} t_b},$$

где t_b — момент времени включения k-го навыка в отработку на N-м этапе обучения для отработанных на $N-1$-м этапе навыков, временно не включенных в процесс обучения:

$$m[S^{(k)}(t_N)] = m[S_{np}^{(k)}] + \{m[S_p^{(k)}] - m[S_{np}^{(k)}]\} e^{-\gamma^{(k)}(t_N - t_b)}$$

для k-х навыков, включенных в отработку на N-м этапе с момента t_b, где $m[S_0^{(l)}]$, $m[S_{np}^{(l,k)}]$, $m[S^{(k)}]$, $m[S_p^{(k)}]$ —начальные, предельные, приобретенные и разрушенные количественные оценки l, k-х навыков на N-м этапе.

Поиск оптимального решения (5) в смысле (2) можно провести, применяя процедуры перехода к методам математического программирования и алгоритмов стохастической аппроксимации [10, 11], разбивая процесс обучения на N-м этапе на дискретные отрезки времени t_N и решая систему дифференциальных урав-

нений численными методами. Причем в ходе решения задачи методом стохастической аппроксимации при отработке новых навыков в процессе обучения ищутся оптимальные значения U_2, U_3, U_4 и Q_1, а при решении задачи на включение в отработку разрушенных навыков методами математического программирования определяется Q_2. Необходимо отметить тесную связь предлагаемого подхода с экспериментальными методами исследования, так как целый ряд параметров модели, таких как например U_5, S_0, S_p и другие можно определить только экспериментально. Поиск U_1 возможно произвести с привлечением структурно-алгоритмических методов анализа реальной деятельности и методов распознания образов. Оптимизация процессов обучения при этом реализует характерные траектории вектора состояния, вид которых показан на рис. 2.

Применение на практике указанного метода позволило сократить время обучения на некоторых этапах обучения на 30%.

Список литературы: 1. *Кораков В. Г.* Автоматизация обработки, передачи и отображения радиолокационной информации. — М. : Сов. радио, 1975.—192 с. 2. *Рыбаков Ф. И., Макухин В. Н., Григорьев В. А.* Методы и средства повышения эффективности работы операторов РЛС. — Зарубежная радиоэлектроника, 1977, № 11, с. 3—27. 3. *Берга А. И., Паск Г.* Обучение, как процесс создания системы управления. — В кн. : Кибернетика и процесс обучения. — М. : Прогресс, 1970, с. 35—41. 4. *Chant V., Luenberger* D. A mathematical theory of instruction pacing.— Journ. of mathematical psychology, 1974, VII, № 2, p. 132—158. 5. *Дарымов Ю. П., Крыжановский Г. А., Цепляев Ю. Ф.* Вопросы совершенствования профессиональной подготовки диспетчеров УВД.— В кн.: Автоматизированные системы УВД.— Л.: ОЛАГА, 1978, с. 112—116. 6. *Крыжановский Г. А., Сухомехов В. П.* О постановке и решении общей задачи параметрической оптимизации.— Вопросы радиоэлектроники. Сер. ТПО, 1971, вып. 2, с. 18—27. 7. *Бехтерева Н. П., Бундзен П. В., Гоголицын Ю. Л.* Мозговые коды психической деятельности.— Л.: Наука, 1977.— 160 с. 8. *Лурье А. И.* Некоторые нелинейные задачи автоматического регулирования.— М.: Гостехиздат, 1951.—112 с. 9. *Гантмахер Ф. Р.* Теория матриц.— М : Наука. 1967.—145 с. 10. *Тобак Д., Куо Б.* Оптимальное управление и математическое программирование.— М.: Наука, 1975.—176 с. 11. *Вазан М.* Стохастическая аппроксимация.— М.: Мир, 1972.—192 с.

Поступила 2 октября 1979 г

УДК 510.62

В. П. ГЛАДУН, Н. Д. ВАЩЕНКО, И. Г. БИБА, Н. И. ГАЛАГАН, Л. В. ХОМЕНКО

ПРОБЛЕМЫ АВТОМАТИЗАЦИИ ПРОЦЕССОВ РЕШЕНИЯ ЗАДАЧ

Известные исследования процессов решения [1, 2] выполнялись на модельных задачах в относительно простых, искусственных средах. Для перехода к реальным задачам требуется совершенствование решающих систем в трех основных направлениях.

1. Разработка средств представления знаний, обеспечивающих работу системы в сложных, многокомпонентных средах.

2. Развитие средств адаптации решающих систем.

3. Развитие стратегий решения.

Статья содержит краткое описание исследований по каждой из названных проблем, выполненных в связи с разработкой системы APROS (Adaptive Problem Solver), созданной в Институте кибернетики АН УССР.

Система APROS предназначена для решения прикладных задач планирования действий, в частности, задач планирования поведения роботов; для выполнения экспериментальных исследований процессов решения задач.

С целью обеспечения экспериментальных исследований система оснащена средствами варьирования параметров, стратегий решения, а также вывода данных, характеризующих процессы в системе.

Система APROS представляет собой набор программ на языке PL/1. Структура системы показана на рис. 1.

В качестве входого языка используется язык описания задач SITPLAN (Situation Transformation Problem Language) [3]. Формулировки задач и описания сред вводятся через блок формирования семантического представления в базу данных, организованную в семантическую сеть. Помимо базы данных, содержащей информацию о задачах и средах, в системе имеются библиотека стратегий решений и библиотека схем решений. Схемой решения в широком смысле называется информация, ограничивающая область поиска при решении задач, принадлежащих одному классу. Обычно в схеме решения выделены основные этапы решения и задана последовательность этих этапов. Если задача принадлежит одному из классов, для которых имеются схемы решения, план формируется с помощью схемы решения. Если готовой схемы решения нет, для составления плана применяется одна из стратегий решения, хранящихся в библиотеке. Выбор стратегий производится с учетом типа задачи. Процесс решения регулируется параметрами, заданными пользователем. Помимо планов в системе выводятся на печать данные, характеризующие процесс планирования.

Схемы решения строятся путем обобщения планов конкретных задач. Для поиска схем решения используются обобщенные определения классов задач, которые формируются и заносятся в базу данных блоком обобщения ситуаций и целей. Система построена по модульному принципу, предусмотрена возможность расширения библиотеки стратегий решений и системы в целом. Ведутся работы по организации процесса решения в режиме диалога на естественном языке.

Семантическое представление. Как уже указывалось выше, переход к реальным задачам связан с существенным усложнением сред, в которых должны работать системы планирования действий.

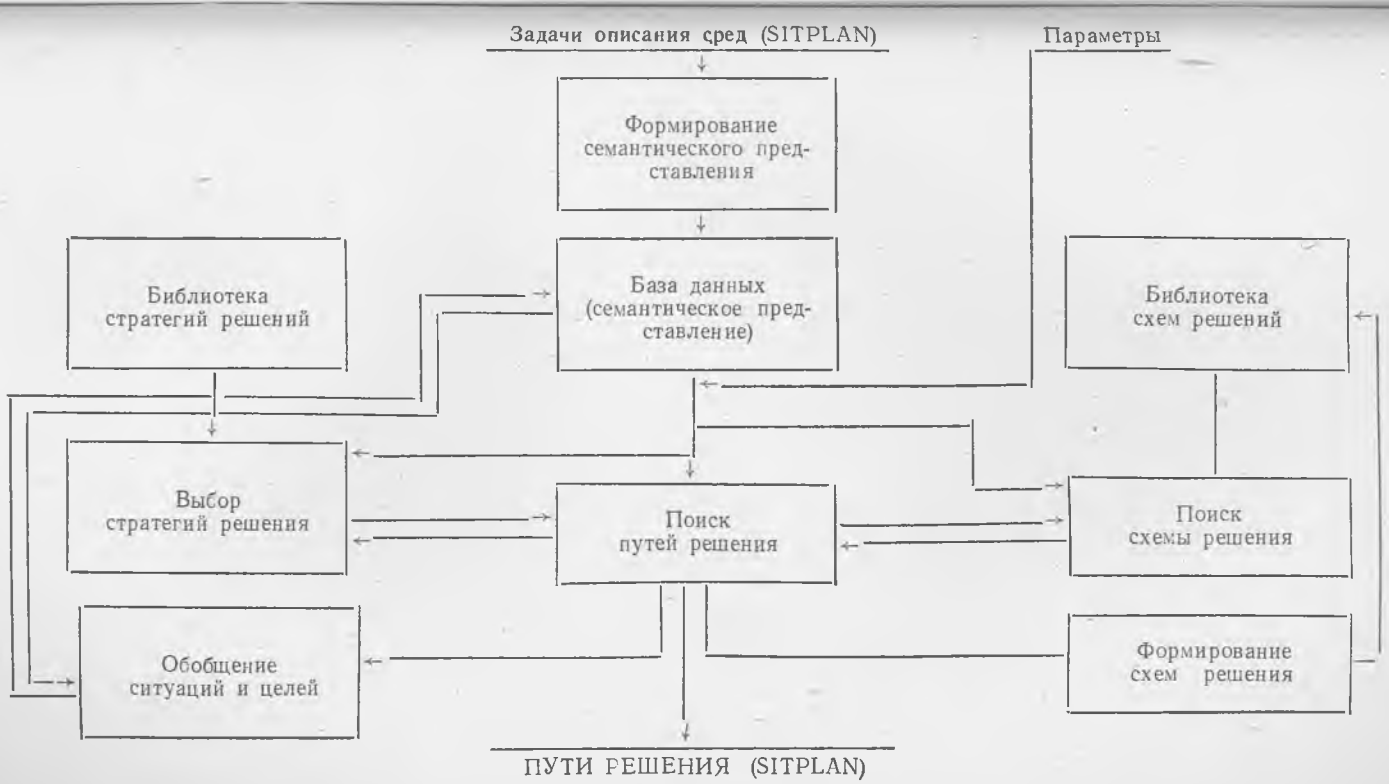

Задачи описания сред (SITPLAN)
Параметры
Формирование семантического представления
Библиотека стратегий решений
База данных (семантическое представление)
Библиотека схем решений
Выбор стратегий решения
Поиск путей решения
Поиск схемы решения
Обобщение ситуаций и целей
Формирование схем решения
ПУТИ РЕШЕНИЯ (SITPLAN)
Данные о процессе решения

Рис. 1.

Среда определяется тройкой $< V, R, Q >$, где V — множество объектов; R — множество отношений, определенных на V (свойства и состояния объектов являются одноместными отношениями); Q — множество действий, которые можно выполнять с элементами множеств V и R. Ситуации задаются подмножествами R.

Представление о сложности среды связано с мощностями множеств V, R, Q и по своей природе относительно. Сложность среды является одним из факторов, определяющих размеры области поиска при составлении плана. У нас нет необходимости вводить условный численный критерий, разграничивающий сложные и простые среды. Будем считать среду сложной, если в этой среде планирование действий с помощью современной техники невозможно без принятия специальных мер, устраняющих простые переборы элементов описания среды.

Проблема устранения простых переборов является одной из главных при построении систем планирования действий, оперирующих большими объемами данных. В связи с этим в таких системах предъявляются высокие требования к ассоциативным свойствам структур, представляющих информацию о среде в памяти системы.

В системе APROS для семантического представления информации в базе данных используются пирамидальные сети [2, 4].

Модель среды, содержащаяся в базе данных, состоит из следующих компонентов: конкретное представление, операторы, обобщенное представление.

Конкретное представление среды формируется по описанию ситуаций в языке SITPLAN. В конкретном представлении сохраняется вся информация о конкретных объектах и отношениях, содержащаяся в описании ситуаций. Описание ситуаций в языке SITPLAN представляет собой последовательность выражений естественного языка, определяющих отношения конкретных объектов. Конкретным представлением служит пирамидальная сеть, в которой входные элементы — рецепторы соответствуют именам объектов, классов объектов, отношений между объектами, именам свойств и состояний объектов.

Операторы, как и ситуации, представлены в базе данных пирамидальной сетью. Описание оператора на языке SITPLAN состоит из нескольких частей: условие применимости, список отношений среды, которые исчезают при выполнении оператора, и список отношений, появляющихся в результате выполнения оператора. В сети операторов каждой из этих частей соответствуют отдельные пирамиды.

В обобщенном представлении в форме пирамидальной сети представлены обобщенные определения классов ситуаций (фреймы), которым соответствуют отдельные схемы решения библиотеки схем решения. Эта сеть в сочетании с библиотекой схем решений отображает глобальные закономерности, присущие

пространству задач, решаемых системой в заданной среде. Сеть формируется при реализации процессов адаптации, описанных в [5].

Пирамидальная сеть удобна для выполнения различных операций ассоциативного поиска. Структурные свойства пирамидальных сетей позволяют успешно преодолеть трудности, возникающие при решении задач в сложных, многокомпонентных средах. Все три вида пирамидальной сети, используемых в базе данных (сеть ситуаций, сеть операторов и сеть обобщенного представления) имеют общий набор рецепторов. Это позволяет осуществлять поиск применимых операторов и классификацию ситуаций ассоциативно, через рецепторы, без просмотра всех операторов или определений классов ситуаций.

Исследование избирательности системы. Решающим требованием к планирующим системам, работающим с описанием сред большого объема, является максимальное сокращение части описания среды, просматриваемой при построении плана. В связи с этим можно было бы оценивать и сравнивать системы по эффективности отбора информации при решении задач.

Объем части описания среды, которая просматривается при составлении плана, зависит от внесистемных и внутрисистемных факторов. К внесистемным факторам относятся сложность среды и задачи, из внутрисистемных факторов на эффективность отбора информации наибольшее влияние оказывают структура модели среды, процедуры поиска, определенные на этой структуре, стратегии решения. Для упрощения сравнения систем следует стремиться к разработке таких критериев эффективности отбора информации, которые бы не зависели от конкретных задач. С помощью критериев такого типа можно сравнивать системы, решающие разные задачи в средах одинаковой сложности, если конечно заданы параметры, позволяющие численно оценивать сложность среды независимо от ее семантики.

Пусть имеется среда $E = < V, R, Q >$. Введем обозначения: n_q, r_q — соответственно число объектов и число отношений в условии применимости оператора $q \in Q$; l_q — среднее число отношений в описании ситуаций, просматриваемых при распознавании применимости оператора в процессе решения задач в среде E; p_q — вероятность применения оператора при решении задач в среде E. Приведенные выше соображения являются обоснованием для ввода следующих определений: 1) отношение $S_E = \frac{|R|}{|V|}$ называется *связностью среды* E. 2) отношение $s_q = \frac{r_q}{n_q}$ называется *связностью оператора* q. 3) отношение $H_E = \frac{M\{s_q\}}{S_E}$, где $M\{s_q\} = \sum_Q s_q p_q$ — математическое ожидание s_q, называется *относительной связностью оператора* в среде E.

Каждый из введенных параметров среды влияет на размер части описания ситуаций, просматриваемой при распознавании применимости оператора. H_E и $|R|$ выделим в качестве основных параметров, определяющих сложность среды. 4) Математическое ожидание $\tau = M\left\{\frac{r_q}{l_q}\right\} = \sum_Q \frac{r_q}{l_q} p_q$ называется *избирательностью системы.*

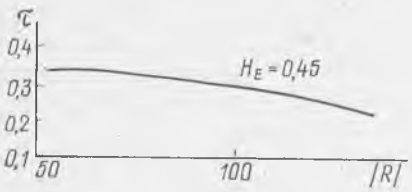


Рис. 2

Избирательность системы является интегральным параметром, оценивающим суммарное воздействие сложности среды и внутрисистемных факторов на объем части описания ситуаций, просматриваемой на каждом шаге процесса построения плана. Основой для сопоставления систем по эффективности отбора информации является график функции $\tau = f(|R|)$ при фиксированной относительной связности оператора.

С помощью таких графиков можно сравнивать планирующие системы, работающие в разных средах с одинаковой относительной связностью оператора. Характеристикой системы являются как отдельные значения τ при разных значениях $|R|$ и H_E, так и сам характер зависимости τ от $|R|$. При проектировании систем, ориентированных на сложные среды, следует стремиться к уменьшению зависимости τ от $|R|$.

Исследование избирательности системы APROS являлось целью первой серии экспериментов с системой. В этих экспериментах система решала задачи планирования действий роботов при перестановках объектов. Задачи ставились по фотографиям реальных сред (комната с лабораторным оборудованием). Описания ситуаций в языке SITPLAN обычно включали свыше 100 выражений. Зависимость τ от $|R|$, построенная в результате этих экспериментов, показана на рис. 2.

Список литературы: 1. *Гладун В. П., Галаган Н. И.* Об одном подходе к классификации и сопоставлению методов поиска решений задач преобразования.— Кибернетика, 1967, № 4, с. 20—25. 2. *Гладун В. П.* Эвристический поиск в сложных средах.— Киев: Наук. думка, 1977.—240 с. 3. *Галаган Н. И.* Язык описания задач SITPLAN.— Кибернетика, 1979, № 2, с. 40—43. 4. *Гладун В. П.* Составление описаний классов объектов на ЦВМ, I, II.— Кибернетика, 1972, № 5, с. 38—44. 5. *Гладун В. П., Ващенко Н. Д.* Адаптация в решающих системах.— Кибернетика, 1979, № 2, с. 61—67.

Поступила 2 октября 1979 г.

УДК 65.506.2

М. Б. АГИШЕВ, А. И. ФОКЕЕВ

ПРИМЕНЕНИЕ ПОЛУНАТУРНОГО МОДЕЛИРОВАНИЯ ДЛЯ ОЦЕНКИ КОДОВЫХ НАИМЕНОВАНИЙ ФОРМАЛИЗОВАННЫХ ТИПОВ СООБЩЕНИЙ, ЦИРКУЛИРУЮЩИХ В АС УВД

В автоматизированной системе управления воздушным движением (АС УВД) большое значение имеет отображение информации на внешних устройствах (дисплеях). В связи с этим необходимо выбирать наиболее удобные для работы диспетчера способы представления информации, определить оптимальный в психолингвистическом отношении способ кодирования формализованных сообщений искусственного языка и, в частности, их постоянной части, которая должна отображаться на табличнознаковом индикаторе (ТЗИ) в кодированной форме.

В качестве кодов, выполняющих коммуникативные функции, может быть использована любая система знаков, включая знаки естественного языка. Под кодом понимается любой условно заданный знак с присвоенным ему в момент создания определенным значением.

Оптимальность знака или знаковой системы является относительной, так как она всегда связана с задачами конкретной деятельности, с условиями функционирования знаковой системы, с внутренними свойствами самих кодов, а также психофизиологическими особенностями человеческого восприятия.

В реальных условиях деятельности оператор пользуется динамическим алфавитом сигнала, т. е. при переходе от различения к индентификации и т. д. алфавит меняется [1].

Другими словами, в разных процессах восприятия акцент диспетчерского внимания направляется на разные характеристики сигнала в пределах одного алфавита. Поэтому разработка методов оптимального кодирования включает: подбор оптимального алфавита (или алфавитов), которыми кодируются отдельные элементы сообщений; установление оптимального соотношения между различными алфавитами в пределах одного сообщения; нахождение оптимальной логической структуры закодированного сообщения.

Принимая во внимание опыт инженерной психологии в изучении систем кодирования [2—4], в настоящей статье рассматриваются различные системы кодирования одних и тех же ситуаций, представленных наименованиями типов сообщений единого банка, с точки зрения их семантической структуры, т. е. с учетом принципов и правил построения алфавита знаков по отношению к кодируемой ситуации, а также с прагматической точки зрения — с учетом свойств этих знаков, влияющих на основные психофизиологические и технологические показатели деятельности

диспетчера (запоминание, скорость и точность декодирования, скорость обучения и т. д.).

Учитывая рекомендации ИКАО и СЭВ, необходимо выбрать оптимальную структуру для трехзнакового кода ситуаций, описанных номинативными словосочетаниями на естественном языке. При этом в нашем случае кодируется не объект или предмет, выраженный одним понятием — словом, а целая ситуация, т. е. код является многомерным по своей информационной емкости.

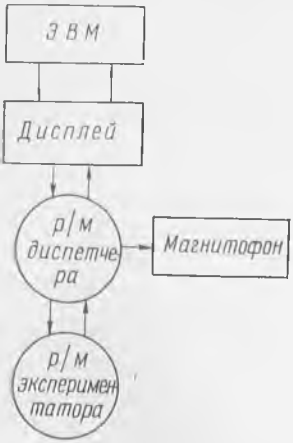


Рис. 1

Для изучения систем кодирования и подбора оптимального алфавита использовалось полунатурное моделирование, которое проводилось на базе диспетчерского тренажера ГОСНИИ ГА в аэропорту Внуково.

Схема проведения эксперимента приведена на рис. 1.

В процессе моделирования испытуемому на таблично-знаковом индикаторе предъявлялись различные системы кодирования одних и тех же ситуаций (табл. 1), а результаты идентификации записывались на магнитофонную ленту.

Так, условный трехбуквенный код сформирован в результате транслитерации (передачи русскими буквами английских сокращений) и контракции, а также с помощью двухбуквенного индентификатора ЛД (летно-диспетчерская группа сообщений) и условно приписанной буквой русского алфавита (например: [АЦП]→ [АСР]→ acceptance → принятие; ЛДА — сообщение о состоянии аэродрома).

Аббревиатурный код образован с помощью инициальных аббревиатур, фрагментарного отсечения, а также контракции в сочетании с инициальным фрагментом (например, ЗНА — плановая заявка на аэродромные полеты; ВЫК — информационное сообщение о выключении средств связи и РТО; ЗВЛ — запрос разрешения на вылет).

Символьные коды сформированы с целью максимальной изобразительной и ассоциативной мотивированности, что приводит к глубинным ассоциациям с кодируемой ситуацией. Так, например: ↑ стрелка вверх приводит к мысли о взлете и формирует соответствующие вербальные ассоциации; ⊤ — в контексте символизирует ограниченность пространства и приводит к вербальной ассоциации *аэродром* и т. д.

Ассоциативные вербальные коды сформированы с помощью трехбуквенных слов естественного языка, которые имеют отдельную ассоциативную связь с кодируемыми ситуациями.

Условные вербальные коды сформированы также с помощью трехбуквенных слов, но приписываемых кодируемым ситуациям в случайном порядке. Эти коды, обладая свойствами слоговости и эстетичности, лишены ассоциативной мотивированности.

Таблица 1

№ п/п	Наименование типовых сообщений	Условные буквенные коды	Аббревиатурные коды	Ассоциативные вербальные коды	Условные вербальные коды
1	2	3	4	5	6
1	Плановая заявка на аэродромные полеты	УТП	ЗНА	иск	пар
2	Информационное сообщение об ограничениях воздушного пространства	ИГВ	ОРЖ	куб	дно
3	Запрос разрешения на вылет	ЛДК	ЗВЛ	шаг	тыл
4	Коррекционное сообщение об изменении времени вылета ЛА	ЛДД	ИВВ	час	дым
5	Информационное сообщение о начале полетов на аэродромах	ЛДЛ	НАП	юла	ток
6	Информационное сообщение об окончании полетов на аэродромах	ЛДО	КАП	дом	род
7	Запрос о состоянии аэродрома	ЛДБ	ЗАС	фон	вал
8	Ответ на запрос о состоянии аэродрома	ЛДА	САС	луг	пуд
9	Экстраординарное информационное сообщение о нарушении режима полета	ЛДТ	НРЖ	аут	зал
10	Приказ-запрещение продолжения полета ЛА, нарушившему режим полета	ЛДУ	ППП	зов	газ
11	Приказ об отмене дополнительных полетов без заявок и утвержденного плана	ЛДС	ПОП	риф	сон
12	Информационное сообщение о средствах обеспечения полетов	ЛДХ	СОП	уют	бок
13	Информационное сообщение о включении средств связи и РТО	ЛДЦ	ВКЛ	ухо	чек
14	Информационное сообщение о выключении средств связи и РТО	ЛДЫ	ВЫК	эхо	дар
15	Ответ-разрешение на перелет государственной границы советским и иностранным ЛА	ЛДЯ	РПГ	вид	луч

Предъявляемые испытуемым алфавиты подбирались в соответствии с определенными критериями, которые представлены на рис. 2 и образуют схему отношений параметров кодов, влияющих на технологические и психофизиологические показатели деятельности диспетчера на всех уровнях реализации АС УВД. На схеме в квадратах указаны основные параметры оптимального кода, а в ромбах — технологические и психофизиологические показатели деятельности диспетчера.

В качестве основных критериев, по которым производилось сравнение систем кодирования и выбор оптимальных способов представления информации диспетчеру УВД, были выбраны скорость декодирования и точность декодирования.

Проведенная серия экспериментов по адаптации и восприятию диспетчером формализованных сообщений, входящих в искусственный язык, показала, что в данном случае, условный буквенный код наиболее близок к оптимальному, так как показатели по скорости и точности декодирования для данного алфа-

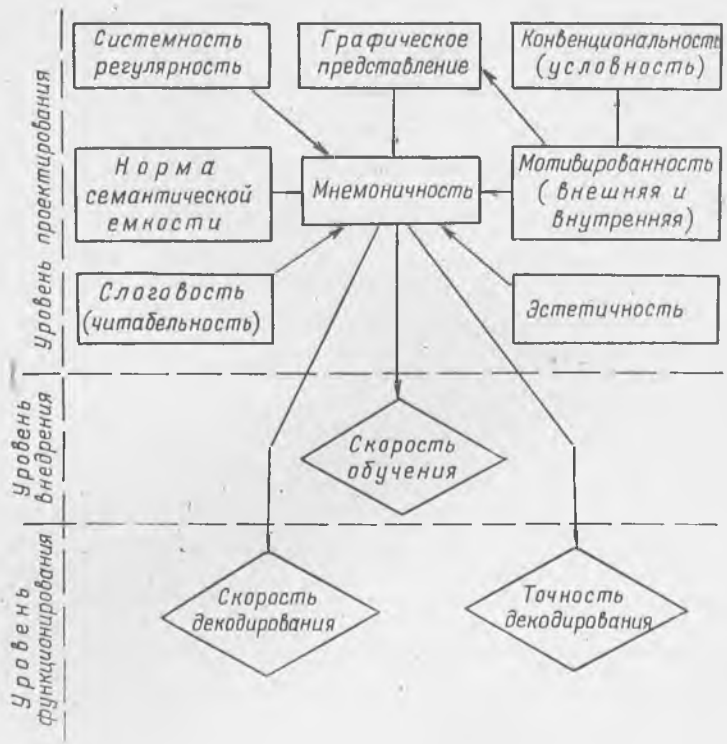


Рис. 2

вита наилучшие (рис. 3) и составляют 3,8с и 6,2% соответственно. Поэтому в качестве системы кодовых обозначений типов сообщений в АС УВД была выбрана система условного буквенного кода (табл. 2).

В то же время результаты полунатурного моделирования показали, что для поставленных задач кодирования в АС УВД может также использоваться аббревиатурный код. Этот код совместно с символьным, вербально ассоциативным и вербально условным кодом необходимо применять на этапе обучения диспетчерского состава, поскольку они способствуют усвоению предлагаемого искусственного языка.

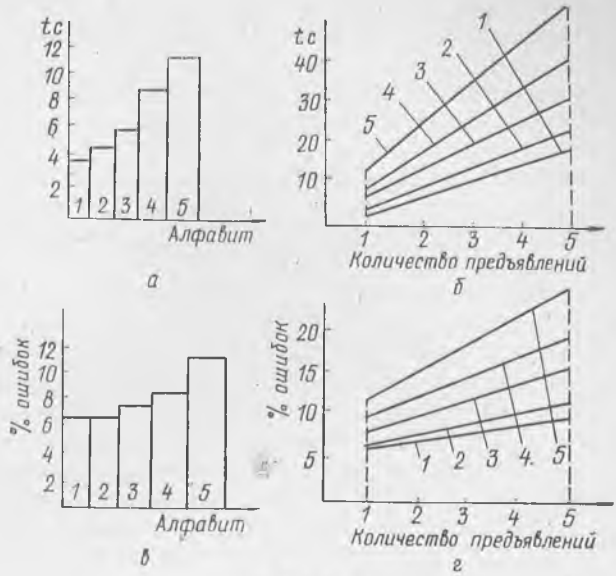


Рис. 3

Таблица 2

№ п/п	Условное обозначение сообщения	Наименование типа сообщения
1	АВС	Аварийное сообщение
2	АОЦ	Принятие управления полетом
3	АЦП	Принятие (сообщение о согласии)
.		
.		
.		
.		
.		
34	УТП	Заявка на аэродромные полеты
35	КПЛ	Координация плана полета
36	ОПП	Отмена плана полета

Список литературы: 1. *Ломов Б. Ф.* О некоторых критериях оценки сигналов передающих информацию человеку-оператору. — В кн.: Проблемы инженерной психологии, вып. 2.— Л., 1965, с. 20—25. 2. *Кринчик Е. П., Киященко Н. К.* К вопросу об оптимальном кодировании сложных сообщений.— В кн.: Проблемы инженерной психологии, вып. 2.— Л., 1965, с. 42—45. 3. *Хавская Р. Н.* Влияние способов кодирования информации на эффективность деятельности диспетчера промышленного производства.— В кн.: Человек и общество, вып. 8.— Л.: Изд-во Ленинград. ун-та, 1971, с. 86—89. 4. *Тутушнина М. К.* Особенности приема и переработки знаковой информации. — В кн.: Психологические проблемы переработки знаковой информации. — М.: Наука, 1977, с. 112—118.

Поступила 2 октября 1979 г.

УДК 62.506.2

В. М. БОНДАРОВСКАЯ, Л. В. ХОМЕНКО, О. Б. ЧМОЛА

О ЛОГИКО-МАТЕМАТИЧЕСКОЙ И ПСИХОЛОГИЧЕСКОЙ ОЦЕНКЕ ЭФФЕКТИВНОСТИ РЕШЕНИЯ

Данная работа является продолжением работы [1], в которой описано психологическое исследование, проведенное с целью совершенствования процесса индуктивного формирования понятий системы «Анализатор» [2].

Этап собственно формирования понятий следует непосредственно за этапом структурирования и запоминания информации об объектах обучающей выборки. В нашем случае структурирование и запоминание исходной информации проводилось в виде растущей пирамидальной сети [1] (РПС), соответствующей той, которую строит система «Анализатор» (3). Задача собственно формирования понятий состояла в выделении специальных элементов, назовем их положительными и отрицательными контрольными элементами (k-элементами), с помощью которых должно осуществляться распознавание объектов, принадлежащих и не принадлежащих понятию [2].

В процессе теоретического анализа указанной задачи введена оценочная функция качества сформированного понятия, как продукта решения задачи:

$$\varphi = \sum_{i=1}^{n} U_i,$$

где U_i — уровень[2], на котором находится 1-й элемент РПС; n — общее число выделенных контрольных элементов.

На рисунке изображена РПС объектов обучающей выборки с указаниями, к каким классам относятся пирамиды объектов. A(B) — обозначение пирамиды объекта принадлежащего (не принадлежащего) понятию, k-элементы объектов классов A и B помечались + и —, соответственно. Окончательное решение должно подчиняться следующему правилу.

Для объектов класса A(B) при спуске от вершин по ветвям их пирамид k-элементы со знаком + (—) должны встречать-

---

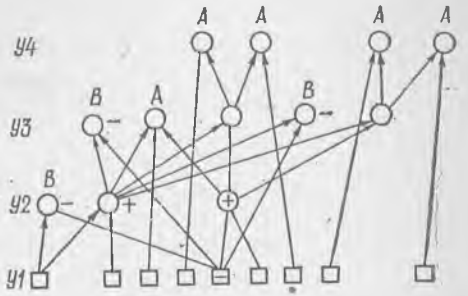
[1] Сеть — это организация памяти об объектах, представленных наборами значений признаков. В сети имеется два типа элементов: рецепторы и ассоциативные элементы. Ввод информации в сеть осуществляется через рецепторы. Каждый рецептор соответствует одному значению признака. Ассоциативный элемент производит выходной сигнал при поступлении сигналов на все его входы. Объекты представлены в сети пирамидами ассоциативных элементов, в основании которых находятся рецепторы, соответствующие значениям признаков из их описаний.

[2] Уровень элемента — это число промежуточных элементов от данного до рецепторов плюс единица, если спускаться по самой длинной ветви данного элемента.

ся раньше, чем k-элементы с — (+), т. е. не должно быть не прикрытых k-элементов с — (+).

Здесь представлены два варианта распределения k-элементов. Оценка первого варианта равна 3, оценка второго — 10.

Предположим, что из двух элементов РПС, участвующих в распознавании одинакового числа объектов обучающей выборки одного класса, выбирается тот, уровень которого выше.



Тогда справедлива следующая теорема.

Теорема. *Чем меньше значение функции φ, тем выше уровень обобщения сформированного понятия.*

Доказательство. Уменьшение значения функции φ является результатом выделения в большинстве случаев меньшего числа k-элементов, причем данные k-элементы принадлежат невысоким уровням РПС.

Так как все объекты обучающей выборки должны быть распознаны правильно, то это означает, что выбраны более общие сочетания признаков, характерные для формируемого понятия.

В случае представительной обучающей выборки это приводит к тому, что вероятность правильного ответа на экзамене (при распознавании объектов, которые не вошли в обучающую выборку, но о которых известно, к какому классу объектов они принадлежат) увеличивается. Теорема доказана.

Решение задачи собственно формирования понятий можно, таким образом, считать эффективным, если в результате получено понятие с возможно меньшим значением функции φ.

Определение оценочной функции предполагает, что эффективное решение должно удовлетворять следующим требованиям: число k-элементов должно быть возможно меньше; k-элементы должны размещаться по возможности на более низких уровнях.

Как нетрудно заметить, здесь речь идет об эффективности самого результата процесса решения, в данном случае выделенного ансамбля k-элементов, который впоследствии будет использоваться для распознавания объектов.

С психологической же точки зрения может рассматриваться еще ряд критериев эффективности решения, которые явно или неявно, сознательно или неосознанно играют роль в процессе решения задачи. Так, принято считать, что эффективной является та стратегия решения задачи, которая приводит к ответу, соответствующему эффективному с точки зрения логико-мате-

матического анализа решению. С другой стороны, такая стратегия эффективна тогда, когда это решение находится за короткое время и на поиск решения затрачивается небольшое количество ходов [3]. Однако данные ряда психологических исследований свидетельствуют о том, что реализация эффективной стратегии в течение одного сеанса участия испытуемого в эксперименте наталкивается на ограничение его ресурсов. В частности зарегистрировано, что в таких случаях испытуемые из-за высокого когнитивного напряжения практически не находят реализации наиболее эффективных стратегий, предполагающих комплексный анализ всей исходной информации, и переходят к использованию менее эффективных, но зато ненапряженных стратегий [3, 4].

Цель нашего исследования — поиск не просто решения задачи собственно формирования понятий, а такого ее решения, которое, базируясь на комплексном анализе всей исходной информации, обеспечивало бы построение более простых, чем в системе «Анализатор», понятий с более высоким уровнем обобщений. Нам нужно выявить способ решения, который бы обеспечивал решение с минимальным значением функции φ.

Мы предположили, что постановка перед испытуемым цели поиска способа получения эффективных решений, ознакомление его с критериями эффективности решения, работа испытуемого в эксперименте не в течение одного сеанса [1], а ряда сеансов будет способствовать успешной реализации им эффективных стратегий решения. Кроме того, мы предположили, что к такому эксперименту целесообразно привлечь лиц, относительно которых можно прогнозировать, что они обычно пользуются в решении задач эвристическими методами сокращения перебора вариантов и поиска решения или способны разрабатывать и применять творческие методы решения.

Методика нашего исследования предусматривала введение и инструкцию для испытуемых информации об оценочной функции решения. Испытуемому сообщалось, какое решение может быть оценено, как эффективное, и по каким критериям. Иными словами, если при традиционной постановке исследования процесса решения задачи испытуемый часто работает с критериями оценки эффективности решения неосознанно, далеко не всегда их формирует и планирует свои действия с ними, то в нашем исследовании испытуемый получал в явном виде сведения об этих критериях. Это, как мы предположили, должно дополнительно стимулировать поиск испытуемыми приемов комплексного анализа исходной информации, обеспечивающих реализацию эффективной стратегии.

Мы применяем следующие приемы фиксации параметров процесса решения испытуемыми экспериментальных задач:

[1] Работу с испытуемыми в течение нескольких сеансов проводила Д. Б. Богоявленская.

запись действий испытуемого, временных параметров решения, направлений поиска, результатов решения, вербального отчета испытуемого, а также регистрацию движений его глаз методом электроокулографии.

В эксперименте участвовало 15 математиков-программистов. Р. В. Горовая в соответствии с методикой креативного поля Д. Б. Богоявленской [5] выделила репродуктивов, которые пользуются, как правило, один раз найденным, надежным способом решения; эвристов, применяющих специальные приемы сокращения поиска и перебора вариантов; и креативов, разрабатывающих и реализующих творческие методы поиска. Каждый из них решил набор задач, который включал 4 типа РПС, всего 9 задач.

Полученные нами экспериментальные данные показали, что испытуемые приняли предложенную им цель решения, осуществляли целенаправленный поиск способов эффективного решения предложенных им задач, приемов комплексного анализа исходной информации, замечали и реализовывали эффективные стратегии решения, т. е. стратегии, которые приводили к получению решений с малым значением функции φ, не требуя при этом от испытуемого значительных затрат ресурсов (время, количество ходов решения, нагрузка на память и т. п.).

Особенно успешно действовали испытуемые, которые в соответствии с методикой креативного поля были отнесены к группе креативов и часть из группы эвристов. Эти испытуемые реализовали эффективные интегративную или интегративно-дифференциальную стратегии.

При этом необходимо отметить, что работа в несколько сеансов позволяла испытуемым находить реализацию намеченных стратегий, при которой затрата ресурсов (времени и когнитивного напряжения) была незначительной.

Рассмотрим, какую роль сыграло в поиске эффективных решений явное представление в получаемой испытуемым инструкции критериев минимизации значений функции φ.

Испытуемые, как уже упоминалось, принимали требование поиска наилучшего решения. После ознакомления с инструкцией они, как правило, старались уяснить приведенные в ней критерии эффективности. Ряд испытуемых дополнял эти критерии, например, такими:

1. Все объекты обучающей выборки должны быть обязательно с помощью полученного решения (выбора k-элементов) распознаны.

2. В определениях не должно быть противоречий.

Таким образом, у испытуемых на вооружении оказывались указанные критерии правильности решения и приведенные в инструкции критерии его эффективности. Те испытуемые, которые реализовали интегративную стратегию и предложили способ эффективного выбора k-элементов в явном виде работали

с критериями (это видно, в частности, из вербального отчета), оценивали их взаимосвязь и соотношение, подсчитывали оценочную функцию. Особенно ярко это проявилось у испытуемого В. П., отнесенного Р. В. Горовой к группе креативов.

С другой стороны, те испытуемые, которые не предложили эффективного способа решения (большей частью это были репродуктивы), не могли, как правило, ориентироваться на критерии эффективности и правильности решения в комплексе. Бывало так, что хотя деятельность испытуемого направлена на поиск эффективного решения, он не дает даже правильного решения. Оказывается, он не учитывал на первых порах требование 2 — в определениях не должно быть противоречий. У таких испытуемых чаще всего проверка решения направлена на выявление неопределенных объектов или противоречий, а не на минимизацию функции φ путем исключения избыточных k-элементов или поиска других вариантов решения.

Интересно, что критерий выбора k-элементов на возможно более низких уровнях РПС очень быстро принимался испытуемыми на вооружение. Многие из них анализировали РПС, двигаясь от нижних уровней к более высоким, что при слабом учете остальных критериев приводит к невозможности проведения комплексного анализа информации и реализации эффективной стратегии решения.

В целом полученные нами данные показали, что такая постановка эксперимента позволяет выявить эффективные способы решения задачи. Введение в эксперимент результатов логико-математического анализа эффективности решения помогает испытуемым осознать критерии качества решения, спланировать свои действия по учету этих критериев, повысить психологическую эффективность получаемых в эксперименте решений.

В данном исследовании нам удалось получить такие способы решения задачи собственно формирования понятий, которые были затем преобразованы в алгоритм, вводимый в настоящее время в систему «Анализатор».

Список литературы: 1. *Бондаровская В. М., Горовая Р. В., Хоменко Л. В.* Исследование процесса поиска эффективного способа решения задачи. — В кн.: Семантические вопросы искусственного интеллекта. — Киев: Знание, 1978, с. 7—8. 2. *Гладун В. П.* Эвристический поиск в сложных средах. — Киев: Наукова думка, 1977. 3. *Bruner T., Goodnow T.* — Gourn. Austin a study of thinking. New York, 1956. 4. *Wetherick N. E.* Structure and content in concept attainment: a full-scate study. — The British Journal of Psychology, 1969, v 60, p. 3. 5. *Богоявленская Д. Б.* Об одном из подходов к исследованию интеллектуального творчества. — Вопросы психологии, 1976, № 4, с. 30—35.

Поступила 2 октября 1979 г.

УДК 510.62

Н. И. ПОВЯКЕЛЬ

ПРОГРАММИРОВАНИЕ КАК СРЕДСТВО ИЗУЧЕНИЯ МЫСЛИТЕЛЬНОЙ ДЕЯТЕЛЬНОСТИ ЧЕЛОВЕКА

Анализ работ по проблеме искусственного интеллекта показывает, что решение ее связано с изучением естественного интеллекта и, в частности с исследованием и моделированием мыслительной деятельности человека. Исследование процесса мышления человека, разрабатывающего программу для ЭВМ, в этом отношении представляет особый интерес: «деятельность человека, составляющего программу для ЭВМ, затрагивает сугубо секретные аспекты человеческого мозга» [1]. Это обусловлено, по нашему мнению, рядом особенностей программирования как человеческой деятельности. Основываясь на результатах психологических исследований, отметим, что человек, решая задачу, не осознает обычно многих операций, которыми он пользуется. В случае же программирования задачи на ЭВМ человек, ориентируя свою деятельность на машину, должен описывать все мельчайшие операции, из которых состоит решение, как бы заглядывая внутрь самого себя, выявляя, а затем и развертывая все детали решения, которые он обычно не осознает [2].

Таким образом, ориентация в деятельности на ЭВМ, с учетом ее особенностей и возможностей «задает» своеобразные требования к деятельности по решению задач. Человек должен сознательно планировать решение, осуществлять логический анализ его, прослеживать до конца все детали решения, и, наконец, описать решение. При этом одной из основных особенностей программирования как человеческой деятельности является использование специфических средств описания алгоритма — языков программирования, а также блок-схемного языка. В этом отношении программирование представляет собой сложную форму деятельности по манипулированию различными средствами — от весьма абстрактных, математических средств до словесно-языковых. По-видимому, в силу этих особенностей и подчеркиваются рядом авторов далеко идущие требования программирования к ограниченным возможностям человека [1].

Необходимость постоянной ориентации на ЭВМ с учетом ограничений и возможностей набора средств * обусловливает особую направленность мышления на собственную деятельность, т. е. обусловливает сложную рефлексивность процессов мышления в ходе осуществления деятельности. Отметим, что рефлексия, применительно к деятельности по решению задач, понима-

* Под средствами решения мы понимаем «материальные, материализованные» и «идеальные» объекты, которые не входят в задачную ситуацию, но привлекаются извне для ее решения». [4].

ется «как осознание оснований собственного движения в предметном содержании задачи с учетом определения средств ее решения» [3]. Исходя при изучении мышления из представления о зависимости мыслительного процесса от его осознанности, мы считаем, что необходимость постоянной рефлексии, «обращения сознания» не только на совершаемую деятельность, но и на возможную деятельность вычислительной машины, а также в связи с этим и на свою деятельность, изменяет не только выполняемые действия, их предметное содержание, но и организацию и продуктивность мыслительного процесса.

Мы полагаем, что отмеченные выше особенности деятельности человека, составляющего программу для ЭВМ, предполагают «овеществление» [3] и экстериоризацию внутренних, психических процессов, и поэтому предоставляют большие возможности для изучения мышления человека. Другими словами, изучаемая деятельность — программирование задачи на ЭВМ, может выступать и выступает в нашем исследовании не только как материал исследования, но как своеобразный методический прием, как средство для изучения и описания мыслительной деятельности человека. Вместе с тем есть основания полагать, что указанные особенности мыслительной деятельности человека, составляющего программу для ЭВМ, в значительной степени определяются, и, соответственно, различным образом выявляются в зависимости от используемого средства деятельности — языка программирования. На определяющее влияние характеристик языков программирования как средств решения, на мышления человека и его понимание задач, неоднократно указывалось виднейшими специалистами в области разработки языков программирования. Так, Питер Наур, применительно к языкам программирования, отмечал, что «нет проблемы вне понимания средства, подходящего для ее решения» [5].

На данном этапе исследования нами изучалась мыслительная деятельность человека-пользователя, разрабатывающего программу для ЭВМ, т. е. осуществляющего деятельность, в основе которой, по мнению ряда авторов [6], «лежат три процесса: осознание задачи, планирование и программирование». При этом изучение проводилось в условиях использования различных языков программирования *.

Мы предполагали, что языки программирования оказывают существенное влияние на мыслительную деятельность человека-пользователя. Более того, мы полагали, что характеристики используемых языков с различной степенью направленности и развернутости позволяют наблюдать и описывать мышление человека.

---

* В исследование не включались ситуации программирования задачи на ЭВМ, в которых этапы ее выполняются разными людьми, а также не рассматривались вопросы отладки.

Исследование осуществлялось на материале решения задач обработки данных с помощью ЭВМ при использовании двух языков программирования: КОБОЛа и ЯОДа. Выбор этих языков в качестве исследуемых был обусловлен: 1) их ориентацией на один и тот же класс задач — обработки данных; 2) спецификой КОБОЛа как языка программирования высокого уровня, наиболее распространенного и широко известного в мировой практике, универсального языка коммерческих задач; 3) спецификой ЯОДа как языка обработки данных, в основу разработки которого было положено психологическое изучение деятельности пользователя-непрофессионала [7].

Испытуемые-пользователи ЭВМ различной профессиональной и общей подготовленности (64 чел.), прошедшие курс обучения вышеназванным языкам в максимально уравненных условиях обучения.

Эксперимент проводился индивидуально с каждым испытуемым. В ходе эксперимента испытуемыми решался набор типовых задач обработки данных, предъявляемых по возрастанию степени сложности *, на указанных языках программирования. Фиксировались и учитывались при анализе: результаты решений (программы), временные показатели, вербальные отчеты испытуемых. Для фиксации и описания процессов решения задач, а также в целях возможной формализации при анализе и оценке деятельности применялась модифицированная модель решения задачи, в которой выделялись микроэтапы решения или нормативные единицы деятельности [4]. При выведении результативно-процессуальной оценки деятельности по решению задач в j-м опыте ($j = 1, 2, \ldots, N$), с использованием как психологических так и формальных методов, была введена численная мера оценки деятельности [8], называемая «эффективность» (y_j) и выражающая количественно соответствие деятельности по решению задач в целом и на отдельных этапах эталонно-нормативной деятельности.

Процедура эксперимента и обработки данных строилась на основе и в соответствии с принципами математической теории планирования эксперимента [9]. Это позволяло нам получить математическое описание эффективности деятельности по решению задач, в зависимости от многофакторного воздействия, в виде уравнения регрессии. Отметим, что данные дисперсионного анализа апробирующего эксперимента позволили предположить, что данное уравнение регрессии целесообразно искать в виде линейной части ряда Тейлора: $Y = \beta_0 + \sum_{i>1}^{n=4} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j$; где n — количество факторов; x_i — значения факторов; β_0, β_i, β_{ij} —

* Степень сложности рассматривалась как относительно объективная характеристика задачи и определялась методом анализа ее с учетом специфики класса задач обработки данных и методом экспертных оценок.

коэффициенты регрессии. В основу настоящего эксперимента был положен факторный план 2^4 (четыре фактора варьировались на двух уровнях) с дублированием опытов. В зависимости $y = f(x_1, x_2, x_3, x_4)$ изучалось парциальное и совместное влияние четырех качественных факторов (x_1 — фактор профессиональной подготовленности пользователя; x_2 — фактор общей подготовленности пользователя; x_3 — фактор используемого языка программирования; x_4 — фактор степени сложности задачи) на функцию отклика y — эффективность деятельности по решению задач.

По обобщении матриц, была получена аппроксимация зависимости $y = f(x_1, x_2, x_3, x_4)$ в виде следующего многочлена:

$$y = 1{,}184 + 0{,}152x_1 + 0{,}162x_2 - 0{,}363x_3 - 0{,}115x_4 - 0{,}022x_1x_2 +$$

$$+ 0{,}154x_1x_3 - 0{,}001x_1x_4 - 0{,}019x_2x_3 - 0{,}005x_2x_4 + 0{,}025x_3x_4.$$

Проверка значимости коэффициентов уравнения регрессии показала, что на 0,10 уровне значимости все выделенные факторы оказывают влияние на эффективность деятельности по решению задач, при этом наибольшее влияние оказывает фактор языка программирования (x_3). Отметим, что заметное влияние оказывает и сочетание факторов x_1x_3, x_3x_4.

Проверка адекватности линейной аппроксимации, проведенная по критерию Фишера [9], показала адекватность такой аппроксимации на 0,05 уровне значимости, что позволяет путем «мысленного эксперимента предвидеть значение функции» [9], эффективности деятельности, в любой точке экспериментального пространства.

Полученные нами экспериментальные данные показывают, что язык программирования оказывает существенное и ведущее влияние как на результат, так и на процесс деятельности по решению задач. При этом влияние языка программирования на результат деятельности проявилось в качестве результатов (программ), в существенных различиях в ошибках, допущенных испытуемыми; во временных показателях решений. Более того, оказалось, что один из языков программирования (ЯОД) повышает эффективность деятельности по решению задач, что подтверждает знак коэффициента ($-a_3$) в полученном многочлене. Данный язык программирования облегчает процесс решения, повышая его эффективность, при использовании этого языка деятельность человека — пользователя по решению задач, становится более удобной, комфортной. Положительное влияние этого языка программирования сказывается, прежде всего, на процессе решения, что проявляется: а) в полноте и точности идентификации объектов; в) во всесторонности анализа объектов в их связях; в) в построении полной и адекватной задачной структуры (модели задачи); г) в последовательности и полно-

те планирования, благодаря чему уменьшилось количество проб и ошибок при устранении различий между объектами; д) в точности и правильности выбора последовательности применения операторов языка; ж) в организованности функций контроля; з) в нахождении более рациональных алгоритмов решения и др.

Язык программирования, как показывает качественный анализ экспериментальных данных, «задает» деятельность, точнее, «задает» способ действий по решению задачи в целом и по этапам. Таким образом, можно сказать, что языки программирования опосредуют операциональную сторону деятельности. При этом, особенности используемого языка, как оказалось предопределяют и «целевое» наполнение действий и операций в их динамике, т. е. видоизменяют и содержательную сторону деятельности по решению задач.

И, наконец, что представляло для нас особый интерес, при использовании различных языков программирования предопределяются, и поэтому, различным образом выявляются, такие особенности мышления человека-пользователя как подробности планирования, степень контроля и самоконтроля в процессе решения, а также степень детализации проблемы. Так, например, в КОБОЛе, значительно меньшая степень детализации по сравнению с машинно-зависимыми языками программирования, и значительно большая степень детализации по сравнению с такими языками программирования как ЯОД. Это связано, прежде всего, со степенью ориентации на ЭВМ при написании программы, а также с особенностями объектов преобразований (операндов) и оперативных единиц преобразований (операторов) в исследуемых языках программирования.

Как оказалось, в исследуемых языках особо существенны различия в крупноблочности операндов и операторов языка. В соответствии с этой особенностью, в одном из языков (КОБОЛ) в сравнении с другим языком (ЯОД), испытуемым в расчете на машину, а также на объект преобразования (запись), приходилось описывать мельчайшие операции, из которых состоит решение. Поэтому степень рефлексии, ее усложнение, в КОБОЛе значительно более высокое, чем в ЯОДе. Во втором языке, ввиду значительно более укрупненного операнда (массив — таблица) и более крупноблочных операторов языка, имели место более свернутые, обобщенные операции осознанного соотнесения и контроля. Другими словами, уровень языка программирования, а также отмеченная многими авторами [10], тенденция к «сжатию» или компрессии элементов языка, оказывает определяющее влияние на мыслительную деятельность человека-пользователя, предопределяя и переструктурируя ее.

Отмеченные выше особенности исследуемых языков программирования, как показывает анализ экспериментальных данных, позволяют различным образом «развернуть», а, значит, и фиксировать мышление человека.

Подводя итоги вышесказанного, отметим, что:

1) Программирование как человеческая деятельность представляет собой богатый материал, а также специфическое средство для наблюдения и изучения мыслительной деятельности человека.

2) Использование различных языков программирования, которые несут в себе различную степень абстракции, сокращенности и обобщенности, позволяет наблюдать и фиксировать, а значит, изучать и описывать с различной степенью «развернутости» и экстериоризации процессов, мышление человека.

Языки программирования как средства деятельности существенно влияют на мыслительную деятельность человека, осуществляющего разработку программы для ЭВМ, поэтому, при разработке языка программирования появляется возможность моделирования тех параметров, моментов мыслительной деятельности человека, которые, в соответствии с целями исследования, представляются необходимыми.

Список литературы: 1. *Ершов А. И.* Aesthetics and the Human Factor in Programming. — Communications of the ACM, 1972, 15, № 7, p. 12—15. 2. *Бондаровская В. М.* Некоторые аспекты психологического анализа деятельности программиста. — В кн.: Очерки психологии труда оператора. — М.: Наука, 1974, с. 45—51. 3. *Алексеев Н. Г.* Познавательная деятельность при формировании осознанного решения задач. — Автореф. дис... канд. психол. наук, М., 1975.—15 с. 4. *Гергей Т., Машбиц Е. И.* К характеристике модели решения учебных задач. — Вопросы психологии, 1973. № 6, с. 9—14. 5. *Naur P.* The place of Programming in a World of Problems Tolls and People. — Froceeding of the IFAP Congress, 1965. 6. *Брукс Р.* Моделирование интеллектуальной деятельности программиста. — Труды IV Международной объединенной конференции по искусственному интеллекту. — М., 1975, с. 25—27. 7. Решение задач обработки данных с помощью ЭВМ. Учебное пособие /Под ред. В. М. Глушкова, А. Л. Стогния, Е. М. Ющенко и др. — Киев: Вища школа, 1975. — 192. 8. *Повякель Н. И., Забара Е. В.* К вопросу о формализации при оценке эффективности деятельности пользователя в системах Человек — ЭВМ. — Психологические аспекты эффективности и надежности систем «Человек-Техника». Ереван, 1979, с. 34—39. 9. *Хикс Ч.* Основные принципы планирования эксперимента. — М.: Мир, 1967.—160 с. 10. *Weinberg G. M.* The psychology of Computer programming. — New York, 1971. — 192 p.

Поступила 2 октября 1979 г.

УДК 510.62

Ю. Ф. ТИТОВ

ВАРИАНТ НЕПРЕРЫВНОЙ ЛОГИКИ, ПРИГОДНЫЙ ДЛЯ АНАЛИЗА СЕТЕВЫХ УСТРОЙСТВ

Достижения в области применения сетевых устройств для решения задач распознавания и автоматического управления [1] позволяют рассматривать данные устройства как перспективные средства вычислительной техники с параллельным принципом обработки аналоговых сигналов. Однако неразработанность формальных методов анализа и синтеза сетевых устройств сущест-

венно сдерживает их широкое использование. Предлагаемый вариант непрерывной логики позволяет исследовать наличие всех статически устойчивых состояний анализируемого сетевого устройства и найти области изменения входных сигналов, приводящих к данным состояниям.

Непрерывная математическая логика — исчисление, оперирующее величинами, заданными в интервале $\{0,1\}$. Известные варианты непрерывной логики Р. Мак-Нотона [2] и С. А. Гинзбурга [3], построенные на основе базиса из трех операций: max, min и дополнения, мало применяются [4].

Излагаемый вариант непрерывной логики отличается тем, что в нем могут использоваться любые операции и функции элементарной алгебры. Чтобы результаты действий принадлежали тому же множеству чисел, что и исходные данные, а именно, изменялись только в интервале $\{0,1\}$, вводится специфическая операция, именуемая двухсторонним ограничением. Благодаря особенностям операции ограничения данный вариант непрерывной логики имеет полезную связь с теорией систем линейных уравнений.

Для обозначения постоянных в непрерывной логике вводится алфавит латинских заглавных букв A, B, C, ... , Z или с индексами A_1, A_2, ... , A_n. Переменные обозначаются строчными латинскими буквами a, b, c, ... , z или с индексами x_1, x_2, ... , x_m. Вводятся скобки круглые, квадратные и фигурные. Определяется, что все возможные значения постоянных и переменных представляют собой линейно-упорядоченное, двухсторонне-ограниченное множество, расположенное в интервале $\{0,1\}$, включая граничные значения.

Вводится основная операция непрерывной логики: **двухстороннее ограничение**. Двухсторонним ограничением по верхнему пределу **1** и нижнему пределу **0** называется операция присваивания некоторой переменной значения **1**, если она больше единицы, и значения **0**, если переменная меньше нуля. Внутри интервала $\{0, 1\}$ значение переменной сохраняется без изменения:

$$L(\alpha) = \begin{cases} 1 \text{ если } \alpha \geqslant 1; \\ \alpha \text{ если } 0 < \alpha < 1; \\ 0 \text{ если } \alpha \leqslant 0. \end{cases} \qquad (1)$$

Вспомогательными операциями непрерывной логики являются операции сложения, вычитания, умножения, деления, а также произвольные алгебраические функции и их комбинации. Все вспомогательные операции имеют тот же смысл, что и в соответствующих разделах математики. В последовательной бесскобочной записи устанавливается следующий порядок выполнения операций непрерывной логики. В первую очередь вычисляются элементарные функции, затем выполняются операции умножения и деления, а потом — сложения и вычитания. Завершающая опе-

рация — ограничение. Если какую-либо операцию требуется выполнять с приоритетом, то ее заключают в скобки (круглые скобки имеют приоритет перед квадратными, а квадратные перед фигурными).

В зависимости от состава вспомогательных операций различают линейные и нелинейные непрерывные логики. Те непрерывные логики, в которых используется не более четырех вспомогательных операций: сложение, вычитание, умножение и деление, называются линейными логиками. Одиннадцать разновидностей линейных логик задаются набором из основной и вспомогательных операций, совместно образующих базис исчисления (см. таблицу). Применение в качестве вспомогательных операций таких функций, как показательная, степенная, логарифмическая и др. дает многочисленные варианты нелинейных логик.

Вариант линейной логики	I	II	III	IV	V	VI	VII	VIII	IX	X	XI
Операции базиса	L + −	L + ×	L + ÷	L − ×	L − ÷	L × ÷	L + − ×	L + − ÷	L + × ÷	L − × ÷	L + − × ÷

Все известные в настоящее время дискретные логики (булева алгебра, мажоритарная, миноритарная, пороговая логики и т. п.) являются частными случаями линейных логик при дискретном изменении переменных. Рассмотрение их выходит за рамки данной статьи. Для анализа сетевых устройств используется седьмой вариант линейной логики, который задается базисом из четырех операций: ограничения, сложения, вычитания и умножения. В дальнейшем будем рассматривать только этот вариант линейной логики. В ней справедливы тождества:

$$L(x) = x;\ L(0) = 0;\ L(1) = 1;\ L(x+1) = 1;\ L(x-1) = 0;$$
$$L(0-x) = 0;\ L(Wx) = Wx;\ L(1-Wx) = 1-Wx; \qquad (2)$$
$$L[1-L(1-Wx)] = Wx;\ L[1-L(1-W_1x_1-W_2x_2)] =$$
$$= L(W_2x_2 - W_1x_1);\ L[1-L(1-W_1x_1)-L(1-W_2x_2)] =$$
$$= L(W_1x_1 + W_2x_2 - 1),$$

где W, W_1, W_2 — константы.

Приводим без доказательства следующую теорему.

Теорема. *Любую функцию линейной логики от произвольного, но конечного числа переменных можно записать в виде системы канонических уравнений вида:*

$$Y_k = L\left[\sum_{i=1}^{I} W_{ik}x_i - \sum_{j=1}^{J} W_{jk}x_j + \sum_{p=1}^{P} V_{pk}y_p - \right.$$
$$\left. - \sum_{q=1}^{Q} V_{qk}y_q + W_{10} - W_{20}\right], \qquad (3)$$

где x_i, x_j — независимые переменные; y_k, y_p, y_q — результирующие переменные; W_{ik}, W_{jk} — постоянные коэффициенты веса независимых переменных; V_{pk}, V_{qk} — постоянные коэффициенты веса результирующих переменных; W_{10}, W_{20} — некоторые константы.

$$0 \leqslant (x_i,\ x_j,\ y_p,\ y_q,\ W_{ik},\ W_{jk},\ V_{pk},\ V_{qk},\ W_{10},\ W_{20}) \leqslant 1. \quad (4)$$

При использовании тождеств линейной логики (2) система канонических уравнений может быть приведена к компактной форме, минимальной по числу уравнений. Наиболее важным свойством канонических уравнений является то, что каждое уравнение содержит только один оператор ограничения.

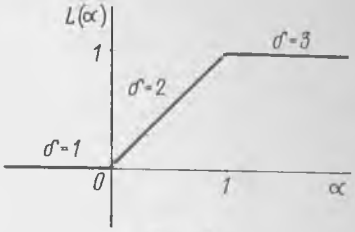


Рис. 1

Произвольное сетевое устройство без СУТ, состоящее из активных узлов и пассивных связей [1], может быть описано некоторой системой канонических уравнений линейной логики (3). Здесь W_{ik}, W_{jk} — веса связей, через которые подаются внешние сигналы на узел k, V_{pk} и V_{qk} — веса связей выходов других узлов с узлом k. При этом предполагается, что каждый узел сетевого устройства обеспечивает суммирование входных сигналов, взятых с весами W_{ik} и W_{jk}, V_{pk}, V_{qk} и двухсторонее ограничение. Исследование данной системы (3) позволяет установить совместность уравнений и получить статические решения. Из-за нелинейности каждого уравнения в системе ее приходится исследовать специальными методами.

Разделим все пространство решений системы на столько областей, сколько потребуется, для того чтобы в каждой области система оказалась линейна. Это возможно потому, что пространство решений имеет кусочно-линейную структуру. Такая структура возникла из-за действия единственного нелинейного оператора в уравнениях — двухстороннего ограничения, которое имеет три линейных участка (см. рисунок). Решение каждого уравнения может находиться в одной из трех областей. Комбинации линейных областей m-уравнений дает 3^m линейных областей пространства решений.

Обозначим некоторую область из 3^m через s, а соответствующее ей решение обозначим через Y_s,

$$Y_s = (y_1^{\delta},\ y_2^{\delta},\ \ldots,\ y_k^{\delta},\ \ldots,\ y_m^{\delta}), \quad (5)$$

где y_k^{δ} — решение k-го уравнения в системе; δ — верхний индекс указывает на принадлежность решения y_k к одной из трех областей:

$$\text{при } \delta = 1 \quad y_k^1 = 0,$$

при $\delta = 2$ $0 < y_k^2 < 1$;

при $\delta = 3$ $y_k^3 = 1$. (6)

В пределах линейной области y_k^2 операторы ограничения не действуют, поэтому в тех канонических уравнениях, где левая часть заведомо находится внутри интервала $(0, 1)$, исключая граничные значения, оператор ограничения может быть устранен,

$$y_k^2 = \sum_{i=1}^{I} W_{ik}x_i - \sum_{j=1}^{J} W_{jk}x_j + \sum_{p=1}^{P} V_{pk}y_p^{\delta} - \sum_{q=1}^{Q} V_{qk}y_q^{\delta} + W_{10} - W_{20}. \quad (7)$$

Для остальных областей, где $\delta = 1$ и $\delta = 3$, решения принимают фиксированные значения $y_k^1 = 0$ и $y_k^3 = 1$ соответственно, что несколько упрощает вид систем уравнений.

Таким образом, исходная система канонических уравнений может быть записана в виде 3^m систем линейных уравнений вида

$$\sum_{l=1}^{m} a_{ls}y_{kr}^{\delta} = b_k, \quad (8)$$

где $r = 1, 2, \ldots, m$; $s = 1, 2, \ldots, 3^m$; $k = 1, 2, \ldots, m$;

$$a_{rs} = \begin{cases} V_{pk}, & \text{если } r = p \neq k; \\ -V_{qk}, & \text{если } r = q \neq k; \\ 1 + V_{pk}, & \text{если } r = p = k; \\ 1 - V_{qk}, & \text{если } r = q = k; \end{cases} \quad (9)$$

$$b_k = \sum_{i=1}^{I} W_{ik}x_i - \sum_{j=1}^{J} W_{jk}x_j + W_{10} - W_{20}. \quad (10)$$

Исследование полученных систем линейных уравнений (8) проводится традиционным методом вычисления определителей. Если определитель одной из систем, вычисляемый из ее коэффициентов a_{rs} не равен нулю, то данная система совместна и имеет решения. В противном случае решения нет. Отметим неоднозначность решения систем на границах линейных областей. Так как границы соприкосновения кусочно-линейных областей принадлежат обеим соприкасающимся областям, то для получения решения на границе областей необходимо потребовать наличия решения в каждой из соприкасающихся областей. Поэтому для дальнейшего анализа выберем из всего многообразия систем линейных уравнений 8, только те, которые совместны при $\delta = 1$, $\delta = 2$ и $\delta = 3$.

В связи с тем, что области изменения решений для каждой выбранной системы нам известны (7), из уравнения (8) можно рассчитать соответствующие области изменения b_k. Зная $b_{k\,\text{макс}}$ и $b_{k\,\text{мин}}$, составим ряд следующих линейных неравенств:

$$b_{k\,\text{мин}} \leqslant \sum_{i=1}^{I} W_{ik}x_i - \sum_{j=1}^{J} W_{jk}x + W_{10} - W_{20} \leqslant b_{k\,\text{макс}}, \quad (11)$$

где $k = 1, 2, \ldots, m$. Разрешая эти неравенства относительно x_i и x_j, получаем области изменений входных переменных, дающих то или иное решение системы канонических уравнений линейной логики.

Анализ сетевых устройств посредством изложенного варианта непрерывной логики более экономичен с точки зрения объема вычислений по сравнению с математической моделью [5]. Объем вычислений при использовании линейной логики, в пересчете на операции сложения, дает число операций, пропорциональное примерно седьмой степени от числа узлов в сетевом устройстве, в то время как методы математического моделирования дают число операций, пропорциональное двенадцатой — пятнадцатой степени от числа узлов в анализируемом сетевом устройстве.

Список литературы: 1. *Амосов Н. М., Касаткин Л. М., Касаткина Л. М.* Активные семантические сети в работах с автономным управлением. — Тр. IV Междунар. конф. по искусственному интеллекту, 1975, т. 9, 41—47. 2. *Мак-Нотон Р.* Теорема о бесконечнозначной логике высказываний.— Кибернет. сб., 1961, № 3, с. 46—54. 3. *Гинзбург С. А.* Непрерывная логика и ее применения. — Автоматика и телемеханика, 1967, № 2, с. 23—30. 4. *Гинзбург С. А.* Математическая непрерывная логика и изображение функций. — М.: Энергия, 1968.— 168 с. 5. *Амосов Н. М.* Автоматы и разумное поведение. — Киев: Наук. думка, 1973.—140 с.

Поступила 2 октября 1979 г.

УДК 510.62

Е. С. КУЗИН, О. Б. РЯДЧЕНКО, И. Б. ФОМИНЫХ, М. С. ЧЕРКАШИН

ОБ ОДНОМ ПОДХОДЕ К РЕШЕНИЮ ЗАДАЧ ПЛАНИРОВАНИЯ

Рассмотрим задачу планирования как задачу синтеза частично упорядоченной во времени последовательности операторов (в общем случае макрооператоров), приводящих к цели [1]. Под целью понимаем частично определенную последовательность требуемых обобщенных ситуаций проблемной среды, доопределение которой производится в процессе решения задачи в зависимости от конкретных условий.

Такая постановка задачи характерна и для задач функционирования некоторой решающей системы (РС) робота в классе квазидинамических неполностью определенных сред. В средах допускаются изменения, независимые от решающей системы, но отсутствует явная зависимость элементов модели проблемной среды от времени. Кроме того, информация, которая доступна системе, может быть неполной и неточной.

Решение задачи планирования можно свести к решению ряда подзадач, основные из которых следующие: разработка принципов представления и организации знаний о проблемной среде и возможностях РС, адекватных классу решаемых задач; разработка на базе некоторой знаковой системы эффективности с

достаточным набором выразительных средств языка представления знаний; разработка механизмов представления и хранения знаний на машинном уровне; разработка эффективных стратегий, позволяющих совместно с декларативно представленными знаниями автоматически формировать план действий РС в заданном классе сред.

Рассмотрим способы решения этих подзадач с помощью подхода, основанного на последовательном использовании смыслового содержания системы понятий, описывающих данный класс проблемных сред и составляющих семантическое знание решающей системы робота.

Информация, необходимая для решения задач планирования, задается РС в виде семиотической проблемно-ориентированной системы знаний (СЗ), представляющей собой целостную совокупность взаимосвязанных сведений, существенных для данного класса задач, о предметах, отношениях, свойствах, процессах и закономерностях реального мира; сведений о целях и возможностях РС, а также сведений о стереотипах, стратегиях и логике рассуждений при решении подклассов данного класса задач планирования. Основные принципы организации СЗ — структурирование и ассоциативность, определяющие целостность СЗ. Структурирование системы знаний имеет следующие аспекты.

Во-первых, в соответствии с двумя типами сведений, представляемых в различных знаковых системах, в СЗ условно выделяются знания, организованные в виде модели проблемной среды (фактуальные знания), и знания, выражаемые с помощью алгоритмических схем рассуждения (актуальные знания). На данном этапе исследований первые представляются на семиотическом уровне в рамках специального декларативно-процедурного языка и на машинном уровне с помощью списковых структур; знания второго рода представляются на алгоритмических языках программирования типа *PL/1*, *LISP*. Они выражают две возможные формы представления знаний: декларативную и процедурную. Причем часть процедурно представленных знаний может быть введено в состав фактуальных знаний (в частности, знание об операторах рассуждений).

Во-вторых, структурирование информации осуществляется на базе выделения в модели среды ядерных знаний, имеющих общий характер и истинных для решения данного класса задач планирования, и ситуативных (контекстных) знаний, истинных для данного контекста. В качестве примера ситуативных знаний назовем описания начальных, промежуточных и целевых ситуаций. В свою очередь, в ядерных знаниях выделяются знания различных типов знания общего характера о предметах, отношениях, свойствах и процессах данного класса проблемных сред (в частности, о причинно-следственных и пространственно-временных отношениях), знания о возможных действиях (операторах) РС (в частности, о физических и логических последствиях

применения операторов. Эти знания служат основой для построения элементов псевдофизической логики: казуальной, пространственно-временной, логики действий. Наконец, в ядерных знаниях могут быть выделены мета-знания, т. е. знания о способах преобразования первичных знаний (например, знания в виде операторов рассуждения).

Следующий аспект структурирования знаний заключается в классификации знаний по семантическим категориям и построении иерархии понятий в рамках каждой выделенной семантической категории по отношениям часть — целое и частное — общее. Набор семантических категорий открытый. В разрабатываемой версии экспериментальной решающей системы введены семантические категории признаков, отношений, предметов, изменений, ситуаций, действий, процедур.

Важно отметить, что структурирование знаний определяется тем, как оно используется в системе алгоритмов, т. е. функциональной структурой всей системы знаний (в этом проявляются свойства ассоциативности и целостности знаний). Это, в частности, означает, что процессы обобщения и укрупнения направляются значимостью смысла понятий для функционирования решающей системы. Иерархия понятий оказывается *субъективно* обусловленной с точки зрения РС и динамически зависящей от класса решаемых задач.

Другим основополагающим принципом при построении СЗ является принцип *ассоциативности* знаний, отражающий свойство связности знаний. Реализация принципа ассоциативности осуществляется на базе структурированной семантической сети, которую можно интерпретировать как совокупность гиперграфов с раскрашенными вершинами и ребрами, допускающих представление n-арных отношений, ветвлений и циклов. Узлами сети являются как атомарные, так и сложные понятия (предложения), а в качестве связей выступает класс понятий, именуемый бинарными отношениями. Причем одни и те же отношения в различных фрагментах семантической сети могут выступать как узлы и как связи. Допускаются ассоциативные связи процедур (алгоритмов) с узлами семантической сети. В этом случае процедуры оформляются как подпрограммы.

В качестве языка представления знаний, организованных в виде модели среды, используется специально разработанный декларативно-процедурный язык описания проблемных сред (ЯПЗ). В этом языке допускается квантификация переменных, множеств и предикатов, за счет чего существенно расширяются выразительные возможности языка. Набор квантификаторов открытый и организован в иерархическую систему.

В разрабатываемой версии РС введены следующие квантификаторы: логические (существования и всеобщности), размытые (несколько, много, мало); численные (один, два, пять и т. д.).

Синтаксис языка основан на правилах построения конструкций и осуществляется индуктивным способом. В качестве классов правильно построенных конструкций ЯПЗ вводятся элементарные конструкции, конструкции на уровне элементов и конструкции на уровне предложений.

Представление знаний, организованных в виде модели среды, на машинном уровне осуществляется на основе цепных списков, реализованных с помощью аппарата базированных массивов структур языка программирования *PL*/1 [2]. Следует отметить, что при интерпретации знаний в виде списковых структур особое внимание должно быть обращено на адэкватность этого представления представлению модели среды на языке представления знаний; другими словами, должно быть обеспечено представимость на основе списков всех правильных конструкций, допустимых в ЯПЗ.

Цепные списки состоят из двух типов объектов: элементов (*EL*) и ассоциаторов (*AS*). Узлам семантической сети сопоставляются элементы, а связям — ассоциаторы. При таком подходе каждое предложение — тройка языка ЯПЗ с помощью цепных списков выражается в виде последовательности типов объектов списка: $\delta_1 := < EL, AS, EL >$. Для представления типов списка используется аппарат базированных массивов структур языка *PL*/1. Эти структуры имеют одинаковую длину и фиксированное число полей. В полях типов объектов указаны адресные ссылки структур друг на друга, а также характеристики элементов семантической сети (например, цвет вершины или ребра, имя квантификатора, имя семантической категории).

Все массивы структур помещены в некоторый одномерный массив, каждая строка которого содержит один тип элементов списка и имеет длину, равную длине структуры. Используется страничный способ хранения модели на магнитных дисках.

Стратегии планирования (алгоритмические схемы рассуждения) в предлагаемом подходе носят преимущественно эвристический характер. Они определены на ассоциативной семантической сети, частично сами декларативно представлены в ней в виде операторов рассуждений и осуществляют в ней поиск по связям элементов с требуемым смысловым содержанием и их преобразование. На каждом этапе рассматривается только часть связей данного понятия с другими, т. е. вычленяется необходимая в данный момент часть его смыслового содержания. Направление процесса решения задачи определяется заложенными в алгоритмах стратегиями и той информацией, которая была получена на предыдущих этапах рассуждения.

Результатом работы системы алгоритмов является формирование близких к оптимальным планов решения конкретных задач данного класса в пределах, содержащихся в системе знаний. План представляет собой графоподобную структуру с ветвлениями и циклами, узлами которой являются промежуточные

ситуации (подцели), а в качестве связей выступают операторы, осуществляющие достижение этих подцелей. Допускается априорное изменение критерия оптимальности. В частности, в качестве критерия оптимальности может выступать требование минимизации числа операторов в плане при условии достижения цели. В общем случае цель, как отмечалось выше, задается в виде обобщенного разветвленного описания целевого процесса. Процесс решения задачи при таком подходе следует понимать как итеративное доопределение целевого процесса. При этом конкретизируются элементы цели, некоторые из них корректируются, выявленные подзадачи включаются в описание целевого процесса как новые подцели. В процессе формирования плана решения задачи цель может пополняться указаниями и рекомендациями оператора-пользователя.

Условием начала функционирования системы алгоритмов является наличие рассогласований и неопределенностей между задаваемыми в конкретной задаче целевой S_z и начальной S_0 ситуациями, выявляющихся в процессе сравнения этих структур. В качестве рассогласований выступают различия в значении признака, в отношениях или, в более общем случае, различия между сравниваемыми подструктурами целевой и текущей ситуаций. Под неопределенностью понимается суждение, для которого в начальной ситуации не нашлось ни удовлетворяющего ему суждения, ни противоречащего. При определении очередности и последовательности устранения рассогласований используется стратегия разветвленного обобщенного планирования. Это означает, что в процессе формирования плана на основании оценки уместности каждого оператора составляются частично упорядоченные последовательности операторов — наброски окончательного плана — которые затем корректируются и дополняются другими необходимыми операторами и конкретизируются с помощью специальных процедур. Под обобщенным понимается планирование с использованием сложных семантических конструкций: обобщенных элементарных и макрооператоров.

Операторы являются обобщенными в том смысле, что они описываются через обобщенные понятия *классов* предметов, признаков, отношений; под макрооператором понимается частично упорядоченная последовательность операторов, содержащая знания о решении отдельных подклассов задач. После итеративного формирования очередного наброска плана решения данной задачи он анализируется системой с целью выбора актуальной подзадачи. При первом проходе наброском плана является искомая задача, т. е. пара (S_0, S_z). Актуальная подзадача выбирается в зависимости от оценки предпочтения, сложности решения и ресурсов. Выбранная подзадача $(S^m_{0_i}, S^m_{z_i})$ анализируется путем сопоставления структур, образующих подзадачу, и если $S^m_{z_i}$ удовлетворяется в $S^m_{0_i}$, то задача считается тривиаль-

ной и исключается из наброска плана. Если при сравнении выявились только одни неопределенности, то через монитор формируется запрос к метасистеме относительно устранения неопределенностей и продолжается планирование по другим ветвям. И, наконец, если при сравнении структур $S^m_{0_i}$, $S^m_{z_i}$ выявилось хотя бы одно рассогласование, то система осуществляет поиск оператора и анализ его типа по квантификатору. Возможны три случая.

А. Если найденный оператор D_i элементарный, то моделируется его применение в обоих направлениях: от начальной и от конечной ситуации. При этом если оператор D_i применим в одной из ситуаций, образующих подзадачу $(S^m_{0_i}, S^m_{z_i})$, то система фиксирует удовлетворение соответственно ситуации $S^n_{D_i}$ в $S^m_{0_i}$ или ситуации $S^k_{D_i}$ в $S^m_{z_i}$, (где $S^n_{D_i}$, $S^k_{D_i}$ являются описаниями начальной и конечной ситуации применения оператора D_i) и переходит к выбору следующей подзадачи. Если же оператор D_i не применим ни в одной из ситуаций $S^m_{0_i}$, $S^m_{z_i}$, то формируются новые подзадачи — достичь из ситуации $S^m_{0_i}$ ситуацию $S^n_{D_i}$, а из ситуации $S^k_{D_i}$ ситуацию $S^m_{z_i}$, и эти подзадачи включаются в очередной набросок плана. И, наконец, если при проверке применимости оператора D_i выявились одни неопределенности, то управления передаются монитору, в котором формируется запрос с целью доопределения ситуаций $S^m_{0_i}$, $S^m_{z_i}$. Это выполняется путем прослеживания последовательности ситуаций от $S^m_{0_i}$ до начальной ситуации искомой задачи S_0 или от $S^m_{z_i}$ до целевой ситуации S_z.

В. В случае, если для выбранной подзадачи с помощью процедуры поиска найден макрооператор, то производится формальное включение его структуры в очередной набросок плана, чтобы промоделировать его применение на элементарном уровне.

С. Если выбранная подзадача образована неопределенным оператором, т. е. известно только некоторое подмножество операторов (в общем случае, это все множество операторов), с помощью которого можно решить данную подзадачу, что с помощью стратегии нелинейного планирования осуществляется формирование и упорядочение подцелей выбранной подзадачи. Результат применения стратегии может быть трояким.

Во-первых, может быть найдено и упорядочено некоторое подмножество операторов (в частном случае, хотя бы один оператор). В этом случае это упорядоченное подмножество операторов включается в очередной набросок плана. Упорядочение происходит на основании оценки уместности оператора. Во-вторых, для всех рассогласований, выявленных при сравнении структур $S^m_{0_i}$, $S^m_{z_i}$, составляющих искомую подзадачу, может не

найтись ни одного оператора — кандидата, хотя эти рассогласования не помечены в семантической сети как неустранимые. Управление передается монитору, в котором формируются соответствующие сообщения о доопределении операторов. И, наконец, третий случай аналогичен второму с той лишь разницей, что рассогласования, составляющие подзадачу помечены в сети как неустранимые. Например, эти рассогласования по значениям постоянных признаков. В этом случае управление также передается монитору, который фиксирует тупик данной ветви и продолжает планирование по другим ветвям.

Итак, при любом исходе результаты решения подзадачи записываются в очередной набросок плана, и он итеративно анализируется до тех пор, пока не будет найдено квазиоптимальное решение искомой задачи либо не исчерпаются отведенные для решения ресурсы.

Описанный подход последовательно реализован при построении информационно-программного обеспечения экспериментальной решающей системы, базирующейся на ЭВМ ЕС 1020. В качестве проблемной среды используются задачи из класса сборки произвольных пространственных конструкций из геометрических тел при их произвольной начальной конфигурации.

Список литературы: 1. *Кузьмин Е. С., Поздняк Г. Е., Фоминых И. Б.* Задача планирования деятельности робота с искусственным интеллектом. — Тр. IV МОКИИ, 1975, т. 1, с. 48—56. 2. *Гриднев А. А.* О машинном представлении модели среды робота в виде «универсальных» цепных списков. — Вопросы радиоэлектроники. Сер. общетехн., 1976, вып. 9, с. 20—25.

Поступила 2 октября 1979 г.

УДК 681.3.06

С. К. КОЛУБАЙ

ПРОГРАММИРОВАНИЕ И РЕАЛИЗАЦИЯ ЦИКЛИЧЕСКИХ ПРОЦЕССОВ НА АСИНХРОННОМ ЯЗЫКЕ ТИПА $<i, k, j, l>$

Способы программирования и средства реализации линейных и разветвляющихся вычислительных процессов на асинхронном языке типа $<i, k, j, l>$ рассматривались в работах [1, 2].

Поскольку язык является асинхронным, то его эффективная реализация может быть обеспечена только путем использования соответствующей аппаратной поддержки процессов диспетчеризации. С этой целью в работе [2] предложен блок ассоциативной процессоро-памяти (Пр-П), состоящий из ячеек и блока управления. Каждая ячейка Пр-П может находиться только в одном из четырех состояний: *свободна*, *не готова*, *готова*, *выполняюсь*.

Самым существенным моментом в организации вычислительных процессов без циклов является то, что при осуществлении

программы каждая ее инструкция выполняется не более одного раза. Поэтому ячейка процессоро-памяти разработана так, чтобы она всегда переходила из состояния *выполняюсь* в состоянии *свободна*, в котором инструкция, хранящаяся в ячейке, становится недоступной для считывания. Кроме этого, процессы корректировки состояний ячеек изменяли содержимое полей j и l инструкции.

Организация циклических процессов требует многократного выполнения инструкций, находящихся в теле цикла. Поэтому после выполнения инструкции ячейка должна переходить из состояния *выполняюсь* в состояние *не готова* для того, чтобы она могла опять быть переведена в состояние *готова*. Кроме этого, процессы корректировки состояний ячеек не должны изменять содержимого полей j и l инструкции. Таким образом, в соответствии с этими требованиями должны быть внесены изменения в граф переходов ячейки процессоро-памяти и в ее операционную схему.

Требование перехода ячейки в состояние *не готова* после выполнения хранящейся в ней инструкции обусловливает также и то, что в начальный момент времени все ячейки, занятые инструкциями программы, должны находиться в состоянии *не готова*. Это условие наиболее просто удовлетворить следующим образом: в инструкциях, которые должны быть готовы при запуске программы на выполнение, в качестве номеров j и l проставляется некоторый произвольный номер — номер параллельных точек входа, не равный ни одному из номеров j и l инструкций программы.

В связи с модификацией ячейки процессоро-памяти представляется целесообразным также заменить инструкцию управления на новую, более соответствующую духу языка типа $<i, k, j, l>$. В работе [1] показано, что каждая инструкция управления $i)$ $L_i : m, n(j, l)$ должна сопровождаться двумя инструкциями пребразования $i)$ $P(m)$ и $i)$ $P(n)$, обеспечивающими перевод ячейки, содержащей эту инструкцию управления, в состояние *свободна* из состояния *выполняюсь* при любом результате m или n. Вводимая ниже инструкция управления позволяет вместо этих трех инструкций записывать в программе только две, практически эквивалентные им по результатам, а в большинстве случаев обойтись только одной инструкцией управления.

Новая инструкция управления записывается в следующем виде:

$$i)\ L_i(j,\ l), \qquad (1)$$

где i — номер инструкции; j, l — номера, задающие связи по данным и управлению; L_i — логическая функция, вырабатывающая логические значения **истина** (*true*) или **ложь** (*false*).

При выполнении инструкции управления (1), если L_i принимает значение **истина**, то вырабатывается номер i, в противном случае — номер $-i$ (номер i с отрицательным знаком).

Эта инструкция управления позволяет естественным образом исключить инструкцию преобразования из необходимого набора инструкций, так как инструкция преобразования $i)\,P\,(j,\ l)$ эквивалентна инструкции управления $i)\,\boldsymbol{true}\,(j,\ l)$. Поэтому множеству $\{i)\,L_i : m,\ n\,(j,\ l);\ i)\,P\,(m);\ i)\,P(n)\}$ со старой инструкцией управления и инструкциями преобразования практически эквивалентно множество $\{m)\,L_i\,(j,\ l);\ n)\,\boldsymbol{true}\,(-m)\}$ с двумя инструкциями управления типа (1).

Таким образом, программа P на асинхронном языке типа $< i,\ k,\ j,\ l >$ состоит из инструкций только двух типов: инструкций обработки, имеющих вид $i)\,k_i\,(j,\ l)$ и инструкций управления — $i)\,L_i\,(j,\ l)$, где k_i — команда обработки или ввода — вывода. *Программой* называется конечное множество $P = \{a_1,\ a_2,\ \ldots\ a_n\}$ *инструкций с номером z параллельных точек входа*. Каждая инструкция $a \in P$ состоит из четырех компонент $a = (a^i,\ a^k,\ a^j,\ a^l)$, причем a^i — номер и a^k — *команда инструкции*, а a^j и a^l — *номера, задающие связи* инструкции a по данным и управлению. Номер z удовлетворяет условию: $\exists\,(a \in P)\,(a^j = z \wedge a^l = z)$. Номера $a_1^j,\ a_2^j,\ \ldots,\ a_n^j$ инструкций программы — натуральные числа, а номера $a_1^l,\ a_1^l,\ a_2^j,\ a_2^l,\ \ldots,\ a_n^j,\ a_n^l$ — целые числа, не равные нулю. Тип инструкции a определяется типом ее команды a^k. Если a^k — команда обработки или ввода — вывода, то a — *инструкция обработки*. Если a^k — команда, определяющая вычисление некоторой логической функции, то a — *инструкция управления*.

Рассмотрим граф переходов и операционную схему ячейки процессоро-памяти, соответствующей сформулированным выше требованием и новой инструкции управления, т. е. обеспечивающей организацию циклических вычислительных процессов при выполнении программ.

На рис. 1 приведен граф переходов ячейки Пр-П, при построении которого приняты следующие обозначения; *свободна, не готова, не готова 1, не готова 2, не готова 3, готова, выполняюсь* — состояния ячейки Пр-П; $a^i,\ a^j,\ a^l$ — номера инструкции $a = (a^i,\ a^k,\ a^j,\ a^l)$, хранящейся в этой ячейке; **сброс**, **запись**, **чтение** — управляющие сигналы, поступающие на ячейку из блока управления Пр-П.

Как видно из графа переходов, ячейка Пр-П из любого состояния по сигналу **сброс** переходит в состояние *свободна*, причем только по этому сигналу. По сигналу **запись** ячейка переходит из состояния *свободна* в состояние *не готова* (рис. 1, *а*) или в состояние *не готова 1* (рис. 1, *б*). Одновременно на ее регистры записывается некоторая инструкция a программы.

На рис. 1, *а* показан граф переходов ячейки для случая $a^j = a^l$, а на рис. 1, *б* — для случая $a^j \neq a^l$. Напомним, что содержимое каждого поля $a^i,\ a^j,\ a^l$ любой инструкции a не равно нулю. Если $a^j = a^l$, то ячейка при поступлении на нее но-

мера a^j переходит из состояния *не готова* в состояние *готова*. Если же $a^j \neq a^l$, то при поступлении на ячейку номера a^j, она переходит из состояния *не готова 1* в состояние *не готова 2*, а при поступлении на нее номера a^l — в состояние *не готова 3*. Из состояния *не готова 2* ячейка переходит в состояние *готова*

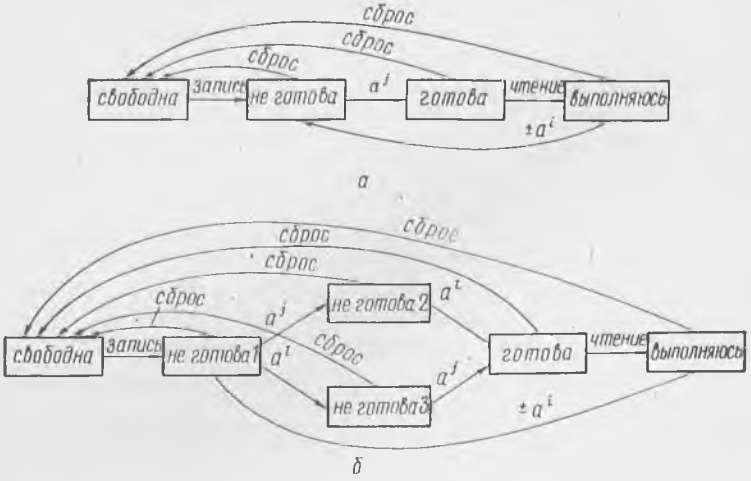


Рис. 1

при поступлении на нее номера a^l, а из состояния *не готова 3* — при поступлении на нее номера a^j.

В состоянии *готова* ячейка находится до тех пор, пока на нее не поступит сигнал *чтение* из блока управления Пр-П. При этом из ячейки считывается номер и команда инструкции, т. е. пара (a^l, a^k), которая передается из блока Пр-П на процессор вычислительной системы для выполнения команды a^k. Сама же ячейка переходит в состояние *выполняюсь*, в котором она остается до тех пор, пока на нее не поступит номер a^i или $-a_i$, сигнализирующий о нормальном завершении выполнения команды a^k. Напомним, что если a^k — команда обработки или ввода — вывода, то на ячейку может поступить только номер a^i, а если a^k реализует некоторую логическую функцию L_i (логическая команда или команда сравнения), то в зависимости от результата ***true*** (или ***false***) может поступить как номер a^i, так и номер $-a^i$. При поступлении на ячейку номера a^i (или $-a^i$) она переходит в состояние *не готова* (рис. 1, *а*) или в состояние *не готова 1* (рис. 1, *б*), т. е. в состояние, в котором она находилась после записи на нее инструкции. Таким образом, это обеспечивает то, что ячейка вновь может перейти в состояние *готова* и т. д.

На рис. 2 показана операционная схема ячейки процессоропамяти, состоящая из регистров R_1, R_2, R_3 и R_4, триггеров T_1,

T_2 и T_3, схем сравнения C_1, C_2 и C_3, элементов ИЛИ ИЛИ$_1$, ИЛИ$_2$ и ИЛИ$_3$, элементов И$_1$ и И$_2$, а также управляемых шин.

Информация о состоянии ячейки формируется на выходах **В** и **Г**. Состояние ячейки характеризуется парой (значение сигнала на выходе **В**, значение сигнала на выходе **Г**), причем (0, 0)

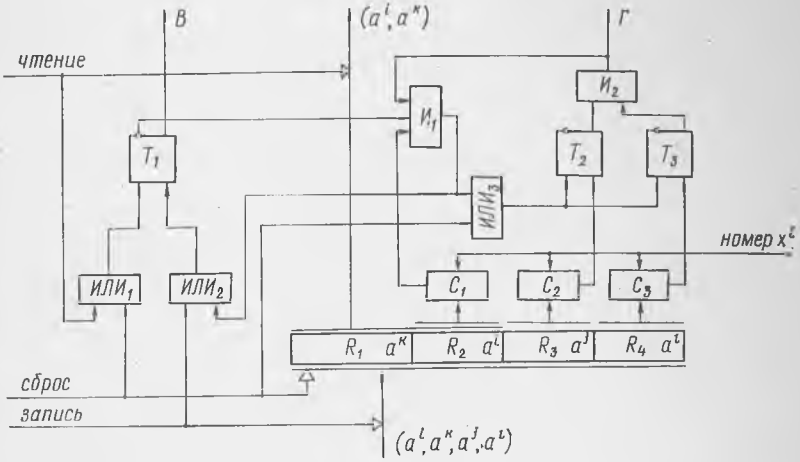


Рис. 2

означает, что ячейка находится в состоянии *свободна*; (1, 0)— *не готова* (а также *не готова 1*, *не готова 2*, *не готова 3*); (1, 1) — *готова*; (0,1) — *выполняюсь*.

Управляется ячейка сигналами **сброс**, **запись** и **чтение** от блока управления Пр-П. Кроме этого, на шины номер x^i всех ячеек передается параллельно через блок управления Пр-П номер каждой очередной выполнившейся инструкции, т. е. номера $\pm a^i$ этих инструкций. На управляемую шину (a^i, a^k, a^j, a^l) из входных регистров Пр-П передается очередная записываемая инструкция программы. Эта шина управляется сигналом **запись**. С управляемой шины (a^l, a^k) на выходные регистры Пр-П передается номер и команда считываемой инструкции. Эта шина управляется сигналом **чтение**.

Работает ячейка следующим образом. В начальный момент времени из блока управления Пр-П параллельно на все ячейки передается сигнал **сброс**. Этот сигнал, поступая на ячейку, устанавливает в нулевое состояние регистры R_1, R_2, R_3 и R_4, а также триггер T_1 через элемент ИЛИ$_1$ и триггеры T_2 и T_3 через элемент ИЛИ$_3$. В результате все ячейки процессоро-памяти перейдут в состояние *свободна* (сигналы (0, 0) на выходах **В** и **Г**).

Запись инструкции в ячейку осуществляется путем передачи содержимого инструкции на шину (a^i, a^k, a^j, a^l) и управляюще-

го сигнала **запись**. В результате компоненты инструкции запишутся на регистры ячейки, причем на регистр R_1 запишется команда a^k инструкции, на регистр R_2 — номер a^i инструкции, а на регистры R_3 и R_4 — номера a^j и a^l, соответственно. При этом ячейка переходит в состояние *не готова* (или *не готова 1*), так как сигнал запись через элемент $ИЛИ_2$ установит триггер T_1 в единичное состояние, т. е. на выходах **В** и **Г** ячейки установится сигнал (1, 0).

Для записи всей программы, содержащей n инструкций, в процессоро-память необходимо n произвольно расположенных ячеек в состоянии *свободна*. В результате записи программы все эти ячейки перейдут в состояние *не готова* (или *не готова 1*).

Инициирование выполнения программы, записанной в процессоро-память, осуществляется передачей на блок управления Пр-П значения номера z параллельных точек входа, что может быть осуществлено, например, с пульта управления вычислительной системы или с ее центрального устройства управления. Значение номера z в блоке управления Пр-П размножается и передается параллельно и одновременно на шины номер x^t всех ячеек, находящихся в состоянии *не готова* и *не готова 1*.

В каждой из ячеек значение номера z поступает на первые входы схем сравнения C_1, C_2 и C_3, вторые входы которых подключены к выходам регистров R_2, R_3 и R_4, соответственно. Если содержимое регистра совпадает со значением номера, поступившего на первый вход схемы сравнения, то эта схема сравнения формирует на выходе сигнал. Номер z параллельных точек входа обязательно совпадает с номером a^j и a^l некоторой ячейки, находящейся в состоянии *не готова*. Следовательно, эта ячейка перейдет в состояние *готова*, так как сигналы со схем сравнения C_2 и C_3 установят триггеры T_1 и T_2 в единичное состояние, а значит на выходе элемента $И_2$ сформируется единичное значение сигнала.

Таким образом, в результате инициирования программы в состояние *готова* перейдут все ячейки, в регистрах R_3 и R_4 которых хранится номер, совпадающий по значению с номером точек входа.

Выполнение программы состоит в передаче номеров и команд инструкций, содержащихся в ячейках в состоянии *готова*, на свободные процессоры системы. Для этого из блока управления Пр-П на каждую такую ячейку поочередно посылаются сигналы **чтение**, которые управляют передачей команд и номеров инструкций с регистров R_1 и R_2 через управляемую шину (a^i, a^k) на выходные регистры процессоро-памяти, с которых они передаются на свободные процессоры системы.

Ячейки, на которые поступили сигналы **чтение**, переходят в состояние *выполняюсь*, так как этот сигнал через элемент $ИЛИ_1$ устанавливает триггер T_1 в нулевое состояние, а значит на выходе **В** сформируется нулевое значение сигнала.

Если в вычислительной системе достаточно процессоров, то все ячейки, которые находились в состоянии *готова*, в результате чтения могут перейти в состояние *выполняюсь*.

Выполнив команду инструкции, процессор передает на блок управления Пр-П номер инструкции, причем возможно с отрицательным знаком, если выполнилась логическая команда или команда сравнения. Процессор же переходит в состояние *свободен*.

Номер инструкции размножается в блоке управления Пр-П и передается параллельно и одновременно на шины номер x^i всех ячеек, находящихся в одном из состояний неготовности или в состоянии *выполняюсь*. В ячейках этот номер сравнивается на схемах сравнения C_1, C_2 и C_3 с содержимым регистров R_2, R_3 и R_4. Если ячейка находилась в состоянии *выполняюсь* и сработала ее схема сравнения C_1, то это означает, что выполнилась команда инструкции, хранящейся в ячейке. При этом сигнал со схемы сравнения C_1 пройдет через элемент $И_1$ и установит триггер T_1 в единичное состояние через элемент $ИЛИ_1$, а через элемент $ИЛИ_3$ установит триггеры T_2 и T_3 в нулевое состояние. В результате ячейка перейдет в исходное состояние неготовности (*не готова* или *не готова 1*). Если же ячейка находилась в одном из состояний неготовности и сработала ее схема сравнения C_2 или C_3 или обе вместе, то, в соответствии с графом переходов (рис. 1), эта ячейка переходит в состояние *не готова 2*, или *не готова 3*, или *готова*.

Описанный процесс выполнения программы продолжается до тех пор, пока все ячейки, занятые инструкциями программы не окажутся одновременно в состоянии неготовности. Такое состояние всех инструкций программы и определяет момент конца счета по этой программе.

Программирование на языке типа $< i, k, j, l >$ циклических вычислительных процессов и реализацию их при помощи процессоро-памяти с ячейками, изложенного выше типа, рассмотрим на примере составления и выполнения программы для решения дифференциального уравнения первого порядка методом Рунге — Кутта.

Пример. Найти решение дифференциального уравнения $y' = x \cdot y$, удовлетворяющее начальному условию $y = y_0$ при $x = x_0$ на отрезке $[x_0, x_n]$ с шагом h, применяя формулу $\Delta y_i = \frac{1}{6} \cdot [k_1 + 4 \cdot k_2 + k_3]$, где $k_1 = h \cdot x_i \cdot y_i$; $k_2 = h\left(x_i + \frac{h}{2}\right)\left(y_i + \frac{k_1}{2}\right)$; $k_3 = h\,(x_i + h)\,(y_i + 2 \cdot k_2 - k_1)$.

В программе команды инструкций записаны в алголоподобной нотации, а каждая инструкция $(a = a^i,\ a^k,\ a^j,\ a^l) \in P$ в виде $a^i \cdot a^k\ (a^j,\ a^l)$, причем, если $a^j = a^l$, то для сокращения записи такая инструкция записывается в виде $a^i \cdot a^k\ (a^j)$.

Программа P, определяющая процесс решения этого дифференциального уравнения, имеет вид:

$P = \{$1. ввод $(x1, y1, h1, n)$ (29);
2. $i := 0$ (29);
3. $h2 := h\ 1/2$ (1);
4. *true* (2, 3);
5. $r1 := x1 \times y1$ (4);
6. $x2 := x1 + h2$ (4);
7. $x3 := x1 + h1$ (4);
8. $i := i + 1$ (4);
9. $k1 := h1 \times r1$ (5);
10. $r1 := k\ 1/2$ (9);
11. $y2 := y1 + r1$ (10);
12. $r1 := x2 \times y2$ (11, 6);
13. $k2 := h1 \times r1$ (12);
14. $r1 := 2 \times k2$ (13);
15. $r2 := y1 - k1$ (9);
16. $y3 := r1 + r2$ (14, 15);
17. $r3 := x3 \times h1$ (7);
18. $k3 := y3 \times r3$ (16, 17);
19. $r4 := k2 \times 4$ (13);
20. $r1 := k1 + k3$ (9, 18);
21. $x1 := x3$ (18);
22. $x[i] := x3$ (7, 8);
23. *true* (21, 22);
24. $r1 := r1 + r4$ (20, 19);
25. $dy := r1/6$ (24);
26. $y1 := y1 + dy$ (25);
27. $y[i] := y1$ (26, 8);
4. $i < n$ (27, 23);
28. печать (x, y) $(-4)\}$.

Здесь число 29 — номер параллельных точек входа. Данная программа имеет две параллельные точки входа: инструкции с номерами 1 и 2.

В программе вводятся значения переменных $x1$, $y1$, $h1$ и n, равные x_0, y_0, h и n условия задачи соответственно. Выводятся на печать два массива с именами x и y, содержащие по n компонент каждый. Эти массивы и представляют искомое решение дифференциального уравнения.

Для размещения в процессоро-памяти этой программы необходимо 29 ячеек. Обозначим эти ячейки $Я_1$, $Я_2$, ..., $Я_{29}$ и будем считать, что инструкции программы P заняли ячейки в том порядке, в котором они записаны, т. е. инструкция 1. ввод $(x1, y1, h1, n)$ (29) находится в ячейке $Я_1$, инструкция 2. $i := 0$ (29) — в ячейке $Я_2$, ..., инструкция 4. $i < n$ (27, 23) — в ячейке $Я_{28}$, и инструкция 28. печать (x, y) (-4) — в ячейке $Я_{29}$.

Динамика изменения состояний ячеек $Я_1$, ..., $Я_{29}$ в процессе реализации программы P отражена в таблице, в которой приведены их состояния в течение первой итерации и в начале второй. Приняты следующие обозначения состояний ячеек: Н — *не готова*, Н1 — *не готова 1*, Н2 — *не готова 2*, Н3 — *не готова 3*, Г — *готова*, В — *выполняюсь*.

Для улучшения наглядности таблицы, состояния ячеек указываются только в моменты их изменения. Для сокращения размеров таблицы, начиная с 21-го момента времени, в ней пропущены строки, соответствующие нечетным моментам времени, так как в эти моменты времени указываются только ячейки в состоянии *выполняюсь*. Пропущенные строки легко могут быть восстановлены при помощи графы «номера, поступившие на Пр-П», в которой указываются номера инструкций, выполнившихся в предыдущий момент времени.

№ ячейки Пр-П / Время	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	26	27	28	29	Номера, поступающие на Пр-П
1	Н	Н	Н	Н1	Н	Н	Н	Н	Н	Н	Н	Н1	Н	Н	Н	Н1	Н	Н1	Н	Н1	Н	Н1	Н1	Н1	Н	Н	Н1	Н1	Н	
2	Г	Г																												29
3	В	В																												
4	Н	Н	Г	Н2																										1,2
5			В																											
6			Н	Г																										3
7				В																										
8				Н1	Г	Г	Г	Г																						4
9					В		В																							
10					Н		Н		Г								Г					Н2								5,7
11						В			В																					
12						Н			Н	Г		Н3			Г					Н2									6,9
13										В							В												
14										Н	Г						Н	Н 3												10,17
15								В			В																			
16								Н			Н	Г			В							Г					Н3			8,11
17												В								Г	Г									
18												Н1	Г		Н	Н3				Н 1	Н									12,15
19													В									В								
20													Н	Г					Г			Н1	Н3							13,22
22														Н		Г			Н					Н3						14,19
24																Н 1		Г												16
26																														18
28																		Н 1					Г	Г						20,21
30																							Н1	Н1	Г					23,24
32																									Н	Г		Н3		25
34																										Н	Г			26
36																											Н 1	Г		27
38					Г	Г	Г	Г																				Н1		+4
40					Н			Н	Г													Н3					Н3			5,8
42							Н		Н	Г					Г		Г			Н2		Г								7,9

Таблица составлена из расчета максимальной загрузки двухпроцессорной вычислительной системы, причем выбор ячеек в состоянии *готова*, на которые вырабатывался сигнал *чтение*, осуществлялся произвольно. Предполагается также, что процессоро-память позволяет реализовать двумерный вычислительный процесс, и для выполнения любой команды (включая и команды ввода — вывода) необходимо одну единицу времени.

Как видно из таблицы, инструкции, находящиеся в ячейках $Я_1 \div Я_4$ выполняются однократно и только после запуска программы на счет; инструкции, находящиеся в ячейках $Я_5 \div Я_{28}$ выполняются в каждой очередной итерации (т. е. составляют тело цикла), причем инструкция управления, находящаяся в ячейке $Я_{28}$ определяет число повторений тела цикла; инструкция, находящаяся в ячейке $Я_{29}$, также выполняется один раз, но уже после завершения циклического процесса. Заметим, что после завершения выполнения программы все ячейки $Я_1 \div Я_{29}$ возвращаются в начальное состояние неготовности, которое они имели после записи программы Р в процессоро-память. Поэтому программа Р может быть вновь инициирована без перезаписи в Пр—П, например, с другими данными.

Таким образом, асинхронный язык типа $<i, k, j, l>$ совместно с процессоро-памятью, построенной на ячейках, описанных выше, позволяет программировать и реализовать циклические вычислительные процессы с интрациклическим распараллеливанием.

Список литературы: 1. *Колубай С. К.* Программирование на МК-языке типа $<i, k, j, l>$.— Проблемы бионики. 1979, вып. 23, с. 66—74. 2. *Колубай С. К.* Параллельная вычислительная система для МК-программ.— Проблемы бионики, 1979, вып. 23, с. 85—92.

Поступила 20 декабря 1979 г.