

Государственный комитет Совета Министров СССР  
по делам строительства

Центральный научно-исследовательский и проектно-  
экспериментальный институт автоматизированных  
систем в строительстве  
ЦНИПИАСС

УДК: 681.322.06:721.011:69.003:65.014.011.56

№ гос. регистрации

Изм. №

"Утверждено"

Директор ЦНИПИАСС

Г.Т.Н., доцент

М.А. Гусakov

3003 1977 г.

Технический проект экспериментальной  
системы пакетов прикладных программ  
автоматизированного проектирования  
систем организационного управления  
/логико-интерпретационный блок про-  
ектирования/

Всего томов 3. Том 3, книга I

Шифр 15-2-76

И.о. зам. руководителя отделения АСУС *И.М. Лившиц* И.М. Лившиц

Руководитель темы,  
зав. отделом ИО АСУС

*С.П. Никаноров*

С.П. Никаноров

Ответственный исполнитель,  
с.н.с., к.ф.-м.н.

*Д.Б. Персиц*

Д.Б. Персиц

Москва - 1977 г.

№

АКАДЕМИЯ НАУК УССР  
 ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР ОДЕССКОГО ОТДЕЛЕНИЯ  
 ИНСТИТУТА ЭКОНОМИКИ

ЭКСПЕРИМЕНТАЛЬНАЯ СИСТЕМА ППП АВТОМАТИЗИРОВАННОГО  
 ПРОЕКТИРОВАНИЯ СИСТЕМ ОРГАНИЗАЦИОННОГО УПРАВЛЕНИЯ

Д 4. СОДЕРЖАНИЕ ЗАДАЧ ПРОЕКТИРОВАНИЯ  
 И АЛГОРИТМЫ ИХ РЕШЕНИЯ

на 114 листах

"Согласовано"

Руководитель темы  
С.П. Никаноров С.П. НИКАНОРОВ

Ст. научный сотрудник к.ф.-м.н.  
А.Б. Персиц А.Б. ПЕРСИЦ

Ответственный исполнитель  
 Ст. инженер

А.В. Айзентат А.В. АЙЗЕНТАТ

Ст. инженер

Б.А. Закс Б.А. ЗАКС

Зав. отделом моделирования  
 и системотехники

Г.Я. Портнов Г.Я. ПОРТНОВ

15.2.76

ТП 73, кч 1

## ОГЛАВЛЕНИЕ

	Стр
Д.4. Содержание задач проектирования и алгоритмы их решения .....	3
1. Формирование главного рода структуры (ППП-ГРС)....	4
1.1. Постановка задачи (содержание и метод решения)....	4
1.2. Алгоритм управления функционированием ППП (УПП)....	9
1.3. Описание алгоритмов модулей тела пакета.....	15
1.4. Характеристики информационных массивов ППП и связь с другими ППП .....	50
1.5. Ограничения и возможности расширения ППП .....	59
1.6. Средства программного контроля и контроля результатов .....	60
2. R-интерпретация (ППП R-интерпретации) .....	61
2.1. Постановка задачи (содержание и метод решения) ...	61
2.2. Алгоритм управления функционированием ППП (УПП)....	65
2.3. Описание алгоритмов модулей тела пакета .....	77
2.4. Характеристики информационных массивов ППП и связь с другими ППП .....	103
2.5. Ограничения и возможности расширения ППП .....	112
2.6. Средства программного контроля и контроля результатов .....	115

# I. Формирование главного рода структуры (ППШ ГРС)

## I.I. Постановка задачи (содержание и метод решения).

Метод автоматизированного проектирования систем организационного управления включает в качестве одной из своих главных компонент математический аппарат, позволяющий строить сложные определения систем (главные роды структур).

Основными понятиями этого аппарата являются конститутанты, род структуры, дополнение и отождествляющее отображение. Различают следующие имена конститутант:

- $X_i$  -  $i$ -тое основное базисное множество;
- $C_i$  -  $i$ -тое вспомогательное базисное множество;
- $D$  - родовая структура;
- $M$  - тип родовой структуры;
- $\Pi_i$  -  $i$ -тый терм;
- $K_i$  - тип  $i$ -того термина;
- $A_i$  -  $i$ -тая аксиома;
- $T_i$  -  $i$ -тая теорема.

Конститутанты с именами  $M$ ,  $\Pi_i$ ,  $K_i$ ,  $A_i$ ,  $T_i$  имеют выражения, обозначаемые соответственно  $\bar{M}$ ,  $\bar{\Pi}_i$ ,  $\bar{K}_i$ ,  $\bar{A}_i$ ,  $\bar{T}_i$  и являющиеся некоторыми формулами, выражающими эти конститутанты классов  $X$ ,  $C$ ,  $D$ ,  $\Pi$ .

Под родом структуры  $\Phi$  I типа понимается совокупность: кортежа основных базисных множеств  $\{X_i\}_{i=1, \overline{N_x}}$ , кортежа вспомогательных базисных множеств  $\{C_i\}_{i=1, \overline{N_c}}$ , родовой структуры  $D$ , типа родовой структуры  $M$ , набора термов  $\{\Pi_i\}_{i=1, \overline{N_\pi}}$ , соответствующего набора типов термов  $\{K_i\}_{i=1, \overline{N_k}}$ , множества аксиом  $\{A_i\}_{i=1, \overline{N_a}}$  и множества теорем  $\{T_i\}_{i=1, \overline{N_t}}$ , удовлетворяющая следующим условиям:

1) выражения для конститутант классов  $M$  и  $K$  есть ступени над  $X_1, \dots, X_{N_x}; C_1, \dots, C_{N_c}$  (понятие ступени определяется рекурсивно:

- а)  $X_p$  и  $C_q$  - ступени;
- б) если  $S_1, S_2, \dots, S_n$  - ступени, то  $B(S)$  и  $P(S_1, \dots, S_n)$  - ступени;

2) если  $\Pi_j \in \{\bar{\Pi}_i\}$  (имя  $\Pi_j$  входит в выражение конститутанты  $\Pi_i$ ), то  $j < i$ .

15-2-76

ТПТЗ01

Род структуры II типа отличается от рода структуры I типа тем, что у него отсутствуют конститuentы  $A$  и  $M$ .

Аналогично определяется дополнение.

Дополнением  $E$  называется совокупность:

кортежа основных базисных множеств  $\{x_i\}_{i=1, \overline{N_A}}$ , набора термов  $\{P_i\}_{i=1, \overline{N_P}}$ , соответствующего набора типов термов  $\{K_i\}_{i=1, \overline{N_K}}$ , множества аксиом  $\{A_i\}_{i=1, \overline{N_A}}$  и множества теорем  $\{T_i\}_{i=1, \overline{N_T}}$ , причем выражения для конститuent класса  $K$  являются квази-степенями над  $x_1, \dots, x_{N_A}$  (понятие квазиступени определяется рекурсивно:

- $x_p$  - квазиступень;
- если  $S = S_1, \dots, S_2$  - квазиступени, то  $B(S), B^{-1}(S), P_{2_i}(S) = P(S_1, \dots, S_2)$  - квазиступени).

Отождествляющим отображением  $\Gamma$  называется конечная последовательность ~~и~~ конститuent классов  $x, c, A, \Pi$ .

Для получения главного рода структур применяются специальные операции над родами структур, дополнениями и отождествляющими отображениями. Такими операциями являются операции: свободного произведения ( $OP_1$ ), прямого произведения ( $OP_2$ ), усиления ( $OP_3$ ), родового усиления ( $OP_{3,1}$ ), простого расширения базы ( $OP_5$ ), общего расширения базы ( $OP_7$ ), смешанного произведения ( $OP_9$ ), терм-вложения ( $OP_{11,1}$ ), конкретизирующего вложения ( $OP_{11}$ ), булеанизации ( $OP_{15}$ ).

Кратко охарактеризуем некоторые из этих операций. (Более полное представление об операциях можно получить из описаний алгоритмов соответствующих модулей).

Операция свободного произведения  $\Phi = OP_1(\Phi_1, \Phi_2)$

Кортеж основных базисных множеств  $\{x_i\}_{i=1, \overline{N_\Phi}}$  результата содержит  $N_\Phi = N_1^1 + N_2^2$  конститuent, причем первые  $N_1^1$  конститuent соответствует основным базисным множествам I аргумента, а остальные - основным базисным множествам II аргумента.

Аналогично определяется кортеж вспомогательных базисных множеств  $\{c_i\}_{i=1, \overline{N_c}}$ ,  $N_c = N_c^1 + N_c^2$ .

Выражение конститuent  $M$  находится по разному в зависимости от типов  $\Phi_1$  и  $\Phi_2$ :

$$\bar{M} = \begin{cases} P(M_1^1, M_2^2) & , \text{ если } \Phi_1 \text{ и } \Phi_2 - \text{ I типа} \\ \frac{M_1^1}{M_2^2} & , \text{ если } \Phi_1 - \text{ I типа, а } \Phi_2 - \text{ II типа} \\ \frac{M_2^2}{M_1^1} & , \text{ если } \Phi_1 - \text{ II типа, а } \Phi_2 - \text{ I типа} \end{cases}$$

где  $\bar{M}^2$  - выражение  $\bar{M}_2$  после замен  $x_i$  на  $x_{i+N_x^1}$  и  $c_i$  на  $c_{i+N_c^1}$ .

Число термов результата находится по формуле

$$N_n = \begin{cases} N_n^1 + N_n^2 + 2, & \text{если } \Phi_1 \text{ и } \Phi_2 - \text{I типа} \\ N_n^1 + N_n^2 & \text{в остальных случаях.} \end{cases}$$

Множества аксиом (а также теорем) аргументов после соответствующих замен имен конституэнт объединяются  $N_A = N_A^1 + N_A^2$ ,  $N_T = N_T^1 + N_T^2$ .

Операция прямого произведения  $\Phi = OP_2(\Phi_1, \Phi_2)$

Выполняется лишь при условии  $N_x^1 = N_x^2$ ,  $N_c^1 = N_c^2$ .

Отличается от операции свободного произведения тем, что здесь  $N_x = N_x^1 = N_x^2$ ,  $N_c = N_c^1 = N_c^2$  и конституэнты  $x_i$  и  $c_j$  II аргумента переходят в конституэнты результата с теми же номерами  $x_i$  и  $c_j$ .

Операция смешанного произведения  $\Phi = OP_9(\Phi_1, \Phi_2, \Gamma_1, \Gamma_2)$

Операция смешанного произведения является обобщением двух предыдущих операций. Здесь отождествляющие отображения  $\Gamma_i$  показывают, в какие конституэнты результата переходят конституэнты классов  $x$  и  $c$  родов структур  $\Phi_i$  ( $i = 1, 2$ ).

Операция усиления  $\Phi = OP_3(\Phi_1, E, \Gamma)$

При выполнении операции усиления к наборам термов, типов термов, множествам аксиом и теорем I аргумента добавляются термы, типы термов, аксиомы и теоремы, полученные из соответствующих конституэнт дополнения  $E$  заменой имен конституэнт  $x_i$  на имена, определяемые отображением  $\Gamma$ .

Операция родового усиления.  $\Phi = OP_{3,1}(\Phi_1, \Phi_2, \Gamma)$  отличается от предыдущей операции тем, что вторым аргументом является род структуры и вследствие этого отображение  $\Gamma$  должно определять замену конституэнт не только класса  $x$ , но и классов  $C$  и  $D$ .

Операция простого расширения базы  $\Phi = OP_5(\Phi_1, n, m)$

расширяет основное и вспомогательное множества рода структуры:

$$\begin{aligned} N_x &= n \geq N_x^1 \\ N_c &= m \geq N_c^1 \end{aligned}$$

При этом  $x_i$  аргумента переходит в  $x_i$  результата, и  $c_j$  аргумента переходит в  $c_j$  результата.

Операция общего расширения базы  $\Phi = OP_7(\Phi_1, \Gamma, n, m)$  является обобщением операции  $OP_5$  и отличается от нее тем, что перевод конституэнт  $x_i$  в  $c_j$  аргумента  $\Phi_1$  в конституэнты результата происходит не стандартно, а с помощью отождествляющего отображения.

В задаче формирования главного рода структур можно выделить следующие функции:

1. Задавание операционной схемы.

Для получения главного рода структур проектировщик должен задать так называемую операционную схему, состоящую из ориентированного графа (не имеющего циклов и петель, с единственной конечной вершиной), а также операционного заполнения - отображения множества вершин графа, отличных от начальных, во множество операций.

2. Формирование базового замыкания.

Сущность функции состоит в заполнении начальных вершин графа базовыми родами структур, дополнениями, отождествляющими отображениями. Это заполнение происходит с помощью задания отождествляющего отображения  $\eta$  базовых родов структур, дополнений, отображений во множество начальных вершин графа операционной схемы.

3. Получение промежуточного (или главного) рода структуры.

Для реализации этой функции проектировщик задает номер вершины, соответствующий искомому роду структуры. После этого происходит выделение нижнего замыкания этой вершины, т.е. выделение подграфа операционной схемы, все вершины которого, и только они, предшествуют данной. А затем производятся операции в соответствии с выделенным подграфом до тех пор, пока не будет получен искомый род структуры.

4. Контроль процесса формирования главного рода структуры.

Контроль осуществляется на всех этапах формирования главного рода структуры. При вводе операционной схемы проверяется отсутствие циклов и петель и единственность конечной вершины в операционной схеме.

При вводе базового замыкания проверяется соответствие типов аргументов операциям в операционной схеме. При формировании промежуточного (главного) рода структуры делаются про-

15-2-16  
ТПЗм1

верка на возможность выполнения каждой операции и т.д.

15-2-76  
ТПЗст

### 1.2. Алгоритм управления функционированием ППП.

Управление функционированием ППП ГРС производит управляющая программа пакета (УПП), состоящая из модуля автоматической R-интерпретации и модуля организации процесса формирования ГРС (МОП-ГРС).

В данном разделе описывается работа УПП по управлению процессом формирования ГРС (п.1.2.1.), приводится алгоритм модуля организации процесса формирования ГРС (п.1.2.2.), а также конкретизируется для данного случая описание блок-схемы модуля автоматической R-интерпретации (модуля ППП R-интерпретации), приведенной в п.2.2.3. (п.1.2.3.).

#### 1.2.1. Описание процесса управления формированием ГРС.

Работа ППП ГРС начинается с ввода в ЭВМ задания на работу системы ППП, содержащего управляющий оператор получения главного (промежуточного) рода структуры. ГУП расшифровывает содержание управляющего оператора и передает УПП ГРС (точнее говоря, МОП-ГРС) номер вершины, соответствующий искомому роду структуры.

После этого МОП-ГРС формирует в соответствующем представлении граф, приведенный на Рис.1.2.1.

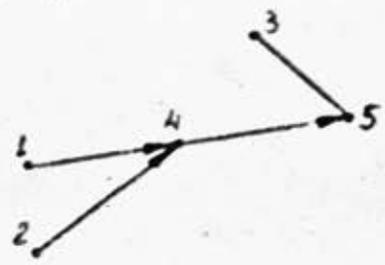


Рис. 1.2.1.

где начальным вершинам соответствуют:

- 1 - массив ОПСХ;
- 2 - массив КОРС;
- 3 - номер вершины операционной схемы, указанный в управляющем операторе,

в вершинах 4 и 5 приписаны следующие модули:

- 4 - модуль удаления

75-2-76

ТПГЗм1

подграфа\*);

5 - модуль формирования нижнего замыкания\*).

Одновременно заполняются каталоги:

W - каталог адресов информации, соответствующей начальным вершинам графа, приведенного выше на Рис. I.2.1.;

V - каталог значений начальных вершин графа (отмечаются вершины 1, 2, 3).

Затем управление передается модулю автоматической R-интерпретации, который сначала удаляет из массива ОПСХ все записи, отмеченные в КОПС, как выполненные, а затем берет нижнее замыкание вершины графа операционной схемы, указанной в управляющем операторе задания. В результате работы модуля получается ВОПСХ (временная операционная схема). Отметим, что массив ОПСХ при этом сохраняется в первоначальном виде.

После окончания работы модуля автоматической R-интерпретации МОП-ГРС вновь подготавливает работу этого модуля, но уже с другим графом - графом временной операционной схемы (массив ВОПСХ). В этом случае в каталог V входят массивы РСД и ОТСТ, а в качестве каталога W вершин графа массива ВОПСХ, полученных интерпретаций, выступает массив КОПС.

Затем управление вновь передается модулю автоматической R-интерпретации, который выполняет все операции над родами структур, для которых аргументы получили значения, и заносит результаты этих операций в массив РСД.

После окончания работы модуля автоматической R-интерпретации МОП-ГРС выдает на печать список начальных вершин массива ВОПСХ, не получивших значений (если такие есть), либо сообщает о том, что требуемый род структуры получен.

### I.2.2. Описание модуля организации процесса формирования ГРС (МОП-ГРС).

Модуль предназначен для настройки модуля автоматической R-интерпретации на обработку операционной схемы.

Исходные данные:

- массив ОПСХ;
- массив КОПС;

\* ) Описание модуля приведено в п. 2.3.

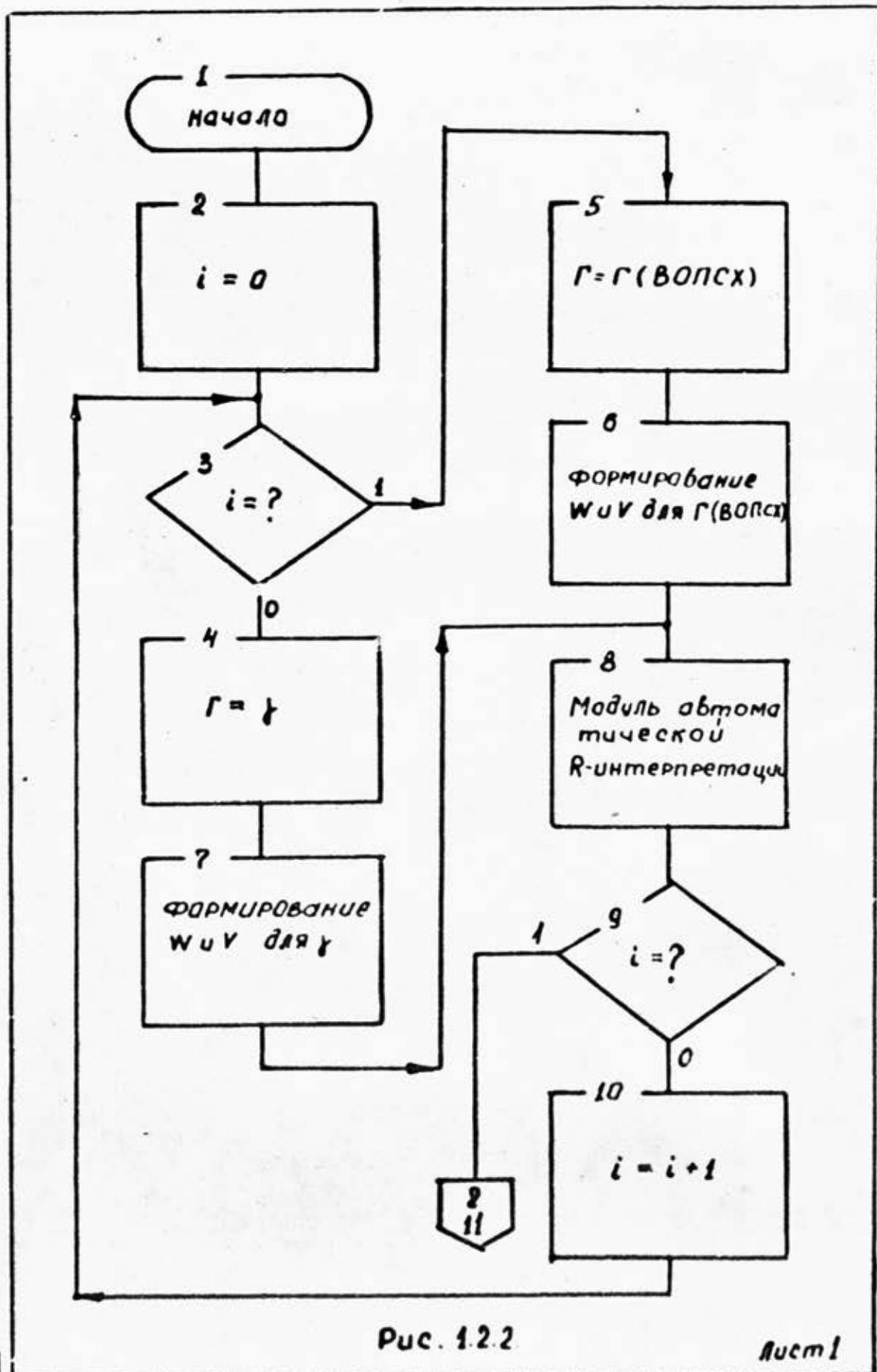


Рис. 12.2

Лист 1

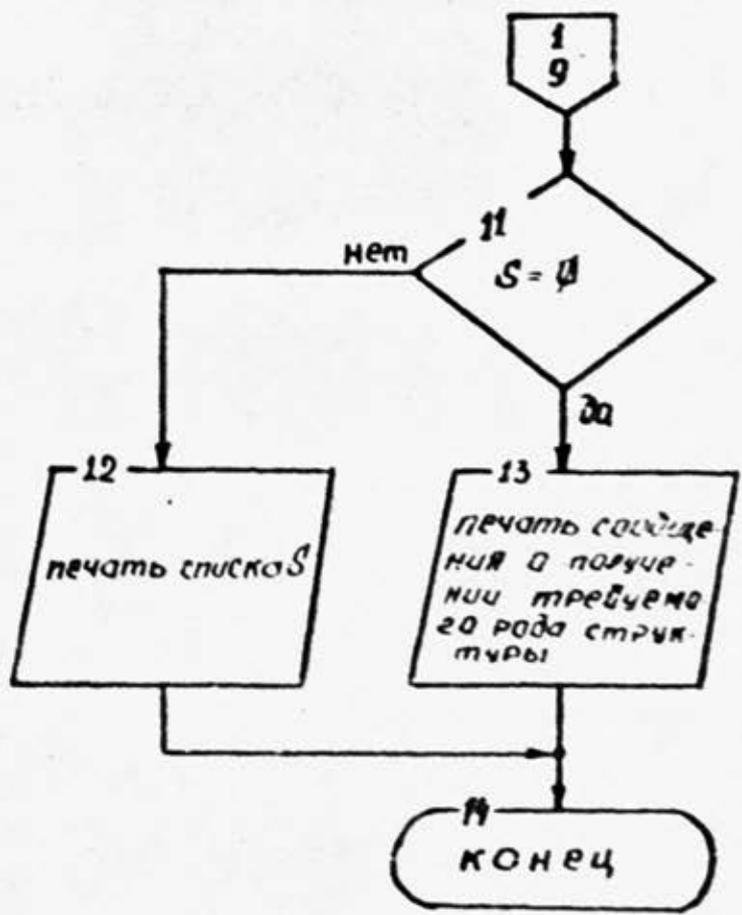


Рис 1.2.2.

- номер вершины операционной схемы, указанные в управляющем операторе задания;

- идентификаторы библиотек модулей ППП ГРС и ППП R-интерпретации.

Блок-схема модуля приведена на Рис. 1.2.2. В блок-схеме используются следующие обозначения:

G - граф, обрабатываемый модулем автоматической R-интерпретации;

ВОПСХ - массив "временная операционная схема", полученный в результате I-ого применения модуля автоматической R-интерпретации;

G(ВОПСХ) - граф, соответствующий массиву ВОПСХ;

$\zeta$  - граф, приведенный на Рис. 1.2.1.;

W - каталог адресов значения конститuent, получивших интерпретацию\*;

V - каталог значений конститuent, получивших интерпретацию\*;

S - множество начальных вершин графа G, не получивших интерпретацию.

### 1.2.3. Особенности работы модуля автоматической R-интерпретации при обработке массива ВОПСХ.

Полное описание работы модуля автоматической R-интерпретации приведено в п.2.2.3. данного документа. Здесь же будут указаны особенности работы модуля автоматической R-интерпретации при обработке массива ВОПСХ.

Отметим прежде всего, что при обработке массива ВОПСХ модулем автоматической R-интерпретации множества SJNT и No всегда пусты, т.е. вычисления не проходят по путям бл.10 → бл.15 и бл.11 → бл.16.

В бл.7 происходит переход к следующей записи массива ВОПСХ.

Бл.8 анализирует, все ли аргументы операции определены.

Бл.9 соответствует поиску аргументов операций в массивах РСД и СТОН.

Бл.12 осуществляет вызов модуля, выполняющего операцию.

\* каталоги W и V используются модулем автоматической R-интерпретации. Подробнее о них см. в п.2.2.1-2.2.3.

указанную в обрабатываемой записи.

В бл.13 происходит отметка выполненности операции в массиве КОПС, а в бл.14 - запись результата в массив РСД.

75-2-76

Т.П.ч.3 д.1

### Г.3. Описание алгоритмов модулей тела пакета.

В тело пакета входят следующие модули:

- модуль, выполняющий операции свободного, прямого и смешанного произведения;
- модуль, выполняющий операции усиления и родового усиления;
- модуль, выполняющий операции простого и общего расширения базы;
- модуль, выполняющий операцию терм-вложения;
- модуль, выполняющий операцию конкретизирующего вложения;
- модуль, выполняющий операцию булеализации.

Введем некоторые обозначения, общие для блок-схем всех модулей тела пакета:

Имя - имя рода структуры - результата операций;

$N_x, N_c, N_d, N_p, N_a, N_t$  - числа конститuent класса  $x, c, d, p, a, t$  результата операции;

$N_x^i, N_c^i, N_d^i, N_a^i, N_p^i, N_t^i$  - числа конститuent класса  $x, c, d, p, a, t$   $i$ -того аргумента операции;

$V(B_i)$  - элемент  $B$  записи массива "Роды структур", соответствующий результату ( $i$ -тому аргументу) операции<sup>\*)</sup>;

$\psi_i$  - отождествляющее отображение конститuent  $i$ -того аргумента в конститuent результата<sup>\*\*)</sup>;

$F_{\psi_i}$  (выражение) - оператор замены конститuent класса  $x, c, d, p$  в выражении в соответствии с отображением  $\psi_i$ ;

$F_{\psi_i}$  (выражение,  $x, c$ ) - модификация предыдущего оператора. Замена подвергаются лишь конститuent указанных классов (в данном случае  $x$  и  $c$ ).

$wir(B)$  - оператор расщепления элемента  $B$ , выделяет из  $B$  элементы  $\bar{m}, \bar{n}, \bar{p}, \bar{a}, \bar{t}$ ;

\*) Структура записи массива "роды структур" описана в п.Г.4 данного документа.

\*\*) Структура записи массива "Т-интерпретация" описана в п.Г.4 данного документа.

$S(\bar{M}, \bar{K}, \bar{P}, \bar{A}, \bar{T})$  - оператор сборки элемента  $B$  из элементов  $\bar{M}, \bar{K}, \bar{P}, \bar{A}, \bar{T}$  ;  
 $\bar{M}_i, \bar{K}_i, \bar{P}_i, \bar{A}_i, \bar{T}_i$  - подэлементы  $\bar{M}, \bar{K}, \bar{P}, \bar{A}, \bar{T}$  элемента  $B_i$ .

$$n_x = N_x ;$$

$$n_c = N_x + N_c ;$$

$$n_d = N_x + N_c + N_d ;$$

$$n_m = N_x + N_c + 2N_d ;$$

$$n_n = N_x + N_c + 2N_d + N_p ;$$

$$n_k = N_x + N_c + 2N_d + 2N_p ;$$

$$n_a = N_x + N_c + 2N_d + 2N_p + N_a ;$$

$$n_t = N_x + N_c + 2N_d + 2N_p + N_a + N_t .$$

### 1.3.1. Модуль, выполняющий операции свободного, прямого и смешанного произведения.

Исходными данными для работы модуля являются:

- код операции ;
- I аргумент ( $\Phi_1$ ) - произвольный род структуры ;
- II аргумент ( $\Phi_2$ ) - произвольный род структуры.

Блок-схема модуля представлена на рис. 1.3.1.

В блок-схеме через  $\beta$  обозначен признак, служащий для разделения случаев

$$a) (N_d^1 = 0) \wedge (N_d^2 = 0)$$

$$b) (N_d^1 = 1) \vee (N_d^2 = 1)$$

В блок-схеме модуля приведены следующие блоки:

#### Бл. I. Контроль аргументов операции.

При выполнении операции прямого произведения ( $OP_2$ ) проверяются равенства

$$N_x^1 = N_x^2, N_c^1 = N_c^2$$

При выполнении операции смешанного произведения проверяются следующие соотношения:

- a)  $N(\Gamma_i) = N_x^i + N_c^i, (i=1,2)$  - где  $N(\Gamma_i)$  - число элементов в отождествляющем отображении  $\Gamma_i$  - ( $i+2$ )-ом аргументе

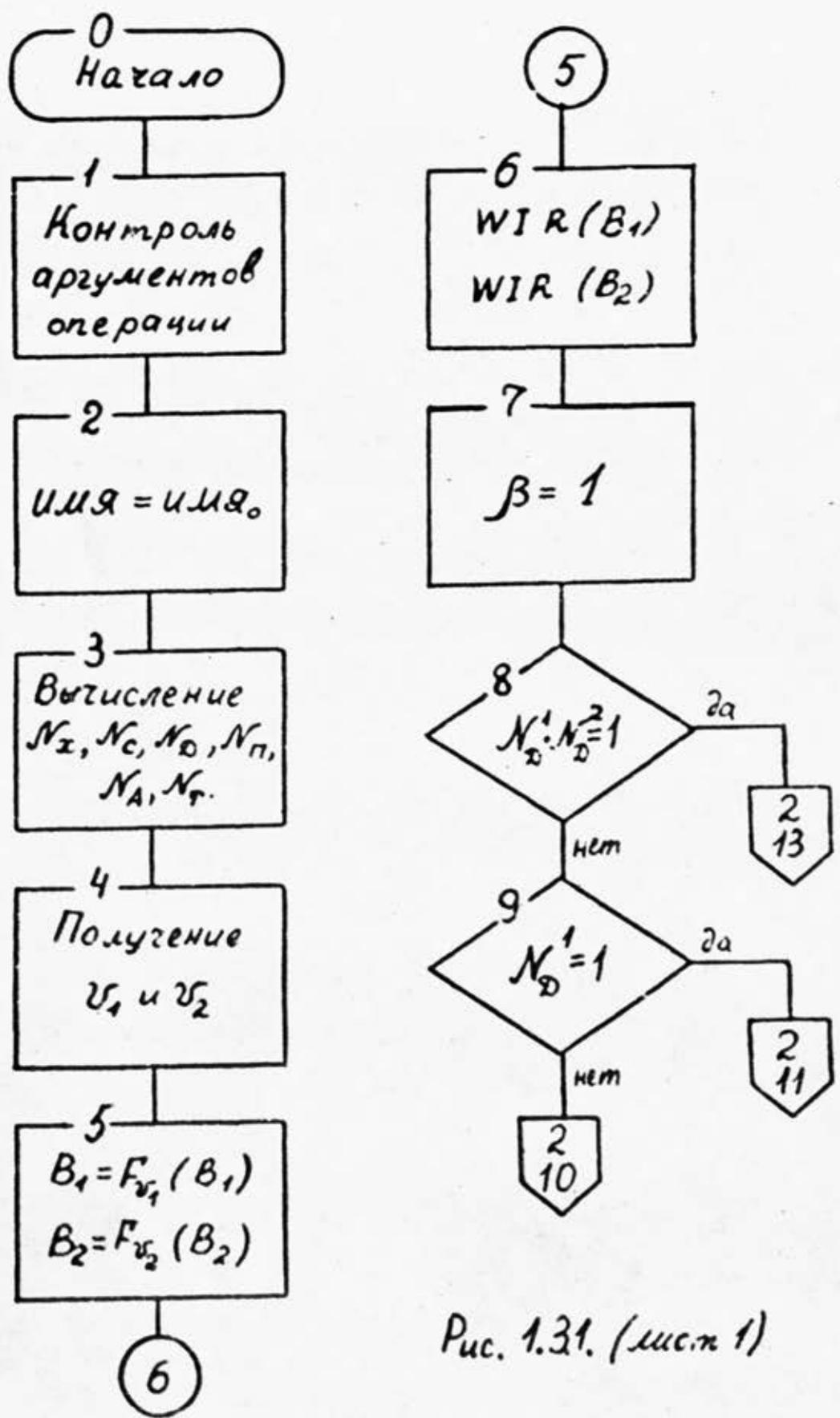


Рис. 1.31. (лист 1)

15.2.86  
ТПрЗмЛ

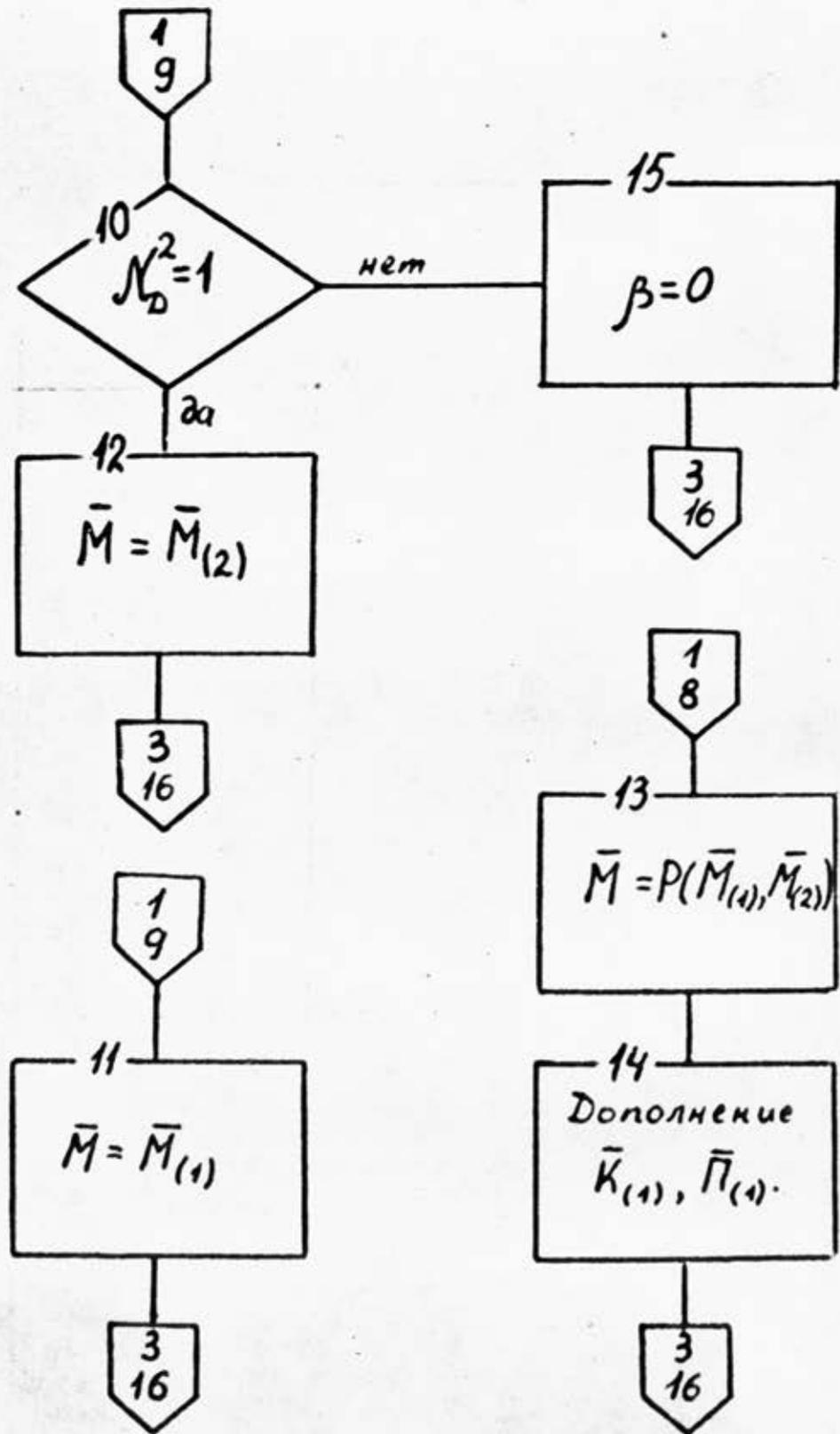


Рис 1.3.1. (лист 2)

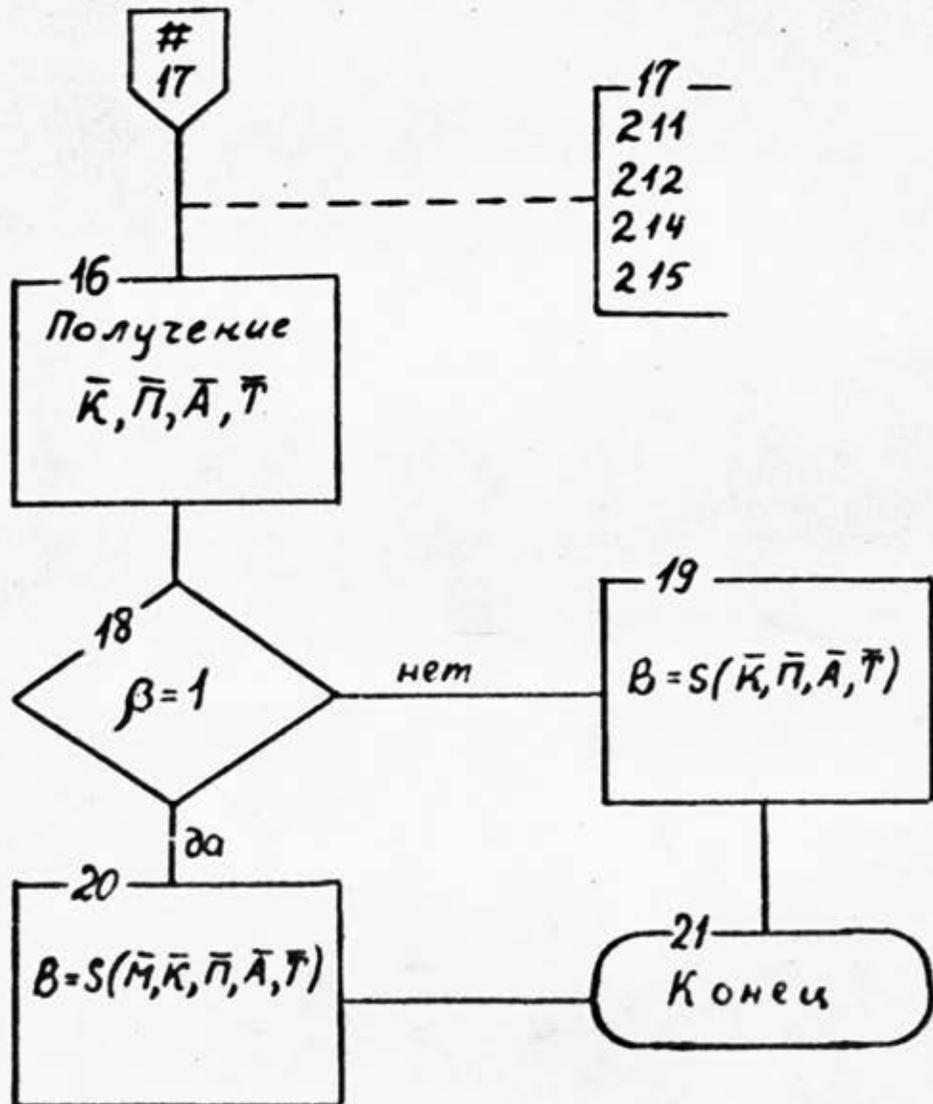


Рис. 1.3.1. (лист 3)



$$v_i^c = \left\{ \left( \binom{n_x^i + 1}{n_x + 1 + d_i \cdot j N_c^i} \right), \left( \binom{n_c^i}{n_x + N_c^i + d_i \cdot j N_c^i} \right) \right\}$$

(формируется в соответствии с  $\Gamma_i$  в случае  $OP_9$ )

$$v_i^A = \left\{ \left( \binom{n_A^i}{n_A + N_A^i \cdot N_A^i (1+i)} \right) \right\}, \text{ если } N_A^i = 1.$$

$$v_i^M = \left\{ \left( \binom{n_M^i}{n_M + N_A^i \cdot N_A^i (N_M + i)} \right) \right\}, \text{ если } N_A^i = 1.$$

$$v_i^n = \left\{ \left( \binom{n_M^i + 1}{n_M + 1 + 2N_A^i N_A^i + d_i N_M^i} \right), \left( \binom{n_M^i}{n_M + N_M^i + 2N_A^i N_A^i + d_i N_M^i} \right) \right\}$$

$$v_i^k = \left\{ \left( \binom{n_M^i + 1}{n_M + 1 + 2N_A^i N_A^i + d_i N_M^i} \right), \left( \binom{n_M^i}{n_M \cdot N_M^i + 2N_A^i N_A^i + d_i N_M^i} \right) \right\}$$

$$v_i^A = \left\{ \left( \binom{n_K^i + 1}{n_K + 1 + d_i \cdot N_A^i} \right), \left( \binom{n_A^i}{n_K + N_A^i + d_i \cdot N_A^i} \right) \right\}$$

$$v_i^T = \left\{ \left( \binom{n_K^i + 1}{n_A + 1 + d_i \cdot N_T^i} \right), \left( \binom{n_T^i}{n_A + N_T^i + d_i \cdot N_T^i} \right) \right\}$$

$i = 1, 2.$

Замечания. 1. Если какое-либо из чисел  $N_x^i, N_c^i, N_M^i, N_A^i, N_T^i$  равно 1, то соответствующая последовательность  $v_i$  ( $v_i^x$  или  $v_i^c$  или  $v_i^M$  или  $v_i^k$  или  $v_i^A$  или  $v_i^T$ ) состоит только из первой пары.

2. Если какое-либо из чисел  $N_c^i, N_A^i, N_M^i, N_A^i, N_T^i$  равно 0, то соответствующая подпоследовательность  $v_i$  отсутствует.

Бл.5. Блок 5 соответствует работе подпрограммы редактирования выражений (функция  $F_5(B)$ ).

Исходные данные для подпрограммы - выражение  $\delta_i$ , отождествляющее отображение  $\sigma_i$ , а также числа  $N_x^i, N_c^i, N_M^i, N_A^i, N_T^i, N_x^i, N_c^i, N_A^i, N_M^i, N_A^i, N_T^i$ , которые служат для

15-2-76

ТПТЗм1

расшифровки  $\nu_i$ . В результате работы подпрограммы в выражениях  $B_i$  происходит замена конституэнт в соответствии с отождествляющими отображениями  $\sigma_i$  ( $i = 1, 2$ ).

Бл.6. Получение  $WIR(B_1)$  и  $WIR(B_2)$

Функция  $WIR$  выделяет из элемента  $B$  записи рода структуры элемента  $\bar{M}, \bar{K}, \bar{P}, \bar{A}, \bar{T}$ .

Бл.12,13,14. Присвоение значения элементу  $\bar{M}$  результата операции.

В блоке II  $\bar{M}$  получается как сцепление

$$'P('x \bar{M}_{(1)} x', 'x \bar{M}_{(2)} x')$$

Бл.14. Дополнение элементов  $\bar{K}_{(1)}, \bar{P}_{(1)}$ .

$$\bar{K}_{(1)} = \bar{M}_{(1)} x' z' x \bar{M}_{(2)} x' z' x \bar{K}_{(1)}$$

$$\bar{P}_{(1)} = 'Pz_1 A' x' z' x' Pz_2 A' x' z' x \bar{P}_{(1)}$$

Бл.16. Получение элементов  $\bar{K}, \bar{P}, \bar{A}, \bar{T}$  результата операции.

$$\bar{K} = \bar{K}_{(1)} x' z' x \bar{K}_{(2)}$$

$$\bar{P} = \bar{P}_{(1)} x' z' x \bar{P}_{(2)}$$

$$\bar{A} = \bar{A}_{(1)} x' z' x \bar{A}_{(2)}$$

$$\bar{T} = \bar{T}_{(1)} x' z' x \bar{T}_{(2)}$$

Бл.19,20. Получение элемента  $B$  результата операции.

С помощью специальной подпрограммы (функция  $S$ ) осуществляется сцепление элементов  $\bar{M}$  (если  $\bar{M}$  присутствует),  $\bar{K}, \bar{P}, \bar{A}, \bar{T}$  в элемент  $B$ .

Результатом работы модуля является запись рода структуры, а также запись вспомогательного массива для Т-интерпретации.

### 1.3.2. Модуль, выполняющий операции усиления и родового усиления.

Исходными данными для работы модуля являются:

- код операции;
- I аргумент операции ( $\varphi_1$ ) - род структуры;
- II аргумент для операции усиления ( $E$ ) - дополнение; для операции родового усиления ( $\varphi_2$ ) - род структуры;
- III аргумент операции ( $\Gamma$ ) - отождествляющее отображение.

Блок-схема модуля представлена на рис. 1.3.2.

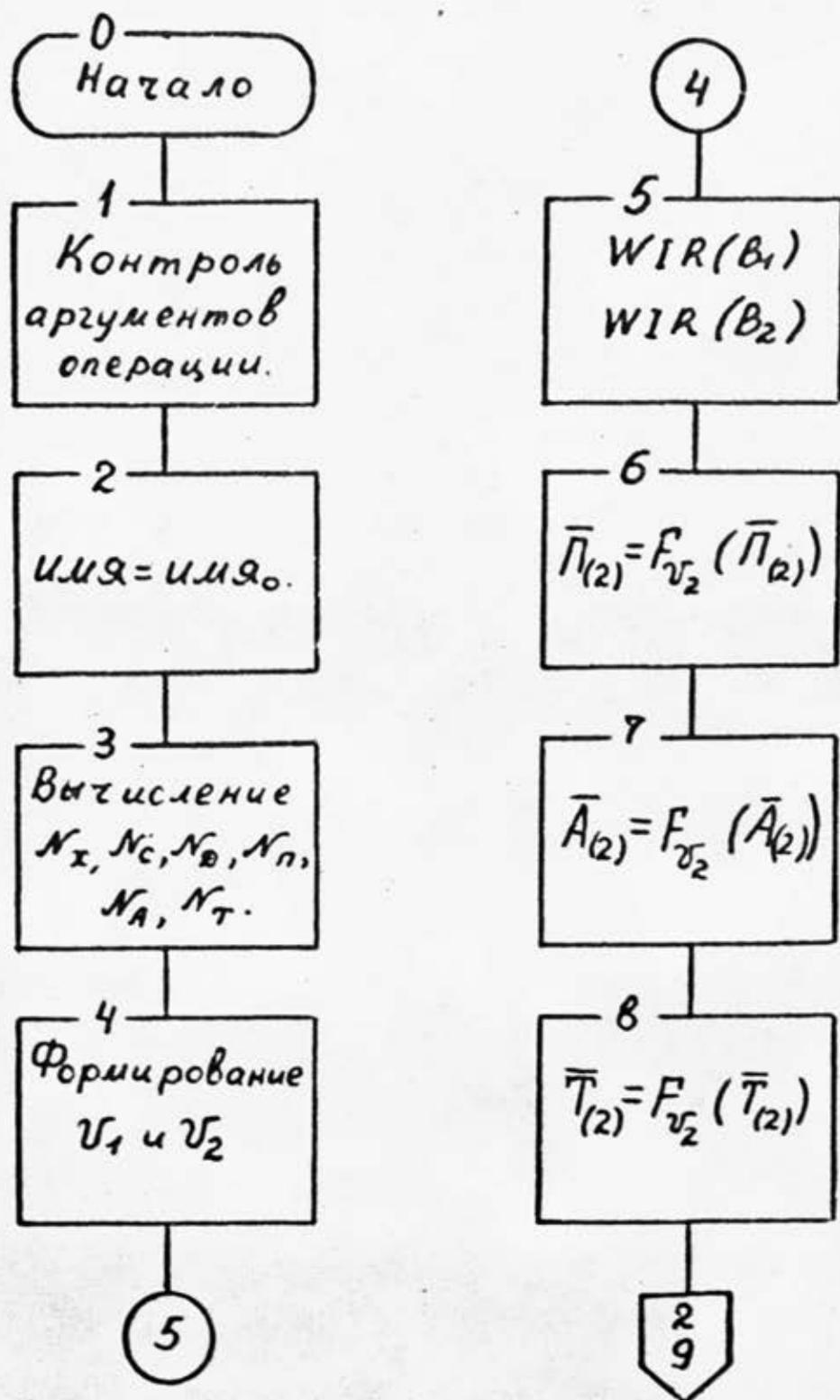


Рис. 1.3.2 (лист 1)

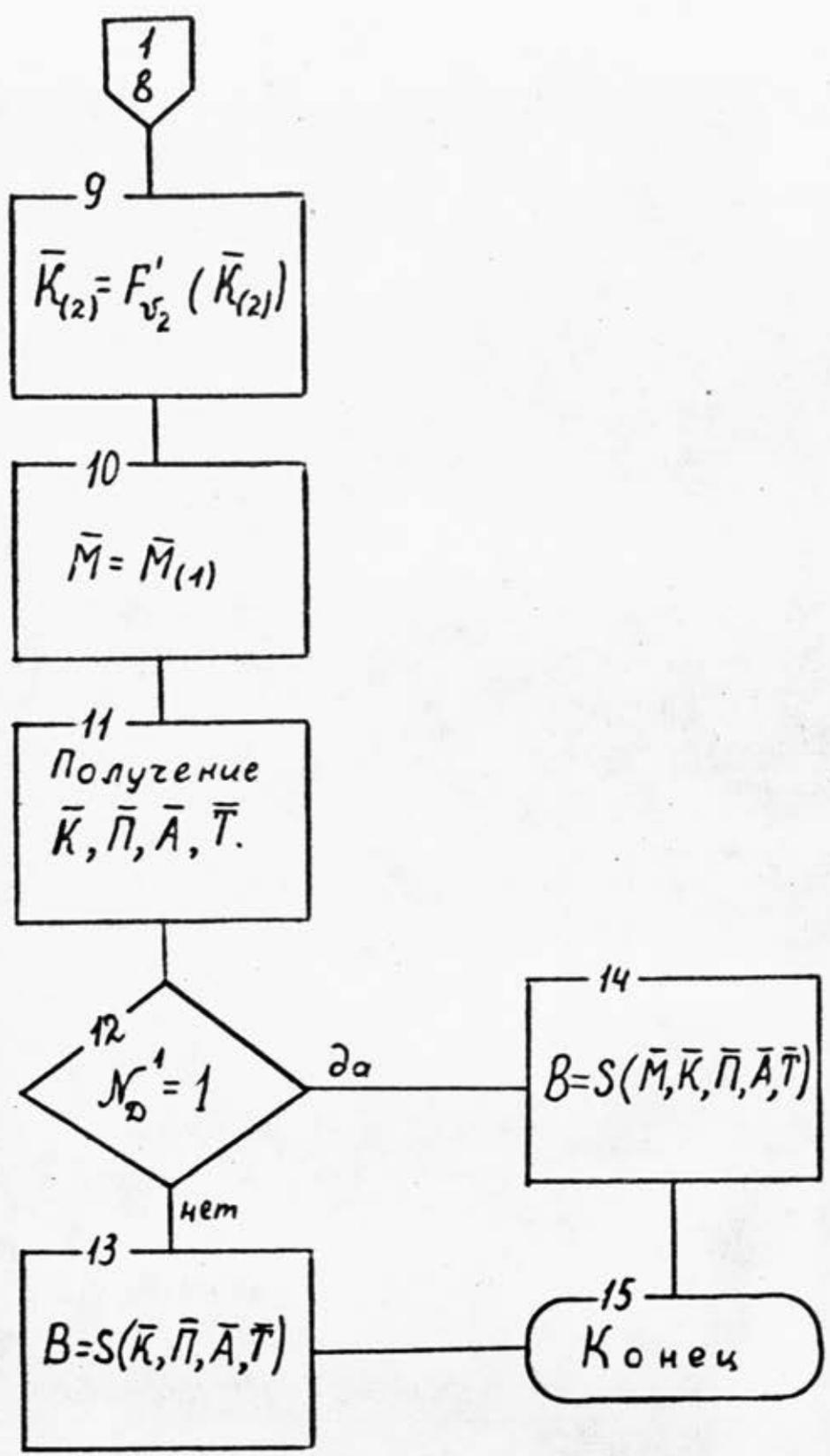


Рис. 1.3.2 (лист 2)

15-2-76  
ТП-301

В блок-схеме модуля приведены следующие блоки.

Бл.1. Контроль аргументов операции происходит путем проверки следующих соотношений:

1) число элементов в отождествляющем отображении  $\Gamma$

$$N(\Gamma) = N_x^2 + N_C^2 + N_D^2$$

2) если  $\Gamma(\rho)$  - конститuentы класса

- X, то ее номер  $L \leq N_x^2$

- C, то ее номер  $\bar{L} \leq N_C^2$

- D, то  $\bar{M} = B(\dots)$

- П, то ее номер  $\bar{L} \leq N_D^2$  и  $\bar{K}_L = B(\dots)$

3) отображение  $\Gamma$  не может определять соответствие

$$C_p \rightarrow X, \gamma$$

При невыполнении указанных соотношений должно даваться сообщение об ошибке.

Бл.2. Результату операции присваивается имя указанное в операционной схеме.

Бл.3. Вычисление  $N_x, N_C, N_D, N_P, N_A, N_T$  производится по следующим формулам:

$$N_x = N_x^1; \quad N_P = N_P^1 + N_P^2;$$

$$N_C = N_C^1; \quad N_A = N_A^1 + N_A^2;$$

$$N_D = N_D^1; \quad N_T = N_T^1 + N_T^2;$$

Бл.4. Получение отождествляющих отображений  $\sigma_1$  и  $\sigma_2$ .

$$\sigma_2^{КСДМП} = \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} n_M^1 + N_n^1 \\ n_M^1 + N_n^1 \end{pmatrix} \right\}$$

$$\sigma_1^K = \left\{ \begin{pmatrix} n_n^1 + 1 \\ n_n + 1 \end{pmatrix}, \begin{pmatrix} n_K^1 \\ n_n + N_n^1 \end{pmatrix} \right\}$$

$$\sigma_1^A = \left\{ \begin{pmatrix} n_K^1 + 1 \\ n_K + 1 \end{pmatrix}, \begin{pmatrix} n_A^1 \\ n_K + N_A^1 \end{pmatrix} \right\}$$

$$\sigma_1^T = \left\{ \begin{pmatrix} n_A^1 + 1 \\ n_A + 1 \end{pmatrix}, \begin{pmatrix} n_T^1 \\ n_A + N_T^1 \end{pmatrix} \right\}$$

$\sigma_2^{КСД}$  - формируется в соответствии с отождествляющим отобра-

женем  $\Gamma$ .

$$U_2^M = \left\{ \begin{pmatrix} n_M^2 \\ 0 \end{pmatrix} \right\}, \quad \text{если } N_A^2 = I$$

$$U_2^N = \left\{ \begin{pmatrix} n_M^2 + 1 \\ n_M + N_M^1 + 1 \end{pmatrix}, \begin{pmatrix} n_N^2 \\ n_N \end{pmatrix} \right\}$$

$$U_2^K = \left\{ \begin{pmatrix} n_N^2 + 1 \\ n_N + N_N^1 + 1 \end{pmatrix}, \begin{pmatrix} n_K^2 \\ n_K \end{pmatrix} \right\}$$

$$U_2^A = \left\{ \begin{pmatrix} n_K^2 + 1 \\ n_K + N_A^1 + 1 \end{pmatrix}, \begin{pmatrix} n_A^2 \\ n_A \end{pmatrix} \right\}$$

$$U_2^T = \left\{ \begin{pmatrix} n_A^2 + 1 \\ n_A + N_T^1 + 1 \end{pmatrix}, \begin{pmatrix} n_T^2 \\ n_T \end{pmatrix} \right\}$$

Замечания. 1. Если в приведенных последовательностях  $U_1^{XСДМП}$ ,  $U_i^K$ ,  $U_i^A$ ,  $U_i^T$ ,  $U_2^N$  вторая пара совпадает с первой, то она не входит в эту последовательность.

2. Если в приведенных последовательностях  $U_1^{XСДМП}$ ,  $U_i^K$ ,  $U_i^A$ ,  $U_i^T$ ,  $U_2^N$  первый элемент второй пары меньше первого элемента первой пары, то эти последовательности не входят в соответствующую последовательность  $U_i$ .

3. Если  $N_A^2 = 0$ , то  $U_2^M$  не входит в  $U_2$ .

Бл.5. Получение  $WIR(B_i)$  и  $WIR(B_i)$

Функция  $WIR$  выделяет из элемента  $B_i$  ( $i = 1, 2$ ) записи рода структуры элементы  $\bar{M}$ ,  $\bar{K}$ ,  $\bar{N}$ ,  $\bar{A}$ ,  $\bar{T}$ .

Бл.6,7,8. Данные блоки соответствуют работе подпрограммы редактирования выражений  $\bar{P}(i)$ ,  $\bar{A}(i)$ ,  $\bar{T}(i)$  (функция  $F_{U_i}(i)$ )

Исходные данные для подпрограммы:

- выражение  $B$  ( $\bar{P}(i)$  или  $\bar{A}(i)$  или  $\bar{T}(i)$ ),
- отождествляющее отображение  $U_2$ , а также
- числа  $N_M^1$ ,  $N_N^1$ ,  $N_A^1$ ,  $N_T^1$ ,  $N_M^2$ ,  $N_N^2$ ,  $N_A^2$ ,  $N_T^2$ ,  $N_M$ ,  $N_N$ ,  $N_A$ ,  $N_T$ , которые служат для расшифровки  $U_2$ .

В результате работы подпрограммы в выражении  $B$  происходит замена конститuent классов  $M$ ,  $N$ ,  $A$ ,  $T$  в соответствии с отождествляющим отображением  $U_2$ .

Бл.9. Редактирование элемента  $\bar{K}_{(2)}$  отличается от редактирования элементов  $\bar{P}_{(2)}$ ,  $\bar{A}_{(2)}$ ,  $\bar{T}_{(2)}$ . Поэтому эту операцию производит специальная подпрограмма  $F'_{\delta_2}$  (выражение).

В результате работы подпрограммы в выражении  $\bar{K}_{(2)}$  происходит замена конститuent классов  $\lambda$  и  $\sigma$ , определяемая отображением  $\delta_2$  следующим образом:

$$\lambda_p \text{ заменяется на } \begin{cases} \lambda_q \text{ (или } \sigma_q), & \text{если } \delta_2 \text{ определяет соответствие } \lambda_p \rightarrow \lambda_q \text{ (или } \lambda_p \rightarrow \sigma_q), \\ \sigma, & \text{если } \delta_2 \text{ определяет соответствие } \lambda_p \rightarrow \sigma \\ & \text{(или } \lambda_p \rightarrow \pi_i) \text{ и } \bar{M} = B(\delta)(\bar{K}_i = B(\delta)) \end{cases}$$

$$\sigma_p \text{ заменяется на } \begin{cases} \sigma_q, & \text{если } \delta_2 \text{ определяет соответствие } \sigma_p \rightarrow \sigma_q \\ \sigma, & \text{если } \delta_2 \text{ определяет соответствие } \sigma_p \rightarrow \sigma \\ & \text{(или } \sigma_p \rightarrow \pi_i) \text{ и } \bar{M} = B(\delta)(\bar{K}_i = B(\delta)) \end{cases}$$

Бл.10. Получение элемента  $\bar{M}$

$$\bar{M} = \bar{M}(1)$$

Бл.11. Получение элементов  $\bar{K}$ ,  $\bar{P}$ ,  $\bar{A}$ ,  $\bar{T}$

$$\bar{K} = \bar{K}(1) \times' 2' \times \bar{K}(2)$$

$$\bar{P} = \bar{P}(1) \times' 2' \times \bar{P}(2)$$

$$\bar{A} = \bar{A}(1) \times' 2' \times \bar{A}(2)$$

$$\bar{T} = \bar{T}(1) \times' 2' \times \bar{T}(2)$$

Бл.13,14. Получение элемента  $B$  результата операции.

С помощью специальной подпрограммы (функция  $S$ ) осуществляется сцепление элементов:  $\bar{M}$  (если  $M$  присутствует),  $\bar{K}$ ,  $\bar{P}$ ,  $\bar{A}$ ,  $\bar{T}$  в элемент  $B$ .

Результатом работы модуля является запись рода структуры, а также запись вспомогательного массива для Т-интерпретации.

1.3.3. Модуль, выполняющий операции простого и общего расширения базы.

Исходными данными для работы модуля являются:

- код операции;
- I аргумент операции ( $\Phi_1$ ) - род структуры;

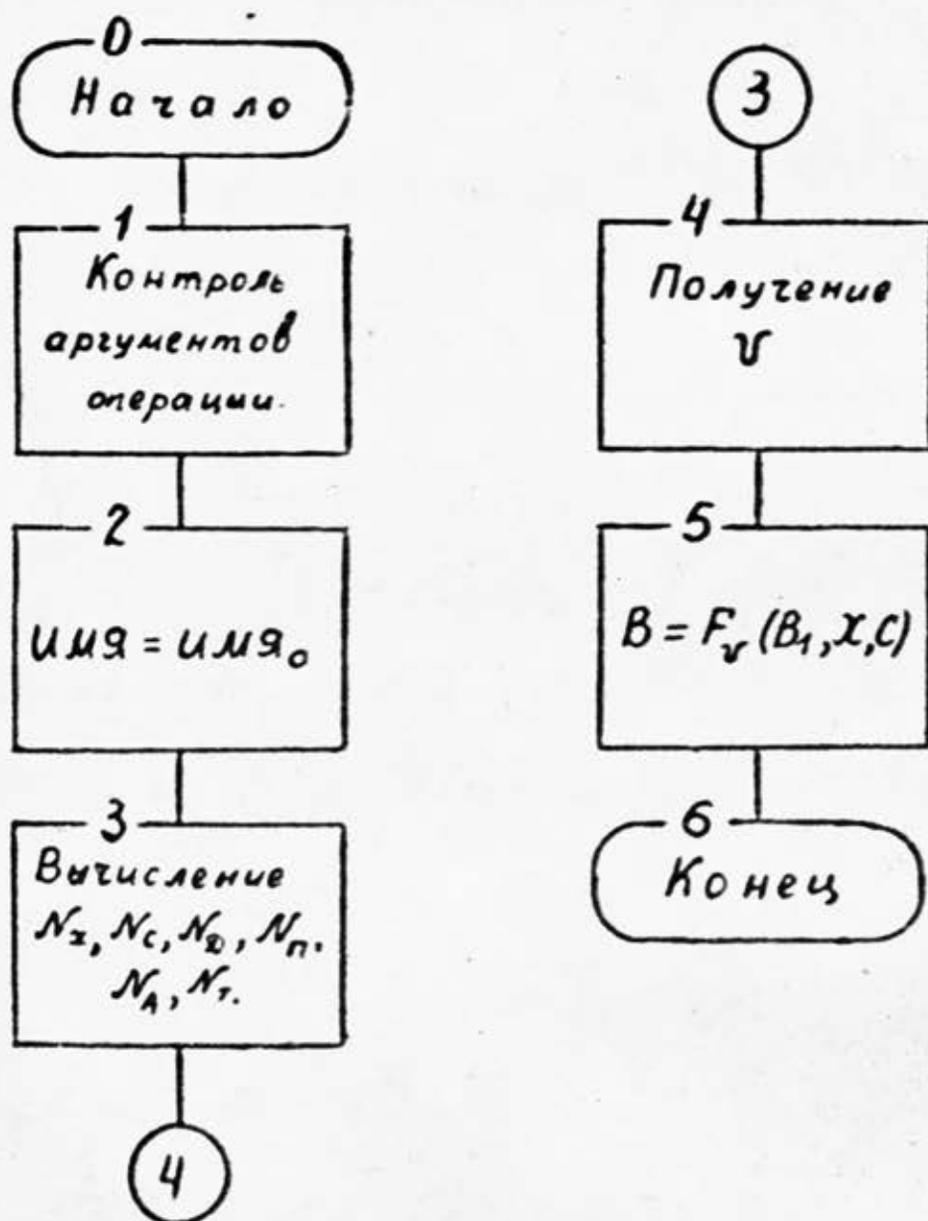


Рис. 1.3.3.

- Параметр операции ( $\Gamma$ ) - отождествляющее отображение (присутствует лишь в операции общего расширения базы);
- I параметр ( $n$ ) - натуральное число, обозначающее количество конститuent класса  $X$  в результате;
- II параметр ( $m$ ) - натуральное число, обозначающее количество конститuent класса  $C$  в результате.

Блок-схема модуля представлена на рис. 1.3.3.

В блок-схеме приведены следующие блоки.

- Бл.1. При выполнении операций простого и общего расширения базы проверяется выполнение следующих соотношений:
- а)  $n \geq N_x^1$ ;  $m \geq N_c^1$ ;
- При выполнении операции общего расширения базы дополнительно проверяется выполнение следующих соотношений:
- б)  $N(\Gamma) = N_x^1 + N_c^1$ , где  $N(\Gamma)$  - число элементов в отождествляющем отображении  $\Gamma$ ;
  - в) первые  $N_x^1$  имен в  $\Gamma$  есть имена  $x_i$ , где  $1 \leq i \leq n$ ;
  - г) последние  $N_c^1$  имен в  $\Gamma$  есть имена  $c_j$ , где  $1 \leq j \leq m$ ;
- Бл.2. Результату операции присваивается имя, указанное в операционной схеме.
- Бл.3. Вычисление  $N_x$ ,  $N_c$ ,  $N_d$ ,  $N_A$ ,  $N_T$ ,  $N_n$  производится по следующим формулам:

$$\begin{aligned} N_x &= n; & N_n &= N_n^1; \\ N_c &= m; & N_A &= N_A^1; \\ N_d &= N_d^1; & N_T &= N_T^1; \end{aligned}$$

- Бл.4. Получение отождествляющего отображения  $\mathcal{V}$ .

$$\mathcal{V}_x = \begin{cases} \left\{ \left( \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} N_x^1 \\ N_T^1 \end{pmatrix} \right) \right\} & \text{в случае операции простого расширения} \\ \text{формируется в соответствии с } \Gamma & \text{базы;} \\ \text{формируется в соответствии с } \Gamma & \text{в случае операции} \\ & \text{общего расширения базы.} \end{cases}$$

$\sigma^c = \left\{ \left\{ \binom{N_x^1+1}{n+1}, \binom{n_c^1}{n+N_c^1} \right\} \right\}$  в случае операции простого расширения базы;  
 формируется в соответствии с  $\Gamma$  в случае операции общего расширения базы.

$$\sigma^{AMPKAT} = \left\{ \binom{n_c^1+1}{n+m+1}, \binom{n_r^1}{n_r} \right\}$$

Бл.5. Получение элемента В результата операции с помощью подпрограммы редактирования выражений (функция  $F\sigma(B)$ ).

Исходные данные для подпрограммы:

- выражение  $B_1$ ;
- отождествляющее отображение  $\sigma$ ;
- числа  $N_x^1, N_c^1, N_d^1, N_p^1, N_A^1, N_T^1, N_x, N_c, N_d, N_p, N_A, N_T$  (для расшифровки  $\sigma$ ).

В результате работы подпрограммы в выражении  $B_1$  происходит замена конституэнт типа  $x$  и  $c$  в соответствии с отождествляющим отображением  $\sigma$ .

#### 1.3.4. Модуль, выполняющий операцию терм-вложения

Исходными данными для работы модуля являются:

- I аргумент ( $\varphi_1$ ) - род структуры;
- II аргумент ( $\varphi_2$ ) - род структуры;
- III аргумент ( $\Gamma$ ) - отождествляющее отображение.

Блок-схема модуля представлена на рис.1.3.4.

В блок-схеме через  $\beta$  обозначен признак, служащий для разделения случаев: а)  $(N_A^1 = 0) \wedge (N_A^2 = 0)$ ; ( $\beta = 0$ )  
 б)  $(N_A^1 = 1) \vee (N_A^2 = 1)$ ; ( $\beta = 1$ )

В блок-схеме приведены следующие блоки.

Бл. I. Контроль аргументов операции происходит путем проверки следующих соотношений:

- а) число элементов в отождествляющем отображении  $\Gamma$

$$N(\Gamma) = N_x^2 + N_c^2;$$

- б) если  $\Gamma(\rho)$  - конституэнты класса

- $x$ , то ее номер  $i \leq N_x^1$ ;
- $c$ , то ее номер  $i \leq N_c^1$ ;
- $d$ , то  $\bar{M} = B(\dots)$ ;
- $p$ , то ее номер  $i \leq N_p^1$       и  $\bar{K}_i = B(\dots)$ ;

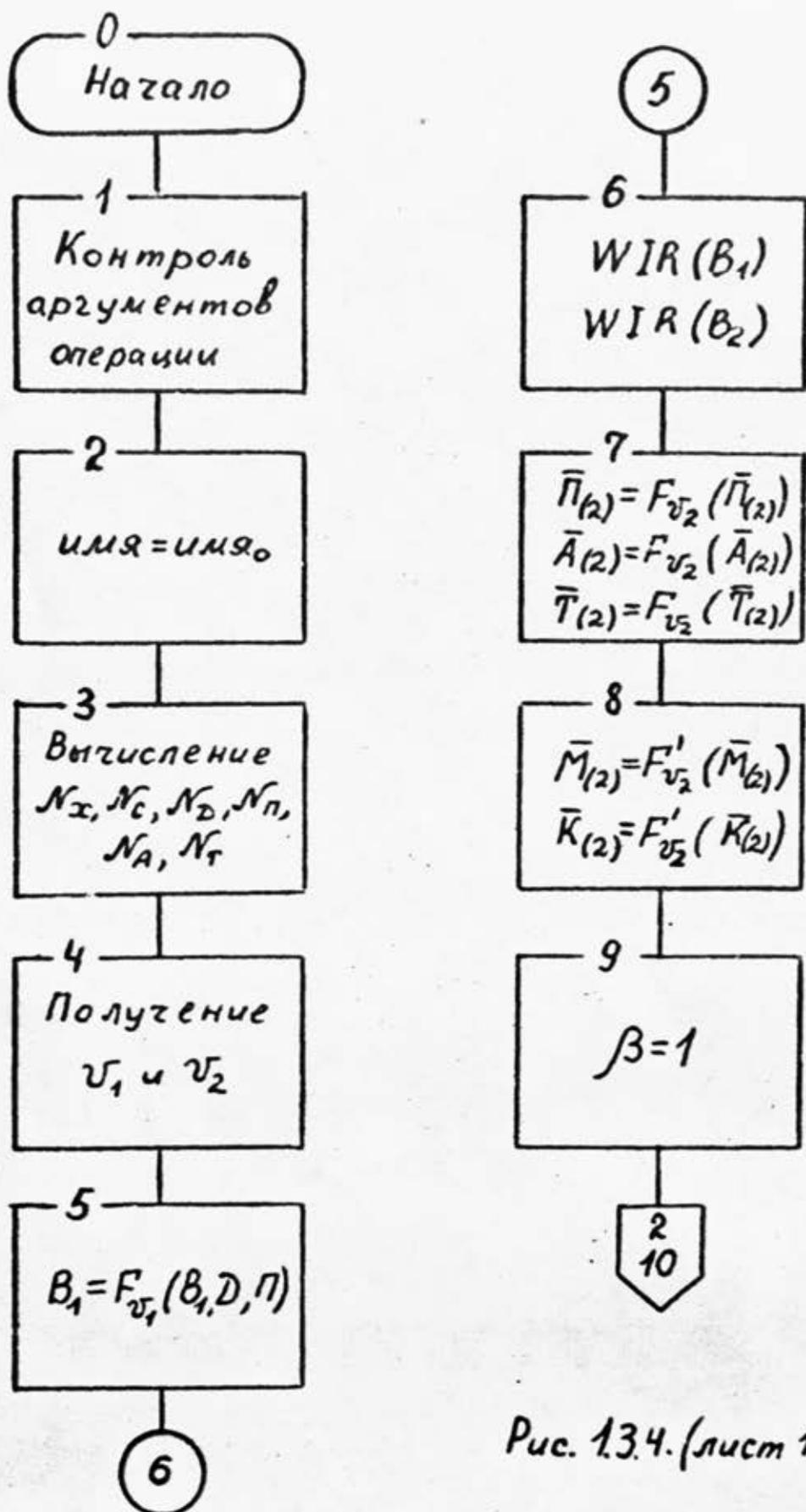


Рис. 1.3.4. (лист 1)

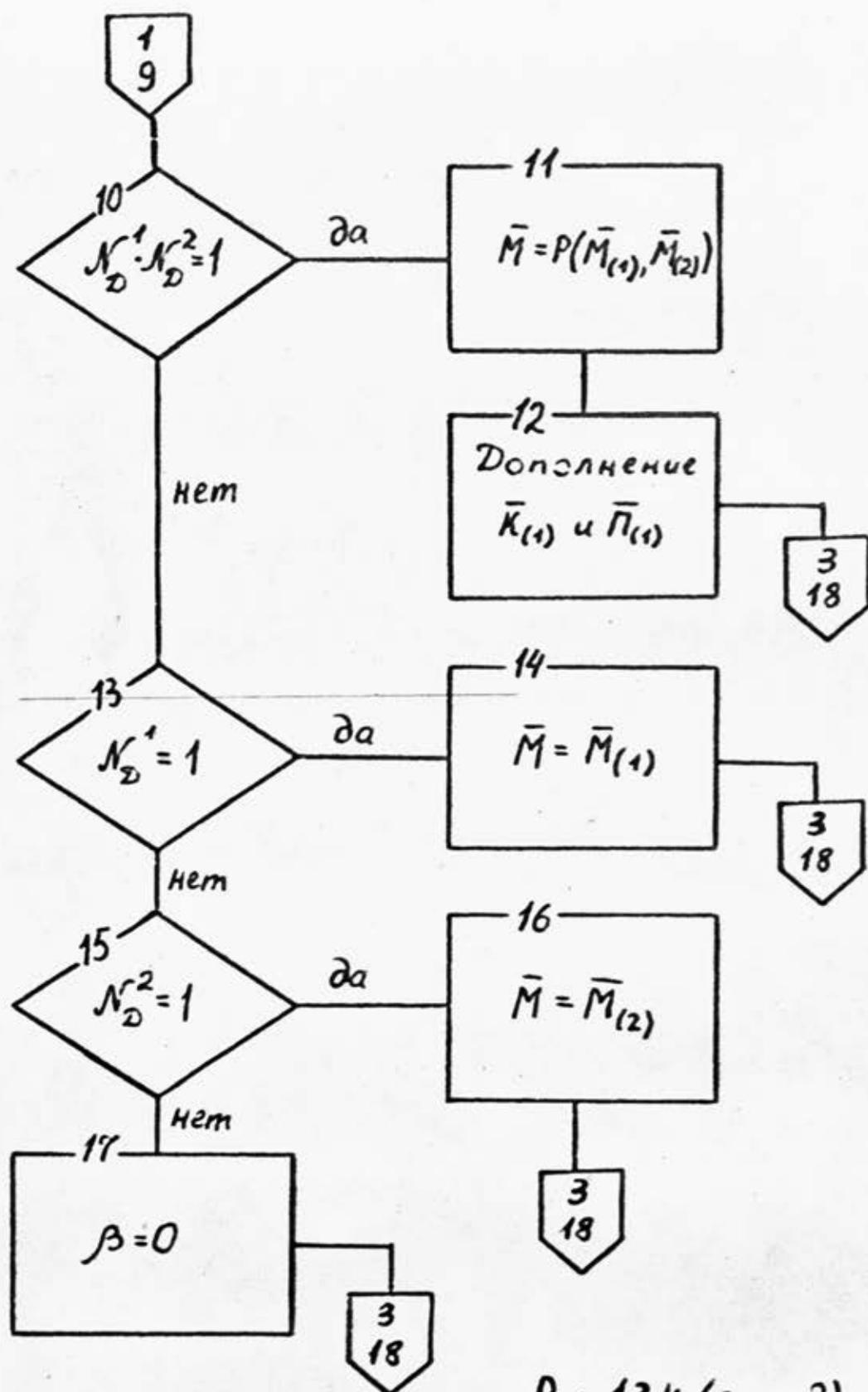


Рис. 1.3.4 (лист 2)

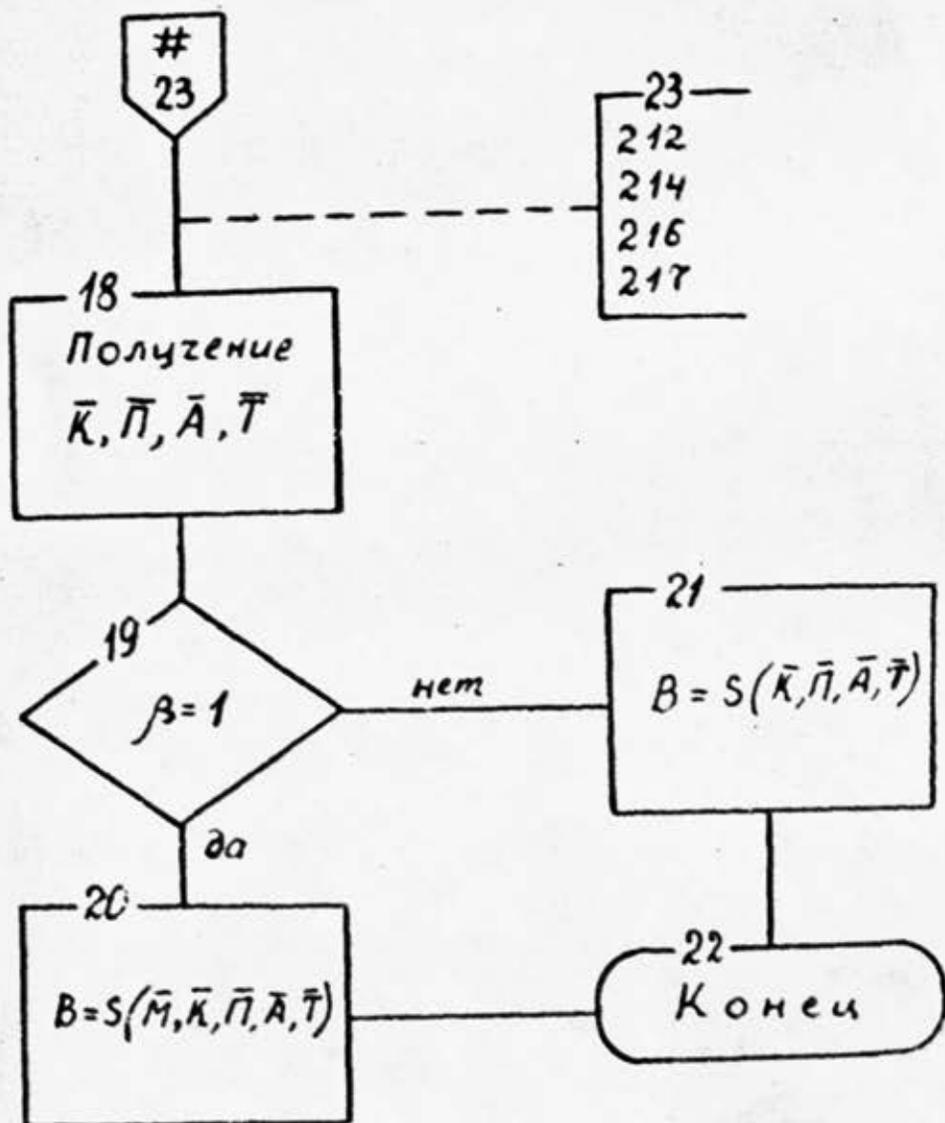


Рис. 1.34. (лист 3)

в) отображение  $\Gamma$  не может определять соответствие  $x_p \rightarrow c_q$

Бл.2. Результату операции присваивается имя, указанное в операционной схеме.

Бл.3. Вычисление  $N_k, N_c, N_d, N_p, N_A, N_T$  производится по следующим формулам:

$$\begin{aligned} N_k &= N_k^1; \\ N_c &= N_c^1; \\ N_p &= N_p^1 + N_p^2 + 2 \cdot N_A^1 \cdot N_A^2; \\ N_A &= \begin{cases} 1, & \text{если } (N_A^1 = 1) \vee (N_A^2 = 1) \\ 0 & \text{в противном случае} \end{cases} \\ N_T &= N_T^1 + N_T^2 \\ N_A &= \begin{cases} N_A^1 + N_A^2 + 1, & \text{если } (N_A^2 = 1) \\ N_A^1 + N_A^2, & \text{если } N_A^2 = 0 \end{cases} \end{aligned}$$

Бл.4. Получение отождествляющих отображений  $\sigma_1$  и  $\sigma_2$

$$\sigma_1^{1c} = \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} n_c^1 \\ n_c^1 \end{pmatrix} \right\}$$

$\sigma_2^{1c}$  определяется отождествляющим отображением  $\Gamma$ .

$$\sigma_i^A = \left\{ \begin{pmatrix} n_A^i \\ n_A + N_A^1 \cdot N_A^2 (1+i) \end{pmatrix} \right\}, \quad \text{если } N_A^i = 1$$

$$\sigma_i^M = \left\{ \begin{pmatrix} n_M^i \\ n_M + N_A^1 \cdot N_A^2 (N_M + 1) \end{pmatrix} \right\}, \quad \text{если } N_A^i = 1$$

$$\sigma_i^N = \left\{ \begin{pmatrix} n_N^i + 1 \\ n_M + N_A^1 \cdot N_A^2 (1+i) + (i-1) N_N^1 + 1 \end{pmatrix}, \begin{pmatrix} n_N^i \\ n_N + N_A^1 \cdot N_A^2 (1+i) + (i-1) N_N^1 + N_N^i \end{pmatrix} \right\}$$

$$\sigma_i^K = \left\{ \begin{pmatrix} n_N^i + 1 \\ n_M + N_A^1 \cdot N_A^2 (1+i) + (i-1) N_N^1 + 1 \end{pmatrix}, \begin{pmatrix} n_K^i \\ n_M + N_A^1 \cdot N_A^2 (1+i) + (i-1) N_N^1 + N_N^i \end{pmatrix} \right\}$$

$$\sigma_i^A = \left\{ \begin{pmatrix} n_K^i + 1 \\ n_K + (i-1) N_A^1 + 1 \end{pmatrix}, \begin{pmatrix} n_A^i \\ n_K + (i-1) N_A^1 + N_A^i \end{pmatrix} \right\}$$

$$\sigma_i^T = \left\{ \begin{pmatrix} n_A^i + 1 \\ n_A + (i-1) N_T^1 + 1 \end{pmatrix}, \begin{pmatrix} n_T^i \\ n_A + (i-1) N_T^1 + N_T^i \end{pmatrix} \right\}$$

$i = 1, 2.$

Бл.5. Блок соответствует работе подпрограммы редактирования выражения  $B_1$  (функция  $F_6(B, A, \Pi)$ ).

Исходные данные для подпрограммы:

- выражение  $B_1$  ;
- отождествляющее отображение  $\mathcal{U}_1$  ;
- числа  $N_1^i, N_2^i, N_3^i, N_4^i, N_5^i, N_6^i, N_7^i, N_8, N_9, N_{10}, N_{11}, N_{12}$  (для расшифровки  $\mathcal{U}_1$ ).

В результате работы подпрограммы в выражении  $B_1$  происходит замена конституэнт классов  $d$  и  $\Pi$  в соответствии с отождествляющим отображением  $\mathcal{U}_1$ .

Бл.6. Получение  $WIR(B_1)$  и  $WIR(B_2)$

Функция  $WIR$  выделяет из элемента  $B_i$  записи рода структуры элементы  $\bar{M}_{(i)}, \bar{K}_{(i)}, \bar{\Pi}_{(i)}, \bar{A}_{(i)}, \bar{T}_{(i)}$   $i = 1, 2$ .

Бл.7. Блок соответствует работе подпрограммы редактирования выражений  $\bar{\Pi}_{(2)}, \bar{A}_{(2)}, \bar{T}_{(2)}$  (функция  $F_6(B)$ ).

Исходные данные для подпрограммы:

- выражения  $B$  ( $\bar{\Pi}_{(2)}$  или  $\bar{A}_{(2)}$  или  $\bar{T}_{(2)}$ ) ;
- отождествляющее отображение  $\mathcal{U}_2$  ;
- числа  $N_1^2, N_2^2, N_3^2, N_4^2, N_5^2, N_6^2, N_7^2, N_8, N_9, N_{10}, N_{11}, N_{12}$ .

В результате работы подпрограммы в выражениях  $\bar{\Pi}_{(2)}, \bar{A}_{(2)}, \bar{T}_{(2)}$  происходит замена конституэнт класса  $x, c, d, \Pi$  в соответствии с отождествляющим отображением  $\mathcal{U}_2$ .

Бл.8. Блок соответствует работе подпрограммы редактирования выражений  $\bar{M}_{(2)}$  и  $\bar{K}_{(2)}$ .

Редактирование этих элементов отличается от редактирования элементов  $\bar{\Pi}_{(2)}, \bar{A}_{(2)}, \bar{T}_{(2)}$ . Поэтому эту операцию производит специальная подпрограмма  $F'_6$  (выражение).

В результате работы подпрограммы в выражении  $\bar{M}_{(2)}$  и  $\bar{K}_{(2)}$  происходит замена конституэнт классов  $y, c$  определяемая отображением  $\mathcal{U}_2$  следующим образом:

$$\begin{aligned}
 & \text{кр заменяется ю} \left\{ \begin{array}{l} \chi \varphi, \text{ если } \beta_1 \text{ определяет соответствие } \chi \rho \rightarrow \chi \varphi \\ S, \text{ если } \beta_1 \text{ определяет соответствие } \chi \rho \rightarrow A \\ \text{(или } \chi \rho \rightarrow \Pi_i \text{) и } M = B(\beta) \text{ (} \bar{K}_i = B(\beta) \text{)} \end{array} \right. \\
 & \text{ср заменяется ю} \left\{ \begin{array}{l} \sigma \varphi, \text{ если } \beta_2 \text{ определяет соответствие } \sigma \rho \rightarrow \sigma \varphi \\ S, \text{ если } \beta_2 \text{ определяет соответствие } \sigma \rho \rightarrow A \\ \text{(или } \sigma \rho \rightarrow \Pi_i \text{) и } M = B(\beta) \text{ (} \bar{K}_i = B(\beta) \text{)} \end{array} \right.
 \end{aligned}$$

Бл. II, I4, I6. Получение значения элемента  $\bar{M}$  результата операции.

В блоке II  $\bar{M}$  получается как сцепление

$$\bar{M} = 'P( \begin{array}{c} \chi \\ \chi \end{array} \bar{M}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} , \begin{array}{c} \chi \\ \chi \end{array} \bar{M}_{(2)} \begin{array}{c} \chi \\ \chi \end{array} )'$$

Бл. I2. Дополнение элементов  $\bar{K}_{(1)}$  и  $\bar{\Pi}_{(1)}$

$$\bar{\Pi}_{(1)} = ' \rho \chi, A \chi \rho \chi \chi \chi \bar{\Pi}_{(1)}'$$

$$\bar{K}_{(1)} = \bar{M}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{M}_{(2)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{K}_{(1)}$$

Бл. I8. Получение элементов  $\bar{K}$ ,  $\bar{\Pi}$ ,  $\bar{A}$ ,  $\bar{T}$  результата операции.

$$\bar{K} = \bar{K}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{K}_{(2)}$$

$$\bar{\Pi} = \bar{\Pi}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{\Pi}_{(2)}$$

$$\bar{A} = \begin{cases} \bar{A}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{A}_{(2)} \begin{array}{c} \chi \\ \chi \end{array} \chi \Pi_2 \in \begin{array}{c} \chi \\ \chi \end{array} \bar{M}_{(2)} \text{ если } (N_A^1=1) \wedge (N_A^2=1) \\ \bar{A}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{A}_{(2)} \begin{array}{c} \chi \\ \chi \end{array} \chi A \in \begin{array}{c} \chi \\ \chi \end{array} \bar{M}_{(2)} \text{ если } (N_A^1=0) \wedge (N_A^2=1) \\ \bar{A}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{A}_{(2)} \quad \text{в остальных случаях} \end{cases}$$

$$\bar{T} = \bar{T}_{(1)} \begin{array}{c} \chi \\ \chi \end{array} \chi \begin{array}{c} \chi \\ \chi \end{array} \bar{T}_{(2)}$$

Бл. 20, 21. Получение элемента  $B$  результата операции.

С помощью специальной подпрограммы (функция  $S$ ) осуществляется сцепление элементов  $\bar{M}$  (если  $\bar{M}$  присутствует),  $\bar{K}$ ,  $\bar{\Pi}$ ,  $\bar{A}$ ,  $\bar{T}$  в элемент  $B$ .

### 1.3.5. Модуль, выполняющий операцию конкретизирующего вложения

Исходными данными для работы модуля являются:

- I аргумент ( $\Phi_1$ ) - род структуры;
- II аргумент ( $\Phi_2$ ) - род структуры;
- III аргумент ( $\Gamma$ ) - отождествляющее отображение.

Блок-схема модуля представлена на рис.1.3.5.

В блок-схеме через  $\beta$  обозначен признак, служащий для разделения случаев:

а)  $(N_A^1 = 0) \wedge (N_A^2 = 0), (\beta = 0)$   
 б)  $(N_A^1 = 1) \vee (N_A^2 = 1), (\beta = 1)$

В блок-схеме приведены следующие блоки.

Бл.1. Контроль аргументов операции происходит путем проверки следующих соотношений:

а) число элементов в отождествляющем отображении  $\Gamma$

$$N(\Gamma) = N_1^1 + N_2^2;$$

б) если  $\Gamma(\rho)$  - конститутенты класса

- $x$ , то ее номер  $i \in N_1^1$ ;
- $c$ , то ее номер  $i \in N_2^2$ ;
- $d$ , то  $M = B(\dots)$ ,
- $p$ , то ее номер  $i \in N_p^1$  и  $\bar{K}_i = B(\dots)$ .

в) отображение  $\Gamma$  не может определять соответствие  $x\rho^{-c}$ ;

При невыполнении указанных соотношений должно выдаваться сообщение об ошибке.

Бл.2. Результату операции присваивается имя, указанное в операционной схеме.

Бл.3. Вычисление  $N_x, N_c, N_d, N_p, N_A, N_T$  производится по следующим формулам:

$$N_x = N_x^1;$$

$$N_c = N_c^1;$$

$$N_d = \begin{cases} 1, & \text{если } (N_A^1 = 1) \vee (N_A^2 = 1) \\ 0 & \text{в противном случае} \end{cases}$$

$$N_p = N_p^1 + N_p^2 + 2 N_d^1 \cdot N_d^2;$$

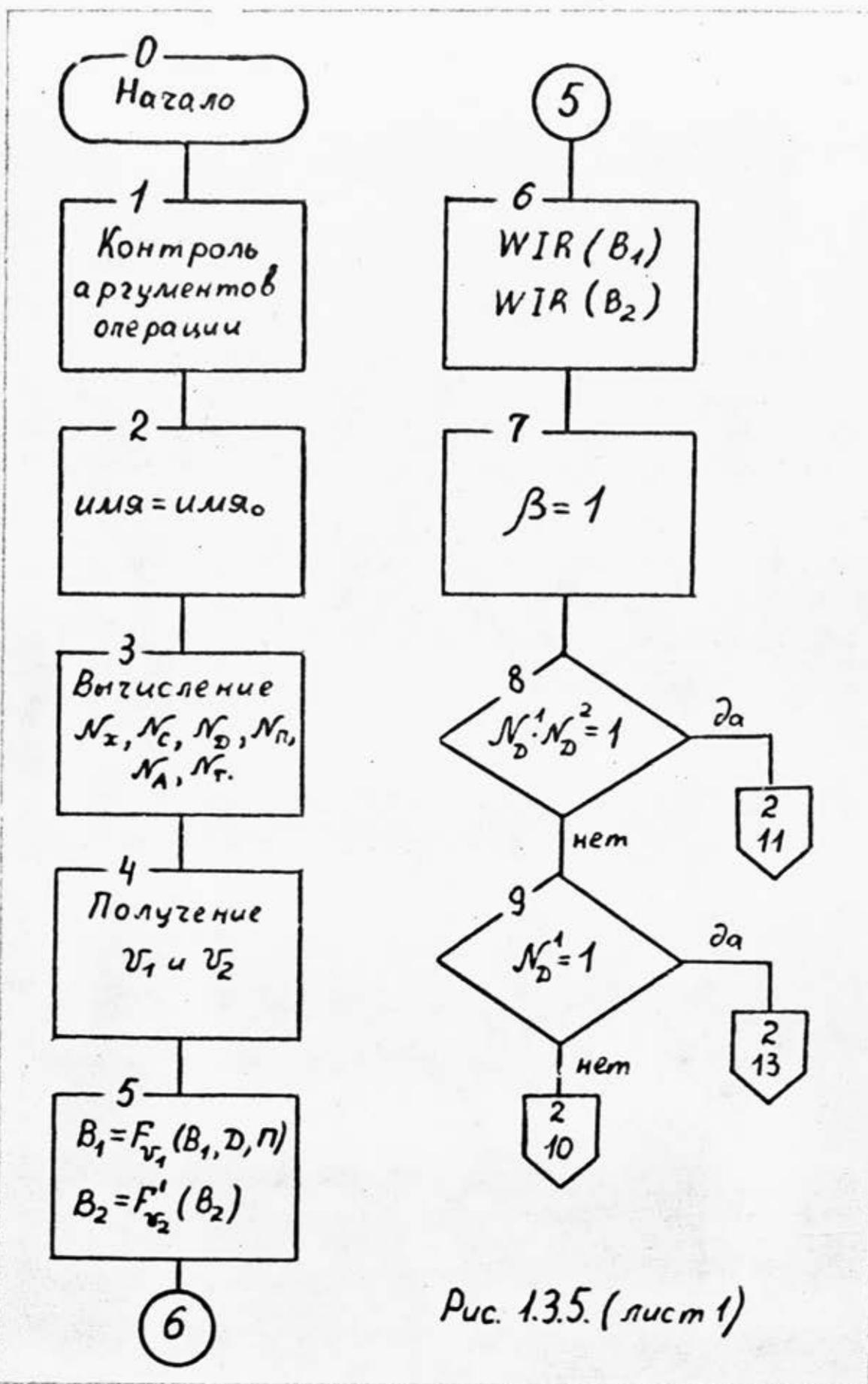


Рис. 1.3.5. (лист 1)

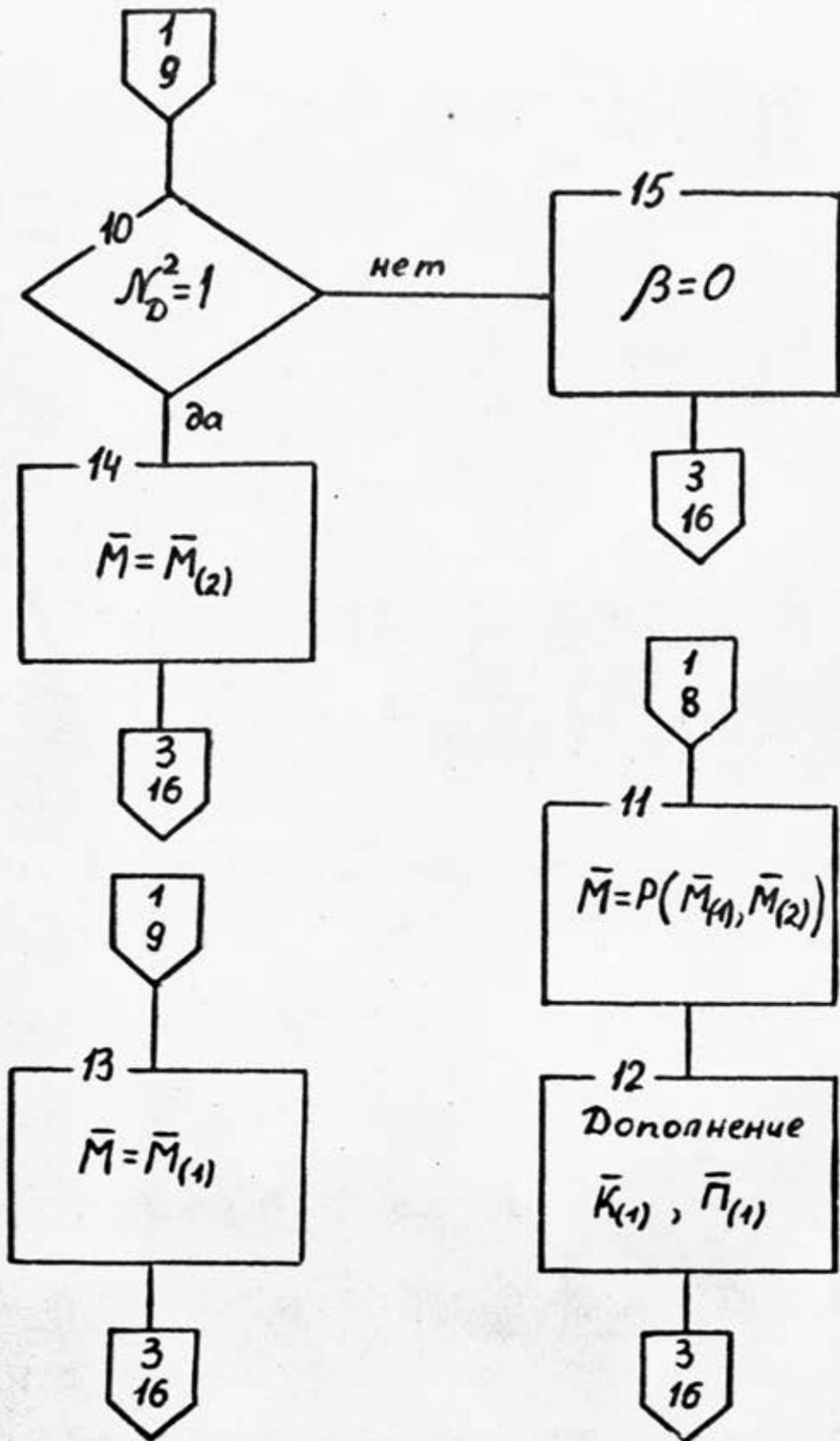


Рис. 1.35 (лист 2)

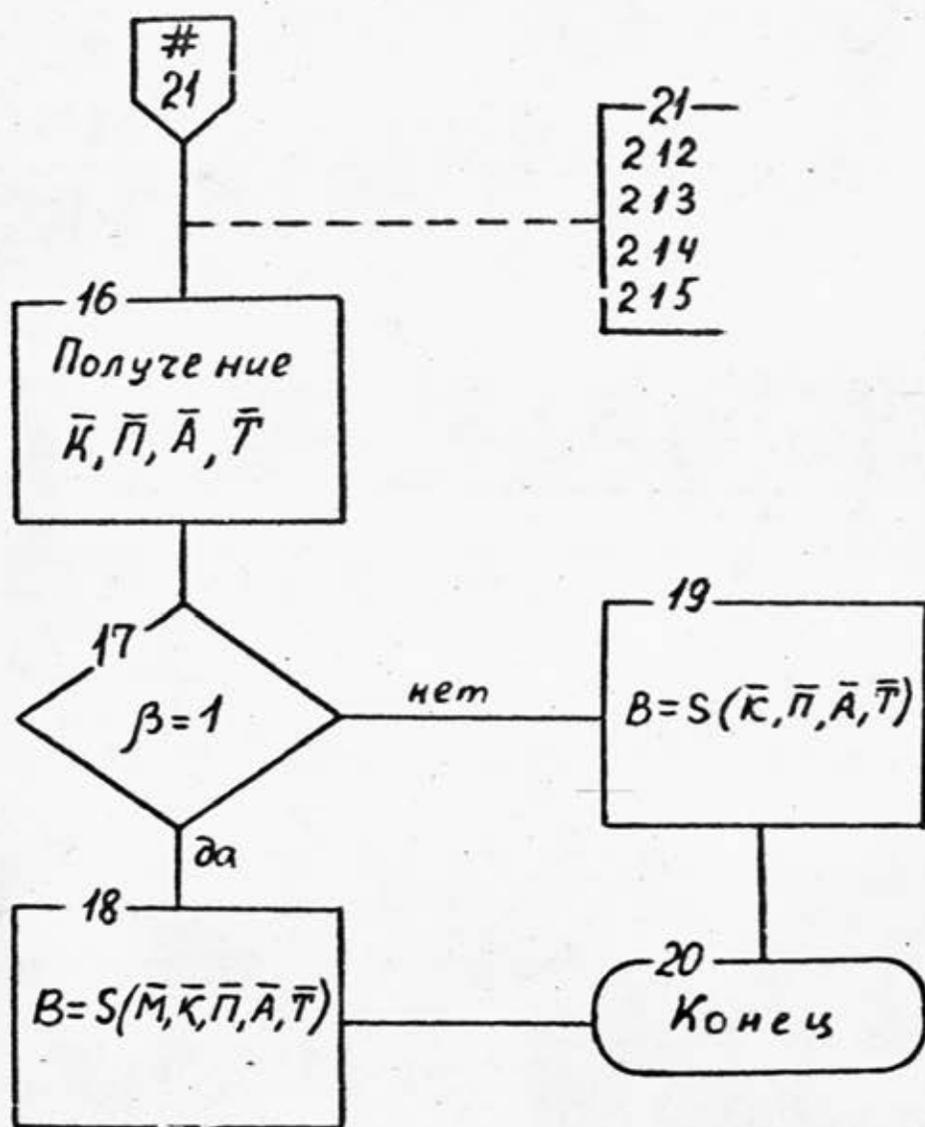


Рис. 13.5. (лист 3)

$$N_A = \begin{cases} N_A^1 + N_A^2 + 1 & \text{, если } (N_A^i = I) \\ N_A^1 + N_A^2 & \text{в остальных случаях} \end{cases}$$

$$N_T = N_T^1 + N_T^2;$$

Бл.4. Получение отождествляющих отображений  $\sigma_1$  и  $\sigma_2$

$$\sigma_1^{1c} = \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} n_c^1 \\ n_c^1 \end{pmatrix} \right\}$$

$\sigma_2^{1c}$  — определяется отождествляющим отображением  $\Gamma$ .

$$\sigma_i^D = \left\{ \begin{pmatrix} n_A^i \\ n_A + N_A^1 \cdot N_A^2 (1 + i) \end{pmatrix} \right\} \text{ если } N_A^i = I$$

$$\sigma_i^M = \left\{ \begin{pmatrix} n_n^i \\ n_m + N_A^1 \cdot N_A^2 (N_n + i) \end{pmatrix} \right\}, \text{ если } N_A^i = I$$

$$\sigma_i^N = \left\{ \begin{pmatrix} n_m^i + 1 \\ n_m + N_A^1 \cdot N_A^2 (1 + i) - (i-1)N_n^1 + 1 \end{pmatrix}, \begin{pmatrix} n_n^i \\ n_m + N_A^1 \cdot N_A^2 (1 + i) + (i-1)N_n^1 + N_n^1 \end{pmatrix} \right\}$$

$$\sigma_i^K = \left\{ \begin{pmatrix} n_n^i + 1 \\ n_n + N_A^1 \cdot N_A^2 (1 + i) - (i-1)N_n^1 + 1 \end{pmatrix}, \begin{pmatrix} n_k^i \\ n_n + N_A^1 \cdot N_A^2 (1 + i) + (i-1)N_n^1 + N_n^1 \end{pmatrix} \right\}$$

$$\sigma_i^A = \left\{ \begin{pmatrix} n_A^i - 1 \\ n_A + (i-1)N_A^1 + 1 \end{pmatrix}, \begin{pmatrix} n_A^i \\ n_A + (i-1)N_A^1 + N_A^1 \end{pmatrix} \right\}$$

$$\sigma_i^T = \left\{ \begin{pmatrix} n_A^i + 1 \\ n_A + (i-1)N_T^1 + 1 \end{pmatrix}, \begin{pmatrix} n_T^i \\ n_A + (i-1)N_T^1 + N_T^1 \end{pmatrix} \right\}$$

$i = 1, 2.$

Бл.5. Блок соответствует работе подпрограммы редактирования выражений  $B_1$  (функция  $F_V(B, A, \Pi)$ ) и  $B_2$  (функция  $F'_V(B)$ ).

Исходные данные для подпрограммы  $F_V(B, D, \Pi)$ :

- выражение  $B_1$ ;
- отождествляющее отображение  $\sigma_1$ ;
- числа  $N_A^1, N_C^1, N_A^2, N_n^1, N_A^1, N_T^1, N_A, N_C, N_A, N_n, N_A, N_T$  (для расшифровки  $\sigma_1$ ).

В результате работы подпрограммы в выражении  $B_1$  происходит замена конституэнт классов  $A$  и  $\Pi$  в соответствии с отождествляющим отображением  $\sigma_1$ .

Исходные данные для подпрограммы  $F'_V(B)$ :

- выражение  $B_2$ ;

- отождествляющее отображение  $\mathcal{U}_2$ ;
- числа  $N_1^i, N_2^i, N_A^i, N_\Pi^i, N_{\bar{A}}^i, N_{\bar{\Pi}}^i, N_{\bar{K}}, N_{\bar{C}}, N_{\bar{A}}, N_{\bar{\Pi}}, N_{\bar{A}}, N_{\bar{\Pi}}$ .

В результате работы подпрограммы в выражении  $\mathcal{B}_2$  происходит замена конститuent классов  $x, c, \Pi$  определяемая отображением  $\mathcal{U}_2$  следующим образом:

$$x_p \text{ заменяется на } \begin{cases} x_q, & \text{если } \mathcal{U}_2 \text{ определяет соответствие } x_p \rightarrow x_q \\ s, & \text{если } \mathcal{U}_2 \text{ определяет соответствие } x_p \rightarrow s \\ & \text{(или } x_p \rightarrow \Pi_i \text{)} \\ & \text{и } \bar{M} = B(S) \text{ (} \bar{K}_i = B(S) \text{)}; \end{cases}$$

$$c_p \text{ заменяется на } \begin{cases} c_q, & \text{если } \mathcal{U}_2 \text{ определяет соответствие } c_p \rightarrow c_q \\ s, & \text{если } \mathcal{U}_2 \text{ определяет соответствие } c_p \rightarrow s \\ & \text{(или } c_p \rightarrow \Pi_i \text{)} \\ & \text{и } \bar{M} = B(S) \text{ (} \bar{K}_i = B(S) \text{)}; \end{cases}$$

$$\Pi_i \text{ заменяется на } \Pi_q, \text{ если } \mathcal{U}_2 \text{ определяет соответствие } \Pi_i \rightarrow \Pi_q$$

$$A \text{ заменяется на } \Pi_2, \text{ если } N_A^1 \cdot N_A^2 = I \text{ (в противном случае } A \text{ замене не подлежит).}$$

Бл.6. Получение  $w_{IR}(B_1)$  и  $w_{IR}(B_2)$ .

Функция  $w_{IR}$  выделяет из элемента  $\mathcal{B}_i$  записи рода структуры элементы  $\bar{M}_{(i)}, \bar{K}_{(i)}, \bar{\Pi}_{(i)}, \bar{A}_{(i)}, \bar{T}_{(i)}$ .

Бл.11,13,14. Получение значения  $\bar{M}$  результата операции.

В блоке II  $\bar{M}$  получается как сцепление

$$\bar{M} = 'P( 'x \bar{M}_{(1)} x', 'x \bar{M}_{(2)} x')$$

Бл.12. Дополнение элементов  $\bar{K}_{(i)}$  и  $\bar{\Pi}_{(i)}$

$$\bar{\Pi}_{(i)} = 'p z_1 A z_2 p z_2 A z_1' x \Pi_{(i)}$$

$$\bar{K}_{(i)} = \bar{M}_{(i)} x 'z_1' x \bar{M}_{(i)} x 'z_2' x \bar{K}_{(i)}$$

Бл.16. Получение элементов  $\bar{K}, \bar{\Pi}, \bar{A}, \bar{T}$  результата операции.

$$\bar{K} = \bar{K}_{(1)} x 'z_1' x \bar{K}_{(2)}$$

$$\bar{\Pi} = \bar{\Pi}_{(1)} x 'z_1' x \bar{\Pi}_{(2)}$$

$$\bar{T} = \bar{T}_{(1)} x 'z_1' x \bar{T}_{(2)}$$

$$\bar{A} = \begin{cases} \bar{A}_{(1)} \times' \bar{z} \times' \bar{A}_{(2)} \times' \bar{z} \Pi_2 \in' \bar{M}_{(2)} & \text{если } (N_A^1 = I) \wedge (N_A^2 = I) \\ \bar{A}_{(1)} \times' \bar{z} \times' \bar{A}_{(2)} \times' \bar{z} \Delta \in' \bar{M}_{(2)} & \text{если } (N_A^1 = 0) \wedge (N_A^2 = I) \\ \bar{A}_{(1)} \times' \bar{z} \times' \bar{A}_{(2)} & \text{в остальных случаях} \end{cases}$$

Бл.18,19. Получение элемента В результата операции.

С помощью специальной подпрограммы (функция S) осуществляется спlicing элементов  $\bar{M}$  (если  $\bar{M}$  присутствует),  $\bar{K}$ ,  $\bar{P}$ ,  $\bar{A}$ ,  $\bar{T}$  в элемент В.

### 1.3.6. Модуль, выполняющий операцию булеанизации

Исходными данными для работы модуля являются:

- код операции;
- I аргумент ( $\Phi_1$ ) - род структуры;

Блок-схема операции булеанизации представлена на рис.

#### 1.3.6.

Введем дополнительные обозначения:

$W_1RA$  (выражение) - оператор, выполняющий расщепление выражения на элементы;

$\bar{z}$  - множество индексов конститuent класса  $\Pi$ , в которых производились замены;

$\bar{z}$  - подмножество  $\bar{z}$  индексов тех конститuent класса  $\Pi$ , которые входят в рассматриваемое выражение;

$FORM$  (выражение) - оператор, формирующий по выражению множество  $\bar{z}$ ;

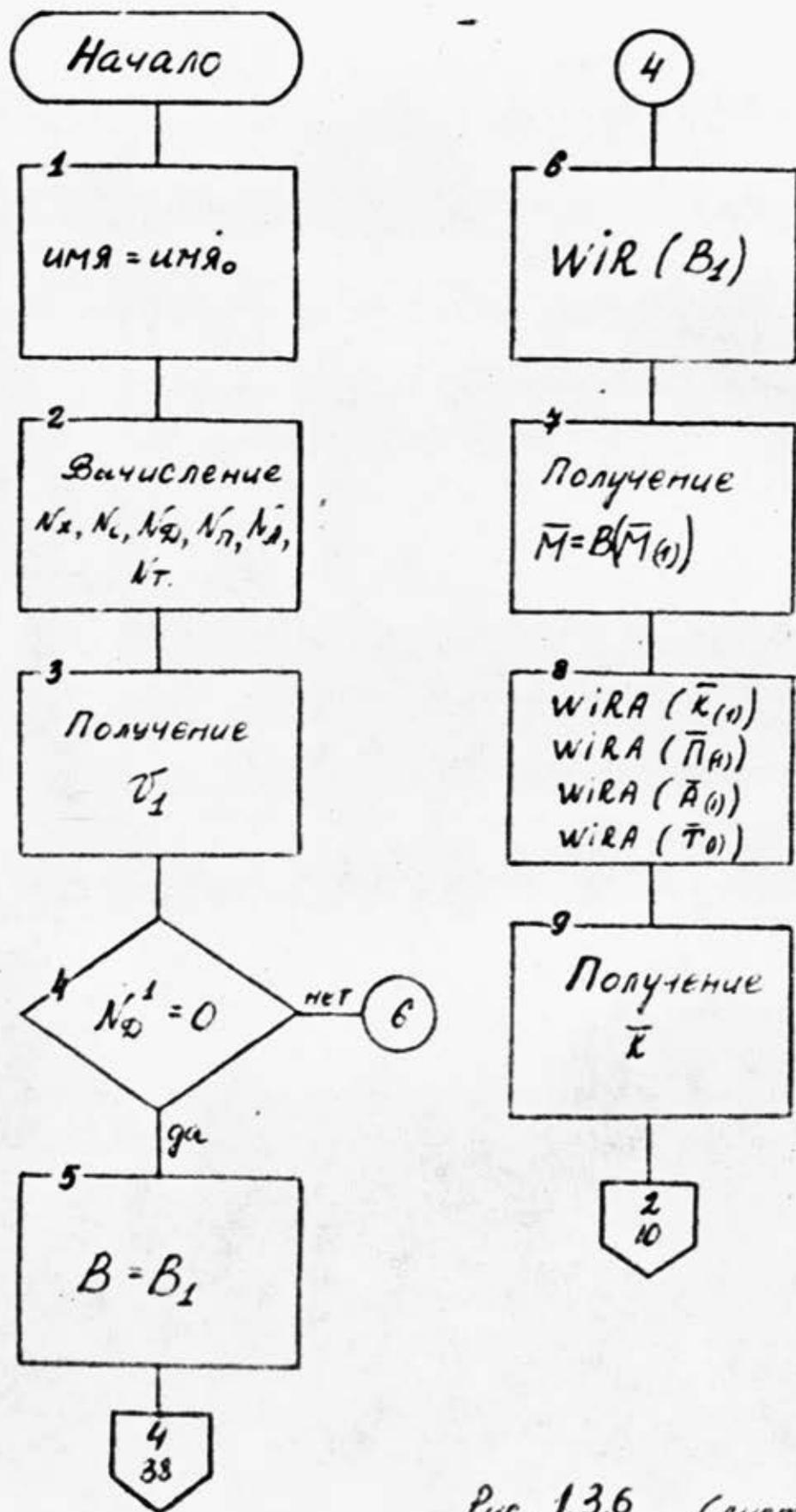
$zAM$  (выражение,  $\bar{z}$ ) - оператор обработки выражения в соответствии со множеством  $\bar{z}$ .

Блок-схема содержит следующие блоки.

Бл.1. Результату операции присваивается имя, указанное в операционной схеме.

Бл.2. Вычисление  $N_x, N_c, N_d, N_n, N_a, N_t$  по формулам:

$$\begin{array}{ll} N_x = N_x^1; & N_n = N_n^1; \\ N_c = N_c^1; & N_a = N_a^1; \\ N_d = N_d^1; & N_t = N_t^1; \end{array}$$



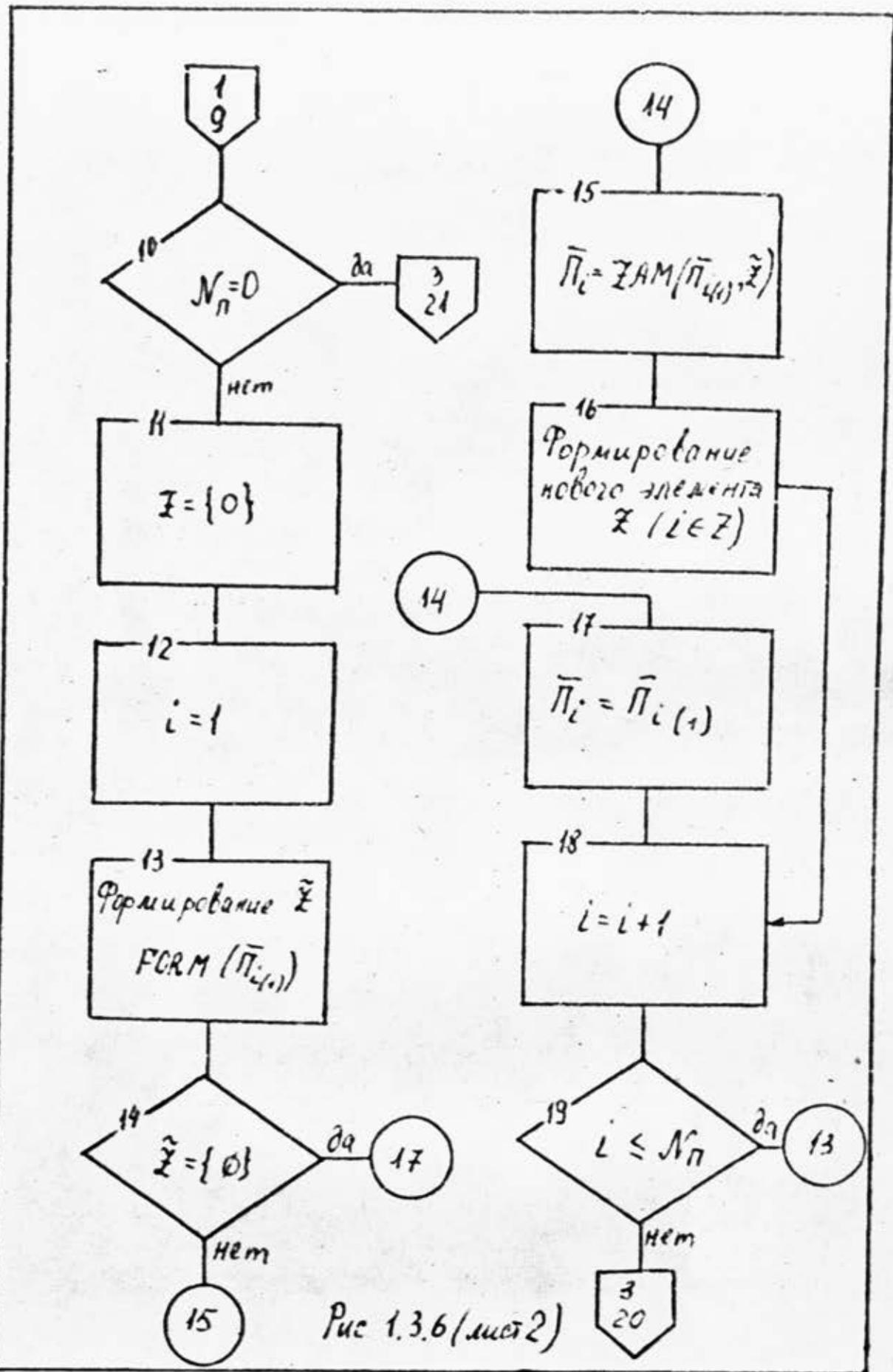


Рис 1.3.6 (лист 2)

15-2-76  
ТПТЗм1

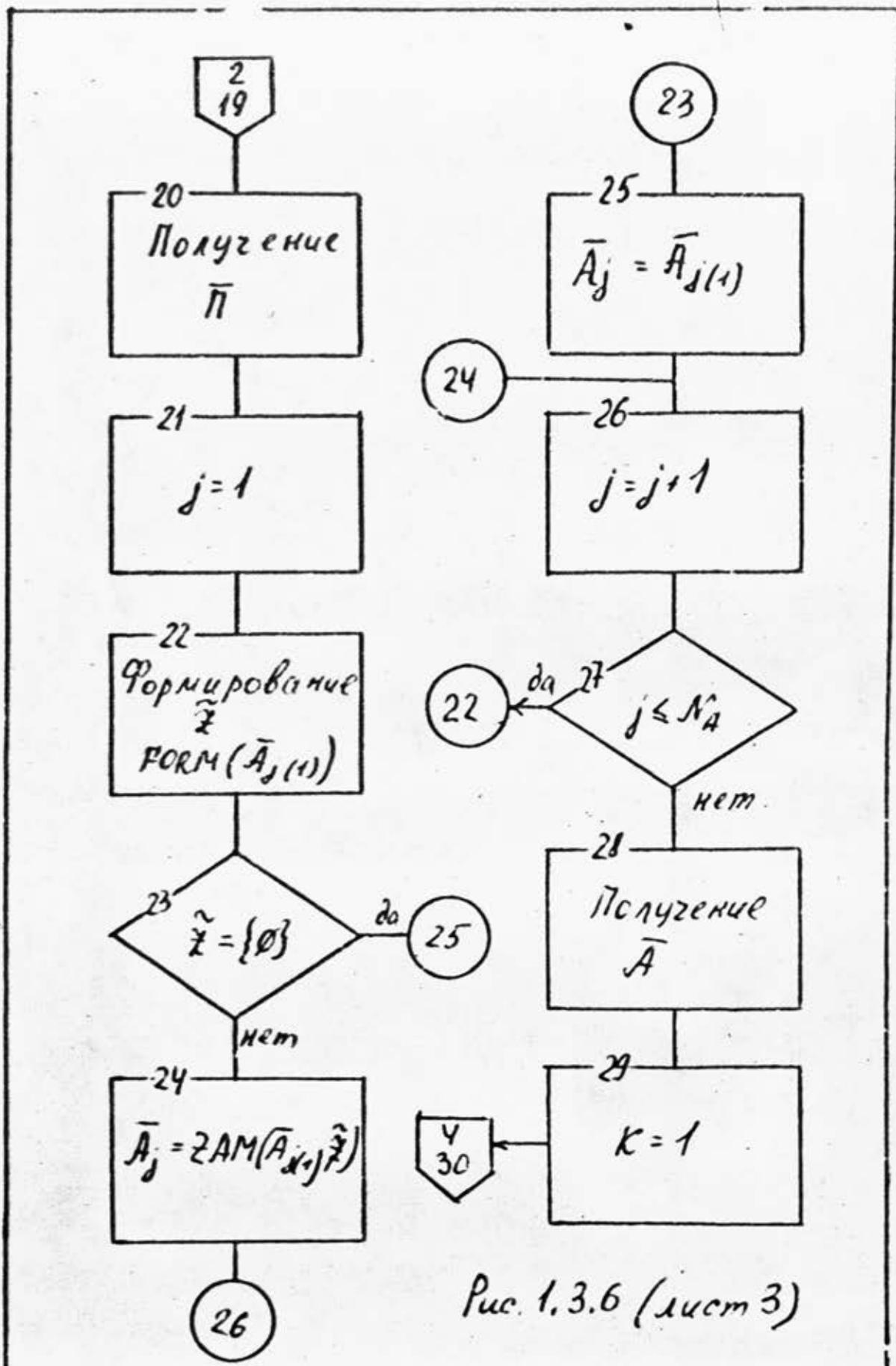


Рис. 1.3.6 (лист 3)

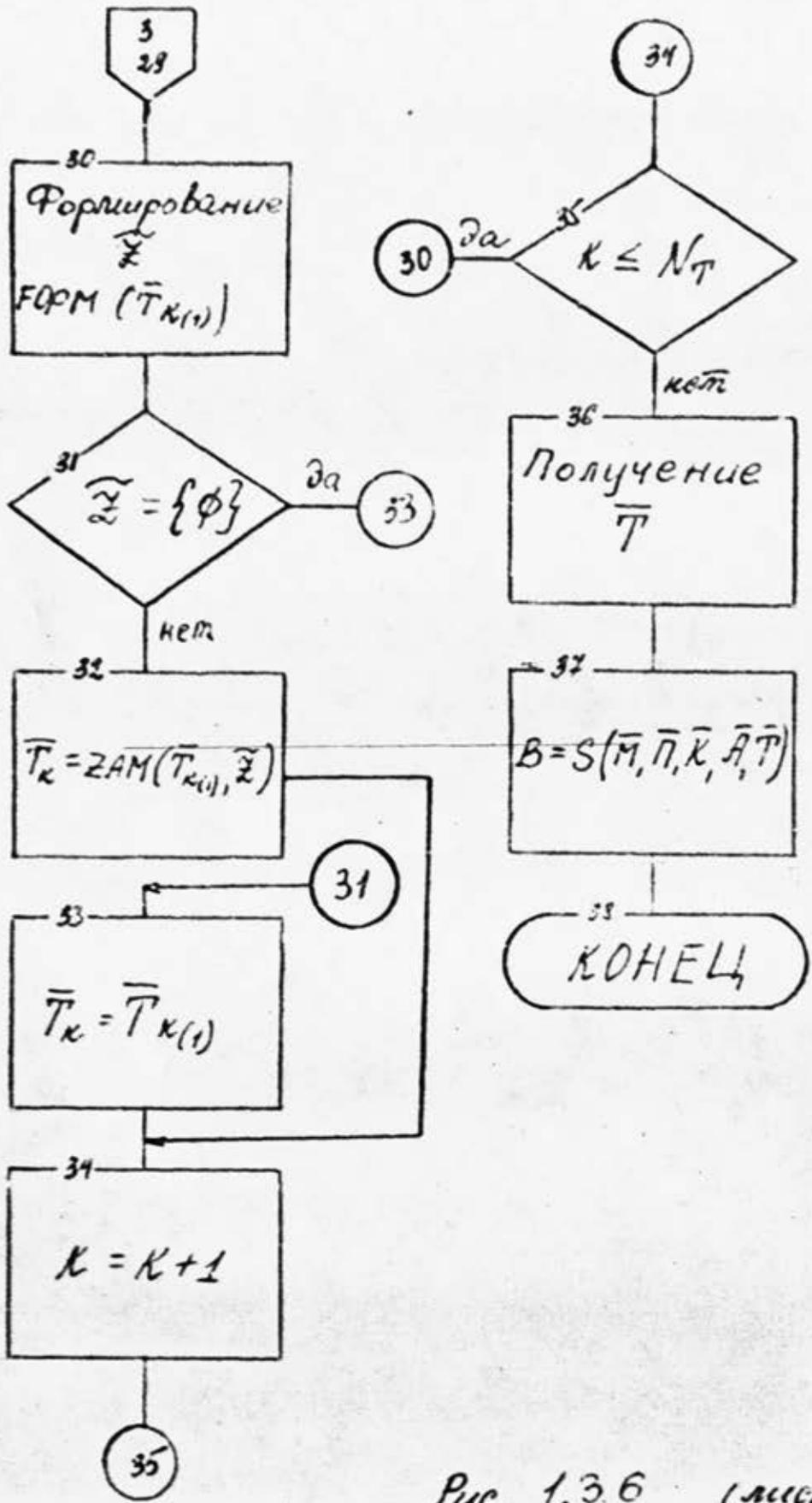


Рис. 1.3.6 (лист 4)

15-2-86  
ТПТЗм1

Бл.3. Получение отождествляющего отображения  $\sigma_1$ .

$$\sigma_1^{\text{скалат}} = \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} n_1 \\ n_1 \end{pmatrix} \right\}$$

Бл.5. Получение элемента  $B$  записи рода структуры результата операции в случае если  $N_A^1 = 0$

Бл.6. Получение  $w|R (B_i)$

Функция  $w|R$  выделяет из элемента  $B_i$  записи рода структуры элементы  $\bar{M}(i)$ ,  $\bar{K}(i)$ ,  $\bar{P}(i)$ ,  $\bar{A}(i)$ ,  $\bar{T}(i)$ .

Бл.7. Получение значения элемента  $\bar{M}$  результата операции сцеплением

$$\bar{M} = 'B(\overset{x}{\bar{M}(i)} \overset{y}{})'$$

Бл.8. Блок соответствует работе подпрограммы расщепления элементов  $\bar{K}(i)$ ,  $\bar{P}(i)$ ,  $\bar{A}(i)$ ,  $\bar{T}(i)$  (функция  $WIRA$  (выражение)).

Исходные данные для подпрограммы:

- выражение ( $\bar{K}(i)$  или  $\bar{P}(i)$  или  $\bar{A}(i)$  или  $\bar{T}(i)$ );
- число элементов в выражении ( $N_n^1$  или  $N_A^1$  или  $N_T^1$ ).

В результате работы подпрограммы выражение

$$\begin{array}{ll} \bar{K}(i) & \text{расщепляется на } \bar{K}_{i(l)} \quad (l = \overline{1, N_n^1}); \\ \bar{P}(i) & \text{на } \bar{P}_{i(n)} \quad (i = \overline{1, N_n}); \\ \bar{A}(i) & \text{на } \bar{A}_{j(n)} \quad (j = \overline{1, N_A}); \\ \bar{T}(i) & \text{на } \bar{T}_{\kappa(n)} \quad (\kappa = \overline{1, N_T}). \end{array}$$

Бл.9. Получение значения элемента  $\bar{K}$  результата операции

$$\bar{K} = \bar{K}_1 \overset{x}{z} \overset{y}{x} \bar{K}_2 \overset{x}{x} \dots \overset{x}{z} \overset{y}{x} \bar{K}_{N_n},$$

где

$$\bar{K}_i = 'B(\overset{x}{\bar{K}_{i(l)}} \overset{y}{})' \quad (i = \overline{1, N_n})$$

Бл.11. Начальное формирование элемента  $\bar{z}$ .

Бл.13,20,30. Формирование множества  $\bar{z}$  с помощью подпрограммы  $FORM$  (выражение).

Исходные данные для подпрограммы:

- выражение ( $\bar{P}_{i(n)}$  или  $\bar{A}_{j(n)}$  или  $\bar{T}_{\kappa(n)}$ );
- множество  $\bar{z}$ .

В результате работы подпрограммы формируется множество  $\bar{z}$  следующим образом:

$$\begin{array}{ll} 0 \in \bar{z} & \text{, если в выражении встречается имя } A; \\ \bar{z} \in \bar{z} & \text{, если } \bar{z} \in \bar{z} \text{ и в выражении встречается} \end{array}$$

имя  $P_z$ .

Бл.17,25,33. Получение элементов  $\bar{P}_i$ ,  $\bar{A}_j$ ,  $\bar{T}_\kappa$  по формулам:

$$\bar{P}_i = \bar{P}_{i(n)},$$

$$\bar{A}_j = \bar{A}_{j(i)};$$

$$\bar{T}_\kappa = \bar{T}_{\kappa(i)};$$

Бл.15,24,32. Получение элементов  $\bar{p}_i$ ,  $\bar{A}_j$ ,  $\bar{T}_\kappa$  с помощью специальной подпрограммы ЗАМ (выражение,  $\bar{z}$ ).

Исходные данные для подпрограммы:

- выражение ( $\bar{p}_{i(i)}$  или  $\bar{A}_{j(i)}$  или  $\bar{T}_{\kappa(i)}$ );
- множество индексов  $\bar{z}$ ,  $\bar{z} = \{l_1, \dots, l_p\}$

В результате работы подпрограммы:

1) в выражении  $\bar{p}_{i(i)}$  происходят замены (аналогично в  $\bar{A}_{j(i)}$  и  $\bar{T}_{\kappa(i)}$ ):

а)  $p_{lm}$  заменяется на  $y_{lm}$ , если  $p_{lm} \in \bar{p}_{i(i)}$  и  $l_m \in \bar{z}$ ;

б)  $d$  заменяется на  $y_0$ , если  $d \in \bar{p}_{i(i)}$

2) элемент  $\bar{p}_i$ ,  $\bar{A}_j$  или  $\bar{T}_\kappa$  получается по формулам:

$$\bar{p}_i = \bar{p}_{i(i)} \wedge' / y_{l_1} \in p_{l_1}, \dots, y_{l_p} \in p_{l_p}, y_0 \in d'$$

$$\bar{A}_j = \wedge' y_{l_1} \in p_{l_1}, \dots, \wedge' y_{l_p} \in p_{l_p}, \wedge' y_0 \in d : \wedge' \bar{A}_{j(i)}$$

$$\bar{T}_\kappa = \wedge' y_{l_1} \in p_{l_1}, \dots, \wedge' y_{l_p} \in p_{l_p}, y_0 \in d \wedge' \bar{T}_{\kappa(i)}$$

где  $z = \{l_1, \dots, l_p\}$  - номера конститuent класса  $\Pi$ , входящих в выражении, в которых производились замены (1а или 1б).

Часть ' $y_0 \in d$ ' записывается в случае, если производилась замена 1б.

Бл.17,25,33. Получение элементов  $\bar{p}_i$ ,  $\bar{A}_j$ ,  $\bar{T}_\kappa$  по формулам:

$$\bar{p}_i = \bar{p}_{i(i)};$$

$$\bar{A}_j = \bar{A}_{j(i)};$$

$$\bar{T}_\kappa = \bar{T}_{\kappa(i)};$$

Бл.16. Формирование нового элемента множества индексов конститuent класса  $\Pi$ , в которых производились замены ( $z$ ) в случае, если  $\bar{z} \neq \{\emptyset\}$ :

$$i \in z \quad (i - \text{индекс конститuentы класса } \Pi).$$

Бл.20,28,36. Получение значения элементов  $\bar{p}$ ,  $\bar{A}$ ,  $\bar{T}$  результата операции по формулам:

$$\bar{p} = \bar{p}_1 \wedge' z' \wedge' \bar{p}_2 \wedge' \dots \wedge' z' \wedge' \bar{p}_{n_p};$$

$$\bar{A} = \bar{A}_1 \wedge' z' \wedge' \bar{A}_2 \wedge' \dots \wedge' z' \wedge' \bar{A}_{n_p};$$

$$\bar{T} = \bar{T}_1 \wedge' z' \wedge' \bar{T}_2 \wedge' \dots \wedge' z' \wedge' \bar{T}_{n_p}$$

Бл.37. Получение значения элемента  $B$  записи рода структуры результата операции с помощью специальной функции  $\delta$  - сцеплением элементов  $\bar{M}$ ,  $\bar{p}$ ,  $\bar{K}$ ,  $\bar{A}$ ,  $\bar{T}$ .

15.2.26

ТПТЗМ1

#### 1.4. Характеристики информационных массивов ППП ГРС и связь с другими ППП.

Пакет прикладных программ формирования главного рода структур использует следующие массивы банка данных:

- 1) массив ОПСХ-массив, задавший операционную схему;
- 2) массив РСД- массив, содержащий роды структур (базисные, промежуточные, главный) и дополнения, привязанные к вершинам графа операционной схемы;
- 3) массив ОТОТ- массив, содержащий отсутствующие отображения, привязанные к вершинам графа операционной схемы;
- 4) массив ТИИТ- массив, используемый при Т- и антиинтерпретации конституэнт;
- 5) массив КОИС- каталог массива ОПСХ;.

Ниже будет дано описание каждого из этих массивов.

##### 1.4.1. Массив ОПСХ.

###### 1) Имя.

Выбор имени массива ОПСХ, употребляемого в программах, ограничивается единственным требованием: это имя должно содержать идентификаторы проекта и варианта. Тем самым в БД должна быть предусмотрена возможность хранения нескольких грсектов и нескольких вариантов по каждому варианту.

###### 2) Информация, содержащаяся в записи массива.

В массиве содержатся записи двух типов.

Каждая запись 1 типа массива ОПСХ соответствует одной вершине операционной схеме. В ней содержится информация о том, какая операция соответствует данной вершине, а также указаны номера вершин, соответствующих аргументам операции. Каждая запись 2 типа содержит информацию о номере слоя, в котором находятся все следующие за ней записи 1-го типа.

###### 3) Организация массива.

Массив ОПСХ имеет записи 2-х типов: запись 1-го типа- переменной длины, запись 2-го типа- фиксированной длины.

###### 4) Метод доступа- индексно-последовательный или прямой.

###### 5) Максимальный размер- 300 записей.

###### 6) Структура записи.

Структура записи I-го типа массива ОПСХ показана на рис. I.4. I.

Признак записи I-го типа	Номер вершины	Код операции	Количество аргументов	Номер вершины I-го аргумента		Номер вершины последнего аргумента
--------------------------	---------------	--------------	-----------------------	------------------------------	--	------------------------------------

Рис. I.4. I. I

Структура записи 2-го типа показана на рис. I.4. I.2.

Признак записи 2-го типа	Номер слоя
--------------------------	------------

Рис. I.4. I.2.

#### 7) Получение массива ОПСХ.

Массив ОПСХ вносится в банк данных (БД) после следующих действий:

- ввода формы I в оперативную память;
- проверки правильности перфорации (контрольные суммы);
- проверки графа операционной схемы;
- записи (в случае отсутствия ошибок) информационной части формы I в БД в виде массива.

Все указанные действия производятся средствами БД.

#### 8) Использование массива ОПСХ в ППП ГРС.

массив ОПСХ используется как входной массив управляющей программой пакета при выборе операции над родами структур, которую нужно выполнить в первую очередь.

#### 9) Использование массива ОПСХ в других ППП.

Массив ОПСХ используется ППП внесения изменений при внесении изменений в операционную схему и в базовое замыкание операционной схемы, а также пакетом перевода представлений при T-и антиинтерпретации конститuent.

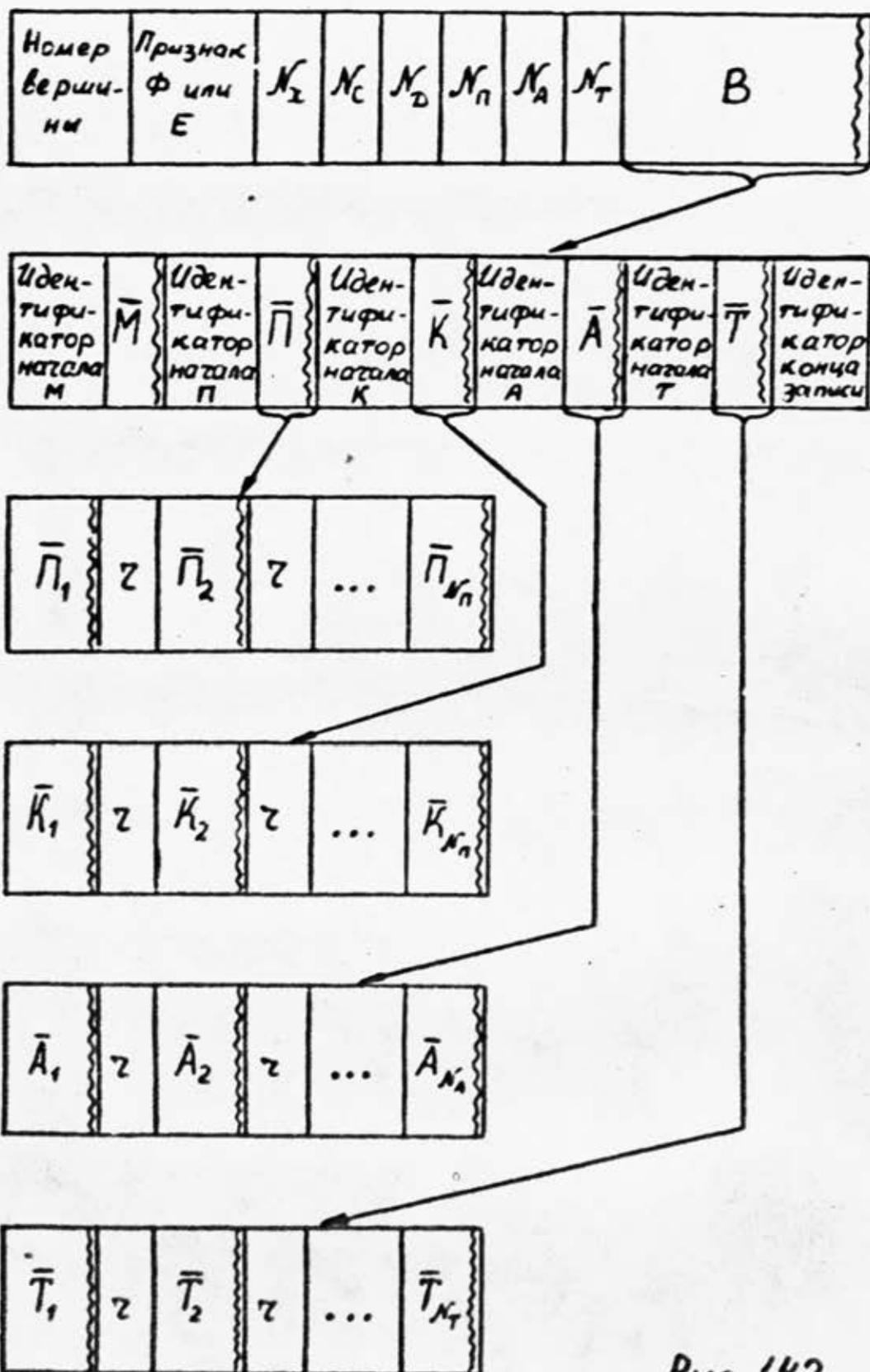


Рис 1.42.

#### 1.4.2. Массив РСД.

##### 1) Имя.

Имя массива РСД, употребляемое в программах, должно содержать идентификаторы проекта и варианта.

##### 2) Информация, содержащаяся в записи массива.

Каждая запись массива РСД соответствует роду структуры (или дополнению), привязанному к вершине графа операционной схемы. В записи содержатся номер вершины, отличительный признак (род структуры или дополнение), числа конститuent всех классов, а также выражения всех конститuent классов  $M, II, K, A, T$ .

##### 3) Организация массива.

Массив РСД имеет записи неопределенной длины.

##### 4) Метод доступа—прямой или индексно-последовательный.

##### 5) Максимальный размер— 200 записей.

##### 6) Структура записи.

Структура записи массива РСД показана на рис. 1.4.2.

В каждой записи массива выделены поля:

Номер вершины— номер вершины графа операционной схемы, соответствующий данному роду структур или дополнению (поле фиксированной длины);

Признак  $\Phi$  или  $E$  — признак, различающий роды структур и дополнения (поле фиксированной длины);

$N_K, N_C, N_{II}, N_A, N_T$  — числа конститuent классов  $K, C, II, A, T$  (поля фиксированной длины)<sup>x)</sup>;

$B$  — поле неопределенной длины, имеющее сложную структуру, показано на рис. 1.4.2 стрелкой 1.

##### Идентификаторы

начал  $M, II, K, A, T$  — идентификаторы начал выражений конститuent  $M, II, K, A, T$  (поля фиксированной длины).

Идентификатор конца записи— поле фиксированной длины.

$\bar{M}, \bar{II}, \bar{K}, \bar{A}, \bar{T}$  — выражения конститuent классов  $M, II, K, A, T$  (поля неопределенной длины);

$\bar{M}_c, \bar{K}_c, \bar{A}_c, \bar{T}_c$  — выражения конститuent  $M_c, K_c, A_c, T_c$  (поля неопределенной длины);

$\bar{2}$  — разделитель, служащий для разделения выражений конститuent одного класса (поле фиксированной длины).

<sup>x)</sup> Для дополнений всегда выполнены равенства  $N_C=0, N_A=0$ .

Замечания:

А) В поле "В" не входят поля:

а) "Идентификатор начала М" и " $\bar{M}$ ", если  $N_D = 0$

б) "Идентификатор начала П", "Идентификатор начала К", " $\bar{P}$ " и " $\bar{K}$ ", если  $N_L = 0$

в) "Идентификатор начала А", и " $\bar{A}$ ", если  $N_A = 0$

г) "Идентификатор начала Т" и " $\bar{T}$ ", если  $N_T = 0$ .

Б) В каждом из полей " $\bar{P}$ ", " $\bar{K}$ ", " $\bar{A}$ ", " $\bar{T}$ " содержится, соответственно,  $N_P$ ,  $N_K$ ,  $N_A$ ,  $N_T - 1$  разделителей "2".

7) Получение массива РСД.

Массив РСД вносится в БД после проверки и ввода в БД отождествляющего отображения  $\eta$ . При этом массив РСД образуется из тех записей массивов БРС (массив базисных родов структур, снабженных базисными индексами) и БДП (массив дополнений, снабженных базисными индексами), которые соответствуют указанным в  $\eta$  базисным родам структур и дополнениям.

Все указанные действия (проверка и ввод БД отождествляющего отображения  $\eta$ , перенос записей из БРС и БДП в РСД) производится средствами БД.

8) Использование массива РСД в ППП ГРС.

Массив РСД является для пакета как входным так и выходным. С одной стороны записи массива РСД используются модулями тела пакета в качестве аргументов операций, а с другой стороны по мере выполнения операций происходит пополнение массива РСД результатами выполненных операций.

9) Использование массива РСД в других ППП.

Массив РСД используется пакетами:

R - интерпретации (при получении R - интерпретации главного рода структуры);

- перевода представлений (при переводе конститuent в представление  $(d_2 d_3 d_4 d_5 d_6)^X$ )

- внесения изменений (при внесении изменений в базовые рода структур и дополнения и отображение  $\eta$ )

Г. 4. У. массив ОТ0Т.

Г) Имя.

Имя массива ОТ0Т, употребляемое в программах, должно со-

X) подробнее о представлениях конститuent см. п. 4. Г. данного документа.

держат идентификаторы проекта и варианта.

2) Информация, содержащаяся в записи массива.

Каждая запись массива ОТОТ соответствует отождествляющему отображению, снабженному сквозным индексом, т.е. привязанному к вершине графа операционной схемы. В записи содержится номер вершины, число имен конститuent в отождествляющем отображении, а также последовательность этих имен.

3) Организация массива.

Массив ОТОТ имеет записи переменной длины.

4) Метод доступа - прямой или индексно-последовательный.

5) Максимальный размер - 50 записей.

6) Структура записи.

Структура записи массива ОТОТ показана на рис. I.4.3.

Номер вершины	$N(\Gamma)$	Первое имя	Второе имя	...	Послед- нее имя
------------------	-------------	---------------	---------------	-----	--------------------

Рис. I.4.3.

В каждой записи массива выделены следующие поля (каждое из которых является полем фиксированной длины):

Номер вершины - номер вершины графа операционной схемы, соответствующий данному отождествляющему отображению;

$N(\Gamma)$  - число имен конститuent в отождествляющем отображении;

$i$ -тое имя - стандартное имя конститuent одного из классов  $1 \leq i \leq N(\Gamma)$  сов X, C, Д, П.

7) Получение массива ОТОТ.

Массив ОТОТ как и массив РСД образуется после ввода в БД отождествляющего отображения  $\eta$ . При этом массив ОТОТ составляется из записей массива БОТ (массив отождествляющих отображений, снабженных базисными индексами), соответствующих указанным в  $\eta$  отождествляющим отображениям.

8) Использование массива ОТОТ в ППП ГРС.

Массив ОТОТ является входным для пакета ГРС. Его записи используются модулями тела пакета в качестве аргументов операций.

9) Использование массива ОТОТ в других ППП.

Массив используется IIII внесения изменений при внесении изменений в отождествляющие отображения.

#### 1.4.4. Массив ТИИТ.

##### 1. Имя.

Имя массива ТИИТ, употребляемое в программах, должно содержать идентификаторы проекта и варианта.

##### 2. Информация, содержащаяся в записях массива.

Запись массива задает отображение  $\beta_i$  конститuent  $i$ -того основного аргумента (род структуры, дополнение) в конститuentу результата этой операции. Таким образом, каждой (отличной от начальной) вершине графа операционной схемы соответствует  $\ell$  записей массива ТИИТ, где  $\ell$  - число основных аргументов соответствующей операции.

Отображение  $\beta_i$  есть последовательность пар неотрицательных чисел:

$$(\alpha_1), (\alpha_2), \dots, (\alpha_{\lambda_i}),$$

удовлетворяющих условиям:

а)  $\alpha_1 = 1 < \alpha_2 < \dots < \alpha_{\lambda_i} = n_i$

б)  $0 \leq \beta_j \leq n_j, \quad j = 1, \dots, \lambda_i$

в) если  $\alpha_{j+1} - \alpha_j > 1$ , то  $\beta_{j+1} - \beta_j$  делится на  $\alpha_{j+1} - \alpha_j$

Для раскодировки отображений  $\beta_i$  введем функцию  $f$  - действующую из множества конститuent рода структуры (или дополнения) во множества натуральных чисел по следующему правилу:

$$\begin{aligned} f(q) &= q, \quad f(cq) = n_c + q, \quad f(A) = n_c + N_A \\ f(M) &= n_c + 2N_A, \quad \text{если } N_A = 1; \quad f(nq) = n_m + q, \\ f(Kq) &= n_n + q, \quad f(Aq) = n_k + q, \quad f(\pi q) = n_\pi + q. \end{aligned}$$

Пусть  $K$  - какая-либо конститuent  $i$ -того аргумента операции,  $f_i$  - функция  $f$  для  $i$ -того аргумента.

Могут возникнуть 2 случая:

а) для векторного  $j$  ( $1 \leq j \leq \lambda_i$ )  $\alpha_j = f_i(K)$ ,

б) для некоторого  $j$  ( $1 \leq j < \lambda_i$ )  
 $\alpha_j < f_i(K) < \alpha_{j+1}$ .

Тогда отображение  $\beta_i$  задает отображение конститuent:

$$K \rightarrow \begin{cases} f_i^{-1}(\beta_j), & \text{в случае а)} \\ f_i^{-1}[(f_i(K) - \alpha_j) \frac{\beta_{j+1} - \beta_j}{\alpha_{j+1} - \alpha_j} + \beta_j], & \text{в случае б)} \end{cases}$$

Замечается: 1) Если числа  $\beta_j$  в случае а) и

$$(f_i(K) - \alpha_j) \frac{\beta_{j+1} - \beta_j}{\alpha_{j+1} - \alpha_j} + \beta_j \quad \text{- в случае б) равны нулю,$$

15-2-76

ТПТЗКМ



5) Максимальный размер- 200 записей.

6) Структура записи.

Структура записи массива КОПС показана на рис. I.4.5.

Номер вершины	Признак Ф, Е или Г
------------------	-----------------------

Рис. I.4.5.

7) Получение массива КОПС.

Массив КОПС образуется при вводе в БД массива ОПСХ. В этот момент 2-ые поля всех записей содержат пробелы. Затем по мере ввода родов структур и дополнений в массив РСД, а также отождествляющих отображений- в массив ОТОТ, эти поля заполняются соответствующими символами. (Ф, Е или Г).

8) Использование массива КОПС в ППП ГРС.

Массив используется УПП для выбора операции, которую можно выполнить в первую очередь.

9) Использование массива КОПС в других ППП.

Массив используется ППП внесения изменений, при внесении изменений в массив ОПСХ, РСД, ОТОТ.

15-2-86

ТПТЗки1

### 1.5. Ограничения и возможности расширения ППП.

При программировании ППП ГРС следует ориентироваться на следующие ограничения:

- 1) число вершин графа операционной схемы не больше 200, из них базовых родов структур не более 50, дополнений не более 50 и отождествляющих отображений не более 50;
- 2) число конституэнт базового рода структуры или дополнения изменяется в пределах 5-50, в среднем-10;
- 3) число конституэнт результата операции не более суммы чисел конституэнт аргументов операции плюс 5;
- 4) число конституэнт главного рода структуры не более 5000;
- 5) запись выражения одной конституэнты занимает не больше 1 к байта.

При создании технического проекта предусматривались следующие возможности расширения ППП:

- 1) введение новых операций над родами структур;
- 2) увеличение числа аргументов имеющих операций над родами структур;
- 3) увеличение числа вершин операционной схемы;
- 4) создание архивов базовых родов структур и дополнений;
- 5) увеличение числа конституэнт главного рода структуры;
- 6) введение новых программных модулей контроля, в частности, модулей контроля синтаксиса выражений.

Для 1) и 2) предусмотрен специальный вид формы 1, допускающий увеличения числа аргументов операции до 9.

Пункт 4) обеспечен созданием массивов БРС и РДП, которые можно пополнять постепенно.

Для 2) и 5) номерам вершин и номерам конституэнт предлагается отводить 4 знака.

Пункты 1) и 6) могут быть обеспечены применением в качестве управляющей программы ППП-ГРС универсального модуля автоматической R-интерпретации, допускающего простое введение новых модулей.

Программирование ППП ГРС следует организовать так, чтобы были легко осуществимы (без большого объема перепрограммирования) расширения пакета в указанных выше направлениях.

### 1.6. Средства программного контроля и контроля результатов.

Контроль информации, которой оперирует ППП ГРС, осуществляется в 2 этапа:

- 1) контроль форм 1, 2, 3 при вводе информации в БД;
- 2) контроль аргументов операций, исходящих из ограничений этих операций.

Первый этап контроля производится программными средствами СУБД<sup>x)</sup>. В него входит контроль перфорации, проверка графа операционной схемы, проверка правильности задания базисных родов структур, дополнений и отображений, проверка правильности задания отображения  $\eta$ , проверка правильности типов аргументов операций и т.д.

Второй этап контроля проводится постепенно по мере выполнения операций над родами структур, дополнениями и отождествляющими отображениями, согласно операционной схеме. Каждый модуль тела пакета содержит блок проверки аргументов операции<sup>xx)</sup>, определяющий, удовлетворяют ли эти аргументы ограничениям этих операций.

x) см. п. 6.2. данного документа.

xx) см. п. 1.3. данного документа.

## 2. R-интерпретация (III R-интерпретации)

### 2.1. Постановка задачи (содержание и метод решения).

Процедура R-интерпретации состоит в вычислении значений конститuent главного рода структур (ГРС) в терминах реальных объектов. Главный род структуры рассматривается как набор конститuent, переменные (конститuent) выражений которых связаны некоторыми соотношениями, как род структуры он описан в постановке задачи для III ГРС (п.1.1. настоящего документа).

В процедуре R-интерпретации можно выделить две основные части:

- 1) формирование вычислительной схемы в виде графа R-интерпретации;
- 2) организация вычислительного процесса по вычислительной схеме.

Формирование вычислительной схемы производится следующим образом. Из III ГРС поступает ГРС, по которому строится граф конститuent, т.е. ориентированный граф  $\langle \Sigma, A \rangle$  с множеством вершин  $\Sigma$  и множеством дуг  $A$ , где  $\Sigma = \{G_{2i}\}_{i=1}^n$  - <sup>ТЕКСТ</sup> ~~список~~ главного рода структуры,  $A = \{ \langle G_{2i}, G_{2j} \rangle \in \Sigma \times \Sigma \mid G_{2i} - \text{выражаемая конститuenta и } G_{2j} \text{ содержится в множестве имен конститuent, входящих в выражение для } G_{2j} \}$  и дуга ведет из  $G_{2i}$  в  $G_{2j}$ , если  $\langle G_{2i}, G_{2j} \rangle \in A$ .

Из графа конститuent по информации, предоставляемой проектировщиком, выделяется подграф. В качестве такой информации может выступать:

- 1) список конститuent, которые подлежат R-интерпретации (список R Int), т.е. тех конститuent, которые должны попасть в проект и составляют абстрактное содержание проекта, *(как правило, высшего уровня)*
- 2) список конститuent, вычисление которых берет на себя проектировщик (эти конститuent называются конститuentaми, подлежащими специальной R-интерпретации) и той части конститuent <sup>выс</sup> выражений для них, которая необходима проектировщику для вычисления. Эти конститuent называются разрешающими терминами.

При получении первого списка из графа выделяется подграф, содержащий все вершины, предшествующие в смысле введенной на

15-2-36  
ТПТЗм1

*всех уровней, а не только разрешающие!*

графе ориентации вершинам списка (этот подграф называется нижним замыканием списка  $R_{Int}$ ). Множество конститuent этого подграфа называется <sup>расширенным</sup> релевантным множеством и обозначается  $REL$ . При получении второго списка (списка  $S_{Int}$ ) из графа  $REL$  удаляются ребра, отвечающие связям между конститuentами специальной  $R$ -интерпретации и их аргументами, не являющимися разрешающими терминами, а также низлежащие подграфы, если они не используются для вычисления других конститuent (которые должны остаться в графе) или не входят в абстрактное содержание проекта.

Затем проектировщик может задать список конститuent, которым по соответствующим им выражениям в ГРС должны быть сопоставлены графы, учитывающие внутреннюю структуру (т.е. аргументы и "элементарные" операции) этих выражений в отличие от "укрупненных" представлений этих выражений в графе конститuent; <sup>этим</sup> "детализированными" графами заменяются соответствующие "укрупненные" графы (конститuent).

После этого проектировщик закрепляет за некоторыми конститuentами программные модули из некоторой исходной библиотеки модулей, которые по значениям аргументов выражений этих конститuent могут вычислить значения самих конститuent.

Получаемый после всей описанной процедуры граф вместе с частичными отображениями конститuent в множество модулей программного обеспечения и множество конститuent специальной

$R$ -интерпретации называется графом  $R$ -интерпретации, он определяет вычислительную схему процесса  $R$ -интерпретации после задания значений начальных вершин. Таким образом, конститuent графа  $R$ -интерпретации могут быть трех типов:

- 1) конститuent, вычисляемые проектировщиком;
- 2) конститuent, вычисляемые некоторым модулем из библиотеки модулей;
- 3) остальные конститuent.

Множество конститuent графа  $R$ -интерпретации называется релевантным множеством графа  $R$ -интерпретации и обозначается  ~~$R_{Int}$~~ .  $Rel$  ?

Для организации вычислительного процесса по вычислительной схеме проектировщик сначала задает значения начальных

15-2-76  
ТПЗкн1

несколько →  
→ входить в  $S_{Int}$ !

?  
здесь  
возможно  
и укрупнен-  
ные !!

вершин (конституэнт) графа R -интерпретации, после чего машина анализирует подграфы, соответствующие выражениям, аргументы которых вычислены. Если верхней вершине такого подграфа соответствует имеющийся в системе модуль, то управление передается этому модулю, в противном случае формируются два списка: 1) список конституэнт, вычисляемых проектировщиком; 2) список конституэнт, не вычисляемых ни имеющимся программным обеспечением, ни проектировщиком.

После вычисления проектировщиком конституэнт I-го списка (эта процедура называется специнтерпретацией) и пополнения программного обеспечения модулями, соответствующими конституэнтам 2-го списка начинается следующий цикл и т.д., пока не будут проинтерпретированы все вершины графа.

При выполнении R -интерпретации можно выделить следующие функции:

1. Формирование релевантного множества REL. Для выполнения этой функции задается ГРС и запрос на абстрактное содержание проекта, в результате получается расширенное релевантное множество REL.

2. Формирование укрупненной вычислительной схемы. Для выполнения этой функции задается список конституэнт специальной R -интерпретации, в результате получается укрупненная вычислительная схема.

3. Формирование детализированной вычислительной схемы. Для выполнения этой функции задается список конституэнт, подлежащих детализации, в результате получается детализированная вычислительная схема.

4. Формирование списка конституэнт I-й очереди. Для выполнения этой функции задается список модулей автоматической R -интерпретации, в результате получается список конституэнт I-й очереди.

5. Выполнение очередного шага автоматической R -интерпретации. Для выполнения этой функции задается R -интерпретация конституэнт, подлежащих специальной R -интерпретации и пополняется необходимыми модулями библиотека модулей автоматической R -интерпретации, в результате получается список конституэнт следующего шага, подлежащих специальной R -интерпретации, список конституэнт, не вычисляемых ни проекти-

ровщиком, на библиотекой модулей и R-интерпретация конститут, вычисленных на данном шаге.

6. Контроль R-интерпретации.

## 2.2. Алгоритм управления функционированием ППП.

В 2.2.1. описывается процесс управления R-интерпретацией, в 2.2.2. и 2.2.3. приводится описание алгоритма модуля организации вычислительного процесса и алгоритма модуля автоматической R-интерпретации, составляющих управляющую программу ППП R-интерпретации.

### 2.2.1. Описание процесса управления R-интерпретацией.

Для управления процессом R-интерпретации предназначена УПП R-интерпретации, состоящая из управляющего модуля-модуля организации вычислительного процесса по вычислительной схеме (МОВП) и модуля автоматической R-интерпретации.

Вычислительной схемой называется ориентированный граф без циклов и петель, для начальных вершин которого задано частичное отображение в множества имен "алгоритмов", причем часть "алгоритмов" реализуется модулями библиотеки модулей, а часть "алгоритмов" - внешними средствами (проектировщиком).

Процесс R-интерпретации распадается на две части:

- а) формирование вычислительной схемы;
- б) организация вычисления по вычислительной схеме;

В предлагаемой реализации формирование вычислительной схемы осуществляется в результате вычисления по некоторой другой вычислительной схеме, задающей переход от главного рода структуры, полученного в ППП ГРС, к графу R-интерпретации, определяющему вычислительную схему для собственного процесса R-интерпретации.

При этом в качестве модулей библиотеки используются модули, алгоритмы которых приводятся в 2.3.1.-2.3.8. Например, можно задать стандартную вычислительную схему, приведенную на рис. 2.2.1.



Рис. 2.2.1.

где начальным вершинам соответствуют:

- 1 - ГРС
- 2 - RJnt
- 3 - SJnt
- 4 - SAп.

а промежуточным вершинам приписаны модули:

- 5 - модуль формирования графа конститuent<sup>x)</sup>
- 6 - модуль формирования нижнего замыкания<sup>x)</sup>
- 7 - модуль удаления подграфа<sup>x)</sup>

8 - модуль управления детализацией графа<sup>x)</sup>  
 (это макромодуль, использующий модуль формирования графа по формуле<sup>x)</sup>)

9 - модуль приведения графа к расслоенному виду<sup>x)</sup>

10 - модуль удаления дублирующихся подграфов графа<sup>x)</sup>.

Получив такую вычислительную схему в качестве задания (в соответствующем представлении), модуль организации вычислительного процесса по вычислительной схеме после оснащения его модулями СУБД для ввода и вывода необходимой информации и библиотекой модулей, состоящей из модулей, приписанных вершинам представляет управляющую программу для формирования вычислительной схемы по ГРС и построения графа R-интерпретации. Кроме приведенной стандартной вычислительной схемы возможны и другие вычислительные схемы, задаваемые проектировщиком для построения графа R-интерпретации.

Графом R-интерпретации (в отличие от других графов конститuent<sup>xx)</sup>, возникающих в процессе формирования вычислительной схемы) будем называть граф, поступающий на вход модуля организации вычислительного процесса по вычислительной схеме и его модификации при организации вычислительного процесса R-интерпретации.

Модуль организации вычислительного процесса по вычислительной схеме (МОВП) получает через главную управляющую программу идентификатор вычислительной схемы и идентификатор библиотеки модулей.

Вычислительная схема (граф R-интерпретации) должна быть предварительно средствами СУБД введена в БД и приведена

x) Описание алгоритма модуля приведено в 2.3.

xx) Описание представления приводится в 2.3.

16-2-96

ТПТЗм1

к расслоенному представлению<sup>х)</sup>, средствами операционной системы вводятся модули библиотеки модулей.

В вычислительной схеме выделяются начальные вершины, специальная  $R$  - интерпретация всех или некоторых начальных вершин дается проектировщиком и вводится средствами СУБД, эти средства должны быть либо достаточно универсальными, либо могут быть специальными при решении конкретных классов задач.

В специальных каталогах фиксируется список проинтерпретированных вершин с адресами их значений и сами значения, причем при решении конкретных задач значения  $R$  - интерпретации могут находиться в нескольких массивах различной структуры.

Затем управление передается модуль автоматической  $R$  - интерпретации для выполнения  $R$  - интерпретации тех конститuent, которые могут быть вычислены по уже имеющимся значениям модулями библиотеки и формирования списков тех конститuent-аргументы которых уже проинтерпретированы<sup>мо</sup> которые не могут быть вычислены либо поскольку их интерпретация должна быть выполнена проектировщиком, либо поскольку отсутствует вычисляющая их значения модуль. Одновременно в графе  $R$  - интерпретации отмечаются подграфы, верхние вершины которых уже получили значения.

После пополнения модулями библиотеки и выполнения проектировщиком специальной  $R$  - интерпретации всех или части выделенных конститuent, производится ввод соответствующих данных, пополнение каталогов и удаление из графа  $R$  - интерпретации подграфов, верхние вершины которых проинтерпретированы. Затем МОВП проверяет, остались ли подграфы в графе  $R$  - интерпретации. Если подграфы остались, то МОВП снова вызывает модуль автоматической  $R$  - интерпретации, при условии, что перед этим была произведена специинтерпретация или пополнена библиотека модулей. Если же специинтерпретация не производилась и библиотека модулей не пополнялась, то работа прекращается.

Часть процесса  $R$  - интерпретации ( и результаты, полученные при этом), которая начинается после  $n$ -го вызова модуля

<sup>х)</sup> Здесь под графом конститuent понимается произвольный граф связей конститuent, а не только соответствующий ГРС.

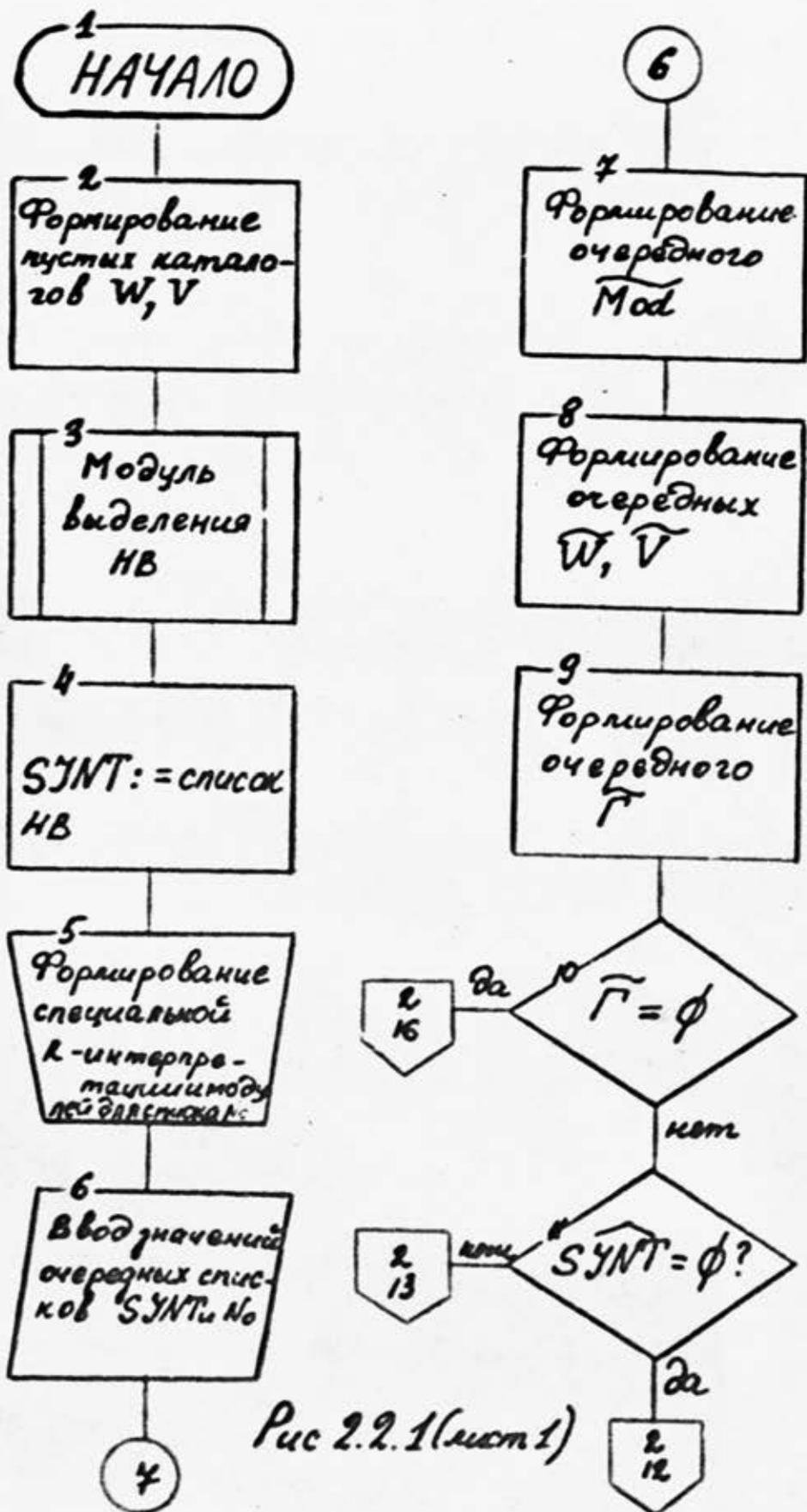


Рис 2.2.1 (лист 1)

15-2-16  
Т/ПТЗ кн1

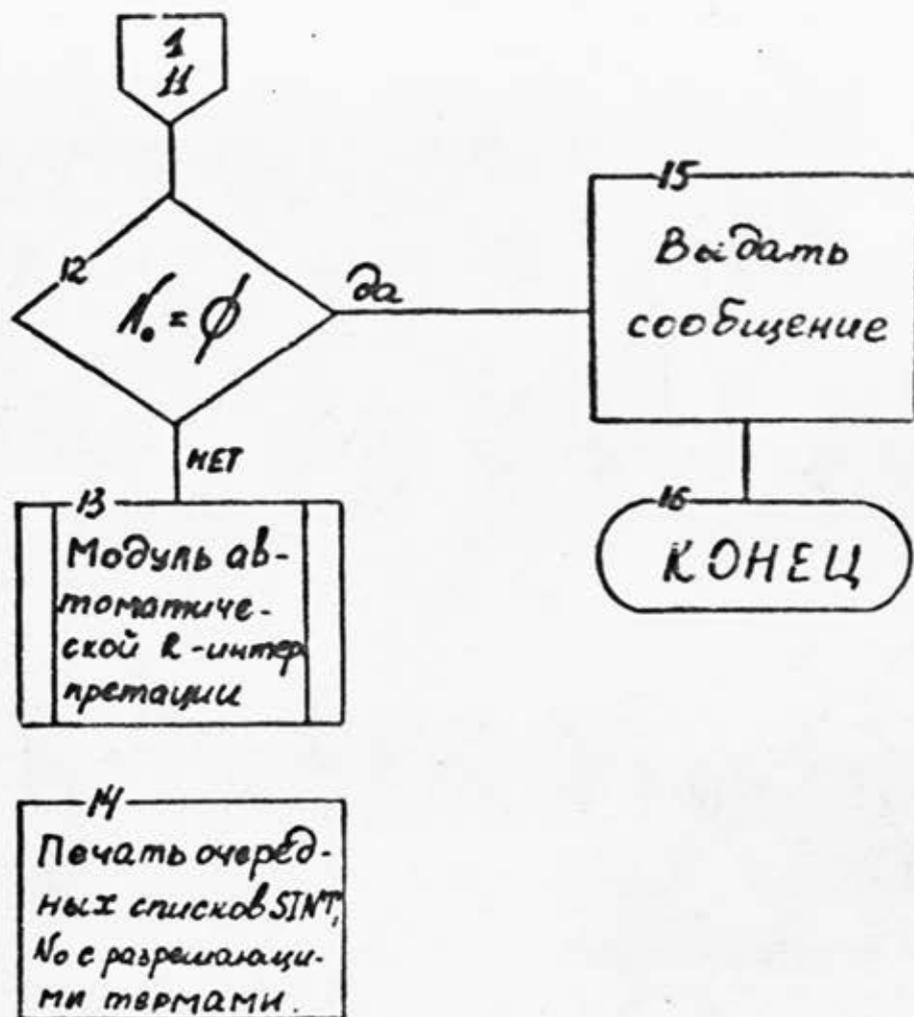


Рис. 2.2.1. (лист 2)

автоматической  $R$ -интерпретации и заканчивается новым вызовом модуля автоматической  $R$ -интерпретации называется  $n$ -ой <sup>очередью</sup> интерпретации, а соответствующие результаты называются результатами  $n$ -й очереди  $R$ -интерпретации.

Если же при проверке, остались ли подграфы в графе  $R$ -интерпретации выясняется, что все подграфы обработаны, то процесс  $R$ -интерпретации завершен.

### 2.2.2. Описание модуля организации вычислительного процесса по вычислительной схеме.

Модуль предназначен для организации обработки <sup>вычислительной</sup> схемы.

Исходные данные: 1-й аргумент - идентификатор вычислительной схемы, задаваемый в виде графа  $R$ -интерпретации (массив  $ГК^x$ ) или ОПСХ<sup>xx</sup>) в <sup>расслоенном</sup> <sup>представлении</sup>  $xxx$ )

2-ой аргумент - идентификатор библиотеки модулей.

В исходных данных должно быть проконтролировано:

1) наличие в библиотеке модулей программ, идентификаторы которых записаны в подграфах вычислительной схемы, если в библиотеке нет соответствующей программы, то она должна быть либо пополнена, либо соответствующий идентификатор должен быть удален из подграфа;

2) расслоенное представление вычислительной схемы, если представление не расслоенное, то вызывается модуль приведения графа к расслоенному виду из СУБД.

В блок-схеме приняты следующие обозначения:

$G$  - граф  $R$ -интерпретации

$W$  - каталог адресов значений конституэнт, получивших  $R$ -интерпретацию

$V$  - каталог значений конституэнт, получивших  $R$ -интерпретацию

$\tilde{W}, \tilde{V}$  - соответствующие каталоги после пополнения

$SInt$  - список данной очереди автоматической  $R$ -интерпретации конституэнт, подлежащих  $R$ -интерпретации

$N_0$  - список конституэнт данной очереди

$R$ -интерпретации, которым не соответствует модули из библиотеки модулей

x) Описание массива приводится в 2.4.

xx) Описание массива приводится в 1.4.

xxx) Описание расслоенного представления ориентированного графа приводится в 2.3.

$S_{int}, N_0$  - список фактически специнтерпретированных вершин и новых модулей на данной очереди  $R$ -интерпретации.

$Mod$  - библиотека модулей автоматической  $R$ -интерпретации,

$F$  - граф  $R$ -интерпретации после удаления обработанных подграфов.

Блок-схема алгоритма представлена на рис. 2.2.1.

Опишем отдельные блоки алгоритма.

Блок 2. Открыть каталоги, эти каталоги соответствуют массивам КРВР, ЗНАЧР соответственно <sup>x/</sup>.

Блок 3. Передача графа  $R$ -интерпретации модулю формирования списка начальных вершин и всех вершин графа<sup>xx/</sup> в результате получается список начальных вершин графа.

Блок 4. Списку  $S_{int}$  /соответствующий массив - ЗАСМ<sup>x/</sup> в качестве начального значения присваивается список НВ графа.

Блок 5. Формирование проектировщиком специальной  $R$ -интерпретации и создание модулей, соответствующих конституэнтам списка  $N_0$  текущей очереди  $R$ -интерпретации.

Блоки 6-8. Ввод модулями СУБД значений специальной  $R$ -интерпретации в массиве ЗАСМ<sup>x/</sup> и идентификаторов модулей в массиве СМОД<sup>x/</sup>, пополнение библиотеки модулей и каталогов  $\tilde{W}, \tilde{V}$ .

Блок 9. Удаление из графа подграфов, обработанных модулями библиотеки автоматической  $R$ -интерпретации и в результате специнтерпретации на текущей очереди  $R$ -интерпретации.

Кроме того вызывается модуль операционного заполнения,<sup>xxx/</sup> который по графу и списку  $N_0$ , заполненному /соответствующий массив СМОД/ проектировщиком, приписывает ВВ подграфов идентификаторы соответствующих им модулей.

Блок 10. Проверка условия  $F = \emptyset$ , т.е. обработаны ли все подграфы графа.

Блоки 11-12, 15, 16. Проверка, было ли на текущей очереди  $R$ -интерпретации сформировано задание на специальную  $R$ -интерпретацию и пополнение библиотеки модулей, если

<sup>x/</sup> Описание информационного массива приводится в 2.4.

<sup>xx/</sup> Описание модуля приводится в 6.2.

<sup>xxx/</sup> Описание модуля приводится в 2.3.

не было<sup>x)</sup>, то выдается сообщение и работа прекращается.

Блок 13. Вызов модуля автоматической  $R$ -интерпретации,<sup>xx)</sup> модуль передается граф  $\tilde{\Gamma}$   $R$ -интерпретации данной очереди, каталоги  $\tilde{W}, \tilde{V}$  данной очереди и идентификатор библиотеки алгоритмов  $Mod$ , выходом модуля являются списки конститuent  $SJnt'$  и  $N_0'$  со значениями разрешающих термов, каталоги  $W, V$  следующей очереди.

Блок 14. Печать списков  $SJnt$  и  $N_0$  с разрешающими термами (массив  $ZASM^x$ ).

В модуле организации вычислительного процесса по вычислительной схеме можно ввести каталог  $S$ -каталог вершин, которым проектировщик по заданиям  $SJnt$  и  $N_0$  предшествующих очередей не составил значения специнтерпретации или вычисляемые модули. В блоке 2 дополнительно формируется пустой каталог  $S$  между блоками 6 и 7 вставляется блок:

$$S := S \cup (SJnt \setminus \hat{SJnt}) \cup (N_0 \setminus \hat{N}_0),$$

а после блока 15 вставляются блоки печати  $S$ , а затем по решению проектировщика, либо работа продолжается (переход на блок 5), либо прекращается (переход на блок 16).

Эти блоки выпечатывают все невыполненные проектировщиком задания (перед этим выполняется  $R$ -интерпретация всех тех вершин, которая возможна) и если проектировщик может проектировать некоторые из них, то он может продолжать работу (переход на блок 5), в противном случае работа завершается.

Каталог  $S$  позволяет работать модулю в случае неполной информации.

Результатом работы модуля является  $R$ -интерпретация конститuent графа вычислительной схемы, которую возможно проинтерпретировать при полученной информации от проектировщика и имеющейся библиотеке модулей.

<sup>x)</sup> Это означает, что вся  $R$ -интерпретация, которую можно выполнить при заданной информации, выполнена, но есть непроинтерпретированные конститuent.

<sup>xx)</sup> Описание модуля приводится в 2.2.3.

### 2.2.3. Описание модуля автоматической R-интерпретации.

Модуль предназначен для R-интерпретации тех конститuent, которые при имеющихся значениях конститuent могут быть вычислены при помощи модулей библиотеки модулей.

Исходные данные:

1-й аргумент - граф R-интерпретации - массив ГР<sup>\*</sup>);

2-й аргумент - каталог W имен конститuent, получивших R-интерпретацию - массив КРВГ<sup>\*</sup>);

3-й аргумент - массив V значения конститuent - массив ЗНАЧР;

4-й аргумент - идентификатор библиотеки модулей.

В блок-схеме приняты следующие обозначения:

SINT - список конститuent, которые должны получить специальную R-интерпретацию;

N<sub>0</sub> - список конститuent, для вычисления которых должны быть представлены новые модули;

n - номер слоя графа;

ПГ - разрешающие термины;

W<sup>(n-1)</sup> - список имен конститuent, получивших R-интерпретацию на предыдущих (n-1)-х слоях графа;

V<sup>(n-1)</sup> - список значения конститuent, получивших R-интерпретацию на предыдущих (n-1) слоях графа;

W<sup>(n)</sup>, V<sup>(n)</sup> - аналогично, W<sup>(n-1)</sup>, V<sup>(n-1)</sup>, но на n-м слое графа.

Блок-схема алгоритма представлена на Рис. 222.

Опишем отдельные блоки алгоритма.

Блок 1,2. Очистка каталогов SINT, N<sub>0</sub>. Этим каталогам соответствует массив ЗАСМ<sup>\*</sup>).

Блоки 4,17. Анализ окончания обработки графа.

Блок 5. Переход к очередному слою, анализ ведется снизу вверх.

Блок 6. Контроль завершения анализа ПГР.

Блок 7. Переход к очередному ПГР слоя, анализ ведется в том порядке, как ПГР записаны в слое.

Блок 8. Проверка, получили ли именные вершины ПГР значения при обработке предшествующих слоев графа.

\*) Описание массива приводится в 24

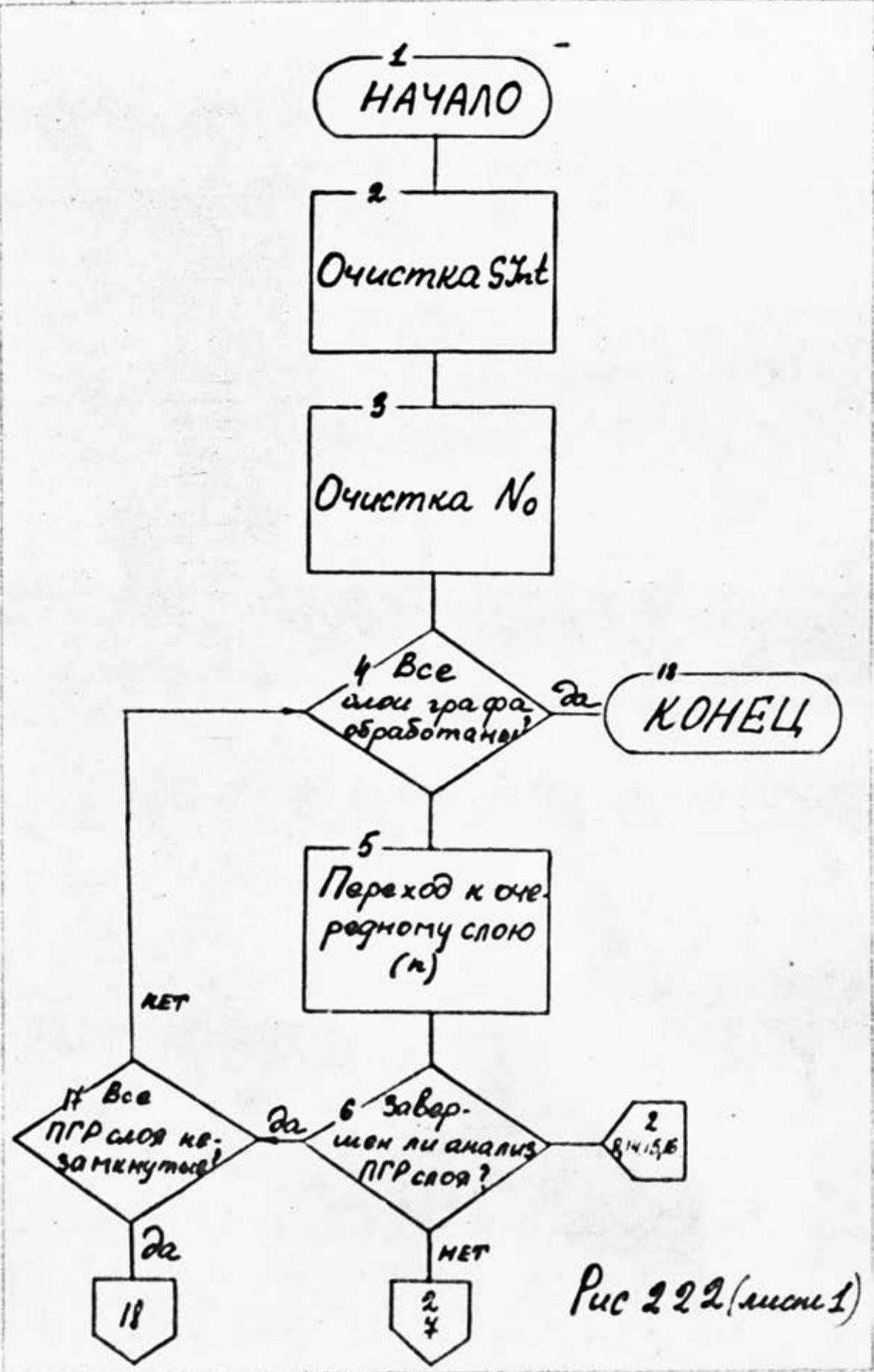


Рис 222 (лист 1)

15.2.86  
Т.П.Зенд

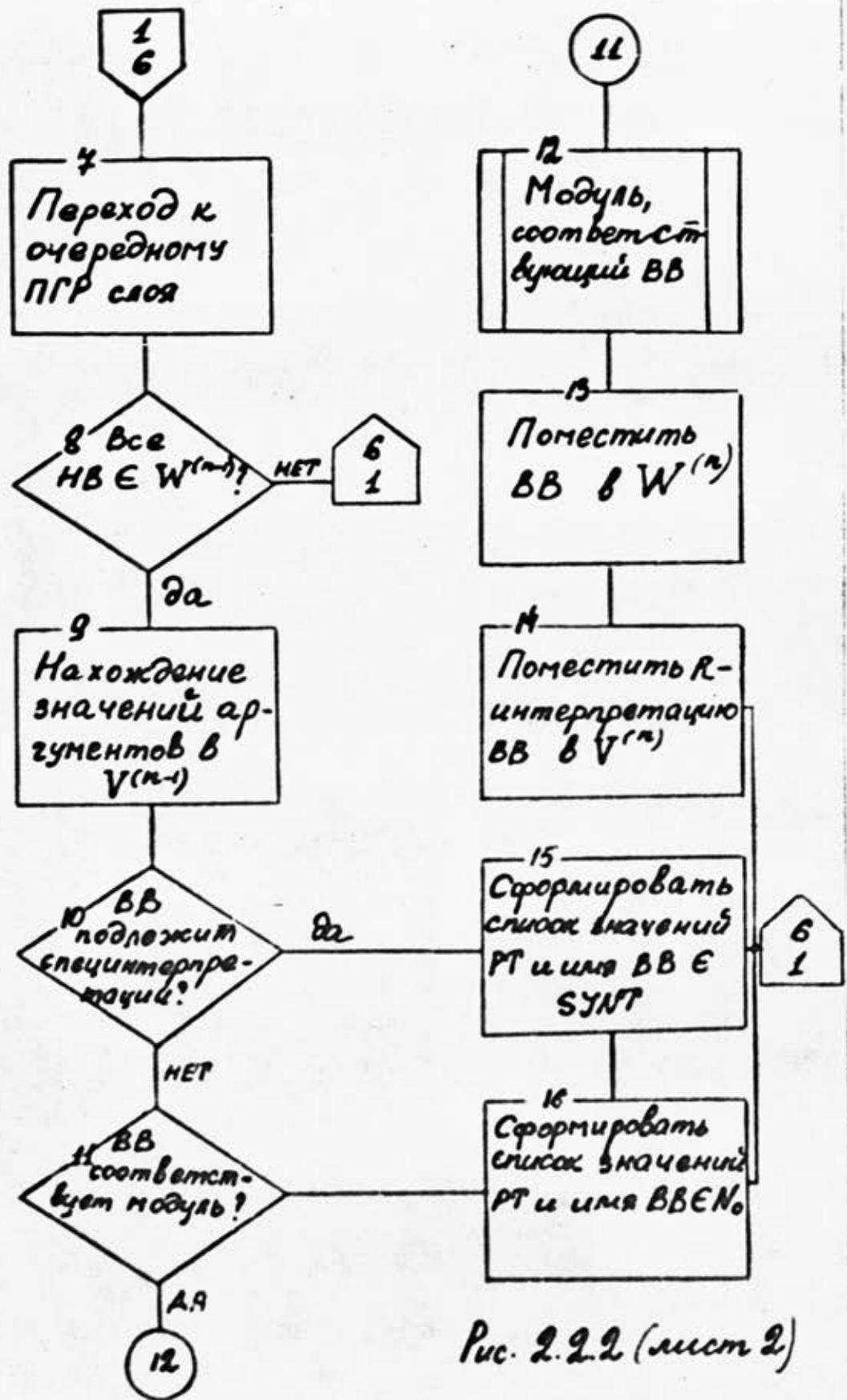


Рис. 2.2.2 (лист 2)

Блок 9. Поиск значения аргументов в каталоге значения в предшествующих слоях графа.

Блок 10. Проверка, подлежит ли верхняя вершина анализируемого ПГР специальной  $R$ -интерпретации, осуществляется по проверке кода идентификатора модуля, отвечающего ВВ ПГР, в случае специнтерпретации - код идентификатора содержит признак специнтерпретации.

Блок 11. Проверка, соответствует ли ВВ ПГР модуль из библиотеки модулей по коду идентификатора модуля в ПГР.

Блок 12. Вызов модуля, соответствующего верхней вершине подграфа, в качестве исходных данных модуль передается значения  $R$ -интерпретации нижних вершин ПГР, в качестве результата получается значение  $R$ -интерпретации верхней вершины.

Блок 13,14. Помещение ВВ ПГР в каталог  $W$  ( в его очередной слой) имен конституэнт, получивших  $R$ -интерпретацию, а самой  $R$ -интерпретации - в каталог  $V$  ( в его текущую очередь) значений конституэнт, получивших  $R$ -интерпретацию.

Блок 15,16. Если ВВ ПГР не подлежит автоматической  $R$ -интерпретации, то формируется список значений разрешающих термов ( в случае, когда  $ВВ \in N_0$ , это будут все аргументы) и имя конституэнты, отвечающей ВВ. в массиве ЗАСМ.

Блок 17. Анализ подграфов слоя на незамкнутость, т.е. есть ли хоть одна верхняя вершина ПГР слоя, принадлежащая  $W^{(n)}$  ( соответствующий подграф называется замкнутым), если есть переход на блок 4, в противном случае - на блок 18.

Результатом работы модуля является  $R$ -интерпретация модулями библиотеки тех конституэнт графа  $R$ -интерпретации, которая возможна при заданных значениях ранее проинтерпретированных конституэнт и заданной библиотеке модулей, и формирование списка конституэнт, подлежащих специальной  $R$ -интерпретации, и значений их разрешающих термов и списка конституэнт, не обеспеченных вычисляемыми их модулями, со значениями аргументов этих конституэнт.

### 2.3. Описание алгоритмов модулей тела пакета

Тело пакета R-интерпретации содержит следующие модули:

- модуль формирования графа конституэнт ;
- модуль формирования нижнего замыкания ;
- модуль удаления подграфа ;
- модуль управления детализацией графа ;
- модуль формирования графа по формуле ;
- модуль операционного замыкания графа ;
- модуль приведения графа к рассленному виду ;
- модуль удаления дублирующихся подграфов ;
- модуль выделения нижних вершин графа ;
- модуль автоматической R-интерпретации.

Введем понятия, необходимые для описания алгоритмов модулей.

Форматизированным представлением рода структуры называется такое его представление в виде упорядоченного набора конституэнт<sup>ж)</sup>, при котором имена всех конституэнт, входящих в выражения для данной конституэнты, должны предшествовать имени данной конституэнты.

Роду структуры сопоставляется граф конституэнт, распределение которого дано в п.2.1.

Полным подграфом длины I ориентированного графа называется подграф, содержащий все вершины, непосредственно предшествующие данной вершине<sup>жж)</sup> и соединяющие их ребра.

Форматизированным представлением ориентированного графа, вершинам которого приписаны имена<sup>жжж)</sup>, называется такое его представление в виде упорядоченного набора полных подграфов длины I, при котором имя нижней вершины некоторого подграфа не может совпадать с именем верхней вершины этого подграфа или следующего подграфа.

<sup>ж)</sup> Описание конституэнт рода структуры приводится в п.1.1.

<sup>жж)</sup> См. определение 3 данного пункта.

<sup>жжж)</sup> Описание массива ГК, соответствующего этому представлению, приводится в

15-2-76  
ТПЗк1

В дальнейшем направление обхода подграфов, совпадающее с их упорядоченностью, называется "снизу вверх", а противоположное "сверху вниз".

Расслоенное представление ориентированного графа без циклов<sup>\*)</sup> является усилением форматизированного представления.

Определение 1. Первым слоем графа называется упорядоченное множество всех полных подграфов длины  $I$  (или длины  $0$ ) с единственной конечной вершиной, начальные вершины которых являются начальными вершинами графа.

Определение 2.  $n$ -ый слой ( $n > 1$ ) графа называется упорядоченное множество всех полных подграфов длины  $I$  с единственной конечной вершиной, начальные вершины которых являются конечными вершинами предыдущих слоев.

Подграфы, фигурирующие в приведенных определениях форматизированного и расслоенного представлений, всюду в описаниях алгоритмов будут называться просто подграфами (кроме специально оговоренных случаев).

Определим понятие нижнего замыкания.

Определение 3. Вершина  $u$  ориентированного графа предшествует вершине  $v$ , если существует упорядоченный путь  $(u_0, u_1, \dots, (u_{k-1}, u_k))$  такой, что  $u_0 = u$ ,  $u_k = v$ . В случае  $k = 1$  говорят, что вершина  $u$  непосредственно предшествует вершине  $v$ .

Определение 4. Нижним замыканием некоторого подмножества вершин ориентированного графа называется минимальный подграф, содержащий это множество и все предшествующие ему вершины.

Графы, обрабатываемые модулями, могут быть в форматизированном или расслоенном представлении. Модули, обрабатывающие граф в форматизированном представлении, должны обрабатывать граф и в расслоенном представлении.

На всех блок-схемах приняты следующие обозначения и сокращения:

ПР - подграф;

ВВ - верхняя вершина;

<sup>\*)</sup> См. описание массива ГК в п. 1.1

- НВ - нижняя вершина ;
- REL<sup>\*</sup>) - множество всех вершин графа ;
- Г<sup>ж</sup>) - граф ;
- Φ - пустой граф (т.е. не содержащий подграфов).

2.3.1. Описание алгоритма модуля формирования графа конституэнт.

Описание приводится в § 6.2.

2.3.2. Описание алгоритма модуля формирования нижнего замыкания.

Модуль предназначен для формирования по графу Г и подмножеству его вершин R<sub>Jnt</sub> нижнего замыкания Г<sub>R<sub>Jnt</sub></sub> множества вершин R<sub>Jnt</sub> и множества REL имен вершин графа Г<sub>R<sub>Jnt</sub></sub>.

Исходные данные:

- 1-й аргумент - список конституэнт R<sub>Jnt</sub>
- 2-й аргумент - граф Г .

Блок-схема представлена на рис. 2.31.

В блок-схеме приняты следующие обозначения:

j - счетчик подграфов в графе.

Опишем отдельные блоки алгоритма.

Блок 2. Контроль аргументов (может производиться по ключу).  
 а) граф Г должен быть представлен в форматизированном представлении.

Блок 3. Присвоение начального значения списку REL.

Блок 4. Граф Г открывается как входной.

Блок 5. Очистка счетчика подграфов j .

Блок 6. Переход к очередному подграфу. Подграфы анализируются сверху вниз.

Блок 7. Проверка верхней вершины очередного подграфа на принадлежность формируемому списку REL.

Блок 8. Пополнение списка REL нижними вершинами очередного подграфа.

\*) Возможно с уточняющим индексом.

15-2-76  
 ГП-3кл1

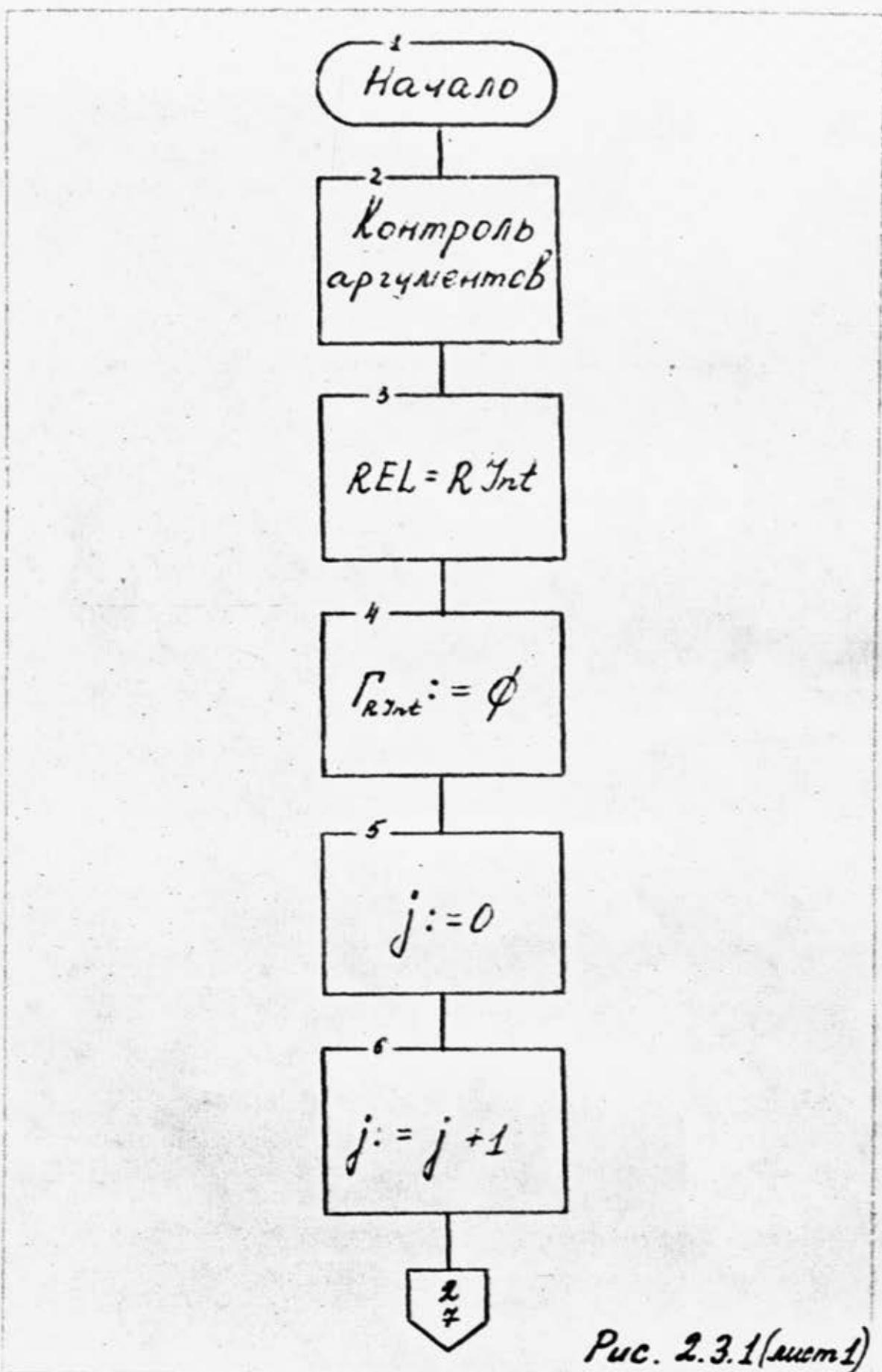
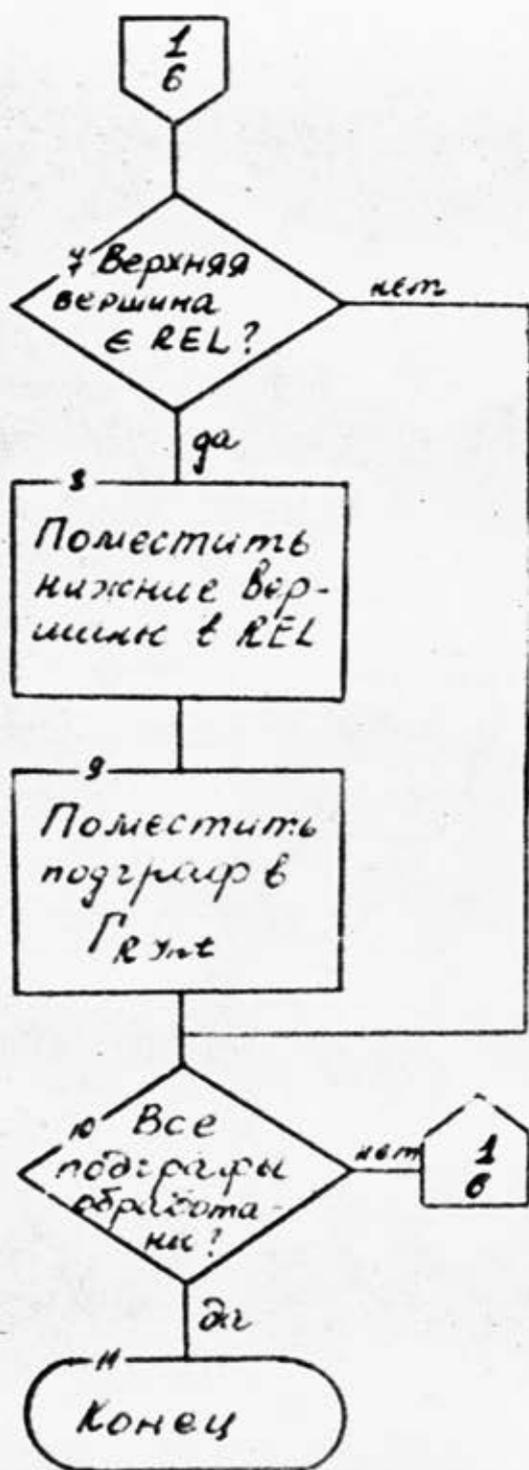


Рис. 2.3.1 (лист 1)



Блок 9. Пополнение формируемого графа  $\Gamma_{RInt}$  очередным подграфом.

Блок 10. Проверка на окончание работы.

Результатом работы модуля является граф  $\Gamma_{RInt}$  и список REL

### 2.3.3. Описание алгоритма модуля удаления подграфа.

Модуль предназначен для удаления из графа  $\Gamma$  по списку конституэнт специальной  $R$ -интерпретации  $SInt$  с разрешающими терминами и списку конституэнт  $RInt$  некоторого подграфа<sup>\*)</sup>, описываемого ниже при описании блок-схемы.

Исходные данные:

- 1-й аргумент - список конституэнт  $SInt$  с разрешающими терминами ;
- 2-й аргумент - список конституэнт  $RInt$  ;
- 3-й аргумент - граф  $\Gamma$  в форматизированном представлении.

Контроль аргументов (включенный при необходимости):

- 1) список  $SInt$  должен быть упорядочен в соответствии с упорядоченностью верхних вершин подграфов графа  $\Gamma$  ;
- 2) граф  $\Gamma$  должен быть представлен в форматизированном представлении<sup>\*\*)</sup>

При работе модуля формируется каталог вершин (КВ), в котором каждой вершине приписывается одно из трех значений 0, 1, 2:

- 0 - вершина возможно неудаляемая ;
- 1 - вершина возможно удаляемая ;
- 2 - вершина обязательно сохраняемая.

Блок-схема алгоритма представлена на рис. 232.

В блок-схеме приняты следующие обозначения:

$ST$  - рабочее поле для хранения очередной конституэнты списка  $SInt$  ;

\*) В данном пакете подграф является произвольным подграфом исходного графа в смысле теории графов.

\*\*) Описание форматизированного представления графа приводится в

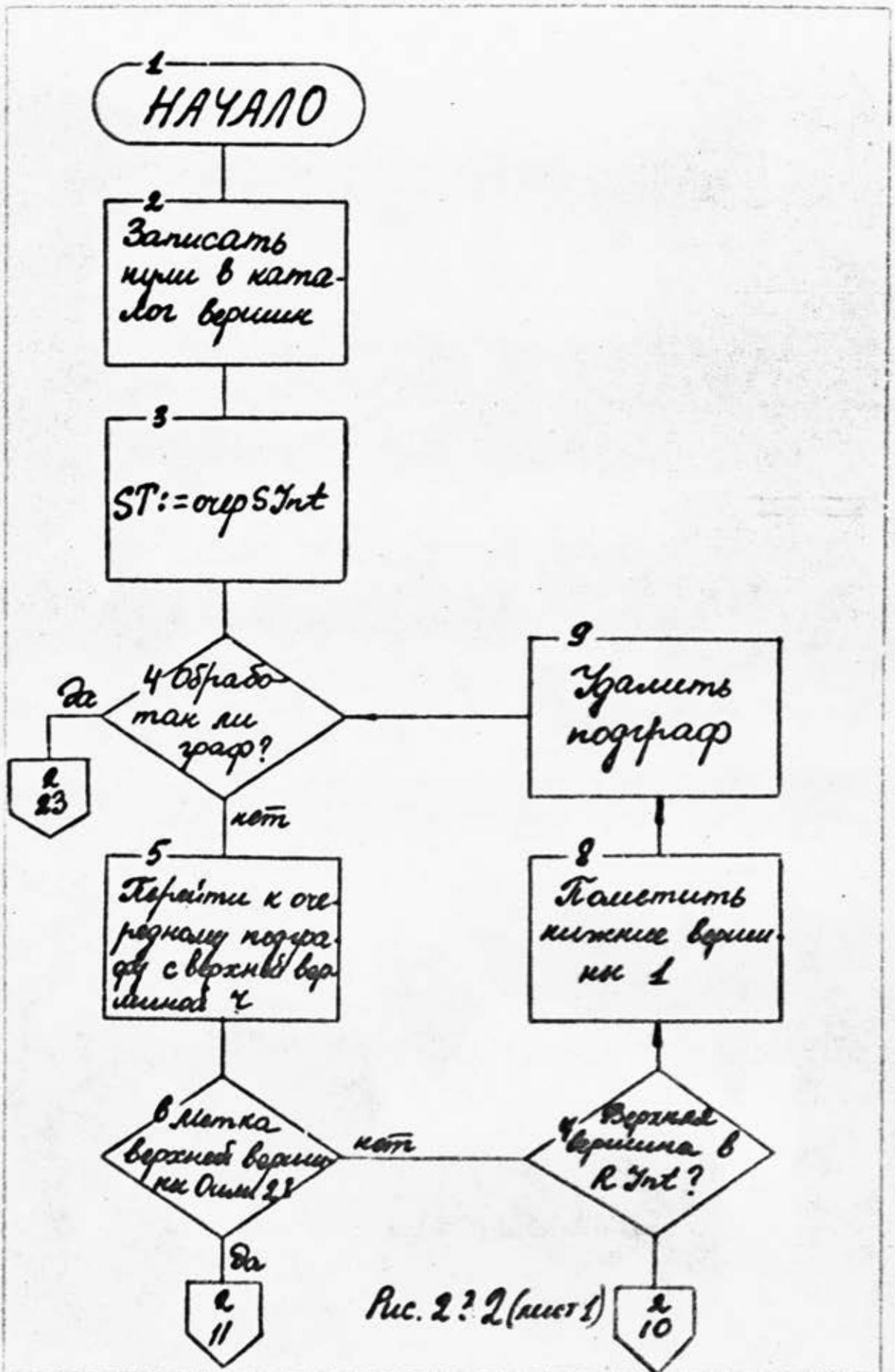


Рис. 2? 2 (лист 1)

15.2.76  
ТПТЗМЛ

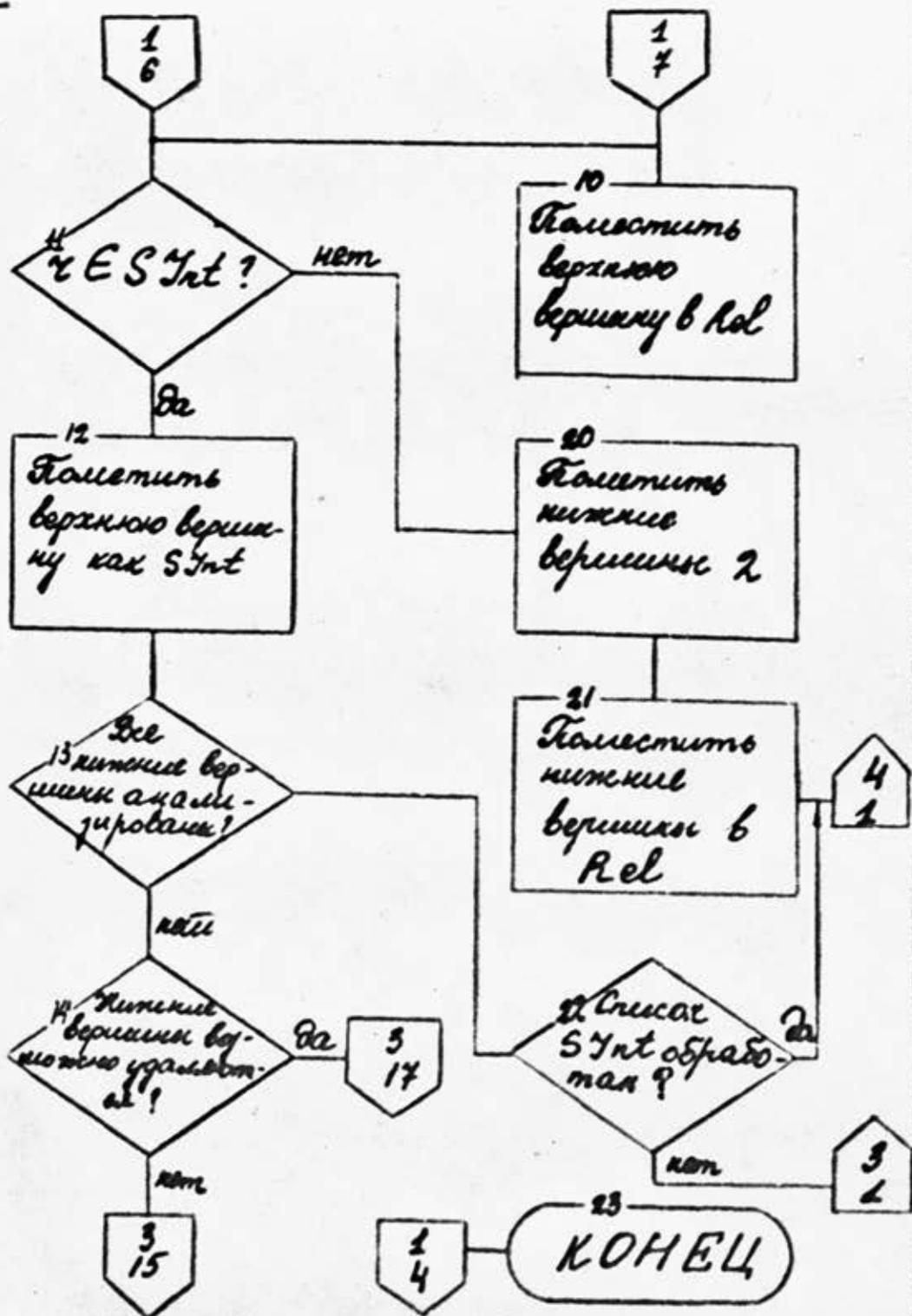


Рис. 2.3.2 (мист)



Рис. 23.2 (машин 3)

$\zeta$  - верхняя вершина (ВВ) подграфа (ПГР);  
 KB - каталог вершин.

Опишем отдельные блоки алгоритма:

- Блок 2. Очистка каталога вершин нулями.  
 Блок 3. Помещение в рабочее поле  $S^T$  очередной сверху вниз конституэнты списка  $S^{Int}$ .  
 Блок 4. Проверка на окончание работы.  
 Блок 5. Переход к очередному подграфу, верхняя вершина которого обозначается  $\zeta$ , анализ подграфов производится сверху вниз.  
 Блок 6. Проверка метки верхней вершины по KB.  
 Блок 7. Проверка принадлежности ВВ списку  $R^{Int}$ , имеющему более высокий приоритет.  
 Блок 8. Пометка в KB нижних вершин анализируемого ПГР как возможно удаляемых.  
 Блок 9. Удаление подграфа (т.е. подграф не переписывается в формируемый граф).  
 Блок 10. Помещение верхних вершин в  $REL$ .  
 Блок 11. Проверка верхней вершины  $\zeta$  на совпадение с заданной конституэнтной списка  $S^{Int}$ .  
 Блок 12. Операция в представлении подграфа отмечается как подлежащая специальной  $R$ -интерпретации.  
 Блок 13. Проверка завершения цикла по нижним вершинам ПГР.  
 Блок 14. Проверка нижней вершины ПГР по списку разрешающих термов для верхней вершины ПГР, если ВВ не попадает в список, то она является возможно удаляемой.  
 Блок 15. Пометка ВВ в KB значением 2.  
 Блок 16. Помещение нижней вершины ПГР в формируемый список  $REL$ .  
 Блок 17. Проверка по KB метки ВВ ПГР на значение 2.  
 Блок 18. Пометка нижней вершины ПГР значением 1 в KB.  
 Блок 19. Удаление из ПГР связи между верхней вершиной и рассматриваемой нижней вершиной.  
 Блок 20. Пометка ВВ меткой 2 в KB.  
 Блок 21. Помещение ВВ в список  $REL$  (если она в ней не встречается).  
 Блок 22. Проверка окончания обработки списка  $S^{Int}$ .

Результатом работы модуля является граф  $G^{Int}$  и список

его вершина  $REL_{\text{int}}$ .

#### 2.3.4. Описание алгоритма модуля управления детализацией графа.

Модуль предназначен для управления детализацией графа по списку  $SA_n$  конститuent, выражениям которых сопоставляются графы связей аргументов и операций, составляющих это выражение.

Исходные данные:

- 1-й аргумент - список конститuent  $SA_n$ ;
- 2-й аргумент - граф  $\Gamma$  в форматизированном представлении.

Блок-схема алгоритма представлена на рис. 2.3.3.

В блок-схеме приняты следующие обозначения:

МФГРФ - модуль формирования графа по формуле

Опишем отдельные блоки алгоритма.

- Блок 2. Ввод списка  $SA_n$ . Контроль: список  $SA_n$  не должен содержать конститuent, подлежащих специнтерпретации.
- Блок 3. Проверка на окончание детализации графа.
- Блок 4. Переход к очередному подграфу графа, анализ графа производится сверху вниз.
- Блок 5. Проверка конститuent на необходимость обработки модулем МФГРФ.
- Блок 6. Поиск в ГРС конститuent, соответствующей рассматриваемой верхней вершине. ГРС анализируется в одном направлении с исходным графом и должен располагаться на другом носителе.
- Блок 7. Передача найденной конститuent и ее выражения в ГРС модулю формирования графа по формуле<sup>\*)</sup>.
- Блок 8. Замена подграфа исходного графа, соответствующего рассматриваемой конститuent, на граф, полученный в результате работы МФГРФ в формируемом "детализированном" графе.

Результатом работы данного модуля является детализированный граф  $\Gamma$ .

\*)

Описание алгоритма этого модуля приводится в п. 2.3.5.

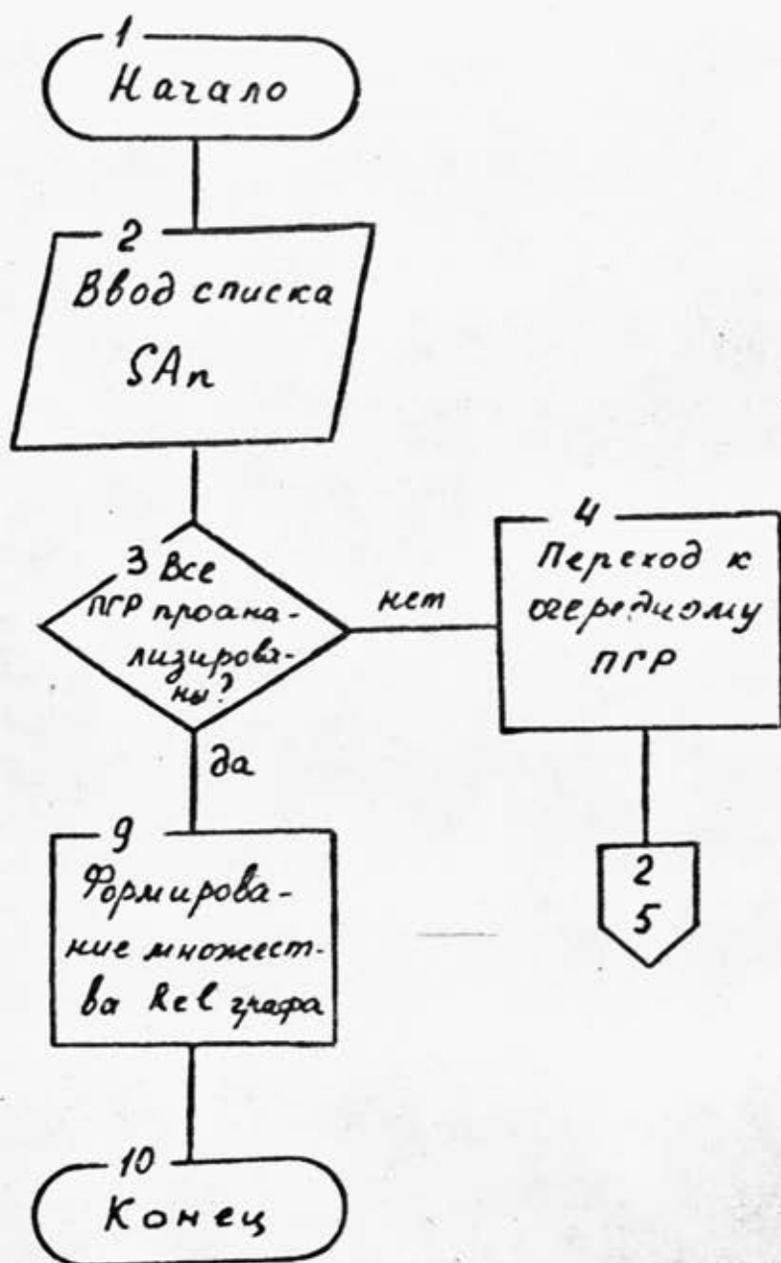
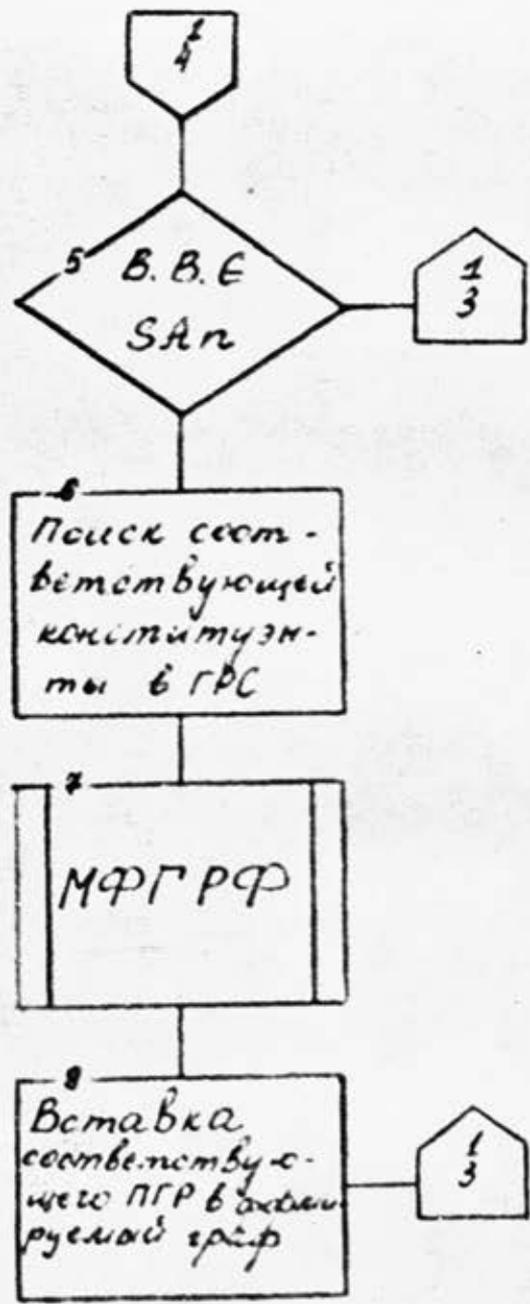


Рис. 2.3.3. (лист 1)



### 2.3.5. Описание алгоритма модуля формирования графа по формуле.

Модуль предназначен для построения по конституенте рода структуры графа связей аргументов и операций, составляющих выражение этой конституэнты.

Исходные данные:

I-й аргумент - конституэнты рода структуры и ее выражение.

Блок-схема алгоритма представлена на рис. 2.3.4.

Опишем отдельные блоки алгоритма.

- Блок 2. Ввод конституэнты и ее выражения. Предполагается, что выражение для конституэнты является правильно построенной формулой.
- Блок 3. Построение на нулевом уровне начального подграфа с исходной конституентой в качестве единственной нижней вершины, этот уровень и начальный подграф являются рабочими для модуля и в окончательный граф не попадает.
- Блок 4. Переход к очередному уровню (в начале работы - это нулевой уровень). Уровни в данном алгоритме отражают глубину анализа и способ организации анализа.
- Блок 5. Контроль на анализ всех подграфов уровня.
- Блок 6. Переход к очередному подграфу уровня.
- Блок 7. Контроль на анализ всех НВ.
- Блок 8. Переход к очередной НВ.
- Блок 9. Контроль конституэнты на терминальность, т.е. отсутствие у нее выражения. Для графа терминальность конституэнт означает, что соответствующая вершина является начальной.
- Блок 10. Поиск выражения конституэнты соответствующей рассматриваемой НВ.
- Блок 11. Выделение внешней операции выражения и ее аргументов. Реализация зависит от языка, на котором записано выражение конституэнты.
- Блок 12. Запоминания конституэнт-аргументов и их выражений. Для идентификации нетерминальных конституэнт вводятся промежуточные обозначения (промежуточные понятия).

15.2.26

ТПтЗкн1

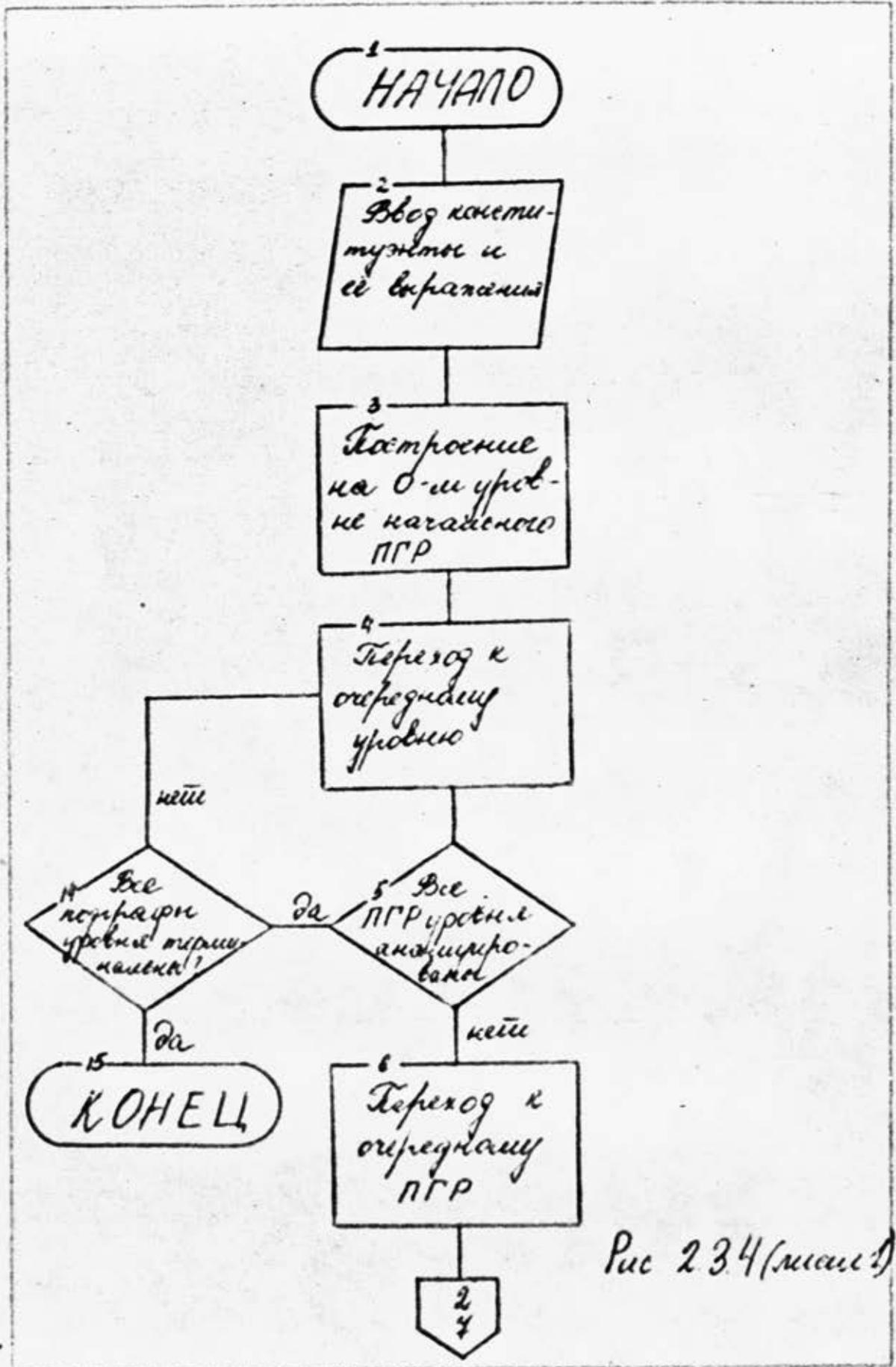


Рис 2.3.4 (мисс.)

15.2.86  
ГПТЗ м 1

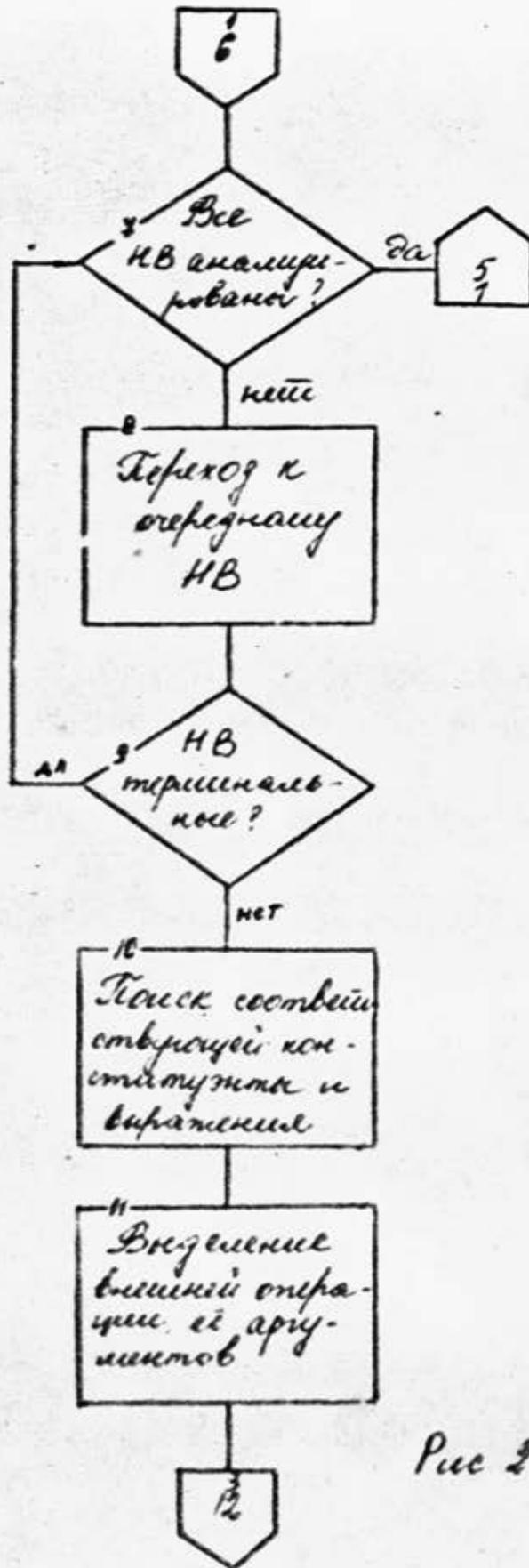


Рис 234 (лист 2)

15.2.26  
ТПТЗКН1



Рис. 23.4 (лист 3)

При этом должна быть обеспечена такая система обозначений промежуточных понятий для различных конституэнт, передаваемых в данный модуль модулем управления детализацией графа, чтобы все промежуточные понятия (конституэнты) имели различные имена. В частности, можно ввести новую стандартную букву В для таких понятий (конституэнт) и передавать через модуль управления детализацией графа число промежуточных понятий, уже имеющихся в графе.

Блок 13. Построение соответствующего подграфа следующего уровня. В этом подграфе верхняя вершина соответствует анализируемому конституэнту, а нижние - ее конституэнтам-аргументам.

Блок 14. Контроль терминальности всех подграфов уровня, здесь под терминальным подграфом понимается ШР, все нижние вершины которого терминальны.

Результатом работы модуля является граф, соответствующий конституэнту и отражающий структуру ее выражения.

### 2.3.6. Описание алгоритма модуля операционного заполнения графа.

Модуль предназначен для операционного заполнения графа, т.е. приписывания некоторым конституэнтам графа вычисляющих их значения модулей.

Исходные данные:

- 1-й аргумент - граф конституэнт;
- 2-й аргумент - список  $Alg$  идентификаторов модулей (соответствующих некоторым конституэнтам графа) и их аргументов-конституэнт.

Блок-схема алгоритма представлена на рис. 2.3.5.

Результатом работы модуля является граф конституэнт, в соответствующих подграфах которого записаны идентификаторы модулей (приписанные их верхним вершинам), а конституэнты переставлены в соответствии с порядком в списке аргументов в  $Alg$ .

15.2.76

Г.П.Зем 1

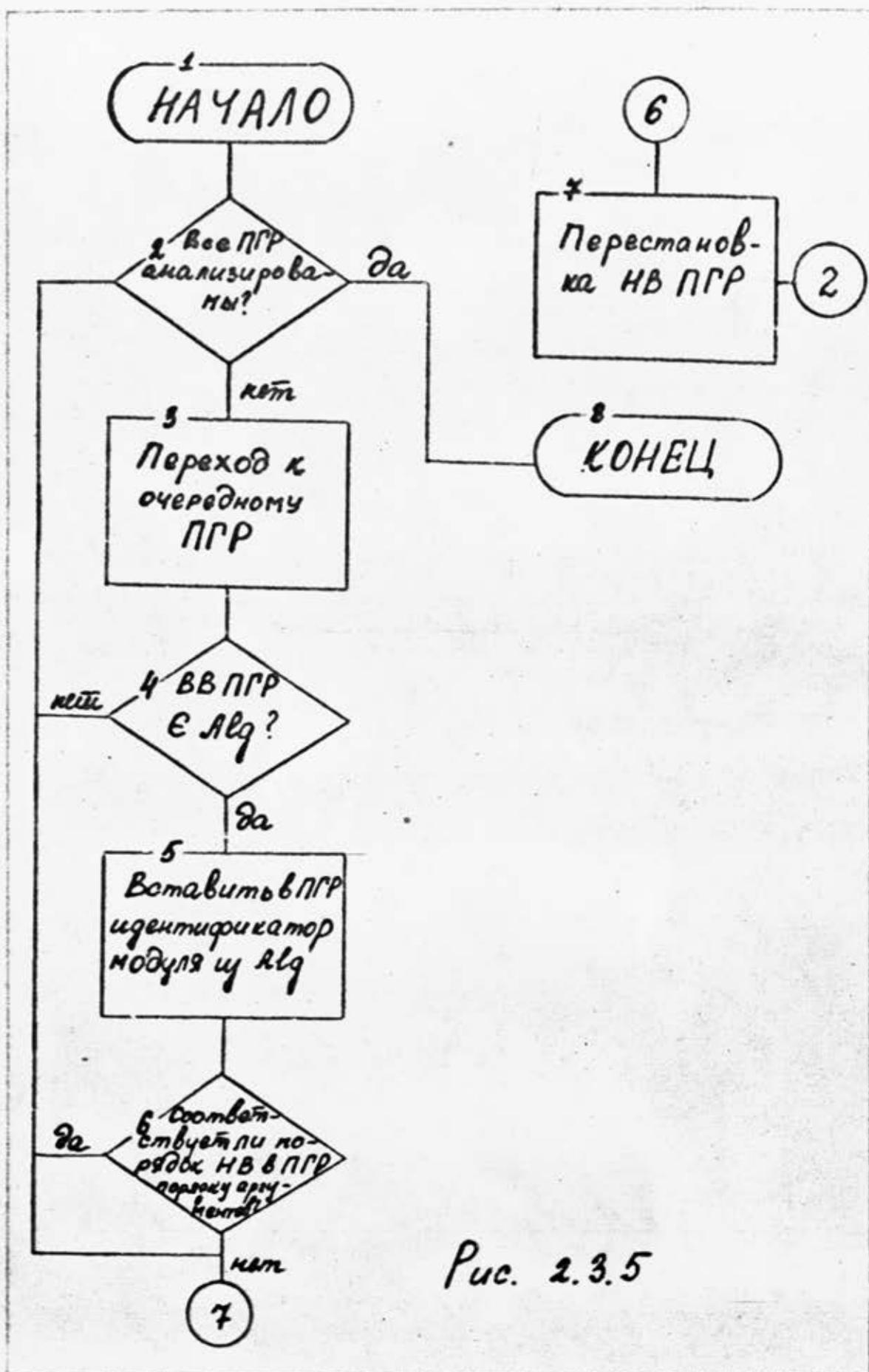


Рис. 2.3.5

### 2.3.7. Описание алгоритма модуля приведения графа к расслоенному виду.

Описание приводится в § 6.2.

### 2.3.8. Описание модуля удаления дублирующихся подграфов.

Модуль предназначен для удаления в графе в расслоенном представлении дублирующихся подграфов.

Исходные данные:

I-й аргумент граф в расслоенном представлении.

Блок-схема алгоритма представлена на рис. 2.3.6.

В блок-схеме алгоритма приняты следующие обозначения:

ТС — таблица соответствия;

C — признак того, что конституэнта, соответствующая верхней вершине ПР, является стандартной ( $C = 1$ ) или нет ( $C = 0$ );

V1, V2 — рабочие ячейки для обработки ТС;

КОПГ — каталог обработанных графов.

Таблица соответствия (ТС) имеет следующую структуру:

Имя	$V_1$	$W_1$	РП	...	$V_k$	$W_k$	РС	$V_{k+1}$	$W_{k+1}$	...	...	...	ПК
-----	-------	-------	----	-----	-------	-------	----	-----------	-----------	-----	-----	-----	----

где имя — имя рабочего поля;

$V_i$  — имя исходной верхней вершины;

$W_i$  — имя соответствующей ей верхней вершины, на которую заменяется имя  $V_i$  при перекодировке;

РП — разделитель пар ( $V_i, W_i$ )

РС — разделитель слоев (слои ТС соответствуют слоям исходного графа);

ПК — признак конца.

Каталог КОПГ представляет перечень вершин обработанных графов, организован также по слоям (между слоями — разделители слоев).

Опишем отдельные блоки алгоритма.

Блок 2. Открыть таблицу соответствия

Блок 3. Контроль завершения анализа слоев.

Блок 4. Переход к очередному слою. Анализ производится снизу вверх.

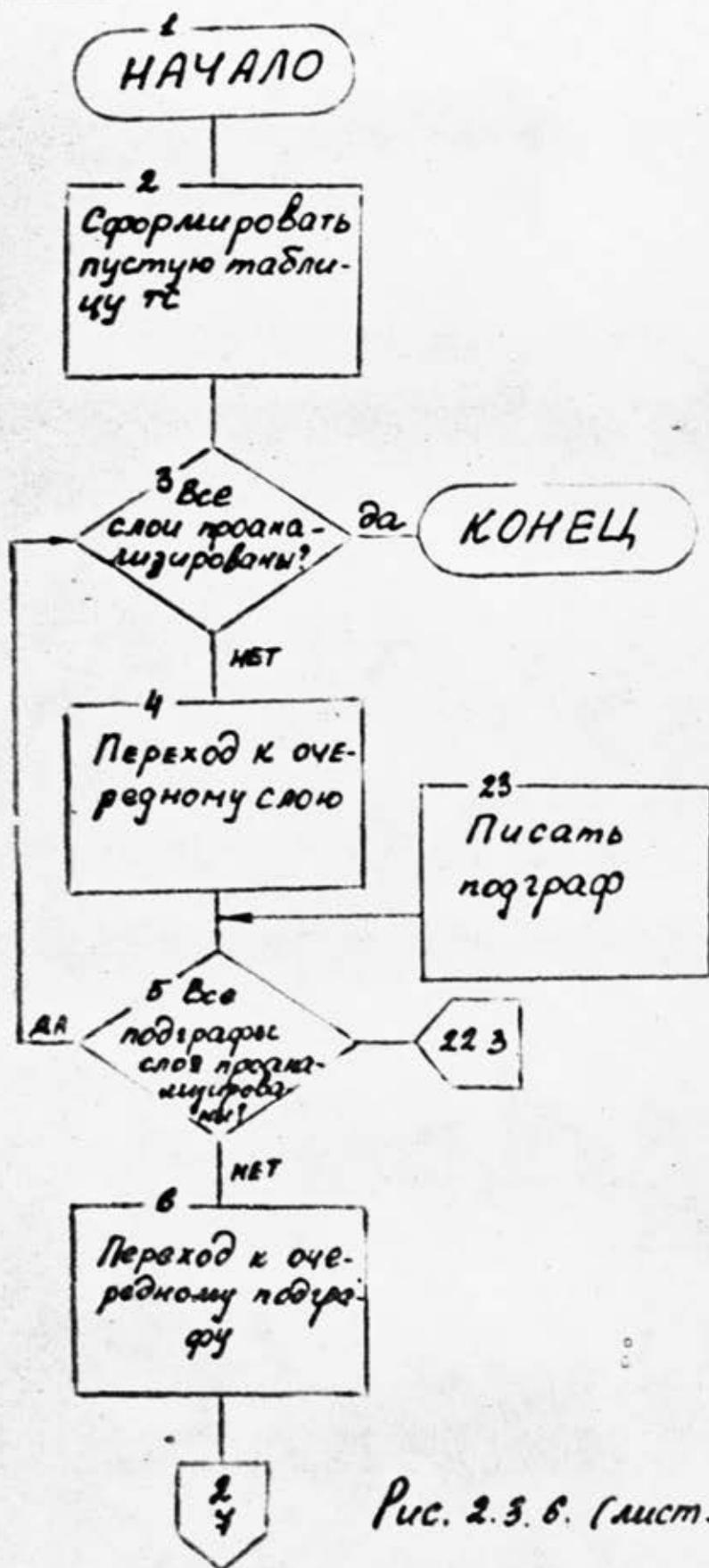


Рис. 2.3.6. (лист 1)

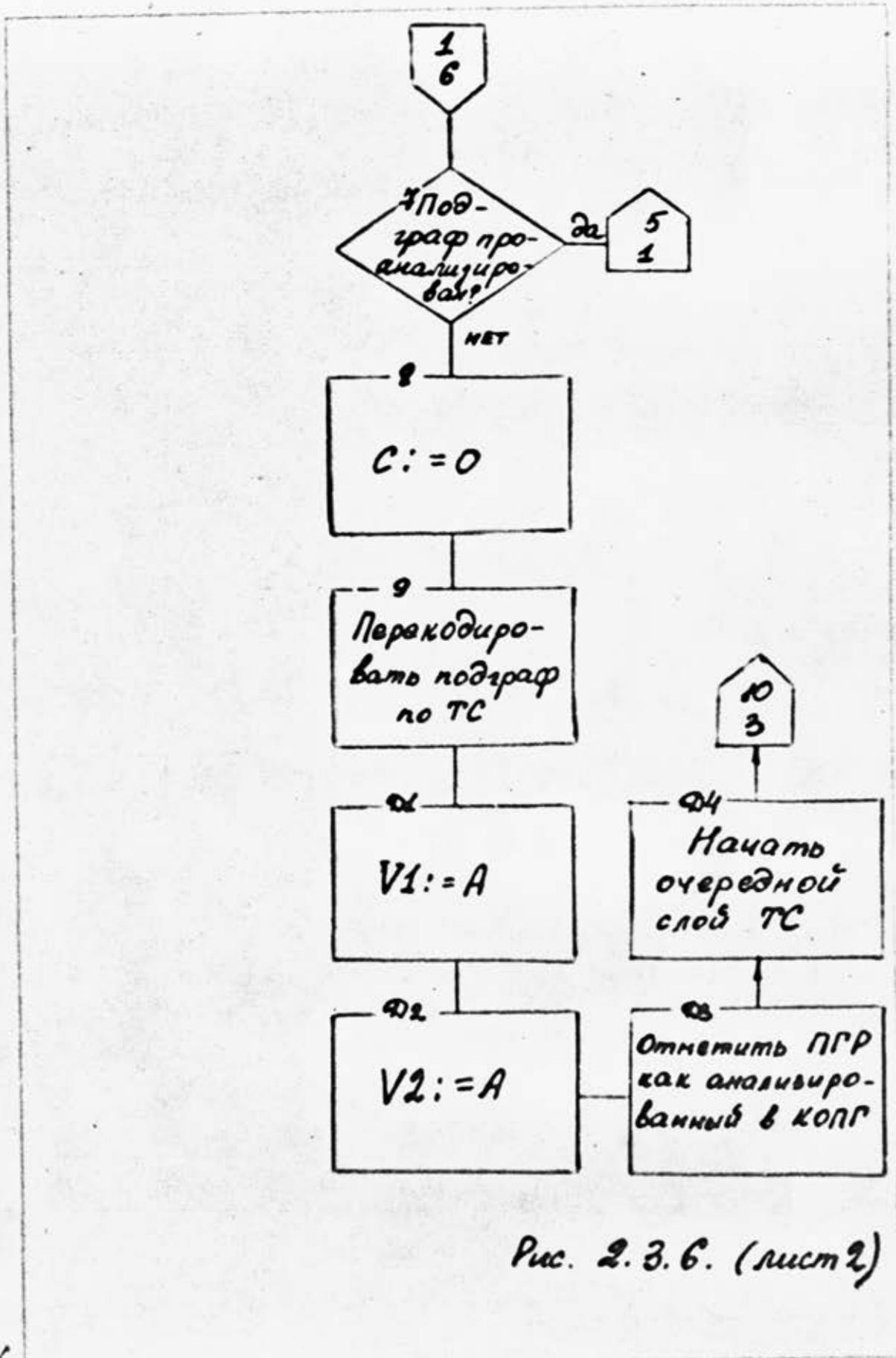


Рис. 2.3.6. (лист 2)

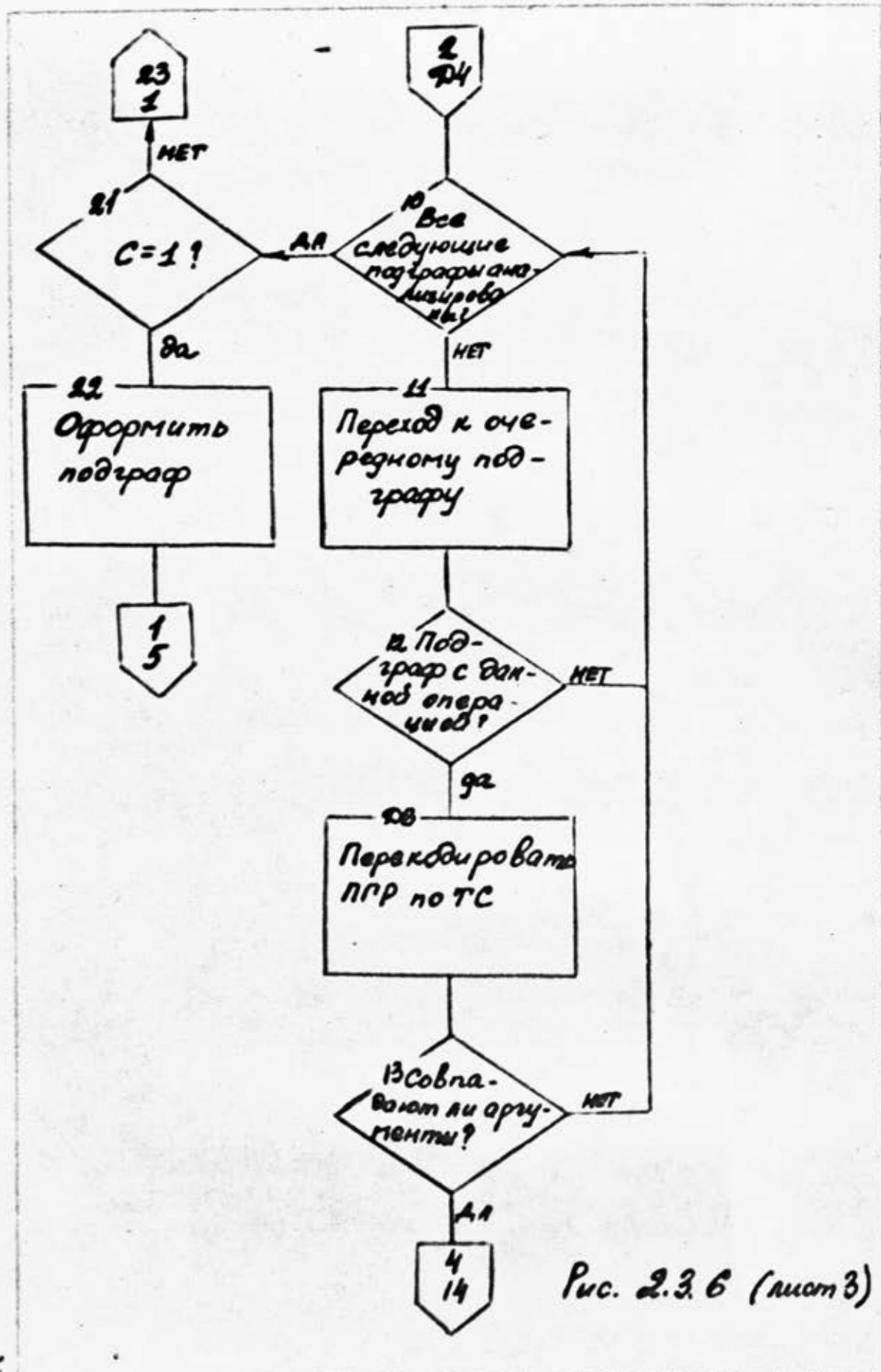


Рис. 2.3.6 (лист 3)

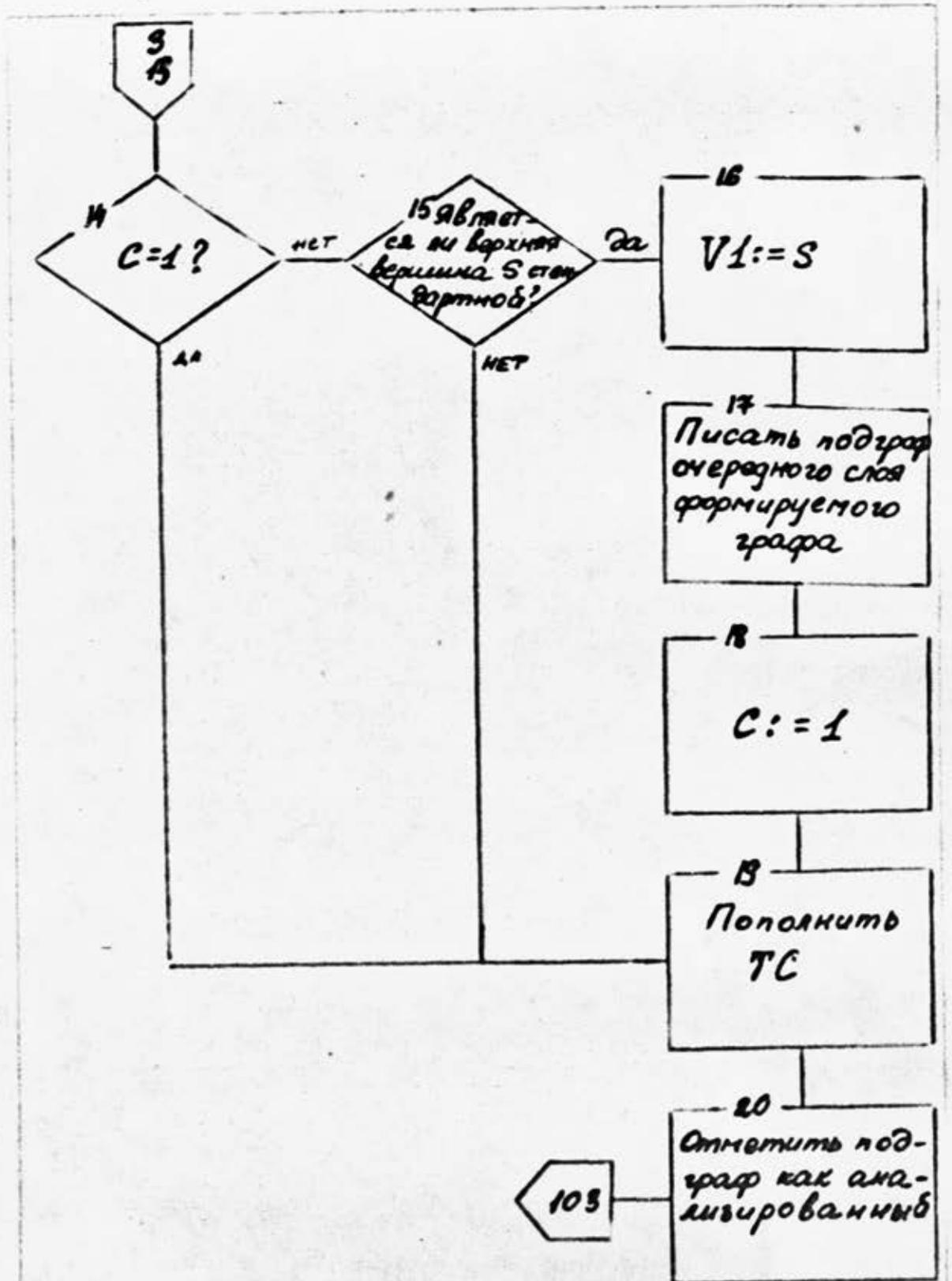


Рис. 2.3.6 (лист 4)

- Блок 5. Контроль завершения анализа ПГР.
- Блок 6. Переход к очередному подграфу данного слоя.
- Блок 7. Контроль подграфа: отмечен ли он как обработанный в КОПГ.
- Блок 8. Очистка признака  $C$ .
- Блок 9. Перекодировка ПГР по ТС, т.е. замена имен всех нижних вершин ПГР на соответствующие им имена по ТС.
- Блок Д1. Занесение в рабочую ячейку имени верхней вершины выделенного ПГР (определение выделенного ПГР дано в описании блока 10).
- Блок Д2. Занесение верхней вершины ПГР в КОПГ.
- Блок Д3. Начало очередного слоя ТС.
- Блок 10. Контроль всех подграфов, следующих за подграфом, выделенным блоками 6 и 7 (будем называть этот ПГР выделенным), отмечены ли они как обработанные в КОПГ.
- Блок 11. Переход к очередному ПГР.
- Блок 12. Проверка выделенного ПГР и очередного ПГР на совпадение операций.
- Блок Д6. Перекодировка ПГР по ТС с учетом только предыдущих слоев (формируемый слой не учитывается).
- Блок 13. Проверка указанных ПГР на совпадение операций.
- Блок 14. Контроль признака  $C$ . Признак  $C$  обеспечивает выполнение следующего условия: если среди всех сравниваемых подграфов есть ПГР со стандартной буквой, то оставляется именно этот подграф. В противном случае остается выделенный подграф.
- Блок 15. Проверка ВВ на стандартность.
- Блок 16. Запоминание стандартной вершины  $S$  для дальнейшей обработки ТС.
- Блок 17. Запись в формируемый граф ПГР со стандартной вершиной.
- Блок 18. Установка признака  $C$  на 1.
- Блок 19. Пополнение ТС, верхней вершине очередного ПГР ставится в соответствие  $V$ .
- Блок 20. Занесение ВВ подграфа в КОПГ.
- Блоки 21-23. Анализ признака  $C$ . Если  $C = 1$ , то в текущем слое ТС все соответствие  $(V, W)$  от пары  $(A, W)$  до пары  $(S, S)$  меняется на  $(V, S)$ .

Если  $C = 0$ , то переписывается выделенный подграф.

Результатом работы модуля является граф, в котором удалены дублирующиеся подграфы. Полученный граф называется графом R-интерпретации.

2.3.9. Описание модуля организации вычислительного процесса по вычислительной схеме - приводится в п.2.2.

2.3.10. Описание модуля автоматической R-интерпретации. Описание приводится в п. 2.3.

## 2.4. Характеристика информационных массивов ППП R-интерпретации и связь с другими ППП.

Пакет прикладных программ R - интерпретации использует следующие массивы:

1) массив РСД- массив, содержащий рода структур (базовые, промежуточные, главный) и дополнения, привязанные к вершинам операционной схемы;

2) массив ГК- массив, содержащий граф конститuent;

3) массив РЕЛ- массив, содержащий коды конститuent, отвечающих вершинам графа конститuent;

4) массив РИИТ- массив, содержащий абстрактное содержание проекта или множество *Rel*;

5) массив САН- массив, содержащий коды верхних вершин подграфов, подлежащих детализации;

6) массив СИИТ- массив, содержащий список конститuent, подлежащих специальной R-интерпретации и их разрешающие термины;

7) массив СМОД- массив, содержащий список конститuent, подлежащих автоматической R-интерпретации, и список конститuent-аргументов;

8) массив ЗАСМ- массив, содержащий значения конститuent и задание на спецификацию и пополнение библиотеки модулей;

9) массив ЗНАЧР- массив, содержащий значение R-интерпретации конститuent;

10) массив КРВГ- каталог адресов R-интерпретированных конститuent.

### 2.4.1. Массив РСД.

Описание приведено в 1.4.2.

### 2.4.2. Массив ГК.

#### 1) Имя.

Имя массива ГК, употребляемое в программах, должно содержать идентификаторы проекта, варианта, номер поколения и код представления (Ф- форматизированное, Р- расслоенное).

Кроме того указывается число записей, число слоев (для расслоенного представления) и число промежуточных понятий, возникших при детализации графа.

И-2.76

ТПРЗки1

2) Информация, содержащаяся в записи массива.

Имеется 2 представления записей ГК:

- 1) форматизированное;
- 2) расслоенное.

В форматизированном представлении все записи одного типа. В расслоенном представлении кроме записей 1-го типа имеются записи 2-го типа.

Каждая запись 1-го типа массива ГК содержит информацию о подграфе графа конститuent. В записи содержится информация о конститuentaх, отвечающих верхней и нижним вершинам, и операции, отвечающей верхней вершине.

Каждая запись 2-го типа содержит информацию о номере слоя, в котором находятся все следующие за ней записи 1-го типа.

3) Организация массива.

Массив ГК имеет записи 2-х типов:

запись 1-го типа-переменной длины,

запись 2-го типа-фиксированной длины.

4) Метод доступа-индексно-последовательный или прямой.

5) Максимальный размер- 5000 записей.

6) Структура записи.

Структура записи 1-го типа показана на рис. 2.4.2.1.

Признак записи 1-го типа <sup>х)</sup>	Код конститuent, соответствующей верхней вершине	Код операции	Число аргументов	Код конститuent, отвечающей 1-й нижней вершине	Код конститuent, отвечающей последней нижней вершине

Рис. 2.4.2.1.

Структура записи 2-го типа<sup>х)</sup> показана на рис. 2.4.2.2.

Признак записи 2-го типа	Номер слоя

Рис. 2.4.2.2.

<sup>х)</sup> Отсутствует в форматизированном представлении.

## 7) Получение массива ГК.

Массив ГК получается как результат обработки записи массива РСД из ППП ГРС, соответствующей ГРС, модулем формирования графа конститuent, а также работы модулей формирования нижнего замыкания, модуля управления детализацией графа, модуля удаления подграфа, модуля приведения графа к расслоенному виду, модуля удаления дублирующихся подграфов, модуля операционного заполнения графа, при этом он получается в расслоенном представлении при работе модуля приведения к расслоенному виду и модуля удаления дублирующихся подграфов, при работе остальных модулей - вообще говоря, в форматизованном представлении (даже если входной массив ГК в модуль был введен в расслоенном представлении).

## 2.4.3. массив РЕЛ.

## 1) Имя.

Имя массива РЕЛ, употребляемое в программах, должно содержать идентификатор проекта и варианта и номер поколения.

## 2) Информация, содержащаяся в записях массива.

Запись массива содержит коды конститuent, отвечающих вершинам графа конститuent.

## 3) Организация массива.

Массив РЕЛ имеет одну логическую запись.

## 4) Метод доступа.

Не ограничивается.

## 5) Максимальный размер - 5000 полей записи.

## 6) Структура записи.

Число полей	Код конститuent	Код конститuent	...	Код конститuent
-------------	-----------------	-----------------	-----	-----------------

## 7) Получение массива РЕЛ.

Массив РЕЛ получается как выходной при работе модулей R - интерпретации или может быть получен по запросу модулем СУБД.

## 8) Использование массива РЕЛ в ППП R - интерпретации.

Массив РЕЛ используется как выходной массив для получения информации о множестве вершин графа конститuent.

## 9) Использование массива РЕЛ в других ППП.

Массив РЕЛ используется ППП внесения изменений при вне-

сении изменений в граф конститuent и ПИП перевода представлений при формировании запроса на перевод представлений.

#### 2.4.4. массив РИИТ.

##### 1) Имя.

Имя массива РИИТ, употребляемое в программах, должно содержать идентификатор проекта и варианта.

##### 2) Информация, содержащаяся в записях массива.

Запись массива содержит коды конститuent, которые образуют абстрактное содержание проекта, т.е. список  $\mathcal{I}nt$ .

##### 3) Организация массива.

Массив РИИТ имеет одну логическую запись.

##### 4) Метод доступа.

Не ограничивается.

##### 5) Максимальный размер - 500 полей записи.

##### 6) Структура записи.

Структура записи представлена на рис. 2.4.3.

Число полей	Код кон-stituenty	Код консти-туенты		Код консти-туенты
	5 байтов	5 байтов		5 байтов

##### 7) Получение массива РИИТ.

Массив РИИТ вносится в БД после следующих действий:

- ввода формы - в оперативную память;
- проверки правильности перфорации (контрольные суммы);
- запись (в случае отсутствия ошибок) форм в БД в виде массива.

8) Использование массива РИИТ в ППП R - интерпретации. массив РИИТ используется как входной массив в модуль формирования нижнего замыкания как список вершин графа, составляющих абстрактное содержание проекта.

##### 9) Использование массива РИИТ в других ППП.

Массив РИИТ используется ППП внесения изменений для внесения изменений в массив ГК.

#### 2.4.5. массив САН.

Структура совпадает со структурой массива РИИТ, приведен описание пунктов, отличное от описаний массива РИИТ.

##### 2) Информация, содержащаяся в записях массива.

Массив САН состоит из одной логической записи, содержащей коды конститuent и служит для представления списка  $S_{Int}$ . Подграфы массива, верхние вершины которых совпадают с кодами конститuent массива САН, подлежат детализации.

8) Использование массива САН в ППП R-интерпретации.

Массив САН используется как входной массив в модуль управления детализацией графа.

9) Использование массива САН другими ППП.

Массив САН используется ППП внесения изменений для изменения списка конститuent, подлежащих обработке модулем управления детализацией графа.

#### 2.4.6. Массив СИНТ.

1) Имя.

Имя массива СИНТ, употребляемое в модуле удаления подграфа, должно содержать идентификатор проекта и варианта.

2) Информация, содержащаяся в записях массива.

Запись массива содержит код конститuent, подлежащей специальной R-интерпретации, число разрешающих термов, и коды разрешающих термов (т.е. массив соответствует списку  $S_{Int}$ ).

3) Организация массива.

Массив состоит из записей переменной длины.

4) Метод доступа.

метод доступа к записям последовательный (если записи упорядочены в соответствии с упорядоченностью подграфов сверху вниз) или индексно-последовательный.

5) Максимальный размер - 500 записей.

6) Структура записи.

Структура записи представлена на рис. 2.4.5.

Код конститuent, подлежащей спец-интерпретации	Число аргументов	Код 1-го разрешающего терма	...	Код 1-го разрешающего терма
5 байтов		5 байтов		5 байтов

Рис. 2.4.5.

7) Получение массива СИНТ.

Массив СИНТ вводится в ЭД после следующих действий:

- ввода формы в оперативную память;

- проверка правильности перфорации (контрольные суммы);
- записи ( в случае отсутствия ошибок) формы - в БД в виде массива.

8) Использование массива СИИТ в ППП R-интерпретации. Массив СИИТ используется как входной в модуль удаления подграфа, для выделения вершин, подлежащих специальной R-интерпретации и их разрешающих термов.

9) Использование массива СИИТ в других ППП. Массив СИИТ используется ППП внесения изменения при внесении изменений в список конститuent, подлежащих специальной R-интерпретации, или в множество их разрешающих термов.

2.4.7. Массив СМОД.

Описание ряда пунктов совпадает с описанием массива СИИТ, приведем описания пунктов, отличных от описания массива ИИТ.

1) Имя массива СМОД, употребляемое в модуле операционного заполнения графа, должно содержать идентификатор проекта и варианта.

2) Информация, содержащаяся в записи массива.

Запись массива содержит код конститuent (которой приписывается вычисляющий ее модуль), идентификатор модуля, коды аргументов в том порядке, как требует модуль (т.е. *Arg*) массив соответствует списку.

6) Структура записи представлена на рис. 2.4.6.

Коды конститuent	Число аргументов	Код I-го аргумента	...	Код I2-го аргумента
------------------	------------------	--------------------	-----	---------------------

Рис.2.4.6.

8) Использование массива СМОД в ППП R-интерпретации. Массив СМОД используется как входной массив в модуль операционного заполнения графа.

9) Использование массива СМОД в других ППП. Массив СМОД используется в ППП внесения изменений при изменении соответствия между конститuentами и соответствующими им идентификаторами модулей.

2.4.8. Массив ЗАСИ.

1) Имя.

15.2.76  
ТПТЗм1

Имя массива ЗАСМ, употребляемых в программах, должно содержать идентификаторы проекта и варианта.

2) Информация, содержащаяся в записи массива.

Массив может быть входным и выходным. Если массив входной, то в записях массива записывается код конституэнты и ее значение для конституэнт, подлежащих специальной R-интерпретации, номер очереди R-интерпретации и номер слоя.

Если массив выходной, то в записях 2-го типа массива записывается код конституэнты и ее значение для разрешающих термов или аргументов конституэнты, код которой записывается в записи 1-го типа с признаком подлежит ли она специнтерпретации (признак S) или отсутствует вычисляющий ее модуль (признак N).

Эти признаки используются как значения соответствующих конституэнт.

3) Организация массива.

Массив имеет записи 2-х типов неопределенной длины.

4) Метод доступа-индексно-последовательный или прямой.

5) максимальный размер- 1000 записей.

6) Структура записи.

Признак записи 1-го типа <sup>x</sup>	Код конституэнты	Номер очереди ин-терпретации	Номер слоя	Значение конституэнты	Разделитель
---------------------------------------	------------------	------------------------------	------------	-----------------------	-------------

Рис. 2.4.8.1.

Признак записи 2-го типа	Код конституэнты	Значение конституэнты	Разделитель
--------------------------	------------------	-----------------------	-------------

Рис. 2.4.8.2.

1) Получение массива ЗАСМ.

Входной массив ЗАСМ вносится в БД при записи (в случае отсутствия ошибок) формы 6

<sup>x</sup> Отсутствует, если массив входной.

выходной массив ЗАСМ формируется при работе модуля автоматической R-интерпретации.

8) Использование массива ЗАСМ в ППП R-интерпретации.

Как входной массив ЗАСМ используется для ввода значений специальной R-интерпретации для работы модуля автоматической R-интерпретации.

Как выходной массив ЗАСМ используется как задание на специальную R-интерпретацию и пополнение библиотеки модулей автоматической R-интерпретации.

9) Использование массива ЗАСМ другими ППП.

Как входной массив ЗАСМ используется в ППП внесения изменений в специальную R-интерпретацию конститuent.

2.4.9. Массив ЗНАЧР.

1) Имя.

Имя массива ЗНАЧР, употребляемое в программах, должно содержать идентификатор проекта и варианта.

2) Информация, содержащаяся в записях массива.

В записях массива содержатся значения R-интерпретации конститuent, очередь R-интерпретации, номер слоя.

3) Организация массива.

Массив имеет записи неопределенной длины.

4) Метод доступа- индексно-последовательный или прямой.

5) Максимальный размер- 6000 записей.

6) Структура записей представлена на рис. 2.4.8. I.

Код конститuent	Номер очереди -интерпретации	Номер слоя	Значение конститuent	Разделитель
-----------------	------------------------------	------------	----------------------	-------------

7) Получение массива ЗНАЧР.

Массив ЗНАЧР формируется как выходной при работе модуля организации вычислительного процесса по вычислительной схеме и модуля автоматической R-интерпретации.

8) Использование массива ЗНАЧР в ППП R-интерпретации.

Массив ЗНАЧР используется как входной и выходной модули организации вычислительного процесса по вычислительной схеме и автоматической R-интерпретации для пополнения множества значений R-интерпретированных конститuent и при по-

15-2-76  
ТПЗин1

иске значения конституэнт.

9) Использование массива другими ППП.  
массив используется ППП внесения изменений.

2.4.10. массив КРВГ.

1) Имя.

Имя массива КРВГ, употребляемое в программах, должно содержать идентификаторы проекта и варианта.

2) Информация, содержащаяся в записях массива.

В каждой записи массива содержится номер очереди R-интерпретации, номер слоя, код конституэнта, получившей R-интерпретацию и адрес, по которому находится R-интерпретация данной конституэнта.

3) Организация массива.

Массив имеет записи фиксированной длины.

4) Метод доступа- индексно-последовательный или прямой.

5) Максимальный размер- 4000 записей.

6) Структура записи.

Структура записи представлена на рис. 2.4.8.

Код конституэнта	Номер очереди -интерпретации	Номер слоя	Адрес -интерпретации конституэнта
------------------	------------------------------	------------	-----------------------------------

Рис.2.4.8.

7) Получение массива КРВГ.

Массив КРВГ формируется при работе модуля организации вычислительного процесса по вычислительной схеме.

8) Использование массива КРВГ в ППП R-интерпретации.

Массив КРВГ используется для анализа, получения ли конституэнта R-интерпретации и для поиска ее значений.

9) Использование массива КРВГ в других ППП.

Массив КРВГ используется в ППП внесения изменений при внесении изменений в массив КРВГ.

## 2.5. Ограничения и возможности расширения ППП.

При программировании ППП R-интерпретации следует учитывать следующие ограничения:

число вершин графа R-интерпретации	$\leq 4000$
из них начальных вершин	$\leq 100$
число строк записи интерпретаций начальных вершин	$\leq 1000$ , в среднем 100

общее число автоматически интерпретируемых конституэнт	$\leq 3500$
--	-------------

число строк записи интерпретации всех конституэнт (вершин) графа R-интерпретации.	$\leq 3 \cdot 10^6$
---	---------------------

Существенной особенностью ППП R-интерпретации является открытость ППП по отношению к используемым методам (модулям автоматической R-интерпретации и специнтерпретации) и информационным массивам (и соответствующим модулям ввода и контроля СУБД). Открытость методов и массивов предъявляет к ППП R-интерпретации требование универсальности средств формирования вызова модулей и определение структуры массива.

При решении конкретных классов задач (например, формирование главного рода структуры) ряд требований может существенно упроститься (например, могут отсутствовать списки  $S_{int}$ , кроме нулевой очереди, и  $N_0$ ), а некоторые каталоги или массивы получить конкретизацию (например, массив ЗНАЧР, о котором в общем случае ничего нельзя сказать, может распаться на несколько массивов различной структуры и т.п.).

При работе модуля организации вычислительного процесса по вычислительной схеме и модуля автоматической R-интерпретации граф R-интерпретации задается в расслоенном представлении, что соответствует определенной организации вычислительного процесса, возможна и другая организация вычислительного процесса, например, по ветвям графа.

Одним из ограничений при использовании ППП R-интерпретации является возможное несоответствие между R-интерпретациями аргументов-конституэнт и требованиями к аргументам программных модулей, использующих их для вычислений. Предлагается следующая обшая схема.

При вводе списка *Synt* и операциональном заполнении графа идентификаторами модулей наряду с самими "разрешающими терминами" (в случае модулей это их аргументы) вводятся требования к аргументам и вид результата специнтерпретации или вычисления. Эти требования заносятся в специальный каталог. Перед началом обработки графа *R*-интерпретации модуль переориентации ссылок собирает по каждой конституэнте в каталоге требования от различных модулей и проектировщика на значение этих модулей и формируется новый каталог-каталог требований.

При специнтерпретации конституэнт 0-й очереди дается интерпретации, удовлетворяющие всем требованиям, либо части требований, но тогда для неудовлетворенных требований либо ППП внесения изменений в соответствующие места графа *R*-интерпретации вставляются модули-преобразователи имеющихся представлений конституэнты в требуемые, либо в особо сложных случаях они остаются непроинтерпретированными. В каталоге требований фиксируются "удовлетворенные" требования (либо непосредственно, либо за счет модулей-преобразователей). При автоматической *R*-интерпретации для модуля проверяется, есть ли интерпретации аргументов, удовлетворяющие его требованиям, если есть интерпретации всех конституэнт-аргументов, но часть из них не удовлетворяет требованиям соответствующего модуля или специнтерпретации, то выдается, наряду со списками *Synt* данной очереди, новый список таких конституэнт вместе с имеющимися интерпретациями их аргументов и список требований к соответствующим аргументам. Проектировщик либо сам дает необходимые представления конституэнт-аргументов, либо ППП внесения изменений в соответствующие места графа *R*-интерпретации вставляются модули-преобразователи имеющихся представлений конституэнты в требуемые, либо соответствующие аргументы в сложных случаях не получают требуемых представлений.

Аналогично обстоит дело с конституэнт, подлежащей специнтерпретации, но в этом случае, если были вставлены модули-преобразователи, то она будет выдана повторно с надлежащими представлениями значений аргументов для специнтерпретации.

В случае не конституэнт, не обеспеченных вычисляющими их модулями, эти модули вместе с модулями-трансформаторами или специнтерпретацией не соответствующих их требованиям аргументов "встраиваются" в вычислительный процесс.

Одновременно с этими процедурами пополняется каталог кон-  
ституэнт и их представлений и каталог требований.

В целом, контроль соответствия представлений конститу-  
энт требованиям модулей имеет целый ряд трудностей, отметим  
две из них:

1) возможность множества вариантов требований к аргумен-  
там и, соответственно, видов представлений значений; это приво-  
дит к увеличению числа проверок соответствия значений требо-  
ваниям;

2) целесообразность создания некоторого языка требова-  
ний для возможности автоматической проверки наличия требуе-  
мых представлений аргументов.

Необходимость анализа, видов представлений и требований,  
введения новых интерпретаций или модулей-преобразователей и  
частая перестройка графа  $R$  -интерпретации существенно ус-  
ложняют процесс  $R$  -интерпретации.

## 2.6. Средства программного контроля и контроля результатов.

Контроль информации, используемой в ППП R-интерпретации, состоит из следующих частей:

- 1) контроль СУБД,
- 2) контроль УПП,
- 3) контроль в соответствующих блоках модулей тела пакета.

При контроле, осуществляемом СУБД, проверяется правильность перфорации соответствующих форм (совпадение контрольных сумм). Кроме того, в СУБД имеются программные блоки, которые осуществляют контроль после внесения изменений в соответствующие входные данные ППП.

При вводе СУБД осуществляет также контроль правильности задания аргументов, например, при вводе списка разрешающих термов для модуля удаления подграфа проверяется, что список разрешающих термов входит в список имен нижних вершин подграфа, соответствующего конституэнте, подлежащей специнтерпретации.

УПП R-интерпретации (модуль организации вычислительного процесса) по вычислительной схеме осуществляет контроль наличия в библиотеке программ модулей, соответствующих идентификаторам в графе R-интерпретации и контроль полноты обработки графа R-интерпретации.

Контроль аргументов осуществляется блоками контроля модулей пакета и описывается в I.3.